Advances in Applied Econometrics: Binary Discrete Choice Models, Artificial Neural Networks, and Asymmetries in the FAST Multistage Demand System

Jason S. Bergtold

Dissertation submitted to the faculty of
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of:

Doctor of Philosophy
in
Economics

Committee:
Daniel B. Taylor (Co-Chair)
Aris Spanos (Co-Chair)
Everett B. Peterson
Anya M. McGuirk
Darrell J. Bosch
Christopher W. Zobel

April 14, 2004
Blacksburg, Virginia

Keywords: Logistic Regression, Bernoulli Regression Model, Artificial Neural Networks, Contingent Valuation, Demand Elasticities, Indirect Separability

Advances in Applied Econometrics: Binary Discrete Choice Models, Artificial Neural Networks, and Asymmetries in the FAST Multistage Demand System

Jason S. Bergtold

(Abstract)

The dissertation examines advancements in the methods and techniques used in the field of econometrics. These advancements include: (i) a re-examination of the underlying statistical foundations of statistical models with binary dependent variables. (ii) using feed-forward backpropagation artificial neural networks for modeling dichotomous choice processes, and (iii) the estimation of unconditional demand elasticities using the flexible multistage demand system with asymmetric partitions and fixed effects across time.

The first paper re-examines the underlying statistical foundations of statistical models with binary dependent variables using the probabilistic reduction approach. This re-examination leads to the development of the Bernoulli Regression Model, a family of statistical models arising from conditional Bernoulli distributions. The paper provides guidelines for specifying and estimating a Bernoulli Regression Model, as well as, methods for generating and simulating conditional binary choice processes. Finally, the Multinomial Regression Model is presented as a direct extension.

The second paper empirically compares the out-of-sample predictive capabilities of artificial neural networks to binary logit and probit models. To facilitate this comparison, the statistical foundations of dichotomous choice models and feed-forward backpropagation artificial neural networks (FFBANNs) are re-evaluated. Using contingent valuation survey data, the paper shows that FFBANNs provide an alternative to the binary logit and probit models with linear index functions. Direct comparisons between the models showed that the FFBANNs performed marginally better than the logit and probit models for a number of within-sample and out-of-sample performance measures, but in the majority of cases these differences were not statistically significant. In addition, guidelines for modeling contingent valuation survey data and techniques for estimating median WTP measures using FFBANNs are examined.

The third paper estimates a set of unconditional price and expenditure elasticities for 49 different processed food categories using scanner data and the flexible and symmetric translog (FAST) multistage demand system. Due to the use of panel data and the presence of heterogeneity across time, temporal fixed effects were incorporated into the model. Overall, estimated price elasticities are larger, in absolute terms, than previous estimates. The use of disaggregated product groupings, scanner data, and the estimation of unconditional elasticities likely accounts for these differences.

Dedication

I dedicate this work to my wife and the Lord, for it was their continual and unbounded love, patience and strength that helped me to complete this work.

Acknowledgements

My sincere gratitude goes out to my advisor Daniel B. Taylor, who has been an advisor, friend, and mentor during my graduate career. Dr. Taylor is a generous and compassionate person, who has a real passion for working with graduate students. In addition, I would like to thank Anya M. McGuirk, Aris Spanos and Everett Peterson. All these individuals have provided guidance and needed support in the writing of this dissertation, as well as in life. I consider them all friends and hope to continue to work with them in the future. Additional thanks are due to Darrell J. Bosch and Christopher Zobel for their helpful comments and suggestions.

The paper in Chapter 3 is a work that began as a term paper assignment in one of my graduate classes. I owe thanks to Ebere Akobundu and Everett Peterson for their help in the writing of this paper. They provided a significant amount of support in the construction and estimation of the model, as well as, the formulation of the final paper.

Table of Contents

List of Figures

Chapter 1:

Chapter 2

Chapter 3

List of Tables

Chapter 1

Chapter 2

Chapter 3

Chapter 1

Statistical Models with Binary Dependent Variables:

A Probabilistic Reduction Approach

Abstract

The a priori imposition of a theoretical structure upon a statistical model can leave the model misspecified, resulting in biased and inconsistent estimates as well as erroneous inferences. The classical approach for specifying statistical models with binary dependent variables in econometrics using latent variables or threshold models does not avoid this criticism. Furthermore, methods for trying to alleviate such problems, such as univariate generalized linear models, have not provided an adequate alternative for specifying statistically adequate models. Thus, this paper re-examines the underlying statistical foundations of statistical models with binary dependent variables using the probabilistic reduction approach, which allows the modeler to efficiently capture the probabilistic information in the observed data being modeled. This re-examination leads to the development of the Bernoulli Regression Model and in turn the Multinomial Regression Model. The construction of these statistical models, without the imposition of a priori theoretical assumptions, allows modelers to reliably test the theories these models are hypothesized to summarize.

1. Introduction

As argued by Spanos (1995), the estimation of theoretically specified statistical models without due consideration to the statistical (systematic) information contained in the observed data used to estimate the model is likely to leave the postulated model misspecified, resulting in biased and inconsistent parameter estimates. He goes on to suggest that the misspecification often results from the fact that the modeler does not attempt to identify the underlying probabilistic structure of the observed data. He concludes by emphasizing the importance of obtaining a statistically adequate model that captures the systematic (statistical) information present in the observed data before using the model to conduct any statistical inference (to evaluate theory):

> *"Armed with a statistically adequate model we can then proceed to consider the question whether the theory in question accounts for the systematic information in the data. The question of 'biasing' the statistical inference results by looking at the data does not arise. When postulating a statistical model the modeler is not looking for a theory in the data, she is looking for 'probabilistic patterns'* (p. 209)*."*

Thus, the statistical model should be viewed separately from the theoretical model in order to obtain reliable inferences concerning the theoretical model.

In the field of econometrics, statistical models with binary dependent variables are commonly specified using utility theory in a probabilistic framework (see Train, 2003). The imposition of such a priori theoretical structures upon statistical models with binary dependent variables without considering the underlying probabilistic structure of the observed data calls into question many of the models found in the applied literature, given these models may be statistically misspecified (Arnold, Castillo and Sarabia, 1999). With the seminal paper by Nelder and Wedderburn (1972), concerning the specification and estimation of generalized linear models, it seemed that a statistical foundation for these types of models was developed. The fact that the link functions for generalized linear models with binary dependent variables can take a wide range of functional forms (that satisfy the properties of a cumulative density function), obscures the fact that the functional form (conditional mean) of these models is dependent upon the joint distribution from which the observed data are generated. Thus, this approach does not provide an adequate method for modeling the probabilistic information from the observed data of a dichotomous choice process.

The purpose of this paper is to re-examine the underlying foundations of statistical models with binary dependent variables using the probabilistic reduction approach developed by Spanos (1986,1995,1999). This examination includes: (i) a formal presentation of the Bernoulli Regression Model (BRM), (ii) specification and estimation of the BRM, and (iii) how to generate a random vector stochastic process for simulations using the BRM. Finally as a simple extension of the BRM, a formal presentation of the Multinomial Regression Model is provided.

The paper is organized as follows. Section two provides a brief background of the traditional approaches for specifying the binary logit and probit regression models to motivate the BRM. Section three then presents a brief description of the probabilistic reduction approach, while section four provides a formal presentation of the Bernoulli Regression Model. Section five presents the Multinomial Regression Model and section six provides some concluding remarks.

2. Traditional Approaches

The evolution of statistical models with binary dependent variables has led to two interrelated approaches for specifying these models. Powers and Xie (2000) refer to these two approaches as the *latent variable* or theoretical approach and the *transformational* or statistical approach. The latent variable approach assumes the existence of an underlying continuous latent or unobservable stochastic process giving rise to the dichotomous choice process being observed. The transformational approach views the observed dichotomous choice process as inherently categorical and uses transformations of the observed data to derive an operational statistical model (Powers and Xie, 2000). This section of the paper provides a brief summary and critique of each approach to motivate the discussion in the remainder of the paper.

2.1 The Latent Variable (Theoretical) Approach

Following Maddala (1983) let Y_i^* denote a continuous unobserved random variable. Furthermore, assume that Y_i^* can be described by the following regression function:

$$Y_i^* = b'\mathbf{x}_i + u_i, \tag{1}$$

where $\mathbf{x}_i$ is a vector of explanatory variables, u_i is independent and identically distributed and $E(u_i) = 0$. In economics, it is sometimes assumed that equation (1) represents the difference in utility between two alternative outcomes for a consumer having to make a particular decision (Train, 2003). Since Y_i^* is unobserved, what is observed is a binary variable (or the decision made by the consumer), Y_i, where:

$$Y_i = \begin{cases} 1 \text{ if } Y_i^* > 0 \\ 0 \text{ otherwise} \end{cases}. \tag{2}$$

Thus, using a probabilistic framework along with equations (1) and (2):

$$\mathbf{P}(Y_i^* > 0) = \mathbf{P}(Y_i = 1) = \mathbf{P}(b'\mathbf{x}_i + u_i > 0) = \mathbf{P}(u_i > -b'\mathbf{x}_i) = 1 - F(-b'\mathbf{x}_i), \tag{3}$$

where $F(.)$ is the cumulative density function (cdf) of u_i, which is assumed to be symmetric. Thus, $1 - F(-b'\mathbf{x}_i) = F(b'\mathbf{x}_i)$ (Gourieroux, 2000). If u_i is distributed IID extreme value, then $F(.)$ is the logistic cdf, and if u_i is IID normal, then $F(.)$ is the normal cdf, giving rise to the binary logit and probit models, respectively (Train, 2003). The estimable model is given by equation (3). The parameters can be estimated using the method of maximum likelihood (see Maddala, 1983).

A significant weakness of this approach was stated by Cosslett (1983):

> “ *There is clearly an unsatisfactory feature of this formulation* [the latent variable approach]. *In order to get a specific functional form for the choice of probabilities, one has to make an assumption about the distribution of the stochastic term, about which we generally have no a priori knowledge.* (p. 766).*”*[1]

If the assumption concerning the stochastic or error term is wrong, the estimable model obtained is misspecified and the parameter estimates inconsistent (Coslett, 1983). This is of particular concern, since the error term in equation (1) is unobservable and the distribution of the error term unverifiable.[2] A second related concern is that the functional form of the regression specified in equation (1) determines the functional form for the argument of $F(.)$, i.e. $b'\mathbf{x}_i$. Fahrmeir and Tutz (1994) state that the binary logit and probit models are estimating $E(Y_i \mid \mathbf{X}_i = \mathbf{x}_i)$, which implies that the functional form of the binary model being estimated is determined by $f(Y_i \mid \mathbf{X}_i; y_1)$, the conditional distribution of Y_i given $\mathbf{X}_i = \mathbf{x}_i$, which is not usually considered under this approach. The a priori imposition of the theoretical structure in equation (1) without considering the underlying probabilistic structure of the observed data, will likely leave the postulated

[1] The prose in square brackets was added by the author for clarification.

[2] Maddala (1983) suggests some other alternatives for $F(.)$, such as the Cauchy or Burr cdfs, but the criticism still holds, given that these additional choices are based on an assumed theoretical assumption.

model in equation (3) misspecified and result in inconsistent estimates and erroneous inferences.

2.2. The Transformational (Statistical) Approach

The transformational approach can be viewed in a number of different ways, but the classical approach is via the use of statistical theory associated with univariate generalized linear models. Following Fahrmeir and Tutz (1994), let $\{Y_i \mid \mathbf{X}_i = \mathbf{x}_i, i = 1,..., N\}$ be an independent stochastic process, where $\mathbf{X}_i$ is a vector of (categorical and/or continuous) explanatory variables, such that $Y_i \mid \mathbf{X}_i = \mathbf{x}_i \sim bin(p_i, 1)$ (i.e. binomial and a member of the simple exponential family of distributions). Furthermore, assume that there exists a linear predictor:

$$h_i = b't(\mathbf{x}_i), \tag{4}$$

where $t(\mathbf{x}_i)$ is a $(S \times 1)$ vector of transformations or functions of the explanatory variables and b is a $(S \times 1)$ vector of parameters. The linear predictor (4) is related to p_i via the following relationship:

$$p_i = E(Y_i \mid \mathbf{X}_i = \mathbf{x}_i) = h(h_i) = h(b't(\mathbf{x}_i)), \tag{5}$$

where $h(.)$ is assumed to be to be a known one-to-one, sufficiently smooth response function. The relationship given by equation (5) implies that:

$$h_i = g(p_i), \tag{6}$$

where $g(.)$ is the inverse of $h(.)$ and is known as the *link function*. The notion of calling this approach the transformational approach stems from the fact that the modeler tries to specify a transformation or functional form for the *link function* to obtain an operational statistical model. If one lets $g(.)$ be the logistic transformation, so that:

$$h_i = \ln\left(\frac{p_i}{1-p_i}\right), \tag{7}$$

then one obtains the traditional binary logistic regression model, whereas if one lets $g(.)$ be the probit (or normit) transformation, Φ^{-1} (the inverse of the normal cdf), then one obtains the traditional binary probit regression model. From equations (5) and (6) it becomes evident that any cumulative density function with an inverse would act as an appropriate link function.[3]

Nelder and Wedderburn (1972) developed this approach in order to construct a unifying approach for specifying regression-like models. Even though in the instances when this approach may lead to a proper statistical model (i.e. one derivable from a proper joint distribution), this approach ignores the fact that the functional form of the link function is dependent upon the joint distribution of Y_i and $\mathbf{X}_i$ and in turn the conditional distribution of $\mathbf{X}_i$ given $Y_i = j$ (Arnold and Press, 1999). Kay and Litte (1987) show that the linear predictor is only appropriate when the conditional distribution of $\mathbf{X}_i$ given $Y_i = j$ satisfies a number of stringent conditions, such as being multivariate normal with homogenous covariance matrix or when the explanatory variables are conditionally independent on Y_i with conditional distributions as specified later in the paper (see Table 2). The second concern with this approach is the fact that the story being told is inconsistent. First of all, the modeler essentially ignores the existence of a proper joint distribution from which the model can be derived, which is a necessary condition

[3] When the link function is the identity function, one obtains the familiar linear probability model (Fahrmeir and Tutz, 1994).

for the existence of any statistical model (Spanos, 1999). Second, the fact that p_i is assumed to be heterogeneous based on i results in an incidental parameters problem.[4]

To alleviate the weaknesses of these two approaches, this paper uses the probabilistic reduction approach proposed by Spanos (1999) to formally specify statistical models with binary dependent variables.

3. The Probabilistic Reduction Approach

The probabilistic reduction approach, as presented by Spanos (1986,1995,1999), is based on re-interpreting the De Finetti representation theorem as a formal way of reducing the joint distribution of all observable random variables involved into simplified products of distributions by imposing certain probabilistic assumptions. This decomposition provides a formal and intuitive mechanism for constructing statistical models, with the added benefit of identifying the underlying probabilistic assumptions of the statistical model being examined. Using this approach, the objective here is to re-examine the statistical foundations of statistical models with discrete dependent variables.

Spanos (1999) defines a statistical model as a set of probabilistic assumptions that adequately capture the systematic information in the observed data in a parsimonious and efficient way. The primary goal of the probabilistic reduction approach is to obtain statistically adequate models, where the "adequacy of a statistical model is judged by the appropriateness of the [probabilistic] assumptions (making up the model) in capturing the systematic information in the observed data (Spanos, 1999b; p.544)." Thus, the construction of a statistical model begins with the observed data. The observed data,

[4] In section 4, its shown that p_i is implicitly being estimated by the intercept term of the logistic regression model through the term k. If k varies with i, then there exists more parameters than numbers of observations, resulting in an inestimable model.

$(z_1,...,z_N)'$, is viewed as one particular realization of the vector stochastic process, $\{Z_i, i=1,...,N\}$, where $Z_i = (Y_i, \mathbf{X}_i')'$ is a $(K+1\times 1)$ random vector and $\mathbf{X}_i$ is a $(k\times 1)$ random vector.. All of the systematic (and probabilistic) information contained in $\{Z_i, i=1,...,N\}$ is captured by the Haavelmo Distribution, which can be represented by the joint density function: $f(Z_1,....,Z_N; f)$, for all $(z_1,...,z_N)' \in \mathbf{R}_Z^{(K+1)N}$, which represents our universe of discourse (Spanos, 1986 and 1999).

Based on a weaker version of De Finetti's representation theorem, by specifying a set of reduction assumptions from three broad categories:

(D) Distributional, **(M)** Memory/Dependence, and **(H)** Heterogeneity, concerning the vector stochastic process $\{Z_i, i=1,...,N\}$, the modeler can reduce the Haavelmo distribution or joint density function into an operational form, giving rise to an operational statistical model and probabilistic model assumptions. By specifying particular reduction assumptions, the modeler is essentially partitioning the space of all possible statistical models into a family of operational models (indexed by the parameter space). For example, if the modeler assumes that $\{Z_i, i=1,...,N\}$ is independent (I) and identically distributed (ID), then:

$$f(Z_1,...,Z_N; f) \overset{I}{=} \prod_{i=1}^{N} f_i(Z_i; j_i) \overset{ID}{=} \prod_{i=1}^{N} f(Z_i; j) = \prod_{i=1}^{N} f(Y_i \mid \mathbf{X}_i; y_1) f(\mathbf{X}_i; y_2), \quad (8)$$

where the last inequality results from the fact that $f(Z_i; j) = f(Y_i, \mathbf{X}_i; j)$. It is the final product of the reduction in the last equality that provides the modeler with a method for defining a proper statistical model.

The conditional distribution $f(Y_i \mid \mathbf{X}_i; y_1)$ (derived from the above reduction) allows the modeler to define a statistical generating mechanism (SGM), which is viewed as an idealized representation of the true underlying data generating process (which is unknown). Furthermore, the SGM helps to bridge the gap between the theoretical and statistical model (see Spanos, 1999). The statistical generating mechanism amounts to a set of conditional moment functions from $f(Y_i \mid \mathbf{X}_i; y_1)$. The SGMs of many statistical models are specified using the regression function:

$$Y_i = E(Y_i \mid \mathbf{X}_i = \mathbf{x}_i) + u_i, \qquad (9)$$

where $E(Y_i \mid \mathbf{X}_i = \mathbf{x}_i)$ represents the systematic component and u_i the nonsystematic component (the error term). The orthogonal decomposition in equation (9) arises when $\mathrm{cov}(Z_i) < \infty$ (see Spanos, 1999). It should be noted that the functional form of $E(Y_i \mid \mathbf{X}_i = \mathbf{x}_i)$ is dependent upon all of the reduction assumptions. For example, if the dependence assumption was relaxed to be Markov dependence of order *q*, then the conditioning set (as well as the reduction in equation (8)) would change from $\{\mathbf{X}_i = \mathbf{x}_i\}$ to $\{\mathbf{X}_i = \mathbf{x}_i, ..., \mathbf{X}_{i-q} = \mathbf{x}_{i-q}\}$ (see Spanos, 1986 and 1999 for other examples). The SGM can contain higher order conditional moment functions when they capture systematic information in the data. These can be specified using u_i, in the following manner:

$$u_i^s = E(u_i^s \mid \mathbf{X}_i = \mathbf{x}_i) + v_{i,s}, \qquad (10)$$

where *s* denotes the *s*th order conditional moment function. When $s = 2,3,4$, equation (10) represents the skedastic (conditional variance), clitic (conditional skewness) and kurtic (conditional kurtosis) functions, respectively.

4. Bernoulli Regression Models

This section of the paper examines the specification and estimation of the Bernoulli Regression Model (BRM). The first sub-section derives the conditional model using the probabilistic reduction approach developed in section 3. The second sub-section then examines how one goes about specifying a BRM. Sub-section three then examines estimation of the parameters of the BRM, while sub-section four examines how to generate sequences of random vectors for simulations using the BRM and asymptotic properties of the maximum likelihood estimators of the parameters of the BRM.

4.1 Theoretical Foundations

Let $\{Y_i, i = 1,..., N\}$ be a stochastic process defined on the probability space $(S, \Im, P(.))$, where $Y_i \sim bin(0,1)$ (Bernoulli), $E(Y_i) = p$ and $Var(Y_i) = p(1-p)$ for $i = 1,..., N$. Furthermore, let $\{\mathbf{X}_i = (X_{1,i},..., X_{K,i}), i = 1,..., N\}$ be a vector stochastic process defined on the same probability space with joint density function $f(\mathbf{X}; y_2)$, where y_2 is an appropriate set of parameters. Furthermore, assume that $E(X_{k,i}^2) < \infty$ for $k = 1,..., K$ and $i = 1,..., N$, making $\{Y_i, i = 1,..., N\}$ and $\{\mathbf{X}_i, i = 1,..., N\}$elements of $L_2(\mathbf{R}^N)$, the space of all square integrable stochastic processes over $\mathbf{R}^N$.

The joint density function of the joint vector stochastic process $\{(Y_i, \mathbf{X}_i), i = 1,..., N\}$ takes the form:

$$f(Y_1,...,Y_N, \mathbf{X}_1,..., \mathbf{X}_N; f), \tag{11}$$

where f is an appropriate set of parameters. Assuming that the joint vector stochastic process $\{(Y_i, \mathbf{X}_i), i = 1,..., N\}$ is independent (I) and identically distributed (ID), the joint distribution given by equation (11) can be reduced (decomposed) in the following manner:

$$f(Y_1,...,Y_N, \mathbf{X}_1,...,\mathbf{X}_N; f) \overset{I}{=} \prod_{i=1}^{N} f_i\left(Y_i, \mathbf{X}_i; j_{\,i}\right) \overset{ID}{=} \prod_{i=1}^{N} f\left(Y_i, \mathbf{X}_i; j\right), \qquad (12)$$

where $j_{\,i}$ and j are appropriate sets of parameters. The last component of the reduction in equation (4-2) can be further reduced so that:

$$f(Y_1,...,Y_N, \mathbf{X}_1,...,\mathbf{X}_N; f) \overset{IID}{=} \prod_{i=1}^{N} f\left(Y_i, \mathbf{X}_i; j\right) = \prod_{i=1}^{N} f\left(Y_i \mid \mathbf{X}_i; y_{\,1}\right) \cdot f\left(\mathbf{X}_i; y_{\,2}\right), \qquad (13)$$

where $y_{\,1}$ and $y_{\,2}$ are appropriate sets of parameters.

It is the reduction in (13) that provides us with the means to define an operational statistical model, but the decomposition of $f\left(Y_i, \mathbf{X}_i; j\right)$ into the product $f\left(Y_i \mid \mathbf{X}_i; y_{\,1}\right) f\left(\mathbf{X}_i; y_{\,2}\right)$ for statistical models with binary dependent variables has not yet been explicitly derived in the statistical literature (searched by the author). For the reduction in equation (13) to give rise to a proper statistical model, it is necessary that the joint density function $f\left(Y_i, \mathbf{X}_i; j\right)$ exist. The existence of $f\left(Y_i, \mathbf{X}_i; j\right)$ is dependent upon the compatibility of the conditional density functions, $f\left(Y_i \mid \mathbf{X}_i; y_{\,1}\right)$ and $f\left(\mathbf{X}_i \mid Y_i; h_1\right)$ (where h_1 is an appropriate set of parameters) (Arnold and Castillo, 1999), i.e.

$$f\left(Y_i \mid \mathbf{X}_i; y_{\,1}\right) \cdot f\left(\mathbf{X}_i; y_{\,2}\right) = f\left(\mathbf{X}_i \mid Y_i; h_1\right) \cdot f\left(Y_i; p\right) = f\left(Y_i, \mathbf{X}_i; j\right), \qquad (14)$$

where $f\left(Y_i; p\right) = p^{Y_i}(1-p)^{1-Y_i}$.

Using condition (14), consider the following relationship:

$$\frac{f(\mathbf{X}_i \mid Y_i = 1; h_1)}{f(\mathbf{X}_i \mid Y_i = 0; h_1)} \cdot \frac{f(Y_i = 1; p)}{f(Y_i = 0; p)} = \frac{f(Y_i = 1 \mid \mathbf{X}_i; y_1)}{f(Y_i = 0 \mid \mathbf{X}_i; y_1)} \cdot \frac{f(\mathbf{X}_i; y_2)}{f(\mathbf{X}_i; y_2)}.^{5} \tag{15}$$

Furthermore, assume that $f(Y_i \mid \mathbf{X}_i; y_1)$ is a conditional Bernoulli density function with the following functional form:

$$f(Y_i \mid \mathbf{X}_i; y_1) = g(\mathbf{X}_i; y_1)^{Y_i}[1 - g(\mathbf{X}_i; y_1)]^{1-Y_i}, \tag{16}$$

where $g(\mathbf{X}_i; y_1): \mathbf{R}^K \times \Theta_1 \rightarrow [0,1]$ and $y_1 \in \Theta_1$, the parameter space associated with y_1.[6] The density function specified by equation (16) satisfies the usual properties of a density function, i.e. following the properties of the Bernoulli density function in Spanos (1999):

(i) $f(Y_i| \mathbf{X}_i; y_1) \geq 0$ for $Y_i = 0,1$ and $\mathbf{X}_i = \mathbf{x}_i \in \mathbf{R}^K$,

(ii) $\sum_{Y_i=0,1} f(Y_i \mid \mathbf{X}_i; y_1) = g(\mathbf{X}_i; y_1) + (1 - g(\mathbf{X}_i; y_1)) = 1$, and

(iii) $$F(b \mid \mathbf{X}_i; y_1) - F(a \mid \mathbf{X}_i; y_1) = \begin{cases} 0 & \text{if } a < 0 \text{ and } b < 0 \\ 1 - g(\mathbf{X}_i; y_1) & \text{if } a < 0 \text{ and } 0 \leq b < 1 \\ g(\mathbf{X}_i; y_1) & \text{if } 0 < a < 1 \text{ and } b \geq 1, \\ 1 & \text{if } a \leq 0 \text{ and } b \geq 1 \\ 0 & \text{if } a > 1 \text{ and } b > 1 \end{cases} \text{ for } (a,b) \in \mathbf{R},$$

where (i) follows from the nonnegativity of $g(\mathbf{X}_i; y_1)$ and $F(. \mid \mathbf{X}_i; y_1)$ represents the cumulative conditional Bernoulli density function, which takes the following functional form:

$$F(z \mid \mathbf{X}_i; y_2) = \begin{cases} 0 & \text{for } z < 0 \\ 1 - g(\mathbf{X}_i; y_1) & \text{for } 0 \leq z < 1 \\ 1 & \text{for z} \geq 1 \end{cases}.$$

[5] The variable X_i is expected to take the same value in the numerator and denominator in order to solve for $g(X_i; y_1)$ in later derivations.

[6] The conditional Bernoulli distribution is based on the traditional approach for specifying binary discrete models, i.e. by letting $p_i = g(\mathbf{X}_i; y_1)$ (see Chen, 1997 and Spanos, 2000).

Substituting equation (16) into (15) and letting $p_j = p^j(1-p)^{1-j}$ for $j = 0,1$ gives:

$$\frac{f(\mathbf{X}_i \mid Y_i = 1; h_1)}{f(\mathbf{X}_i \mid Y_i = 0; h_1)} \cdot \frac{p_1}{p_0} = \frac{g(\mathbf{X}_i; y_1)}{1 - g(\mathbf{X}_i; y_1)} \cdot \frac{f(\mathbf{X}_i; y_2)}{f(\mathbf{X}_i; y_2)}, \tag{17}$$

which implies that:

$$g(\mathbf{X}_i; y_1) = \frac{p_1 \cdot f(\mathbf{X}_i \mid Y_i = 1; h_1)}{p_0 \cdot f(\mathbf{X}_i \mid Y_i = 0; h_1) + p_1 \cdot f(\mathbf{X}_i \mid Y_i = 1; h_1)}. \tag{18}$$

Given the general properties of density functions and that $p_j \in (0,1)$ for $j = 0,1$, the range of $g(\mathbf{X}_i; y_1)$ is $[0,1]$, justifying the assumption that $g(\mathbf{X}_i; y_1): \mathbf{R}^K \times \Theta_1 \to [0,1]$.

A more intuitive and practical choice for $g(\mathbf{X}_i; y_1)$ can be found by using results from Kay and Little (1987). Using the transformation $x = \exp\{\ln(x)\}$ and after rearranging some terms, $g(\mathbf{X}_i; y_1)$ becomes:

$$g(\mathbf{X}_i; y_1) = \frac{\exp\{h(\mathbf{X}_i; h_1)\}}{1 + \exp\{h(\mathbf{X}_i; h_1)\}} = [1 + \exp\{-h(\mathbf{X}_i; h_1)\}]^{-1}, \tag{19}$$

where $h(\mathbf{X}_i; h_1) = \ln \dfrac{f(\mathbf{X}_i \mid Y_i = 1; h_1)}{f(\mathbf{X}_i \mid Y_i = 0; h_1)} + k$ and $k = \ln(p_1) - \ln(p_0)$. Written as the composite function, $g(h(\mathbf{X}_i; h_1))$, $g(.)$ represents the logistic cumulative density function (the transformation function) and $h(.;.)$ represents the traditional index function. Equation (19) illustrates that functional relationship between y_1 and h_1 (i.e. $y_1 = G(h_1)$), as well. It is the relationships given by equations (18) and (19) that form the basis for defining a proper statistical model when the dependent variable is binary.

The reduction of the joint density function (11) gives rise to a family of statistical models for conditional dichotomous choice processes. Given that $Var(Y_i \mid \mathbf{X}_i = \mathbf{x}_i) < \infty$, the stochastic process $\{Y_i \mid \mathbf{X}_i = \mathbf{x}_i, i = 1,..., N\}$ can be decomposed orthogonally into a systematic and nonsystematic component giving rise to the following regression function (see Spanos, 1999):

$$Y_i = E(Y_i \mid \mathbf{X}_i = \mathbf{x}_i) + u_i = g(\mathbf{x}_i;y_1) + u_i = [1 + \exp\{-h(\mathbf{x}_i;h_1)\}]^{-1} + u_i, \qquad (20)$$

where the last inequality follows by substituting in equation (19) and $u_i \sim bin(0,1)$ with variance $g(\mathbf{x}_i;y_1)(1-g(\mathbf{x}_i,y_1))$.[7] Furthermore, equation (20) is the stochastic generating mechanism (SGM) for this family of statistical models, because the conditional variance (or skedastic function) is completely specified in terms of the conditional mean, $g(\mathbf{x}_i;y_1)$ (Spanos, 2000). The reduction of (11) along with the SGM given by equation (20) give rise to the family of Bernoulli Regression Models specified in Table 1. The first three model assumptions, i.e. distributional, functional form and heteroskedasticity, arise from the derivations provided above. The homogeneity and independence assumptions are a result of the IID reduction assumptions made about the joint vector stochastic

Table 1: Bernoulli Regression Model

SGM:	$Y_i = g(\mathbf{x}_i;y_1) + u_i,\ i = 1,...,N$, where $u_i \sim bin(0,1)$ with variance $g(\mathbf{x}_i;y_1)(1-g(\mathbf{x}_i,y_1))$.
Assumptions	
Distributional:	$Y_i \mid \mathbf{X}_i = \mathbf{x}_i \sim bin(g(\mathbf{x}_i,y_1),1)$, (conditional Bernoulli).
Functional Form:	$E(Y_i \mid \mathbf{X}_i = \mathbf{x}_i) = g(\mathbf{x}_i;y_1) = [1 + \exp\{-h(\mathbf{x}_i;h_1)\}]$, where $h(\mathbf{x}_i;h_1) = \ln\left[\frac{f(\mathbf{X}_i \mid Y_i = 1;h_1)}{f(\mathbf{X}_i \mid Y_i = 0;h_1)}\right] + k$ and $y_1 = G(h_1)$.
Heteroskedasticity:	$Var(Y_i \mid \mathbf{X}_i = \mathbf{x}_i) = g(\mathbf{x}_i;y_1)(1-g(\mathbf{x}_i;y_1))$.
Homogeneity:	$y_1 = G(h_1)$ is not a function of $i = 1,...,N$.
Independence:	$\{Y_i \mid \mathbf{X}_i = \mathbf{x}_i, i = 1,...,N\}$ is an independent stochastic process.

[7] Alternatively, u_i can be viewed as is as a random variable having a diatomic distribution, taking the value $-g(\mathbf{x}_i;y_1)$ with probability $1-g(\mathbf{x}_i;y_1)$ or the value $1-g(\mathbf{x}_i;y_1)$ with probability $g(\mathbf{x}_i;y_1)$, with mean equal to 0 and variance equal to $g(\mathbf{x}_i;y_1)(1-g(\mathbf{x}_i;y_1))$ (see Arnold, Castillo and Sarabia, 1999).

process $\{(Y_i, \mathbf{X}_i), i = 1,..., N\}$.

The regression function given by equation (20) is similar to the traditional binary logistic regression model, but above derivations show that it arises naturally from the joint density function given by (11), suggesting it as an obvious candidate for modeling discrete choice processes when the dependent variable is distributed Bernoulli(*p*). An important observation is that the functional forms for both $g(\mathbf{X}_i; y_1)$ and $h(\mathbf{X}_i; h_1)$ are both dependent upon the functional form of $f(\mathbf{X}_i \mid Y_i; h_1)$ and in turn the joint distribution of Y_i and $\mathbf{X}_i$. It would seem other binary statistical models, such as the probit model, may not be proper statistical models, in that the regression functions being estimated may not be derivable from a proper joint density function. In fact, given that binary discrete choice models are somewhat robust to the choice of the transformation function (Amemiya, 1981), statistical models such as the binary probit, may act as close approximations to the BRM, when the index function $h(\mathbf{x}_i; h_1)$ is specified as in Table 1.

4.2 Specification of the BRM

In this section, specification of the BRM is examined when the distribution of the explanatory variables ($\mathbf{X}_i$) conditional on Y_i takes different (distributional) functional forms. Two conditional BRMs are derived in the following two examples, to illustrate the case when an explicit functional form for the conditional distribution $f(\mathbf{X}_i \mid Y_i; h_1)$ can be identified.

Example 1: X_i given Y_i has a Normal Distribution.

Consider the stochastic process $\{X_i, i = 1,..., N\}$, where $X_i \sim N(m, s)$ for $i = 1,..., N$. Hence:

$$f\left(X_i \mid Y_i;h_1\right)=\frac{1}{s\sqrt{p}}\exp\left\{-\frac{1}{2s^2}\left(X_i-a_0-a_1Y_i\right)^2\right\}. \qquad (21)$$

Substituting equation (21) into equation (19) and simplifying gives:

$$h\left(x_i;h_1\right)=-\frac{\left(a_1^2+2a_0a_1\right)}{2s^2}+\frac{a_1}{s^2}x_i+k \text{ , and} \qquad (22)$$

$$g\left(x_i;y_1\right)=\left[1+\exp\left(-b_0-b_1x_i\right)\right]^{-1}, \qquad (23)$$

where $b_0=k-\frac{\left(a_1^2+2a_0a_1\right)}{2s^2}$ and $b_1=\frac{a_1}{s^2}$ (see also Kay and Little, 1987).

Example 2: A Dynamic Bernoulli Regression Model

Consider the case where $i=1,...,N$ denotes time and assume that $\left\{X_i, i=1,...,N\right\}$ is a first-order autoregressive stochastic process, i.e. first order Markov dependent. Furthermore assume that the joint stochastic process $\left\{\left(Y_i, X_i\right), i=1,...,N\right\}$ is stationary. Then the reduction of the joint density function, $f\left(Y_1,...,Y_N,X_1,...,X_N;f\right)$ takes the following form:

$$\begin{aligned}
f\left(Y_1,...,Y_N,X_1,...,X_N;f\right) &\overset{\substack{\textit{Markov}\\ \textit{Dependent}}}{=} f_1\left(Y_1,X_1;j_1\right)\prod_{i=2}^{N} f_i\left(Y_i,X_i \mid X_{i-1};j_i\right) \\
&\overset{\textit{Stationarity}}{=} f\left(Y_1,X_1;j_1\right)\prod_{i=2}^{N} f\left(Y_i,X_i \mid X_{i-1};j\right) \qquad (24) \\
&= f\left(Y_1,X_1;j_1\right)\prod_{i=2}^{N} f\left(Y_i \mid X_i,X_{i-1};y_1\right)f\left(X_i \mid X_{i-1};y_2\right).
\end{aligned}$$

Performing the same type of derivations that gave rise to the general BRM, the index function for this particular case is:

$$h\left(X_i,X_{i-1};h_1',h_1''\right)=\ln\left(\frac{f\left(X_i \mid X_{i-1},Y_i=1;h_1'\right)}{f\left(X_i \mid X_{i-1},Y_i=0;h_1'\right)}\right)+\ln\left(\frac{f\left(X_{i-1} \mid Y_i=1;h_1''\right)}{f\left(X_{i-1} \mid Y_i=0;h_1''\right)}\right)+k \text{ ,} \qquad (25)$$

because $f\left(X_i,X_{i-1} \mid Y_i;h_1\right)=f\left(X_i \mid X_{i-1},Y_i;h_1'\right)\cdot f\left(X_{i-1} \mid Y_i;h_1''\right)$.

The conditional density:

$$f\left(X_i \mid X_{i-1}, Y_i; h_1'\right) = \frac{1}{s_a\sqrt{2p}}\exp\left\{-\frac{1}{2s_a^2}\left(X_i - a_0 - a_1 X_{i-1} - a_2 Y_i\right)^2\right\}, \qquad (26)$$

and the conditional density:

$$f\left(X_{i-1} \mid Y_i; h_1''\right) = \frac{1}{s_g\sqrt{2p}}\exp\left\{-\frac{1}{2s_g^2}\left(X_{i-1} - g_0 - g_1 Y_i\right)^2\right\}. \qquad (27)$$

Substituting equations (26) and (27) into equation (25) and then into equation (19) gives:

$$g\left(x_i, x_{i-1}; y_1\right) = \left[1 + \exp\left(-b_0 - b_1 x_i - b_2 x_{i-1}\right)\right]^{-1}, \qquad (28)$$

where $b_0 = k - \frac{\left(2a_0a_2 + a_2^2\right)}{2s_a^2} - \frac{\left(2g_0g_1 - g_1^2\right)}{2s_g^2}$, $b_1 = \frac{a_2}{s_a^2}$ and $b_2 = \frac{g_1}{s_g^2} - \frac{a_1a_2}{s_a^2}$.

Substituting equation (28) into the BRM gives rise to a type of dynamic BRM, where the independence assumption is replaced by a dependence assumption where X_i follows an AR(1) process.

These two examples provide explicit functional forms for $f\left(\mathbf{X}_i \mid Y_i; h_1\right)$, but for many cases such functional forms are not readily derived.[8] A potential alternative is to assume that:

$$f\left(\mathbf{X}_i \mid Y_i; h_1\right) = f\left(\mathbf{X}_i; h_1\left(Y_i\right)\right). \qquad (29)$$

In this sense, one is treating the moments of the conditional distribution of $\mathbf{X}_i$ given Y_i as functions of Y_i or as heterogeneous based on the value of Y_i. That is $h_1\left(Y_i = j\right) = h_{1,j}$ for $j = 0,1$. Lauritzen and Wermuth (1989) use a similar approach to specify conditional Gaussian distributions, and Kay and Little (1987) use this approach to specify the logistic regressions models in their paper.

[8] For help with such derivations, the work by Arnold, Castillo and Sarabia (1999) may be of assistance.

Table 2 provides the functional forms for $g(x_i;y_1)$ needed to obtain a properly specified BRM with one explanatory variable for a number of different conditional distributions of the form $f(X_i;h_{1,j})$. All of the cases examined in Table 2 have index functions that are linear in the parameters, which allow one to use traditional computer packages to estimate the parameters of the resulting logistic regression model. Naturally, the question arises of what does one do if the conditional distribution, $f(X_i;h_{1,j})$, does not result in an index function linear in the parameters? Examples include situations when $f(X_i;h_{1,j})$ is distributed F, extreme value, logistic or Weibull. In such cases, one option is to explicitly specify $f(X_i;h_{1,j})$ and estimate the model using equation (18), which can be difficult numerically due to the existence of both $h_{1,0}$ and $h_{1,1}$ in $h(x_i;h_{1,j})$. Another approach would be to transform X_i so that one obtains one of the distributions in Table 2. To illustrate this latter approach, consider the following example.

Example 3: Let $f(X_i;h_{1,j})$ be a conditional Weibull distribution, i.e.

$$f(X_i;h_1) = \frac{g \cdot X_i^{g-1}}{a_j^g} \exp\left\{-\left(\frac{X_i}{a_j}\right)^g\right\}, \tag{30}$$

where $(a_j, g) \in \mathbf{R}_+^2$ and $X_i > 0$. That is $X_i \mid Y_i = j \sim W(a_j, g)$. According to Spanos (1999), if $X_i \sim W(a, g)$ then $X_i^g \sim Exp(a)$ (i.e. exponential). Thus, $X_i^g \mid Y_i = j \sim Exp(a_j)$, and using the results from Table 2:

Table 2: Specification of $g(x_i;h_1)$ with one explanatory variable and conditional distribution, $f(X_i;h_{1,j})$, for $j=0,1$.

Distribution of X_i **given** Y_i	$f(X_i;h_{1,j}) =$ [2]	$g(x_i;y_1) =$
Beta[1]	$\frac{X_i^{a_j-1}(1-X_i)^{g_j-1}}{\mathbf{B}[a_j,g_j]}$, where $(a_j,g_j)\in \mathbf{R}_+^2$ and $0\le X_i\le 1$.	$[1+\exp\{b_0+b_1\ln(x_i)+b_2\ln(1-x_i)\}]^{-1}$, where $b_0=\left[k+\ln\left(\frac{\mathbf{B}[a_0,g_0]}{\mathbf{B}[a_1,g_1]}\right)\right]$, $b_1=(a_1-a_0)$ and $b_2=(g_1-g_0)$.
Binomial[1]	$\binom{n}{X_i}q_j^{X_i}(1-q_j)^{n-X_i}$, where $0<q_j<1,\ X_i=0,1$ and $n=1,2,3,\ldots$	$[1+\exp\{-b_0-b_1x_i\}]^{-1}$, where $b_0=\left[k+n\ln\left(\frac{1-q_1}{1-q_0}\right)\right]$ and $b_1=\ln\left(\frac{q_1}{q_0}\right)-\ln\left(\frac{1-q_1}{1-q_0}\right)$.
Chi-square	$\frac{2^{-\frac{v_j}{2}}}{\Gamma\left[\frac{v_j}{2}\right]}X_i^{\frac{v_j-2}{2}}\exp\left\{-\frac{X_i}{2}\right\}$, where $v=1,2,3,\ldots$ and $x\in\mathbf{R}_+$.	$[1+\exp\{-b_0-b_1x_i\}]^{-1}$, where $b_0=\left[k+\left(\frac{v_0-v_1}{2}\right)\ln(2)+\ln\left(\frac{\Gamma\left[\frac{v_0}{2}\right]}{\Gamma\left[\frac{v_1}{2}\right]}\right)\right]$ and $b_1=\frac{v_1-v_0}{2}$.
Exponential	$\frac{1}{q_j}\exp\left\{-\frac{X_i}{q_j}\right\}$, where $q_j\in\mathbf{R}_+$ and $X_i\in\mathbf{R}_+$.	$[1+\exp\{-b_0-b_1x_i\}]^{-1}$, where $b_0=\left[\ln\left(\frac{q_0}{q_1}\right)+k\right]$ and $b_0=\left(\frac{1}{q_0}-\frac{1}{q_1}\right)$.
Gamma[1]	$\frac{1}{g_j\Gamma[a_j]}\left(\frac{X_i}{g_j}\right)^{a_j-1}\exp\left\{-\frac{X_i}{g_j}\right\}$, where $(a_j,g_j)\in\mathbf{R}_+^2$ and $X_i\in\mathbf{R}_+$.	$[1+\exp\{b_0+b_1x_i+b_2\ln(x_i)\}]^{-1}$, where $b_0=\left[k+\ln\left(\frac{b_0\Gamma[a_0]}{b_1\Gamma[a_1]}\right)+(a_0-1)\ln(b_0)-(a_1-1)\ln(b_1)\right]$, $b_1=(a_1-a_0)$ and $b_2=\left(\frac{1}{b_0}-\frac{1}{b_1}\right)$.
Geometric	$q_j(1-q_j)^{X_i-1}$, where $0\le q_j\le 1$ and $X_i=1,2,3,\ldots$	$[1+\exp\{-b_0-b_1x_i\}]^{-1}$, where $b_0=\left[k+\ln\left(\frac{q_1}{q_0}\right)-\ln\left(\frac{1-q_1}{1-q_0}\right)\right]$ and $b_1=\ln\left(\frac{1-q_1}{1-q_0}\right)$.

Table 2 continued.

Logarithmic	$a_j \left(\frac{q_j^{X_i}}{X_i} \right)$, where $a_j = -\left[\ln(1-q_j)\right]^{-1}$, $0 < q_j < 1$ and $\mathrm{X_i} = 1,2,3,...$	$\left[1+\exp\{-b_0 - b_1 x_i\}\right]^{-1}$, where $b_0 = \left[k + \ln\left(\frac{a_1}{a_0}\right)\right]$ and $b_1 = \ln\left(\frac{q_1}{q_0}\right)$.
Log-Normal	$\frac{1}{X_i} \cdot \frac{1}{s_j \sqrt{2p}} \exp\left\{-\frac{\left(\ln(X_i) - m_j\right)^2}{2s_j^2}\right\}$, where $m_j \in \mathbf{R}$, $s_j^2 \in \mathbf{R}_+$ and $X_i \in \mathbf{R}$.	$\left[1+\exp\{b_0 + b_1 \ln(x_i) + b_2 (\ln(x_i))^2\}\right]^{-1}$, where $b_0 = \left[k + \ln\left(\frac{s_0}{s_1}\right) + \left(\frac{m_0^2}{2s_0^2} - \frac{m_1^2}{2s_1^2}\right)\right]$, $b_1 = \left(\frac{m_1}{s_1^2} - \frac{m_0}{s_0^2}\right)$ and $b_2 = \left(\frac{1}{2s_0^2} - \frac{1}{2s_1^2}\right)$.
Normal[1]	$\frac{1}{s_j \sqrt{p}} \exp\left\{-\frac{1}{2s_j^2}(X_i - m_j)^2\right\}$, where $m_j \in \mathbf{R}$, $s_j^2 \in \mathbf{R}_+$ and $X_i \in \mathbf{R}$.	$\left[1+\exp\{b_0 + b_1 x_i + b_2 x_i^2\}\right]^{-1}$, where $b_0 = \left[k + \ln\left(\frac{s_0}{s_1}\right) + \left(\frac{m_0^2}{2s_0^2} - \frac{m_1^2}{2s_1^2}\right)\right]$, $b_1 = \left(\frac{m_1}{s_1^2} - \frac{m_0}{s_0^2}\right)$ and $b_2 = \left(\frac{1}{2s_0^2} - \frac{1}{2s_1^2}\right)$.
Pareto	$q_j x_0^{q_j} X_i^{-q_j - 1}$, where $q_j \in \mathbf{R}_+$, $x_0 > 0$ and $X_i \geq x_0$.	$\left[1+\exp\{-b_0 - b_1 \ln(x_i)\}\right]^{-1}$, where $b_0 = \left[k + \ln\left(\frac{q_1}{q_0}\right) + (q_1 - q_0)\ln(x_0)\right]$ and $b_1 = (q_0 - q_{1.})$
Poisson[1]	$\frac{e^{-q_j} q_j^{X_i}}{X_i!}$, where $q_j > 0$ and $X_i = 1,2,3,...$	$\left[1+\exp\{-b_0 - b_1 x_i\}\right]^{-1}$, where $b_0 = [k + q_0 - q_{\text{`}}]$ and $b_1 = \ln\left(\frac{q_1}{q_0}\right)$.

[1] *Source*: Kay and Little (1987).

[2] *Source*: Spanos (1999). **B**[ ] represents the beta function and Γ[ ] represents the gamma function.

$$g(x_i;y_1)=[1+\exp\{-b_0-b_1x_i^g\}]^{-1}, \tag{31}$$

where $b_0=\left[k+\ln\left(\frac{a_0}{a_1}\right)\right]$ and $b_1=\left(\frac{1}{a_0}-\frac{1}{a_1}\right)$.

In example 3, $g(X_i;y_1)$ is nonlinear in the parameters and the traditional programs for estimating logistic regression models may not be usable. The likelihood function and associated derivatives for these cases are provided in section 4.3. These types of transformations can usually be found in traditional econometric and statistical textbooks and provide a more simplified approach for specifying $g(x_i;y_1)$ using the conditional distributions presented in Table 2.

What about the cases where there are $K>1$ explanatory variables, i.e. $\mathbf{X}_i=(X_{1,i},...,X_{K,i})$? Kay and Little (1987) show that if $f(\mathbf{X}_i;h_{1,j})$ is multivariate normal with homogenous covariance matrix, then:

$$g(\mathbf{x}_i;y_1)=\left[1+\exp\left(-b_0-\sum_{k=1}^{K}b_kx_{k,i}\right)\right]^{-1}. \tag{32}$$

On the other hand, if the covariance matrix exhibits heterogeneity (based on Y_i), then:

$$g(\mathbf{x}_i;y_1)=\left[1+\exp\left(-b_0-\sum_{j=1}^{K}\sum_{k\geq j}^{K}b_{jk}x_{j,i}x_{k,i}\right)\right]^{-1}. \qquad (33).$$

Based on the results obtained by Kay and Little (1987), there are a limited number of multivariate distributions that exist in the literature which would give rise to a readily estimable and tractable BRM. The following three examples present three additional conditional bivariate distributions of the form $f(\mathbf{X}_i;h_{1,j})$ from which a proper BRM can be derived.

Example 4: Let $f(X_{1,i}, X_{2,i}; h_{1,j})$ be a conditional bivariate binomial distribution, i.e.

$$f(X_{1,i}, X_{2,i}; h_{1,j}) = \frac{n!}{X_{2,i}!\,X_{2,i}!(n - X_{1,i} - X_{2,i})!} q_{1,j}^{X_{1,i}} q_{2,j}^{X_{2,i}} (1 - q_{1,j} - q_{2,j})^{n - X_{1,i} - X_{2,i}}, \quad (34)$$

where $X_{1,i} = 0,1,2,3,....$, $X_{2,i} = 0,1,2,3,...$, $n = 1,2,3,....$, $(q_{1,j}, q_{2,j}) \in [0,1]^2$ and $X_{1,i} + X_{2,i} \leq n$ (Spanos, 1999). Then:

$$g(x_{1,i}, x_{2,i}; y_1) = [1 + \exp\{- b_0 - b_1 x_{1,j} - b_2 x_{2,j}\}]^{-1}, \quad (35)$$

where $b_0 = \left[k + n \ln\left(\frac{1 - q_{1,1} - q_{2,1}}{1 - q_{1,0} - q_{2,0}}\right)\right]$, $b_1 = \left[\ln\left(\frac{q_{1,1}}{q_{1,0}}\right) - \ln\left(\frac{1 - q_{1,1} - q_{2,1}}{1 - q_{1,0} - q_{2,0}}\right)\right]$ and

$b_2 = \left[\ln\left(\frac{q_{2,1}}{q_{2,0}}\right) - \ln\left(\frac{1 - q_{1,1} - q_{2,1}}{1 - q_{1,0} - q_{2,0}}\right)\right]$.

Example 5: Let $f(X_{1,i}, X_{2,i}; h_{1,j})$ be a conditional bivariate beta distribution, i.e.

$$f(X_{1,i}, X_{2,i}; h_{1,j}) = \left(\frac{\Gamma(a_j + d_j + g_j)}{\Gamma(a_j)\Gamma(d_j)\Gamma(g_j)}\right)\left[X_{1,i}^{a_j - 1} \cdot X_{2,i}^{d_j - 1} \cdot (1 - X_{1,i} - X_{2,i})^{g_j - 1}\right], \quad (36)$$

where $X_{1,i} \geq 0$, $X_{2,i} \geq 0$ and $X_{1,i} + X_{2,i} \leq 1$ for $i = 1,...,N$; $(a_j, d_j, g_j) > 0$ for $j = 0,1$; and $\Gamma(.)$ is the gamma function (Spanos, 1999). Then:

$$g(x_{1,i}, x_{2,i}; y_1) = [1 + \exp\{- b_0 - b_1 \ln(x_{1,i}) - b_2 \ln(x_{2,i}) - b_3 \ln(1 - x_{1,i} - x_{2,i})\}]^{-1}, \quad (37)$$

where $b_0 = \ln(l) + k$, $b_1 = a_1 - a_0$, $b_2 = d_1 - d_0$, $b_3 = g_1 - g_0$ and

$l = \frac{\Gamma(a_1 + d_1 + g_1)\Gamma(a_0)\Gamma(d_0)\Gamma(g_0)}{\Gamma(a_0 + d_0 + g_0)\Gamma(a_1)\Gamma(d_1)\Gamma(g_1)}$.

Example 6: Let $f(X_{1,i}, X_{2,i}; h_{1,j})$ be a conditional bivariate gamma distribution, i.e.:

$$f(X_{1,i}, X_{2,i}; h_{1,j}) = \frac{a_j q_{1,j} q_{2,j}}{\Gamma[q_{1,j}]\Gamma[q_{2,j}]} e^{-a_j X_{2,i}} X_{1,i}^{q_{1,j} - 1} (X_{2,i} - X_{1,i})^{q_{2,j} - 1}, \quad (38)$$

where $\Gamma[.]$ is the gamma function, $X_{2,i} > X_{1,i} \geq 0$ and $\left(a_j, q_{1,j}, q_{2,j}\right) \in \mathbf{R}_{+}^{3}$ (Spanos, 1999). Then:

$$g\left(x_{1,i}, x_{2,i}; y_1\right) = \left[1 + \exp\left\{-b_0 - b_1 x_{2,i} - b_2 \ln\left(x_{1,i}\right) - b_3 \ln\left(x_{2,i} - x_{1,i}\right)\right\}\right]^{-1}, \qquad (39)$$

where $b_0 = \left[k + \ln\left(\frac{a_1 q_{1,1} q_{2,1} \Gamma\left[q_{1,0}\right]\Gamma\left[q_{2,0}\right]}{a_0 q_{1,0} q_{2,0} \Gamma\left[q_{1,1}\right]\Gamma\left[q_{2,1}\right]}\right)\right]$, $b_1 = \left(a_0 - a_1\right)$, $b_2 = \left(q_{1,1} - q_{1,0}\right)$ and $b_3 = \left(q_{2,1} - q_{2,0}\right)$.

Avenues for deriving more complicated models using multivariate joint distributions might consider results by Arnold, Castillo and Sarabia (1999).

Another approach for specifying a BRM when $K > 1$ is to decompose $f\left(\mathbf{X}_i \mid Y_i; h_1\right)$ or $f\left(\mathbf{X}_i; h_{1,j}\right)$ into a product of simpler conditional density functions. An example that decomposed $f\left(\mathbf{X}_i \mid Y_i; h_1\right)$, a bivariate normal density function with $\mathbf{X}_i = \left(X_i, X_{i-1}\right)$, was provided in Example 2. Thus, the remainder of this sub-section will focus on using $f\left(\mathbf{X}_i; h_{1,j}\right)$.

Following Kay and Little (1987), consider the case where the explanatory variables are independent of each other conditional on Y_i, based on the re-parameterization in equation (29). In this case:

$$f\left(\mathbf{X}_i; h_{1,j}\right) = \prod_{k=1}^{K} f\left(X_{k,i}; h_{1,k,j}\right), \qquad (40)$$

which makes the index function in section 4.1:

$$h\left(\mathbf{x}_i; h_1\right) = \sum_{k=1}^{K} \ln\left(\frac{f\left(X_{k,i}; h_{1,k,1}\right)}{f\left(X_{k,i}; h_{1,k,0}\right)}\right) + k\ . \qquad (41)$$

The index functions of the models specified in Table 2 can be used here to specify equation (41). By specifying the (sub) index functions, $h(x_{k,i};h_{1,k}) = \ln\left(\frac{f(X_{k,i};h_{1,k,1})}{f(X_{k,i};h_{1,k,0})}\right)$, (without k) for each $X_{k,i}$ using the results in Table 2, one can obtain equation (41) and in turn $g(\mathbf{x}_i;y_1)$. To illustrate this approach, consider the following example:

Example 7: Let $f(X_{1,i},X_{2,i},X_{3,i};h_{1,j}) = \prod_{k=1}^{3} f(X_{k,i};h_{1,k,j})$, where $f(X_{1,i};h_{1,1,j})$ is conditional log-normal, $f(X_{2,i};h_{1,2,j})$ is conditional geometric, and $f(X_{3,i};h_{1,3,j})$ is conditional Weibull. Using the results in Table 2 and equation (41):

$$g(\mathbf{x}_i;y_1) = \left[1+\exp\left\{-b_0 - b_1\ln(x_{1,i}) - b_2(\ln(x_{1,i}))^2 - b_3x_{2,i} - b_4x_{3,i}^{g}\right\}\right]^{-1}, \qquad (42)$$

where $b_0 = \sum_{k=1}^{3} b_{0,k} = \left[\ln\left(\frac{s_{1,0}}{s_{1,1}}\right) + \left(\frac{m_{1,0}^2}{2s_{1,0}^2} - \frac{m_{1,1}^2}{2s_{1,1}^2}\right) + \ln\left(\frac{q_{2,1}}{q_{2,0}}\right) - \ln\left(\frac{1-q_{2,1}}{1-q_{2,0}}\right) + \ln\left(\frac{q_{3,0}}{q_{3,1}}\right)\right]$, $b_{0,k}$ is the intercept of the index function for the BRM associated with $f(X_{k,i};h_{1,k,j})$ minus k , $b_1 = \left(\frac{m_{1,1}}{s_{1,1}^2} - \frac{m_{1,0}}{s_{1,0}^2}\right)$, $b_2 = \left(\frac{1}{2s_{1,0}^2} - \frac{1}{2s_{1,1}^2}\right)$, $b_3 = \ln\left(\frac{1-q_{2,1}}{1-q_{2,0}}\right)$ and $b_4 = \left(\frac{1}{q_{3,0}} - \frac{1}{q_{3,1}}\right)$.

If some or none of the explanatory variables are independent conditional on Y_i, then another approach for decomposing $f(\mathbf{X}_i;h_{1,j})$ is sequential conditioning, i.e.

$$f(\mathbf{X}_i;h_{1,j}) = f(X_{1,i};h_{1,1,j})\prod_{k=2}^{K} f(X_{k,i} \mid X_{k-1,i},\ldots,X_{1,i};x_{k,j}), \qquad (43)$$

where $x_{k,j}$ is an appropriate set of parameters. Given the potential complexity of this approach, it can be combined with the case in equation (41) to reduce the dimensionality and increase the tractability of the problem (Spanos, 1999). To illustrate this alternative consider the following example.

Example 8: Decomposing $f\left(\mathbf{X}_i;h_{1,j}\right)$ for $\mathbf{X}_i=\left(X_{1,i},X_{2,i},X_{3,i},X_{4,i}\right)$.

Consider the following decomposition of $f\left(\mathbf{X}_i;h_{1,j}\right)$:

$$f\left(X_{1,i},X_{2,i},X_{3,i};X_{4,i};h_{1,j}\right)=f\left(X_{1,i},X_{2,i};h_{2,j}\right)f\left(X_{3,i},X_{4,i};h_{3,j}\right), \tag{44}$$

where $X_{1,i}$ and $X_{2,j}$ are conditionally independent on Y_i of $X_{3,i}$ and $X_{4,j}$. Now let $X_{1,j}$ given $Y_i=j$ be *bin*(1, r_j), $X_{2,i}$ given $X_{2,j}=l$ and $Y_i=j$ exponential,

$$f\left(X_{1,i};x_{j,l}\right)=\frac{1}{q_{j,l}}\exp\left\{-\frac{X_{1,i}}{q_{j,l}}\right\}, \text{ and} \tag{45}$$

and $X_{3,i}$ and $X_{4,i}$ given $Y_i=j$ bivariate beta (see equation (36)). Now, note that:

$$\begin{aligned}f\left(X_{1,j},X_{2,j};h_{2,j}\right)&=f\left(X_{1,j};x_{j,l}\right)\cdot f\left(X_{2,j};h_{2,2,j}\right)\\&=\left[\frac{r_j}{q_{j,1}}\exp\left\{-\frac{X_{1,i}}{q_{j,1}}\right\}\right]^{X_{2,i}}\left[\frac{\left(1-r_j\right)}{q_{j,0}}\exp\left\{-\frac{X_{1,i}}{q_{j,0}}\right\}\right]^{1-X_{2,i}}\end{aligned} \tag{46}$$

(see Kay and Little, 1987). Based on the above decomposition of $f\left(X_{1,j},X_{2,j};h_{2,j}\right)$:

$$\begin{aligned}g\left(\mathbf{x}_i;y_1\right)=[1+\exp\{&-b_0-b_1x_{1.i}-b_2x_{2,i}-b_3x_{1,i}x_{2,i}-b_4\ln\left(x_{3,i}\right)\\&-b_5\ln\left(x_{4,i}\right)-b_6\ln\left(1-x_{3,i}-x_{4,i}\right)\}]^{-1}\end{aligned} \tag{47}$$

where $b_0=\left[k+\left(\frac{\left(1-r_1\right)q_{0,0}}{\left(1-r_0\right)q_{1,0}}\right)+\ln(l)\right]$, $b_1=\left(\frac{1}{q_{0,0}}-\frac{1}{q_{1,0}}\right)$,

$b_2=\left[\ln\left(\frac{r_1q_{0,1}}{r_0q_{1,1}}\right)+\ln\left(\frac{\left(1-r_1\right)q_{0,0}}{\left(1-r_0\right)q_{1,0}}\right)\right]$, $b_3=\left(\frac{1}{q_{0,1}}-\frac{1}{q_{1,1}}-\frac{1}{q_{0,0}}+\frac{1}{q_{1,0}}\right)$, $b_4=a_1-a_0$,

$b_5=d_1-d_0$, $b_6=g_1-g_0$ and $l=\frac{\Gamma\left(a_1+d_1+g_1\right)\Gamma\left(a_0\right)\Gamma\left(d_0\right)\Gamma\left(g_0\right)}{\Gamma\left(a_0+d_0+g_0\right)\Gamma\left(a_1\right)\Gamma\left(d_1\right)\Gamma\left(g_1\right)}$.

Kay and Little (1987) provide a number of other examples involving discrete and continuous variables. If equation (46) involved the decomposition of an unknown multivariate distribution conditional on Y_i of continuous variables, then it becomes considerably more difficult to derive the specification of $g(\mathbf{x}_i;y_1)$. Guidelines and results presented by Arnold, Castillo and Sarabia (1999) provide a means for attempting these specifications, and are beyond the current scope of this paper.

4.3 Estimation

In order to utilize all of the information present in the distribution of the sample, given by equation (11), one should use the method of maximum likelihood to estimate the parameters of the BRM (Spanos, 1999). Given the independence of the sample, the log-likelihood function for the logistic form of the BRM (primarily used in the paper) takes the form:

$$\ln L(j\,;(\mathbf{y},\mathbf{x})) = \sum_{i=1}^{N}\left[y_i \ln(g(h(\mathbf{x}_i;y_1))) + (1-y_i)\ln(1-g(h(\mathbf{x}_i;y_1)))\right], \qquad (48)$$

where $g(.)$ is the logistic cdf (or other potential alternative) and $y_1 = G(h_1)$.[9] Now let $\partial\mathbf{h}_i$ denote the gradient of $h(\mathbf{x}_i;y_1)$ with respect to the vector y_1 (e.g. b), $\partial^2\mathbf{h}_i$ the Hessian, and $g'(.)$ the logistic pdf (or other potential alternative). Then:

$$\frac{\partial \ln L(j\,;(\mathbf{y},\mathbf{x}))}{\partial y_1} = \sum_{i=1}^{N}\left[\left(\frac{y_i - g(h(\mathbf{x}_i;y_1))}{g(h(\mathbf{x}_i;y_1))(1-g(h(\mathbf{x}_i;y_1)))}\right)g'(h(\mathbf{x}_i;y_1))\partial\mathbf{h}_i\right], \text{ and} \qquad (49)$$

[9] The case in example 2 also results in the same log-likelihood function.

$$\frac{\partial^2 \ln L(j,(\mathbf{y},\mathbf{x}))}{\partial y_1 \partial y_1'} = \sum_{i=1}^{N}\left[\left(\frac{y_i - g(h(\mathbf{x}_i;y_1))}{g(h(\mathbf{x}_i;y_1))(1-g(h(\mathbf{x}_i;y_1)))}\right)^2 (g'(h(\mathbf{x}_i;y_1)))^2(\partial\mathbf{h}_i)(\partial\mathbf{h}_i)'\right] \qquad (50)$$

$$+\sum_{i=1}^{N}\left[\left(\frac{y_i - g(h(\mathbf{x}_i;y_1))}{g(h(\mathbf{x}_i;y_1))(1-g(h(\mathbf{x}_i;y_1)))}\right)\left(g'(h(\mathbf{x}_i;y_1))(\partial\mathbf{h}_i)(\partial\mathbf{h}_i)' + g(h(\mathbf{x}_i;y_1))(\partial\mathbf{h}_i)(\partial^2\mathbf{h}_i)\right)\right].$$

When the index function is linear in the parameters, $\partial\mathbf{h}_i = \mathbf{t}(\mathbf{x}_i)$, a vector of transformations of the vector $\mathbf{x}_i$ (e.g. $\mathbf{t}(\mathbf{x}_i) = (\ln(x_{1,i}), \ln(x_{2,i}), \ln(1 - x_{1,i} - x_{2,i}))'$ for example 5) making $\partial^2\mathbf{h}_i = 0$. To estimate the parameters of the BRM, one must solve the first order conditions, $\frac{\partial L(j,(\mathbf{y},\mathbf{x}))}{\partial y_1} = \mathbf{0}$, given by (49). In general no closed form solution of the parameter estimates exists to this system of equations, requiring the use of a numerical optimization procedure, such as the Newton-Raphson method or the method of scoring, to estimate the model parameters of the BRM. (For these procedures see Fahrmeir and Tutz, 1994; Gourieroux, 2000; Spanos, 2000.) The asymptotic properties of consistency and asymptotic normality of the MLE estimates talked about in the next sub-section follow if certain regularity conditions are satisfied (see Spanos, 1999).

4.4 Simulation and Statistical Inference

A significant benefit of using the probabilistic reduction approach for developing the BRM is that it provides a mechanism for randomly generating the vector stochastic process, $\{(Y_i, \mathbf{X}_i), i = 1,..., N\}$ using the relationship given by equation (14) for simulations involving the BRM. The process involves performing two steps:

Step 1: Generate a realization of the stochastic process $\{Y_i, i = 1,..., N\}$ using a binomial random number generator.

Step 2: Using the regression function(s) associated with $f\left(\mathbf{X}_i \mid Y_i;\boldsymbol{h}_1\right)$ or $f\left(\mathbf{X}_i;\boldsymbol{h}_{1,j}\right)$ generate a realization of the vector stochastic process, $\left\{\mathbf{X}_i, i=1,...,N\right\}$ using appropriate random number generators for specified values of $\boldsymbol{h}_1$.

This approach is enticing in that no a priori theoretical interpretation need be imposed on the generation process, it is purely statistical in nature. Furthermore, the parameters y_1 can be easily determined from the parameters $\boldsymbol{h}_{1,j}$, via $y_1 = G(\boldsymbol{h}_1)$. Examples 1 thru 8 provide a number of these types of mappings.

To see how this procedure works, consider generating simulated data for a Monte Carlo experiment using the conditional density for $f\left(\mathbf{X}_i \mid Y_i;\boldsymbol{h}_1\right)$ specified in example 1.[10]

Example 1 continued: The stochastic process $\left\{(Y_i, \mathbf{X}_i), i=1,...,N\right\}$ was generated 10,000 times, letting $p = 0.6$, $a_0 = 2$, $a_1 = 1$ and $s^2 = 1$ for $T = 50, 100, 250, 500, 1000$. For each generation, the BRM specified in Example 1 was estimated and the parameter estimates saved. The simulation results are summarized in Table 3. These results provide support for the theoretical derivations done in section 4.1, given the closeness of the mean values of the estimated model parameters to their true values. Furthermore, the sample statistics provide support for the asymptotic properties of the estimators encountered in the literature, i.e. consistency and asymptotic normality (see Fahrmeir and Tutz, 1994; Gourieroux, 2000). As the sample size was increased, the means of the estimates approached the true values, the standard errors decreased, and the sample skewness and kurtosis statistics approached 0 and 3 (the values for the normal

[10] MATLAB was used to conduct all simulations. MATLAB code and results are available upon request from the author.

distribution) , respectively. Density plots of the sample distribution of the estimators, b_0 and b_1, for *T = 50* and *T = 500* are provided in Figures 1 thru 4. Again, these figures support the asymptotic normality results found in the literature.

Two more simulations were conducted using the conditional distributions in Table 2. Both provide additional support for the consistency and asymptotic normality of the parameters of the BRM when the index function is linear in the parameters.

Example 9: Monte Carlo simulation examining the BRM when $f\left(X_i;h_1\right)$ is Exponential. The stochastic process $\{(Y_i, \mathbf{X}_i), i = 1,...,N\}$ was generated 10,000 times, letting $p = 0.6$, $q_0 = 1.5$ and $q_1 = 2.0$, for $T = 50, 100, 500, 1000, 2500$. For each generation, the appropriate BRM specified in Table 2 was estimated and the parameter estimates saved. The simulation results are summarized in Table 4. Again, the sample statistics provide support for the asymptotic properties of consistency and asymptotic normality of the

Table 3: Summary Results for Monte Carlo Simulation when $f\left(X_i \mid Y_i;h_1\right)$ is Normal.

		Sample Statistics			
Parameter		**Mean**	**Standard Deviation**	**Skewness**	**Kurtosis**
b_0	**True Value**	**-2.095**			
	T = 50	-2.320	1.100	-0.963	5.89
	T = 100	-2.200	0.708	-0.516	3.80
	T = 250	-2.133	0.422	-0.262	3.23
	T = 500	-2.112	0.295	-0.199	3.17
	T = 1000	-2.103	0.208	-0.122	3.00
b_1	**True Value**	**1.0000**			
	T = 50	1.101	0.439	1.26	8.43
	T = 100	1.046	0.276	0.571	3.82
	T = 250	1.017	0.163	0.295	3.22
	T = 500	1.008	0.114	0.246	3.19
	T = 1000	1.004	0.080	0.135	2.93

Figure 1: Density Plot for $\hat{b}_0$ for $T = 50$

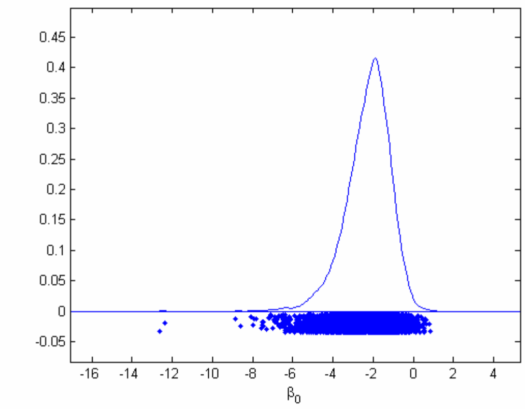


Figure 2: Density Plot for $\hat{b}_1$ for $T = 50$

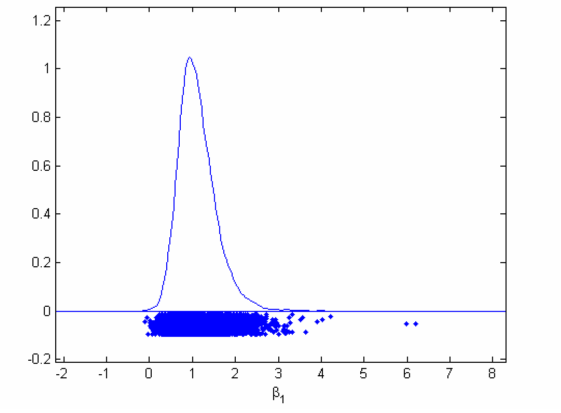


Figure 3: Density Plot for $\hat{b}_0$ for $T = 500$

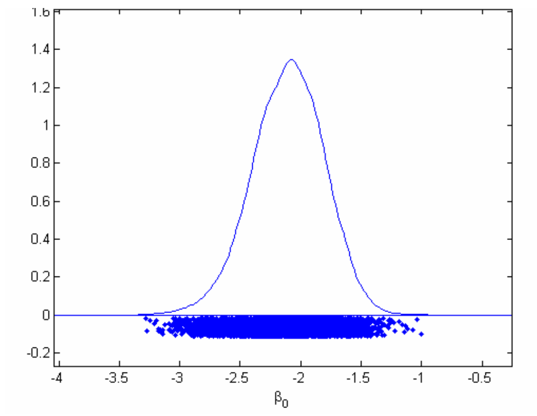


Figure 4: Density Plot for $\hat{b}_1$ for $T = 500$

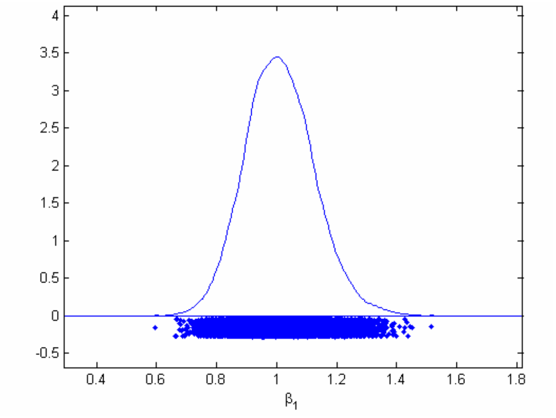


Table 4: Summary Results for Monte Carlo Simulation when $f\left(X_i;h_{1,j}\right)$ is Exponential.

		Sample Statistics			
Parameter		**Mean**	**Standard Deviation**	**Skewness**	**Kurtosis**
b_0	**True Value**	**0.118**			
	T = 50	0.076	0.450	-0.015	3.27
	T = 100	0.105	0.306	-0.064	3.19
	T = 500	0.114	0.132	-0.034	3.03
	T = 1000	0.116	0.094	-0.043	3.00
	T = 2500	0.118	0.059	-0.033	2.99
b_1	**True Value**	**0.167**			
	T = 50	0.209	0.226	0.760	5.08
	T = 100	0.183	0.142	0.539	4.15
	T = 500	0.170	0.058	0.217	3.17
	T = 1000	0.168	0.041	0.182	3.19
	T = 2500	0.167	0.026	0.089	3.10

estimators encountered in the literature (see Fahrmeir and Tutz, 1994; Gourieroux, 2000). Density plots of the sample distribution of the estimators, b_0 and b_1, for *T* = *50* and *T* = *500* are provided in Figures 5 thru 8. Again, these figures support the asymptotic normality results found in the literature.

Example 10: Monte Carlo simulation examining the BRM when $f(X_i;h_1)$ is Log-Normal. The stochastic process $\{(Y_i,\mathbf{X}_i), i=1,...,N\}$ was generated 10,000 times, letting $p=0.6$, $m_0=2.0$, $m_1=2.4$, $s_0=1.2$ and $s_1=1.0$ for $T=50, 100, 250, 500, 1000$. For each generation, the appropriate BRM specified in Table 2 was estimated and the

Figure 5: Density Plot for $\hat{b}_0$ for *T* = *50*

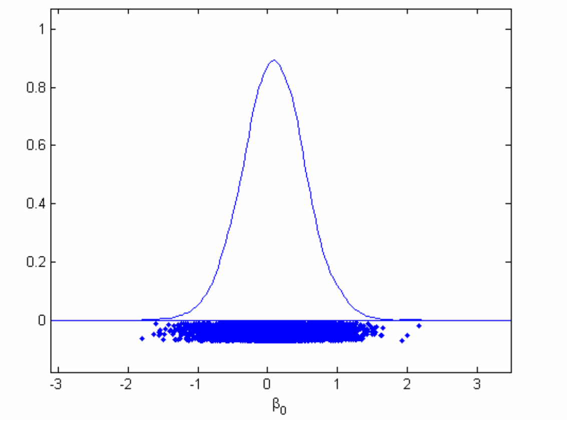


Figure 6: Density Plot for $\hat{b}_1$ for *T* = *50*

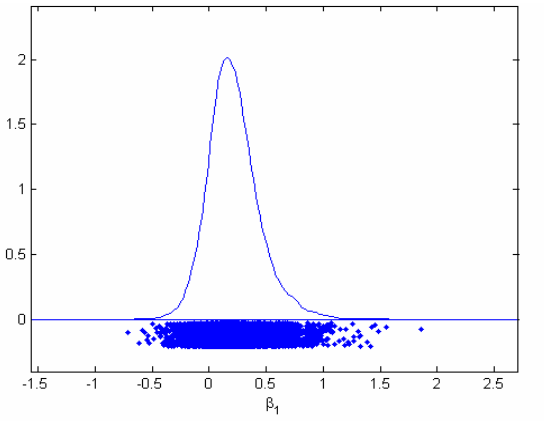


Figure 7: Density Plot for $\hat{b}_0$ for *T* = *1000*

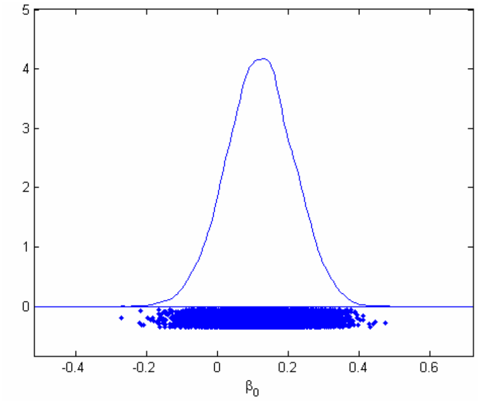
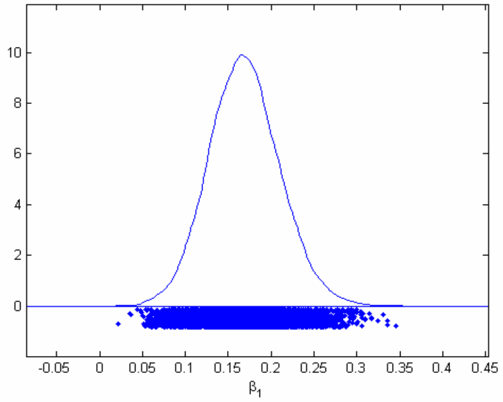


Figure 8: Density Plot for $\hat{b}_1$ for *T* = *1000*



parameter estimates saved. The simulation results are summarized in Table 5. Again, the sample statistics provide support for the asymptotic properties of consistency and asymptotic normality of the estimators encountered in the literature (see Fahrmeir and Tutz, 1994; Gourieroux, 2000).Density plots of the sample distribution of the estimators, b_0, b_1 and b_2, for *T = 50* and *T = 500* are provided in Figures 9 thru 14. Again, these figures support the asymptotic normality results found in the literature.

All of these examples provide evidence that the model parameters of the BRM, when the index function is linear in the parameters, are consistent and asymptotically normal. These properties allow modelers to conduct statistical inferences using the BRM,

Table 5: Summary Results for Monte Carlo Simulation when $f\left(X_i;h_{1,j}\right)$ is Log-Normal.

		Sample Statistics			
Parameter		**Mean**	**Standard Deviation**	**Skewness**	**Kurtosis**
b_0	**True Value**	**-0.903**			
	T = 50	-1.085	1.458	-1.78	30.8
	T = 100	-1.015	0.849	-0.522	4.53
	T = 250	-0.948	0.476	-0.317	3.32
	T = 500	-0.922	0.326	-0.279	3.25
	T = 1000	-0.911	0.224	-0.186	3.15
b_1	**True Value**	**0.960**			
	T = 50	1.160	1.314	0.983	16.2
	T = 100	1.102	0.752	0.384	4.23
	T = 250	1.047	0.424	0.299	3.41
	T = 500	1.026	0.289	0.267	3.23
	T = 1000	1.018	0.197	0.200	3.12
b_2	**True Value**	**-0.153**			
	T = 50	-0.174	0.285	-0.254	9.96
	T = 100	-0.167	0.159	-0.18	4.09
	T = 250	-0.159	0.090	-0.185	3.47
	T = 500	-0.155	0.061	-0.188	3.16
	T = 1000	-0.154	0.042	-0.129	3.09

Figure 9: Density Plot for $\hat{b}_0$ for $T = 50$

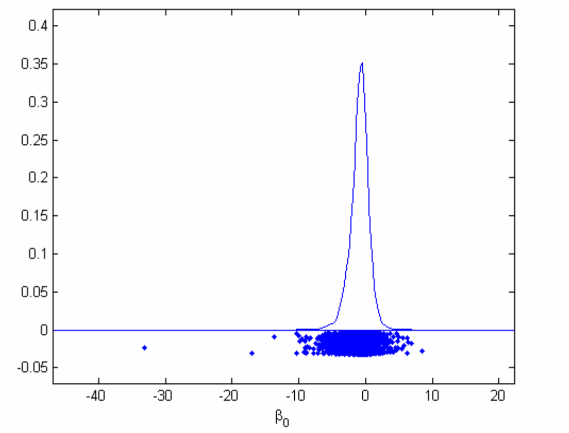


Figure 10: Density Plot for $\hat{b}_1$ for $T = 50$

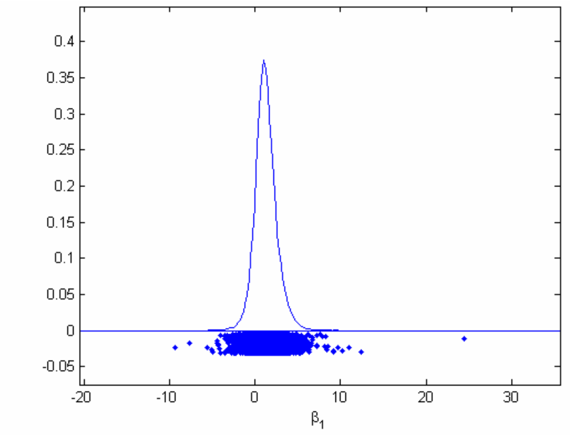


Figure 11: Density Plot for $\hat{b}_2$ for $T = 50$

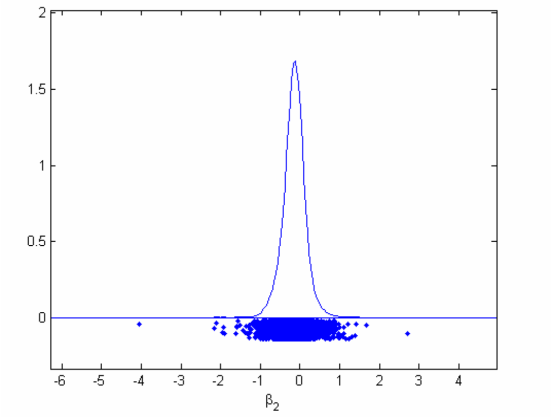


Figure 12: Density Plot for $\hat{b}_0$ for $T = 500$

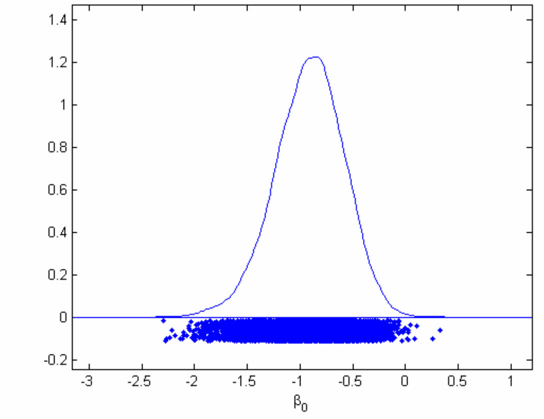


Figure 13: Density Plot for $\hat{b}_1$ for $T = 500$

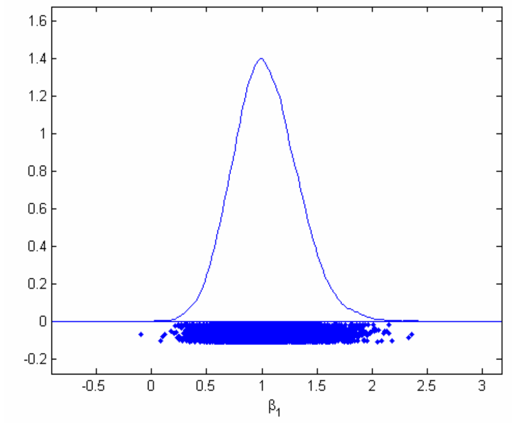
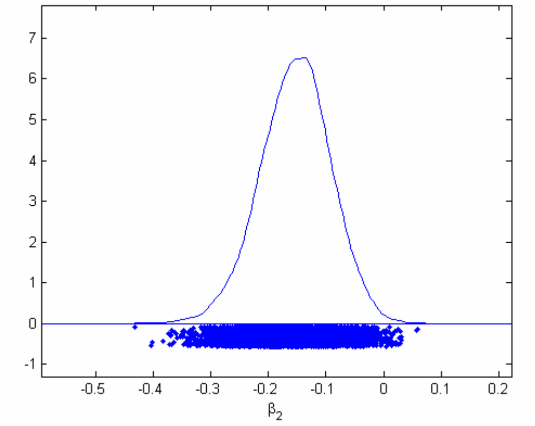


Figure 14: Density Plot for $\hat{b}_2$ for $T = 500$



such as testing the hypotheses $H_0 : b_i = 0$ against $H_1 : b_i \neq 0$ using the asymptotic likelihood ratio test.

5. The Multinomial Regression Model

Using the probabilistic reduction approach and theoretical results in section 4.1, it is a relatively straightforward extension to provide a formal presentation of the multinomial regression model (MRM). Let $\{C_i ; i = 1,..., N\}$ be a stochastic process on the probability space $(S, \Im, P(.))$, where C_i takes values $\{0,1,2,3,...., R\}$ and the ordering of the values in not important (i.e. C_i is a nominal variable). Now consider recoding C_i using $Y_{r,i}$, for $r = 1,..., R$, such that:

$$Y_{r,i} = \begin{cases} 1 \text{ if } C_i = r \\ 0 \text{ otherwise} \end{cases}. \tag{51}$$

When $C_i = 0$, $Y_{i,r} = 0$ for all r. Thus, if $\mathbf{Y}_i = (Y_{1,i},..., Y_{R,i})'$, then when $C_i = r$, $\mathbf{Y}_i = (0,...,1,...,0)' = \mathbf{e}_r$, where the '1' appears in the rth position of the vector. Given that each $Y_{r,i}$ is distributed Bernoulli(p_r) and defined on the same probability space:

$$\mathbf{P}(C_i = r) = \mathbf{P}(Y_{r,i} = 1) = p_r . \tag{52}$$

Fahrmeir and Tutz (1994) state that the distribution of $\mathbf{Y}_i$ and in turn C_i is:

$$f(\mathbf{Y}_i ; \mathbf{p}) = p_1^{Y_{1,i}} p_2^{Y_{2,i}} \cdots p_R^{Y_{R,i}} \left(1 - \sum_{r=1}^{R} p_r \right)^{1 - \sum_{r=1}^{R} Y_{r,i}}, \tag{53}$$

where $\sum_{r=1}^{R} p_r = 1$ and $\sum_{r=1}^{R} Y_{r,i} = 1$. Thus, $\mathbf{Y}_i$ (or C_i) is distributed multinomial, i.e. $\mathbf{Y}_i \sim M(\mathbf{p},1)$, where $\mathbf{p} = (p_1,..., p_R)'$. In addition, based on the above information

$E(Y_{r,i}) = p_r$ and $Var(Y_{r,i}) = p_r(1 - p_r)$, $E(\mathbf{Y}_i) = \mathbf{p}$ (an $(R \times 1)$ vector) and $Cov(\mathbf{Y}_i) = \text{diag}(\mathbf{p}) - \mathbf{pp}'$ (an $(R \times R)$ matrix) (Fahrmeir and Tutz, 1994).

A modeler is usually interested in the MRM, because they are interested in determining the probability that $C_i = r$ or equivalently $Y_{r,i} = 1$. It turns out that the latter interpretation will provide us with a workable statistical model. The joint density function of the joint vector stochastic process, $\{\mathbf{Y}_i, \mathbf{X}_i, i = 1,..., N\}$ takes the form:

$$f(\mathbf{Y}_1,..., \mathbf{Y}_N, \mathbf{X}_i,..., \mathbf{X}_N; f), \tag{54}$$

where f is an appropriate set of parameters. Again, assuming that $\{\mathbf{Y}_i, \mathbf{X}_i, i = 1,..., N\}$ is both an independent and identically distributed joint vector stochastic process, the joint density function (54) can be decomposed in a similar manner to (12) and (13) giving:

$$f(\mathbf{Y}_1,..., \mathbf{Y}_N, \mathbf{X}_i,..., \mathbf{X}_N; f) = \prod_{i=1}^{N} f(\mathbf{Y}_i \mid \mathbf{X}_i; y_1) f(\mathbf{X}_i; y_2). \tag{55}$$

From section 4.1 we know that the decomposition in (55) arises only when:

$$f(\mathbf{Y}_i \mid \mathbf{X}_i; y_1) \cdot f(\mathbf{X}_i; y_2) = f(\mathbf{X}_i \mid \mathbf{Y}_i; h_1) \cdot f(\mathbf{Y}_i; \mathbf{p}) = f(\mathbf{Y}_i, \mathbf{X}_i; j). \tag{56}$$

Again, using condition (56) consider the set of following relationships: for $r = 1,..., R$

$$\frac{f(\mathbf{X}_i \mid \mathbf{Y}_i = \mathbf{e}_r; h_1)}{f(\mathbf{X}_i \mid \mathbf{Y}_i = \mathbf{0}; h_1)} \cdot \frac{f(\mathbf{Y}_i = \mathbf{e}_r; \mathbf{p})}{f(\mathbf{Y}_i = \mathbf{0}; \mathbf{p})} = \frac{f(\mathbf{Y}_i = \mathbf{e}_r \mid \mathbf{X}_i; y_1)}{f(\mathbf{Y}_i = \mathbf{0} \mid \mathbf{X}_i; y_1)} \cdot \frac{f(\mathbf{X}_i : y_2)}{f(\mathbf{X}_i; y_2)}, \tag{57}$$

where $\mathbf{e}_r$ was defined above and $\mathbf{0}$ is a $(R \times 1)$ vector of zeros. Furthermore, assume that $f(\mathbf{Y}_i \mid \mathbf{X}_i; y_2)$ is a conditional multinomial distribution with the following functional form:

$$f(\mathbf{Y}_i \mid \mathbf{X}_i; y_1) = \left(\prod_{r=1}^{R} [g_r(\mathbf{X}_i; y_1)]^{Y_{r,i}} \right) \left(1 - \sum_{r=1}^{R} g_r(\mathbf{X}_i; y_1) \right)^{1 - \sum_{r=1}^{R} Y_{r,i}}, \tag{58}$$

where $g_r(\mathbf{X}_i;y_1)=\mathbf{R}^K \to [0,1]$, $y_1 \in \Theta_1, \sum_{r=1}^{R} g_r(\mathbf{X}_i;y_1)=1$ and $\sum_{r=1}^{R} Y_{r,i}=1$. Again it is relatively easy to verify that the density function given by equation (58) is a proper density function. Substituting equation (58) into equation (57) and letting $p_r = f(\mathbf{Y}_i=\mathbf{e}_r;\mathbf{p})$ and $p_0 = f(\mathbf{Y}_i=\mathbf{0};\mathbf{p})$, gives: for $r=1,...,R$

$$\frac{f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{e}_r;h_1)}{f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{0};h_1)}\cdot\frac{p_r}{p_0}=\frac{g_r(\mathbf{X}_i;y_1)}{1-\sum_{r=1}^{R} g_r(\mathbf{X}_i;y_1)}\cdot\frac{f(\mathbf{X}_i:y_2)}{f(\mathbf{X}_i;y_2)}. \tag{59}$$

Solving the system of *R* equations given by (59) gives: for $r=1,...,R$

$$g_r(\mathbf{X}_i;y_1)=\frac{p_r f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{e}_r;h_1)}{p_0 f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{0};h_1)}\left[1+\sum_{r=1}^{R}\left(\frac{p_r f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{e}_r;h_1)}{p_0 f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{0};h_1)}\right)\right]^{-1}. \tag{60}$$

Again using the transformation, $x=\exp\{\ln(x)\}$ and after rearranging some terms provides a more familiar form:

$$g_r(\mathbf{X}_i;y_1)=\frac{\exp\{h_r(\mathbf{X}_i;h_{1,r})\}}{1+\sum_{r=1}^{R}\exp\{h_r(\mathbf{X}_i;h_{1,r})\}}, \tag{61}$$

where $h_r(\mathbf{X}_i;h_{1,r})=\ln\left(\frac{f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{e}_r;h_1)}{f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{0};h_1)}\right)+k_r$, $k_r=\ln\left(\frac{p_r}{p_0}\right)$ and

$y_1=(y_1,...,y_R)=G(h_1)$. As was done in the previous section, one can also write:

$$f(\mathbf{X}_i \mid \mathbf{Y}_i=\mathbf{e}_r;h_1)=f(\mathbf{X}_i;h_{1,r}). \tag{62}$$

The formulation given by equation (61) is reminiscent of the multinomial logit model commonly found in the literature.

The above reduction gives rise to a family of statistical models for a number of discrete choice processes where the dependent variable is a nominal variable that takes on more than two values. Given that

$$Cov(\mathbf{Y}_i \mid \mathbf{X}_i = \mathbf{x}_i) = \mathrm{diag}(\mathbf{g}(\mathbf{x}_i;y_1)) - \mathbf{g}(\mathbf{x}_i;y_1)\mathbf{g}(\mathbf{x}_i;y_1)' < \infty, \qquad (63)$$

where $\mathbf{g}(\mathbf{x}_i;y_1) = (g_1(\mathbf{x}_i;y_1),..., g_R(\mathbf{x}_i;y_1))'$, the stochastic process $\{\mathbf{Y}_i \mid \mathbf{X}_i = \mathbf{x}_i, i = 1,..., N\}$ can be decomposed orthogonally giving rise to the following system of regression functions:

$$\mathbf{Y}_i = E(\mathbf{Y}_i \mid \mathbf{X}_i = \mathbf{x}_i) + \mathbf{u}_i = \mathbf{g}(\mathbf{x}_i;y_1) + \mathbf{u}_i, \qquad (64)$$

where $\mathbf{Y}_i$, $E(\mathbf{Y}_i \mid \mathbf{X}_i = \mathbf{x}_i)$, $\mathbf{u}_i$ and $\mathbf{g}(\mathbf{x}_i, y_1)$ are all $(R \times 1)$ vectors, and $\mathbf{u}_i \sim M(\mathbf{0},1)$ with $Cov(\mathbf{u}_i)$ given by equation (63). Thus, equation (64) represents the statistical generating mechanism of the MRM. The multinomial regression model is more formally specified with assumptions in Table 6.

As stated by Spanos (2002), the multinomial regression model does not take account of the natural ordering of the values of the dependent variable, C_i, given it is

Table 6: Multinomial Regression Model

SGM:	$\mathbf{Y}_i = \mathbf{g}(\mathbf{x}_1;y_1) + \mathbf{u}_i$, $i = 1,..., N$, where $\mathbf{u}_i \sim M(\mathbf{0},1)$ with $Cov(\mathbf{u}_i) = \mathrm{diag}(\mathbf{g}(\mathbf{x}_i;y_1)) - \mathbf{g}(\mathbf{x}_i;y_1)\mathbf{g}(\mathbf{x}_i;y_1)'$.
Assumptions	
Distributional:	$Y_i \mid \mathbf{X}_i = \mathbf{x}_i \sim M(\mathbf{g}(\mathbf{X}_i;y_1),1)$, (conditional Multinomial).
Functional Form:	$E(Y_i \mid \mathbf{X}_i = \mathbf{x}_i) = \mathbf{g}(\mathbf{x}_i;y_1)$, where $\mathbf{g}(\mathbf{x}_i;y_1) = (g_1(\mathbf{x}_i;y_1),..., g_R(\mathbf{x}_i;y_1))$, $g_r(\mathbf{x}_i;y_1) = \dfrac{\exp\{h_r(\mathbf{x}_i;h_{1,r})\}}{1 + \sum_{r=1}^{R} \exp\{h_r(\mathbf{x}_i;h_{1,r})\}}$, $h_r(\mathbf{x}_i;h_{1,r}) = \ln\left[\dfrac{f(\mathbf{X}_i \mid \mathbf{Y}_i = \mathbf{e}_r;h_1)}{f(\mathbf{X}_i \mid \mathbf{Y}_i = \mathbf{0};h_1)}\right] + k_r$ and $y_1 = G(h_1)$.
Heteroskedasticity:	$Var(\mathbf{Y}_i \mid \mathbf{X}_i = \mathbf{x}_i) = \mathrm{diag}(\mathbf{g}(\mathbf{x}_i;y_1)) - \mathbf{g}(\mathbf{x}_i;y_1)\mathbf{g}(\mathbf{x}_i;y_1)'$.
Homogeneity:	$y_1 = G(h_1)$ is not a function of $i = 1,..., N$.
Independence:	$\{\mathbf{Y}_i \mid \mathbf{X}_i = \mathbf{x}_i, i = 1,..., N\}$ is an independent stochastic process.

assumed to be of nominal scale. If such an ordering matters, then an alternative model should be utilized that can account of the ordering.

The specification and estimation of this model are currently beyond the discussion in the present paper. This section was meant as a simple extension of the BRM. These other issues will be addressed in a future paper by the author.

6. Concluding Remarks

The Bernoulli Regression Model can provide a parsimonious description of the probabilistic structure of conditional binary choice processes examined in the field of economics. Contrary to the limitations encountered by the latent variable or transformational approaches for specifying econometric models with binary choice variables, the Bernoulli Regression Model imposes no a priori theoretical or ad hoc restrictions (or assumptions) upon the model, thereby providing a theory-free statistical model of the conditional binary choice process being examined. As noted by Spanos (1995), this freedom allows the modeler to conduct statistical inferences (if the statistical assumptions made about the underlying stochastic process are appropriate) that can be used to examine if the theory in question can account for the systematic information in the observed data.

The latent variable approach can be used to derive a (parametric) theoretical model that can be associated with the estimated statistical model. In this sense, the modeler tries to find a mapping from the parameters of the statistical model to the parameters of the theoretical model (i.e. the modeler tries to *identify* the theoretical parameters) (Spanos, 1986). Once this is accomplished, the statistical model can be used to evaluate the theoretical model and perform meaningful statistical inferences. Thus, the latent variable approach provides us with a mechanism for relating theory of choice to applied statistical models. The modeler must be careful not to impose the theoretical structure upon the statistical model, given that any inferences taken from such a model would be questionable, in that one would be using the same theory to answer questions about that theory.

References

1. Amemiya, T. “Qualitative Response Models: A Survey.” *Journal of Economic Literature*. 19(December 1981): 1483 – 1536.
2. Arnold, B.C., E. Castillo and J.M. Sarabia. *Conditional Specification of Statistical Models*. New York: Springer Verlag, 1999.
3. Arnold, B.C. and S.J. Press. “Compatible Conditional Distributions.” *Journal of the American Statistical Association*. 84(March, 1989): 152 – 156.
4. Chen, S.X. and J.S. Liu. “Statistical Applications of the Poisson-Binomial and Conditional Bernoulli Distributions.” *Statistica Sinica*. 7(1997): 875 – 892.
5. Coslett, S.R. “Distribution-Free Maximum Likelihood Estimator of the Binary Choice Model.” *Econometrica*. 51(May 1983): 765 – 782.
6. Fahrmeir, L. and G. Tutz. *Multivariate Statistical Modeling Based on Generalized Linear Models.* New York: Springer-Verlag, 1994.
7. Gourieroux, C. *Econometrics of Qualitative Dependent Variables*. Cambridge, UK: Cambridge University Press, 2000.
8. Kay, R. and S. Little. “Transformations of the Explanatory Variables in the Logistic Regression Model for Binary Data.” *Biometrika*. 74(September, 1987): 495 – 501.
9. Lauritzen, S.L. and N. Wermuth. “Graphical Models for Association Between Variables, Some Which Are Qualitative and Some Quantitative.” *Annals of Statistics*. 17(1989): 31 – 57.
10. Maddala, G.S. *Limited Dependent and Qualitative Variables in Econometrics*. Cambridge, UK: Cambridge University Press, 1983.

11. Nelder, J.A. and R.W.M. Wedderburn. “Generalized Linear Models.” *Journal of the Royal Statistical Society, Series A (General).* 3(1972): 370 – 384.

12. Powers, D.A. and Y. Xie. *Statistical Methods for Categorical Data Analysis*. San Diego, CA: Academic Press, 2000.

13. Spanos, A. “Chapter 20: Regression Like Models.” Department of Economics, Virginia Polytechnic Institute and State University, mimeo. 2000.

14. Spanos, A. “On Theory Testing In Econometrics: Modeling with Nonexperimental Data.” *Journal of Econometrics*. 67(1995): 189 – 226.

15. Spanos, A. *Probability Theory and Statistical Inference: Econometric Modeling with Observational Data*. Cambridge, UK: Cambridge University Press, 1999.

16. Spanos, A. *Statistical Foundations of Econometrics Modeling*. Cambridge, UK: Cambridge University Press, 1986.

17. Train, K.E. *Discrete Choice Methods with Simulation*. Cambridge, UK: Cambridge University Press, 2003.

Chapter 2

Networking Your Way to a Better Fit: Using Artificial Neural Networks to Model Dichotmous Choice Contingent Valuation Survey Data

Abstract

Dichotomous choice contingent valuation models, such as the binary logit and probit, are subject to potential functional form misspecifications when modelers impose a priori theoretical interpretations on the models prior to estimation, without considering the underlying probabilistic structure of the observed data. Feed-forward backpropagation artificial neural networks (FFBANNs) provide a semi-nonparametric alternative to the traditional binary logit and probit models. These types of neural networks can act as universal approximators and provide a robust classification tool for out-of-sample prediction, allowing the modeler to potentially avoid problems arising from an incorrect functional form. The purpose of this paper is to empirically compare the predictive capabilities of feed-forward backpropagation artificial neural networks and binary logit and probit models using contingent valuation survey data. The paper provides a statistical perspective for viewing artificial neural networks as binary statistical models and neural network modeling guidelines, as well. Results show that the artificial neural networks performed marginally better than the binary logit and probit models, but these differences tended not to be statistically significant.

1. Introduction

Calculating theoretically consistent willingness-to-pay (WTP) and willingness-to-accept (WTA) measures using contingent valuation survey data in nonmarket valuation studies requires that the underlying econometric model satisfy the utility maximization hypothesis. The most widely used models for this purpose are the logit and probit type models. To satisfy the utility maximization hypothesis, the argument (or index function) of these models must be able to be interpreted as the difference in utility between two states of existence defined by the dependent variable (Hanemann, 1984). Hanneman (1984) purports that this requirement provides a practical procedure for specification of the functional form of the index function of the corresponding model, by postulating a priori the underlying functional form of the utility function. The a priori imposition of such a theoretical structure on the statistical model without considering the underlying probabilistic structure of the observed data is likely to leave the postulated model misspecified statistically.

In order to guarantee that a statistical model is properly specified it should be viewed as isolated from the theory it is purporting to explain (Spanos, 1999). From this viewpoint, Kay and Little (1987) show that when the joint density function of the explanatory variables conditional on the dependent variable is multivariate normal with homogenous variance/covariance matrix the resulting index function of the logistic model is linear, which seems unlikely in the majority of empirical cases. Arnold, Castillo and Sarabia (1999) find such observations make many of the logistic regression models used in applied literature questionable. The probit model represents even more of a conundrum, given that to date no one has derived conditions for this model analogous to those for the logit. Thus, assuring that the functional forms of the logit and probit models are properly specified poses a significant problem. One solution would be to weaken the assumed distributional assumptions of the models and rely upon semi-nonparametric techniques.

An innovative semi-nonparametric technique used for classification problems is the artificial neural network (ANN). These networks have the ability to learn arbitrary and highly nonlinear functional mappings using finite data (Mehrotra, Mohan and Ranka, 1997). White, Hornik and Stinchcombe (1992) show that feed-forward backpropagation artificial neural networks (FFBANNs) can act as universal function approximators under fairly general conditions, and Ripley (1994) explains that this result is easily extended to networks that are used to model binary choice processes. Furthermore, the outputs of these latter types of neural networks can be interpreted as conditional probabilities (Ripley, 1996), providing a semi-nonparametric approach for modeling dichotomous choice problems.

Studies comparing ANNs and logit and/or probit models have examined problems in political science (see Zeng, 1996), medicine (see Arana, Delicado and Marti-Bonmati, 1999), management science (see Tam and Kiang, 1992), and finance (see Zurada et. al., 1999). Overall, these studies suggest that ANNs perform better than or comparably to binary logit and probit models based on out-of-sample prediction power. Furthermore, ANNs were able to predict cases never correctly predicted by traditional statistical approaches. The problem dependent nature of neural networks tends to be their major drawback, because the topological structure of an ANN can be significantly different for two similar problems. Only “rules of thumb” for constructing ANNS are provided in the literature and these guidelines tend to be applicable under certain conditions and problem-settings (Ripley, 1996). These studies suggest that further research continue to compare artificial neural networks to traditional statistical techniques across a variety of fields and problem situations, in order to help understand this drawback and develop general modeling guidelines.

The purpose of this paper is to empirically compare the predictive capabilities of feed-forward backpropagation artificial neural networks and traditional dichotomous choice logit and probit models using contingent valuation (CV) survey data. The objectives of the study are: (i) to provide a statistical context in which to apply FFBANNs for modeling dichotomous choice survey data, (ii) to provide modeling guidelines for the construction and simulation of artificial neural networks using survey data, (iii) to compare the out-of-sample predictive capabilities of feed-forward ANNs to traditional dichotomous choice logit and probit models, and (iv) to provide an algorithm for determining consistent WTP and WTA measures using FFBANNs.

The remainder of the paper is organized as follows. Section two examines and critiques the traditional modeling approach to analyzing CV survey data using dichotomous choice models. Section three then presents a brief but comprehensive overview of FFBANNs and how they can be interpreted as statistical models for modeling CV survey data. Section four proposes general guidelines for constructing FFBANNs by conducting simulations examining a number of decisions that must be made during the modeling process. Section five then examines the predictive capabilities of the FFBANNs to the traditional logit and probit models using the optimal network architectures determined in section four. Section six then examines how consistent WTP and WTA measures can be estimated using FFBANNs. Finally section seven provides concluding remarks and avenues for future research.

2. A Critique of Traditional Modeling Approaches to Analyzing CV Survey Data.

This section of the paper examines the traditional approach of analyzing contingent valuation survey data with binary responses. The traditional viewpoint, a latent variable approach, provides a theoretically consistent approach to modeling CV data, but fails to adequately take account of the underlying statistical assumptions of the proposed estimable model. In contrast, by viewing the response data from a CV survey as inherently categorical (where no latent variable is invoked) and using a purely statistical vantage point for constructing the model, a number of pertinent issues that fail to be addressed by traditional econometric approaches are revealed (Powers and Xie, 2000).

2.1 A Theoretically Consistent Dichotomous Choice Contingent Valuation Model

Hanemann's (1984) seminal paper on welfare evaluations with discrete responses provides the theoretical basis for modeling contingent valuation survey data with binary responses. Following Hanemann (1991), consider an individual who derives utility from the supply of some environmental amenity (e.g. water quality) and let q denote the level of supply of that amenity. Furthermore, let y denote the individual's income and s denote a vector of variables representing the consumption of other market commodities, prices, demographic characteristics and other attributes of the individual.

It is assumed that the researcher does not completely know the functional form of the individual's indirect utility function, thus the unknown components are treated as stochastic. The indirect utility function is given by:

$$V_i(q_i, y, s, e_i) = v_i(q_i, y, s) + e_i, \qquad (1)$$

where $v(.)$ represents the observable component (mean) of the indirect utility function, e is an IID random variable with zero mean representing the unobservable component, and i denotes the level of q being consumed (An, 2000; Hanemann, 1984).[11]

Consider the situation where the individual is now faced with the opportunity of increasing her consumption of *q* from q_0 to q_1. If an increase in *q* is seen as desirable by the individual then $V_1(q_1, y, s, e_1) \geq V_0(q_0, y, s, e_0)$ (Cooper, 2002). If the individual is told that the increase in *q* will cost $*C*, then the individual will pay that amount if:

$$V_1(q_1, y - C, s, e_1) \geq V_0(q_0, y, s, e_0). \qquad (2)$$

The individual's maximum WTP, C_p is equal to the compensating variation measure of the change in *q*, which is found by solving for *C* using (2) with the inequality replaced by equality (Hanemann, 1991). On the other hand, if the individual is told that she will be paid $C to forego an increase in *q*, then the individual will accept that amount if:

$$V_0(q_0, y + C, s, e_0) \geq V_1(q_1, y, s, e_1). \qquad (3)$$

The individual's minimum WTA, C_a, is equal to the equivalent variation measure of the change in *q*, which is found by solving for *C* using (3) with the inequality replaced by equality (Hanemann, 1991). For further reference, the situation characterized by equation (2) will be denoted the WTP case and the situation characterized by equation (3) as the WTA case.

The researcher does not observe the actions of the individual, but only if the individual pays or accepts the offer of $*C* or not (Hanemann, 1991), so the response of

[11] The reference to calling this approach the latent variable approach stems from the fact that utility is not directly observable, and the researcher is trying to reveal the underlying utility (see Powers and Xie, 2000).

the individual can be empirically viewed in a probabilistic framework, where *p* represents the probability that the offer is accepted. In the WTP case:

$$\begin{aligned} p &= \mathbf{P}(\text{individual pays \$C to increase } q) \\ &= \mathbf{P}(V_1(q_1, y-C, s, e_1) \geq V_0(q_0, y, s, e_0)) \\ &= \mathbf{P}(v_1(q_1, y-C, s) + e_1 \geq v_0(q_0, y, s) + e_0) \\ &= \mathbf{P}(\Delta v \geq h) \end{aligned}, \tag{4}$$

where $\Delta v = v_1(q_1, y-C, s) - v_0(q_0, y, s)$ and $h = e_0 - e_1$. In the WTA case:

$$\begin{aligned} p &= \mathbf{P}(\text{individual accepts \$C to forego an increase in } q) \\ &= \mathbf{P}(V_0(q_0, y+C, s, e_0) \geq V_1(q_1, y, s, e_1)) \\ &= \mathbf{P}(v_0(q_0, y+C, s) + e_0 \geq v_1(q_1, y, s) + e_1) \\ &= \mathbf{P}(\Delta v \geq h) \end{aligned}, \tag{5}$$

where $\Delta v = v_0(q_0, y+C, s) - v_1(q_1, y, s)$ and $h = e_1 - e_0$ (Hanemann, 1984). Under the assumption that e_0 and e_1 are IID, h has the same distribution, whether $h = e_0 - e_1$ or $h = e_1 - e_0$. Based on this result, Hanemann (1984) states that equations (4) and (5) can be written as:

$$p = F_h(\Delta v), \tag{6}$$

where $F_h(.)$ is the cumulative density function of h. Thus, as stated by Hanemann (1984), "if the statistical binary response model is to be interpreted as the outcome of a utility-maximizing choice, the argument of $F_h(.)$ … must take the form of a utility difference [i.e. Δv] (p. 334)." As seen below, this result provides a mechanism to determine if a given statistical model is compatible with the utility maximization hypothesis, and provides a procedure for specifying a theoretically consistent functional form for a given model (Hanemann, 1984). Once Δv has been specified, the modeler need only specify $F_h(.)$, which is dependent upon the assumed distributions of e_0 and e_1.[12]

The common case of a linear indirect utility function is illustrated in the following example. Following Hanemann (1984) suppose that:

$$v_i(q_i, y, s) = a_i + by\,, \tag{7}$$

where $a_i = a_i(s)$, $b > 0$ and $i = 0{,}1$. Under the WTP case and using equation (7):

$$\Delta v = a - bC\,, \tag{8}$$

where $a = (a_1 - a_0)$.[13] Thus, a theoretically consistent estimable discrete choice model is gjven by:

$$p = F_h(a - bC). \tag{10}$$

Under the WTA case, equation (10) becomes:

$$p = F_h(a + bC), \tag{11}$$

where $a = (a_0 - a_1)$.[14]

If one assumes that for any fixed (q_i, s, e_i) that $V_i(q_i, y, s, e_i)$ is monotone increasing in y, then there exists an inverse function $Y_i(V_i, q_i, s, e_i)$ such that $Y_i(V_i(q_i, y, s, e_i), q_i, s, e_i) = y$ for all $y \geq 0$. Thus, under the WTP case:

$$C_p = C_p(q, y, s, e) = y - Y_1(V_0(q_0, y, s, e_0), q_1, s, e_1), \tag{12}$$

[12] When e_0 and e_1 are IID extreme value, then $F_h(.)$ is the logistic cdf. If instead e_0 and e_1 are IID normal, then $F_h(.)$ is the normal cdf (Train, 2003).

[13] The vector s can be explicitly included in the following manner: let $a_0(s) = l's$ and $a_1(s) = g's$, where l, g and s are all $(k \times 1)$ vectors of explanatory variables. Then $a = a_1 - a_0 = (l - g)' s$, which leaves equation (8) as a linear function of the explanatory variables. In this case, the data can only be used to identify the difference $(l - g)$ or a (Hanemann, 1984).

[14] For the WTA case, $a = (g - l)' s$.

where $q = (q_0, q_1)$ and $e = (e_0, e_1)$ (An, 2000); and under the WTA case:

$$C_a = C_a(q, y, s, e) = Y_0(V_1(q_1, y, s, e_1), q_0, s, e_0) - y\,. \tag{13}$$

Continuing the example presented earlier, from equations (7) and (12):

$$C_p = (a + h)/b\ , \tag{14}$$

where $a = (a_1 - a_0)$ and $h = (e_1 - e_0)$; and $C_a = C_p$ with $a = (a_0 - a_1)$ and $h = e_1 - e_0$ (Hanemann, 1984).

Equations (12) and (13) provide an alternative formulation for viewing the dichotomous choice model, which is theoretically consistent as well. Under the WTP case, the probability that an individual will be willing to pay \$*C* for an increase in *q* can be equivalently stated as:

$$p = \mathbf{P}(C_p \geq C). \tag{15}$$

Continuing with the example of a linear indirect utility function, substituting equation (14) into equation (15) gives:

$$\begin{aligned} p &= \mathbf{P}((a + h / b) \geq C) \\ &= \mathbf{P}(h \geq bC - a) \\ &= \mathbf{P}(h \leq a - bC) \\ &= \mathbf{P}(\Delta v \geq h) \end{aligned} \tag{16}$$

(Cameron and James, 1987; Hanemann, 1984). Thus, if the researcher assumes that $F_h(.)$ is the cdf of h , then equation (16) leads to the same dichotomous choice model given by equation (10). Under the WTA case, the probability that an individual would be willing to accept \$*C* to forego an increase in *q* can be equivalently stated as:

$$p = \mathbf{P}(C_a \leq C). \tag{17}$$

Substituting equation (14) into equation (17) with $a = (a_0 - a_1)$ and $h = e_1 - e_0$ gives rise to the dichotomous choice model given by equation (11) (Hanemann, 1984). This latter approach to formulating an estimable model is commonly used in place of specifying Δv directly. Thus, for equations (15) and (17) to constitute a theoretically consistent model the arguments of these models must be able to be derived from a utility difference as in equation (6).

Even though the above approaches provide theoretically consistent methods for specifying an estimable model, the researcher still needs to worry about potential model misspecifications. In order to guarantee that the proposed statistical model is properly specified it should be viewed as isolated from the theory it is purporting to explain (Spanos, 1999). In the next section, the dichotomous choice models given by equations (10) and (11) are re-examined from a purely statistical vantage point in order to shed light on some of the potential statistical problems that arise from using these models for CV analysis.

2.2 Functional Forms for the Dichotomous Choice CVM

A weakness of the theoretical approach to specifying dichotomous choice models, as stated by Cosslett (1983), is that:

> "*There is clearly an unsatisfactory feature of this formulation* [the latent variable approach]. *In order to get a specific functional form for the choice of probabilities, one has to make an assumption about the distribution of the stochastic term, about which we generally have no a priori knowledge. Normally distributed stochastic terms could perhaps be justified by an informal appeal to the central limit theorem, but if the stochastic terms are in fact non-normal then the choice probability model is misspecified and the parameter estimates may be inconsistent* (p. 766)."[15]

[15] The prose in square brackets was added by the author for further clarification.

The potential for model misspecification due to incorrect functional form becomes evident when viewing the dichotomous choice CVM model from a purely statistical vantage point.

The researcher in the previous section only observes the response of the individual to the offer of $*C* to increase (or forego an increase in) *q* from q_0 to q_1. This response comes in the form of a ‘yes/no’ or ‘accept/reject’ answer. Thus, the response should be empirically viewed as a Bernoulli random variable with parameter *p*, which represents the probability of a response of ‘yes’ or ‘accept’ (Davis and Xie, 2000). Let R_i denote the response by the i^{th} individual, where $R_i = 1$ for ‘yes’ or ‘accept’ and $R_i = 0$ otherwise.

Given that $\mathrm{var}(R_i) = p(1-p) < \infty$, R_i can be decomposed orthogonally into a systematic and nonsystematic component, giving rise to the following regression function:

$$R_i = E(R_i) + u_i,\ i = 1,\ldots,N \tag{18}$$

where $E(R_i) = p$ and $u_i \sim \mathrm{bin}(1,0)$ with variance $p(1-p)$ (Spanos, 1999 and 1986).[16] To capture the changes in the probability of a ‘yes’ response across individuals, the statistical model represented by equation (18) is traditionally assumed to be heterogeneous in the parameter p, i.e. the probability that the i^{th} respondent answers ‘yes’ is dependent upon the individual being asked, making $E(R_i) = p_i$. This assumption gives rise to the alternative heterogeneous regression function:

[16] The expectation in equation (18) is taken with respect to the marginal distribution of R_i. The distribution of the error term u_i is such that u_i takes the value $1-p$ with probability p and the value $-p$ with probability $1-p$, so that $E(u_i) = 0$ and $Var(u_i) = p(1-p)$ (Maddala, 1983).

$$R_i = p_i + u_i \, , \; i = 1,\ldots, N \qquad (19)$$

where $E(R_i) = p_i$ and u_i ~ $\text{bin}(1,0)$ with variance $p_i(1-p_i)$. In its present form equation (19) is not operational since there are as many observations, i.e. survey respondents, as there are parameters to be estimated (Spanos, 2000). To alleviate this problem, researchers assume that p_i is dependent upon some vector of explanatory variables, denoted here as X_i, via the following relationship:

$$p_i = F[h(x_i;q)], \; i = 1,\ldots, N \, , \qquad (20)$$

where $F(.) : \mathbf{R} \rightarrow [0,1]$, $h(.) : \mathbf{R}^m \rightarrow \mathbf{R}$, *m* denotes the cardinality of (or number of elements in) the vector *X*$_i$, and q is a $(m \times 1)$ vector of unknown parameters.[17] $F(.)$ is referred to as the transformation function, while $h(.)$ is referred to as the index function. Furthermore, the function $F(.)$ is usually chosen to be a cdf and $h(.)$ a linear combination of the elements of the vector x_i (Amemiya, 1981; Davidson and MacKinnon, 1993; Spanos, 2000). Common choices used for $F(.)$ are the logistic and standard normal due to the fact that these functions provide " a reasonable approximation to relations likely to occur in practice and … [they are] manageable mathematically (Cox, 1958: p.216)."[18]

In an experimental context, X_i in equation (20) is usually treated as fixed (or being controlled by the experimenter), justifying the substitution of equation (20) directly into equation (19), giving rise to a proper regression function (in the same manner as the

[17] Using the example is section 2.1, the vector of explanatory variables with a constant added would be $X_i = (\mathbf{1}, C, s)$.

[18] The prose is square brackets was added by the author for further clarification. Other cumulative distribution functions have been proposed in the literature, such as the Cauchy and Burr (Maddala, 1983), but this sentiment still tends to be held by some econometricians.

Gauss linear model as represented by Spanos (1986)).[19] In the field of econometrics, it is highly suspect that X_i can be treated as "fixed in repeated samples" given that econometricians primarily analyze observational data, which tend to be stochastic in nature. Thus, substituting equation (20) into equation (19) makes R_i conditionally dependent upon X_i, giving rise to the alternative statistical generating mechanism:

$$R_i = E(R_i \mid X_i = x_i) + u_i, \tag{21}$$

where $E(R_i \mid X_i = x_i) = p_i = F[h(x_i;q)]$ and $u_i \sim \text{bin}(1,0)$ with variance $p_i(1-p_i)$ (Fahrmeir and Tutz, 1994; Spanos, 1999). Equation (21) is derived from a conditional Bernoulli distribution. To be able to interpret equation (21) as a proper regression function, it is necessary that the conditional Bernoulli distribution underlying equation (21) be derived from a proper joint density function of R_i and X_i (Spanos, 1999).

Following Arnold and Press (1989), the joint density function $f(R_i, X_i)$ exists if and only if:

$$f(R_i, X_i) = f_1(R_i \mid X_i) \cdot f_4(X_i) = f_2(X_i \mid R_i) \cdot f_3(R_i), \tag{22}$$

where $f(R_i \mid X_i)$ represents the conditional density of R_i given X_i, $f(X_i \mid R_i)$ represents the conditional density of X_i given R_i, $f_3(R_i)$ represents the marginal density of R_i, and $f_4(X_i)$ represents the marginal density of X_i. Thus, any specification of equation (21) requires the compatibility between the two conditional distributions $f(R_i \mid X_i)$ and $f(X_i \mid R_i)$. To see what types of restrictions this places on the functional forms that equation (21) can take, the traditional dichotomous choice logit model is examined.

Consider the following relationship using equation (22):

[19] The expectation is still taken with respect to the marginal distribution of $R_{i.}$

$$\frac{f_1(R_i = 1 \mid X_i)\cdot f_4(X_i)}{f_1(R_i = 0 \mid X_i)\cdot f_4(X_i)} = \frac{f_2(X_i \mid R_i = 1)\cdot p_1}{f_2(X_i \mid R_i = 0)\cdot p_0}, \tag{23}$$

where $p_j = f_3(R_i = j) = \mathbf{P}(R_i = j)$ for $j = 0,1$. Substituting in $f_1(R_i = j \mid X_i) = (p_i)^j(1-p_i)^{1-j}$ for $j = 0,1$ and taking the natural log of both sides of equation (23) gives:

$$\ln\left(\frac{f_2(X_i \mid R_i = 1)}{f_2(X_i \mid R_i = 0)}\right) = \ln\left(\frac{p_i}{1-p_i}\right) - \ln\left(\frac{p_1}{p_0}\right) \tag{24}$$

(Kay and Little, 1987). Kay and Little (1987) show that if:

$$\ln\left(\frac{f_2(X_i \mid R_i = 1)}{f_2(X_i \mid R_i = 0)}\right) = a_0 + a'g(x_i), \tag{25}$$

then:

$$\ln\left(\frac{p_i}{1-p_i}\right) = b_0 + b'g(x_i), \tag{26}$$

where $b_0 = a_0 + \ln\left(\frac{p_1}{p_0}\right)$, $b = a$, and $g(x_i)$ is a vector of suitable transformations of the vector x_i. Solving equation (26) for p_i gives rise to the conditional mean in equation (21) for the standard logit model:

$$E(R_i \mid X_i = x_i) = [1 + \exp(-b_0 - b'g(x_i))]^{-1}. \tag{27}$$

Kay and Little (1987) provide the transformations $g(x_i)$ that are required for the members of the exponential family of distributions to ensure that the logistic model given by equation (27) can be derived from a proper joint distribution.

For example, when the conditional distribution of X_i given $R_i = j$ is multivariate normal with heterogeneous covariance matrix dependent upon *j*, the index function

$h(x_i;q)$ is a quadratic function of the vector x_i. On the other hand, when the covariance matrix is homogeneous, the index function is linear in x_i (Kay and Little, 1987). Kay and Little (1987) state that "in cases other than multivariate normality, however, little can be said since there are few other multivariate distributions which could act as appropriate models (p. 498);" and that provide linear index functions.

The above example illustrates the rigid conditions that must be satisfied in order for the index function, $h(x_i;q)$ to be linear (or quadratic) in x_i. Kay and Little (1987) provide some other conditions under which one might obtain a linear or quadratic index function in x_i. Two of these conditions include: (i) independent explanatory variables conditional on R_i with conditional distributions in the exponential family and (ii) dependent explanatory variables conditional on R_i with a binomial or normal conditional multivariate distribution function. In light of the above observations, many of the logistic models used in the literature, and constructed based on the theory in section 2.1 are questionable statistically. Furthermore, many applied studies provide no indication that the functional form assumptions concerning the index and transformation functions are statistically valid (Arnold, Castillo and Sarabia, 1999).[20]

Similar derivations for the probit model have not been conducted, in part due to the fact that the standard normal cumulative density function cannot be expressed in terms of a finite number of additions, multiplications, root extractions or subtractions (Weisstein, 1999). This result does not rule out the existence of $f(R_i, X_i)$for the probit model, but increases the complexity of deriving results analogous to those presented by

[20] Many of the functional form tests considered in the literature compare alternative models, such as the binary logit to the binary probit model, making them model selection tests not misspecfication tests. It could be the case that the alternative model considered is misspecified as well.

Kay and Little (1987). Arnold, Castillo and Sarabia (1999) provide a number of algorithmic approaches to verify the existence of a proper joint distribution by examining the compatibility of $f_1(R_i \mid X_i = x_i)$ and $f_2(X_i \mid R_i = j)$ for $j = 0,1$ to verify that equation (22) holds. To the author's knowledge no one has yet attempted to verify the existence of a proper joint distribution from which the probit model can be derived.

The logit and probit formulations of the dichotomous choice CVM are chosen primarily out of tradition and for ease of computation, but the logistic and standard normal cdfs are not the only functional forms that meet the requirements of a transformation function. A number of other potentially valid functional forms have appeared in the literature on dichotomous choice models. If $F(.)$ is chosen to be the uniform cdf and $h(x_i;q)$ is linear in x_i, then equation (21) gives rise to the linear probability model. This model allows the use of ordinary least squares, but requires that the domain of the conditional mean be restricted to the interval $[0,1]$, which tends to be difficult in practice. The Cauchy cumulative density function gives rise to the arctangent model, where

$$E(R_i \mid X_i = x_i) = F[h(x_i;q)] = \frac{1}{2} + \frac{1}{p}\tan^{-1}(h(x_i;q)) \qquad (28)$$

(McFadden, 1984). Other specifications used in bioassay include the Wilson-Worcester sigmoid:

$$E(R_i \mid X_i = x_i) = F[h(x_i;q)] = \frac{1}{2}\left\{1 + \frac{h(x_i;q)}{\sqrt{1 + h(x_i;q)^2}}\right\}, \qquad (29)$$

and the angle sigmoid:

$$E(R_i \mid X_i = x_i) = F[h(x_i;q)] = \sin^2(h(x_i;q)) \qquad (30)$$

(Finney, 1978). All of these models provide potential alternatives to the logit and probit specifications discussed above, provided that the conditional regression function of such a statistical model can be derived from a proper joint distribution.

Kay and Little (1987) emphasize the dependency of the index function on the choice of the transformation function. The choice of which functional form to use for the index and transformation functions concerns the parameterization of contemporaneous dependence between R_i and X_i (Spanos, 1999). Given that researchers have the ability to vary the functional form of $h(x_i;q)$, Amemiya (1981) states that the importance of having $F(.)$ correctly specified is lessened. If one can approximate $h(x_i;q)$ for a given choice of $F(.)$, then the particular choice of $F(.)$ need only satisfy the conditions of a transformation function. As compelling as this argument may be, it may be the case that a particular choice of $F(.)$ may not give rise to a proper statistical model, in the sense that the conditional Bernoulli distribution based upon $F(.)$ cannot be derived from a proper joint density function. A choice of $F(.)$that does allow one to approximate $h(x_i;q)$ is the logistic cdf.[21] The reverse approach, i.e. fixing $h(x_i;q)$ and approximating $F(.)$, is another possibility, but again it may be the case that a particular choice of $h(x_i;q)$ may not work for any choice of $F(.)$. Finally, one may choose to approximate both $h(x_i;q)$ and $F(.)$ simultaneously. Thus, a way of weakening the functional form assumptions based on these above approaches is to employ semi-nonparametric (SNP) estimation methods. In this sense, the researcher is choosing to weaken the underlying probabilistic

[21] The use of the logistic cdf is supported by results in Kay and Little (1987). Furthermore, in cases where the explanatory variables are highly correlated with each other and there exist a significant number of explanatory variables, the approach presented by Kay and Little may be intractable, thus requiring the use of an approximation to $h(x_i;q)$.

assumptions of the model to avoid potential model misspecification and inconsistencies, at the cost of obtaining statistical inferences that tend to be less precise, in the sense that a semi-nonparametric estimator is likely to be insensitive to some of the systematic information in the data (Spanos, 1999).

Semi-nonparametric methods are a semi-distribution-free approach that avoids restricting $F(.)$ and/or $h(x_i;q)$ in equation (21) by trying to estimate the compound function $F(h(x_i;q))$ (Cooper, 2002). Following Cooper (2002), the modeler can replace $F(.)$, $h(x_i;q)$ or both with a flexible SNP functional form. A common replacement for $h(x_i;q)$ is a version of the Fourier functional form for nonperiodic functions (see Gallant, 1982). If the modeler is interested in relaxing the assumption concerning the choice of cdf, then Gallant and Nychka (1987) construct a SNP functional form for $F(.)$ that is the product of a Hermite polynomial expansion and density with moment generating function. Using this SNP distribution, the modeler can approximate any smooth density function, and in turn smooth cdf, arbitrarily closely. In addition, the researcher can combine the SNP functional forms mentioned above for $F(.)$ and $h(x_i;q)$ as well (see Heng and Randall, 1997; Cooper, 2002). Thus, in conjunction with dichotomous choice models the term semi-nonparametric refers to the fact that the researcher is relaxing the functional form assumptions (and therefore distributional assumption) of the statistical model, by only requiring that equation (21) represent the conditional mean of a conditional Bernoulli distribution from the family of such distributions, instead of a specific element of the family (such as the logit or probit specification with linear index function).

Gabler, Laisney and Lechner (1993) perform a Monte Carlo experiment comparing dichotomous choice probit and semi-nonparametric (SNP) models. They find that when the probit specification is incorrect, the SNP specification can reduce, at some computational cost, the bias associated with an incorrect transformation functional form assumption, i.e. the wrong choice of a cdf. Horrowitz (1993) finds similar results for the case of the dichotomous choice logit specification. Using a modified approach to that proposed by Gallant and Nychka (1987), Gabler, Laisney and Lechner (1993) find that the SNP specification performs substantially better than the probit specification, especially when $F(.)$ is asymmetric. These results suggest that a semi-nonparametric estimation method may help in avoiding potential model misspecifications due to an incorrect functional form.

An alternative semi-nonparametric approach, that has yet to be widely applied in the field of natural resource and environmental economics, is feed-forward back-propagation artificial neural networks. Artificial neural networks provide another potentially powerful SNP tool for modeling dichotomous choice CVMs. A detailed mathematical and statistical presentation of this approach is presented in the next section.

3. Feed-Forward Backpropagation Artificial Neural Networks

An innovative semi-nonparametric technique used for various classification problems in a number of fields, which includes dichotomous choice models, is the feed-forward backpropagation artificial neural network (FFBANN). This section of the paper presents recent applications in related fields, as well as, pertinent mathematical and statistical theory pertaining to FFBANNs.

3.1 Recent Applications

FFBANNs have been used and compared to traditional statistical methods (primarily regression) in a number of fields of study. These fields include: agriculture, economics, finance, forestry, hospitality and tourism management, management science, marketing, medicine, natural resources, political science, and the list goes on. In a number of these studies, FFBANNs were found to be superior to traditional statistical techniques for classification problems on the basis of out-of-sample predictive accuracy.

West, Brockett and Golden (1997) compared artificial neural networks to traditional linear-additive statistical methods (e.g. discriminant analysis and logistic regression) for predicting consumer choice. This comparison was based on a numerical simulation and application to empirical data, in order to compare the within-sample and out-of-sample predictive accuracy of discriminant analysis, FFBANNs and logistic regression. West, Brockett and Golden (1997) found that

> “*practically speaking, on the average, the “best trained” neural network always out performed both discriminant analysis and logistic regression in terms of both within- and out-of-sample predictive accuracy for the noncompensatory decision rules. … All three modeling procedures performed exceptionally well in capturing* [a] *compensatory decision rule. Thus, you cannot go wrong by using neural networks in linear settings and can gain substantially in nonlinear (or unknown) settings* (p. 382).”[22]

Furthermore, the authors claim that their results suggest that FFBANNs did not produce as much within-sample over-fitting of the training data when compared to the traditional statistical models analyzed (West, Brockett and Golden, 1997).[23] A similar study by Dasgupta, Dispensa and Ghose (1994) compared the same models’ abilities to segment consumers based on financial risk aversion and if the consumer would purchase a risky

[22] The prose in square brackets was added by the author for further clarification.

[23] Given that training of a FFBANN was terminated based upon the performance of a validation data set, it could be the case that the neural networks were under-fitted, but this is desirable if the researcher is interested in out-of-sample prediction performance or generalizable statistical inferences.

financial product. The FFBANNs were better able to predict out-of-sample than the logit model and discriminant analysis based on the number of out-of-sample predictions correctly classified, but the FFBANNs superiority was not found to be statistically significant based on a chi-square test for equality of proportions. The results of this study are particularly interesting as the authors noted that “if [their] data set is representative, it would appear that neural network models have the “potential” to perform at comparatively superior levels for this kind of domain of application (p. 243),” i.e. disaggregate level consumer survey response data.

Kastens and Featherstone (1996) used FFBANNs to predict decision makers’ responses to subjective questions concerning agricultural risk, given certain demographic and financial information of the decision makers. Their examination compared the out-of-sample predictive accuracy of ordered multinomial logit models with FFBANNs for predicting out-of-sample survey responses of Kansas farmers to risk-related questions. Kastens and Featherstone (1996) found that the ordered multinomial logit model predicted “well those response categories that are substantially represented in the data, with little allowance for a categorical response that may be of interest to a researcher (p. 414).” On the other hand, FFBANNs coupled with a rigorous out-of-sample model testing procedure, allowed for a flexible modeling procedure that could predict categorical responses in which the researcher may be interested. In the case of agricultural risk, the FFBANNs were better able to delineate between those individuals who had weak and strong risk perceptions (Kastens and Featherstone, 1996).

Zeng (1996) examined choice making and classification problems in the field of political science by comparing FFBANNs to the traditional binary logit and probit models. Using empirical data from three prior studies, Zeng found that the out-of-sample predictive accuracy was only statistically greater for one of the studies, when compared to the results obtained from using binary logit and probit models. Furthermore, conducting Monte Carlo simulations, Zeng concluded that the performance of FFBANNs deteriorates as the data becomes noisier and the underlying choice process becomes closer to being linear.

Despite the increasing number of studies showing that FFBANNs outperform traditional statistical regression models (see Arana, Delicado and Marti-Bonmati, 1999; Goss and Ramchandani, 1998; Jeng and Fesenmaier, 1996; Qi, 2001; Ranasinghe, Hua and Barathithasan, 1999; Zurada et al., 1999 for further comparisons), this is not always the case. The relative performance of artificial neural networks when compared to traditional discrete choice statistical approaches is problem and application dependent. The study by Dasgupta, Dispensa and Ghose (1994) provides evidence that the “level” of data used (e.g. aggregate versus individual survey data) can affect the performance capabilities of FFBANNs. Furthermore, Hand and Henley (1997) suggest that artificial neural networks are useful when the modeler has a poor understanding of the underlying structure of the data or choice process giving rise to the data. If the modeler however has such an understanding, modeling approaches that can utilize this information should be used instead.

Despite the shortcomings of using FFBANNs mentioned above, FFBANNs do provide a flexible semi-nonparametric modeling alternative to more traditional dichotomous choice models. Furthermore, the use of FFBANNs does not preclude making inferences about the underlying process giving rise to the data (for example see

Zeng, 1996). Before addressing such issues though, a formal presentation of FFBANNs needs to be provided. This presentation is completed in the next subsection of the paper.

3.2 Feed-Forward Back-Propagation Artificial Neural Networks

The following section formally defines and presents the topological structure of the FFBANN. In addition, training algorithims, approximation results, and various topographical issues are addressed.

3.2.1 Classifiers

Ripley (1994) states that the field of neural networks is dominated by two approaches: (i) feed-forward neural networks used for classification and (ii) recurrent networks used as associative memories. The focus of this paper is concerned with the study of neural networks for classification purposes.

In classification problems, the modeler has a set of measurements or features concerning the object(s) being classified. Following Ripley (1994) denote the vector of measurements for the i^{th} object as $\mathbf{X}_i = \left(X_{1,i},..., X_{K,i}\right)$, where $\mathbf{X}_i$ is an element of the space X, which can be thought of as a *k*-dimensional Euclidean space (but keep in mind that discrete and more complicated structures can be included). Given that we are trying to classify each object, the modeler associates with each $\mathbf{X}_i$ a class or tag $R_{j,i}$ from a set C of *J* classes. Usually the modeler's objective is to obtain a probability distribution over C, providing the probability that a particular X_i belongs to $R_{j,i}$, $j = 1,..., J$. To incorporate this additional structure consider the space Y, where Y is typically given by $[0,1]^J$ (unless the responses are ordered in which case Y might be given by **R**). A classifier is defined as a mapping:

$$f : \mathsf{X} \rightarrow \mathsf{Y} \tag{31}$$

(Ripley, 1994).

Once the structure of the classifier has been decided, the classifier must be trained (estimated or learned) using a collection of measurement and response data $\{(\mathbf{X}_i, R_{j,i}), i = 1,..., N : \mathbf{X}_i \in \mathsf{X}, R_{j,i} \in \mathsf{C}\}$. The purpose of training is either to classify future observations or to understand the relationship between the attributes (or measurements) of the objects being observed and different classes to which the objects belong (Ripley, 1994). The method used to train a classifier differs depending on the nature of the classifier, e.g. the logit model uses the method of maximum likelihood, which is a form of batch learning. The next sub-section examines one type of classifier known as feed-forward artificial neural networks.

3.2.2 Feed-Forward Artificial Neural Networks

Fausett (1994) defines an artificial neural network as "an information-processing system that has certain performance characteristics in common with biological neural networks (p. 3)." Thus, artificial neural networks can be viewed as the parallel interconnection of many simple elements known as neurons (also referred to as nodes) (West, Brokett and Golden, 1997). Information is processed by passing signals between the neurons along arcs, which are weighted according to the usefulness of the information being passed along that particular arc. As the artificial neural network is trained these weights adjust so that useful arcs (pathways) are strengthened, until the neural network learns to recognize the patterns in the data used to train the neural network. The objective is to have the artificial neural network learn the training patterns so that it can generalize and be used to classify new patterns (Fausett, 1994; West, Brokett and Golden, 1997).

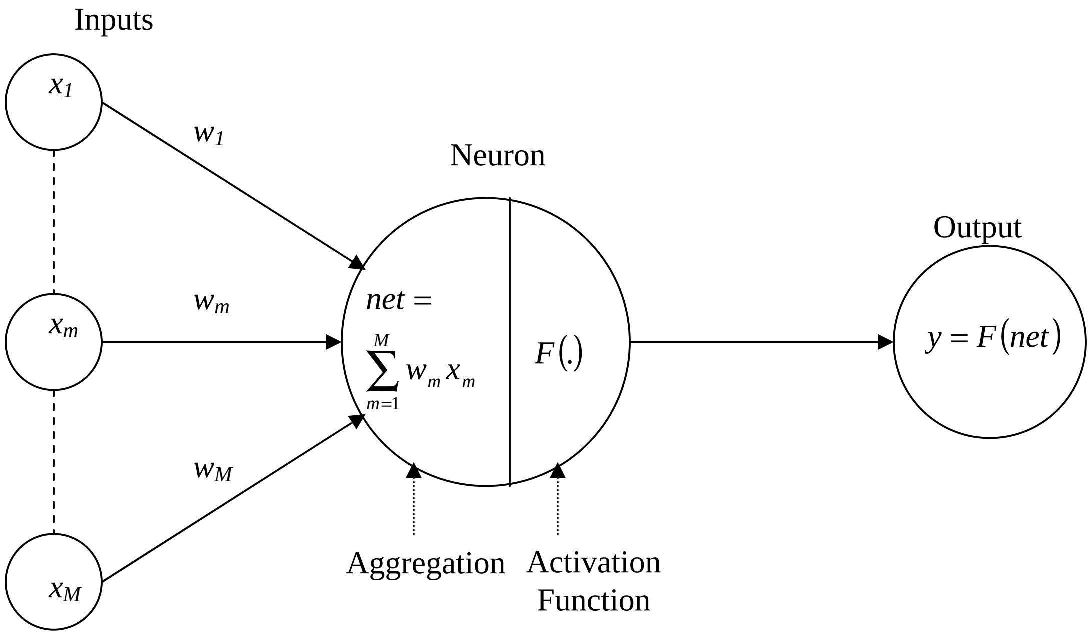
Figure 1 provides a pictorial representation of a single neuron (or node) in the hidden or output layer of an artificial neural network. A neuron takes individual inputs from *M* other neurons, aggregates them in order to obtain a single value, denoted in Figure 1 as *net*, and then performs a nonlinear transformation of *net* using an activation function $F(.)$ to produce an individual output y (West, Brokett and Golden, 1997).

Mathematically, a neuron can be represented by:

$$y = F\left(\sum_{m=1}^{M} w_m x_m\right) = F(net), \qquad (32)$$

where *y* is the output of the neuron, $F(.)$ is the activation function, w_m are the connection weights between neurons and x_m are the inputs (or outputs of the neurons in the previous layer of the network) (Mehorta, Mohan and Ranka, 1997). Two common choices of activation functions are the logistic or (binary sigmoid):

Figure 1: Topology of a Neuron



$$F(net) = \frac{1}{1+\exp\{-net\}}, \tag{33}$$

and the hyperbolic tangent:

$$F(net) = \frac{1-\exp\{-2 \cdot net\}}{1+\exp\{-2 \cdot net\}}. \tag{34}$$

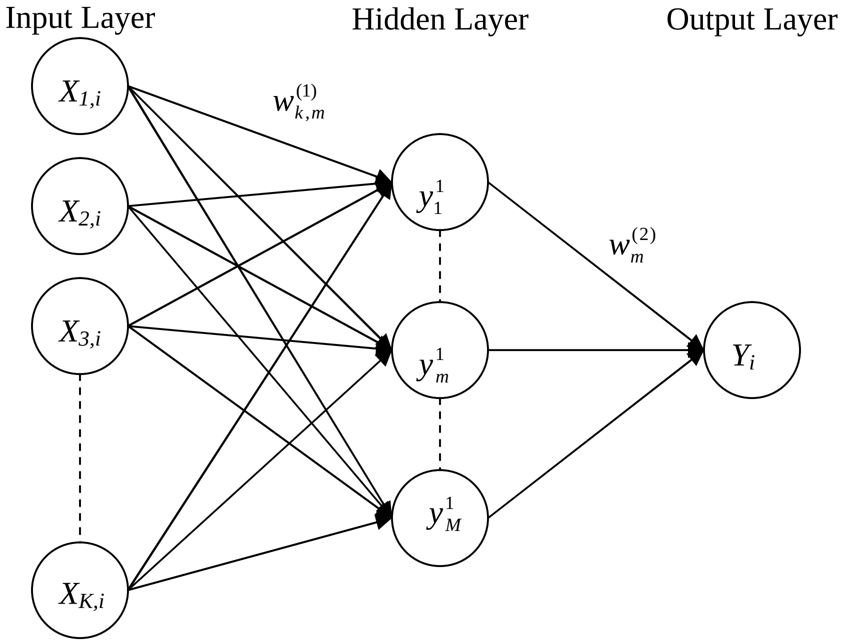
A bias (or intercept) term can be added to the summation of the inputs to the neuron as well. This term is usually treated as another weighted input designated by $x_o = 1$ so that:

$$net = w_0 + \sum_{m=1}^{M} w_m x_m \tag{35}$$

(Fausett, 1994).

The topological structure of a neural network is usually referred to as the net architecture. This architecture is arrayed in a number of different layers. At a minimum there exists an input and output layer, with input and output neurons respectively in each layer. The output of the neurons in the input layer is the input itself (i.e. $F(.)$ is the identity function). For the purpose of approximating highly nonlinear functions, hidden layers or layers with neurons (nodes) between the input and output layers can be added. Figure 2 illustrates the topology of a single-hidden layer feed-forward artificial neural network. The adjective *feed-forward* indicates that input signals travel forward successively from the input layer to the output layer of the network. In a single-hidden layer feed-forward neural network, a pattern $\mathbf{X}_i = \{X_{1,i},..., X_{K,i}\}$ is introduced to the input (or zeroth) layer at which point each neuron in the input layer sends a signal, $w_{k,m}^{(1)} X_{k,i}$, to each neuron in the hidden layer, where *k* designates the input neuron firing the signal, *m* designates the neuron receiving the signal in the hidden layer, and (l) designates that

Figure 2: Net Architecture of a Single-Hidden Layer Feed-Forward Neural Network



these are the connection weights in the $(l-1)^{th}$ to the l^{th} layer of the network. At each neuron in the hidden layer, the input signals are aggregated (net_m) and then transformed using an activation function to obtain the outputs $y_m^1 = F_1(net_m) = F_1\left(\sum_{k=1}^{K} w_{k,m}^{(1)} X_{k,i}\right)$, $m = 1,..,M$. Then each neuron in the hidden layer sends a signal, $w_m^{(2)} y_m^1$, to the output layer, where the weighted sum of the outputs, $net = \sum_{m=1}^{M} w_m^{(2)} y_m^1$ is transformed using another appropriate activation function, producing the output $Y_i = F_2(net)$. Assuming a bias term is included in the formulation, the output (or net architecture) of the artificial neural network presented in Figure 2 can be represented mathematically as:

$$Y_i = F_2\left(w_0^{(2)} + \sum_{m=1}^{M} w_m^{(2)} \cdot F_1\left(w_{0,m}^{(1)} + \sum_{k=1}^{K} w_{k,m}^{(1)} \cdot X_{k,i} \right)\right), \qquad (36)$$

where $F_l, l = 1,2$ designates the activation function in the l^{th} layer of the network (Mehrotra, Mohan and Ranka, 1997; West, Brockett and Golden, 1997).

The representation of a single-hidden layer neural network can be generalized to include feed-forward artificial neural networks with multiple hidden layers. At the l^{th} (hidden) layer of the neural network, the (hidden) neurons receive signals, $w^{(l)}_{m_{l-1},m_l} y^{l-1}_{m_{l-1}}$ from the $(l-1)^{th}$ layer, where m_l represents the m^{th} neuron in the l^{th} layer. In the input layer, $y^0_{m_0} = X_{k,i}$ and in the output layer $y^L_{m_L} = Y_i$, where 0 designates the input layer and L designates the output layer (i.e. $l = 0,1,2,...,L$).[24] Then the inputs at each neuron in the l^{th} layer are aggregated and transformed using an appropriate activation function, producing $y^l_{m_l} = F_l\left(\sum_{m_{l-1}=1}^{M_{l-1}} w^{(l)}_{m_{l-1},m_l} y^{l-1}_{m_{l-1}}\right)$, where M_{l-1} is the number of neurons in the $(l-1)^{th}$ layer. (The weights connecting the $(L-1)^{th}$ hidden layer and the output layer are denoted $w^{(L-1)}_{m_{L-1}}$ when there is one neuron in the output layer.) Thus, a two-hidden-layer network with one output and bias can be given by the following mathematical representation:

$$Y_i = F_3\left(w^{(3)}_0 + \sum_{m_2=1}^{M_2} w^{(3)}_{m_2} \cdot F_2\left(w^{(2)}_{0,m_2} + \sum_{m_1=1}^{M_1} w^{(2)}_{m_1,m_2} \cdot F_1\left(w^{(1)}_{0,m_1} + \sum_{k=1}^{K} w^{(1)}_{k,m_1} \cdot X_{k,i}\right)\right)\right). \quad (37)$$

The motivation behind using the artificial neural networks for pattern classification is that neural networks can be used to approximate highly nonlinear functions. Following Fausett (1994), the Kolmogorov Mapping Neural Network

[24] The output layer can have multiple outputs, in which case $y^L_{m_L} = Y_{m_L,i}$.

Existence Theorem states that any continuous function of *K* variables defined on $[0,1]^K$, where $K \geq 2$, can be represented in the form:

$$f(x_1,\ldots,x_K) = \sum_{m=1}^{2K+1} c_m \left(\sum_{k=1}^{K} y_{k,m}(x_k) \right), \tag{38}$$

where c_m and $y_{k,m}$ are continuous functions of one variable and $y_{k,m}$ are monotonic functions that do not depend on f. In essence, Kolmogorov's theorem tells us that we can approximate any continuous function using univariate functions. The theorem provides the basis for the existence of a two-hidden layer feed-forward artificial neural network that can approximate the true underlying process giving rise to the data, if all the inputs can be scaled between $[0,1]$. The drawback here is that, since the above theorem is an existence theorem it provides no guidelines about the choices of c_m or $y_{k,m}$ (Mehrotra, Mohan and Ranka, 1997). Recall, the goal here is to be able to approximate $E(R_i \mid X_i = x_i)$.

White, Hornik and Stinchcombe (1992) show that any single-hidden-layer feed-forward artificial neural network with a single real-valued node with linear activation function in the output layer (i.e. a function $f : \mathbf{R}^K \to \mathbf{R}$) can approximate any continuous function uniformly on a compact set. This result holds regardless of the choice of activation function, as long as the activation function is a squashing function (i.e. the function is non-decreasing, $\lim_{l\to\infty} F(l) = 1$ and $\lim_{l\to-\infty} F(l) = 0$) and regardless of the dimension of X.

A theorem by Lusin states that if f is a measurable function and X is compact, then f can be closely approximated by continuous functions except on a set of arbitrarily small measure (Fine, 1999). Thus, the results by White, Hornik and Stichcombe (1992) apply to classifiers as well. Furthermore, as long as the activation function is not almost everywhere a polynomial (of which the logistic and hyperbolic tangent activation functions are not), a single-hidden layer feed-forward neural network can approximate any square integrable classifier, which includes $E(R_i \mid X_i = x_i)$ (see section 3.3.2), since the set of all possible single-hidden layer feed-forward networks with a real-valued output node with linear activation function is dense in $L_2(\mathrm{X}, m)$ (the space of all real-valued square integrable functions), with respect to the L_2 metric, where m is a suitable measure defined on L_2 (Fine, 1999).[25] Ripley (1994) notes that the above results apply to neural networks with logistic activation functions in the output layer as long as the outputs produced by the artificial neural network are bounded away from 0 and 1.

These approximation results are primarily illustrative of the approximation properties of artificial neural networks, but tend to be non-constructive in the sense that the theorems provide very broad and insubstantial guidelines for specifying the net architecture of a neural network. The modeler is left with a number of design decisions, such as the number of hidden layers, number of neurons (or nodes) in each hidden layer, and type of activation function (meeting the guidelines in the above theorems) in each layer of the network. These decisions can have a significant affect on the approximation capabilities of the neural network, and are explored in more detail in section 4.

[25] One such measure would be a probability measure that is dominated by (or is absolutely continuous with respect to) a $s-$finite Lebesque or counting measure defined on the Borel field over R^K (Billingsley, 1995).

The next subsection addresses the question of how to estimate or train an artificial neural network, i.e. how does the modeler determine the values of the connection weights in the network? Given that the above approximation theorems are not very helpful in designing an artificial neural network, modelers must specify a net architecture and then use data to train the network by adjusting weights via the minimization of some fitting criterion (Ripley, 1994).

3.2.3 Training Feed-Forward Artificial Neural Networks

For a classifier to effectively classify input data or patterns, the classifier must be trained (or estimated) using a training set of input and output (target) data, i.e. $\{(\mathbf{X}_i, R_{j,i}), i = 1,..., N : \mathbf{X}_i \in \mathcal{X}, R_{j,i} \in \mathcal{C}\}$, to construct a mapping between various input patterns and specified output vectors. During training, the connection weights of the neural network are adjusted according to a learning rule in order to recognize the relationships between the input and output vectors being introduced to the neural network. In feed-forward artificial neural networks, this type of training is sometimes referred to as supervised learning, since the targets or output vectors are given to the network during training (Fausett, 1994).

Given that the connection weights of a feed-forward artificial neural network are adjusted in order to approximate the true underlying functional relationship between the input and output data, the objective of training can be seen as trying to minimize the errors between the output targets given to the neural network during training and the outputs produced by the neural network. In order to achieve this objective, the modeler needs to choose a fitting or error criterion, which will be used as an objective function for the minimization problem just described. For optimization purposes, a desirable property of the fitting criterion is that it be a second-order differentiable function. A common choice that fits this criterion is the mean square error (*MSE*) fitting criterion:

$$E(.)=\frac{1}{N}\sum_{i=1}^{N}\left\|R_i-Y_i\right\|^2 \ , \qquad (39)$$

where R_i is the output target vector and Y_i is the output vector produced by the neural network (Mehrotra, Mohan and Ranka, 1997).[26] Other error criterions suggested, include the L_p norms for $p \geq 1$ and the Kullback-Leibler or cross-entropy criterion:

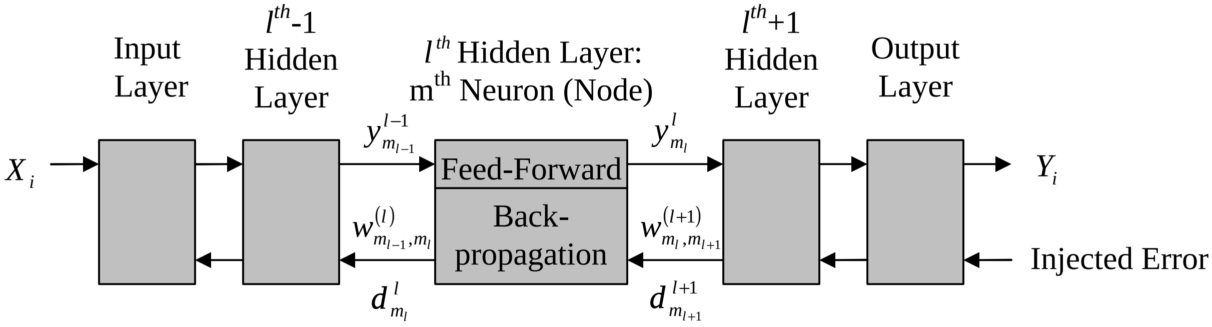
$$E(.)=\sum_{i=1}^{N}\sum_{l=1}^{L}\left[R_{l,i}\cdot\log\left(\frac{R_{l,i}}{Y_{l,i}}\right)+\left(1-R_{l,i}\right)\cdot\log\left(\frac{1-R_{l,i}}{1-Y_{i,j}}\right)\right], \qquad (40)$$

where $R_{l,i}$ is the l^{th} component of the i^{th} output target vector and $Y_{l,i}$ is the l^{th} component of the i^{th} output vector produced by the neural network. The latter critierion provides an information theoretic approach to training, as well as providing an additional basis for a statistical interpretation of these networks as being estimators of the conditional mean from a conditional Bernoulli distribution (Principe, Euliano and Lefebvre, 2000; Ripley, 1994).

Finding the values of the connection weights that minimize the fitting criterion $E(.)$ is an unconstrained optimization problem. The differentiability of the error criterion allows the weights to be updated and calculated recursively during training using the chain rule of differentiation. This procedure is known as backpropagation, hence the use of the acronym FFBANN, which is used from this point on in the paper (Ripley, 1994). Figure 3 illustrates the backpropagation procedure for a FFBANN with up to *L* hidden layers. A net input (X_i) is fed through the neural network producing a net

[26] Equation (39) is equivalent to the L_2 norm.

Figure 3: The Backpropagation Procedure



Source: Principe, Euilaino and Lefebvre, 2000

output (Y_i). The error between the net output and the output target is then computed using the error criterion, where the error is injected recursively into the network to update the weights. The connection weights are updated using a gradient search method, where the update for a given connection weight, $w_{m_{l-1},m_l}^{(l)}$, is a function of the output, $y_{m_{l-1}}^{l-1}$ from the m^{th} node in the $(l-1)^{th}$ layer and the activation error, $d_{m_l}^{l}$ of the m^{th} node in the l^{th} layer of the network. The activation error represents the square error derivative of the fitting criterion associated with the m^{th} node in the l^{th} layer of the network (Principe, Euiliano and Lefebvre, 2000; West, Brockett and Golden, 1997). The procedure illustrated in Figure 3 allows the modeler to easily simulate neural networks by decoupling the computations needed to train a FFBANN. Such a structure allows general procedures to be created for a large number of networks, that only require the modeler to specify the activation functions at each layer, fitting criterion and its derivatives, respectively (Principe, Euliano and Lefebvre, 2000).

Each time all the input data patterns have been presented to the network and the weights updated, the network has been said to complete one epoch (or iteration). Weight updates can be done in batches or online. If done in batches, each input pattern is introduced and the corresponding error and weight update is calculated and saved. The weight updates calculated after all the input patterns have been introduced are then summed and applied at the end of each epoch. In online training, the weights are updated immediately after an input pattern has been introduced to the network, thus the weights will be updated *N* times each epoch (Fine, 1999).

There are a number of iterative algorithms that can be used to minimize the fitting criterion and train FFBANNs. These include steepest descent algorithms, conjugate gradient algorithms, quasi-Newton algorithms and Levenberg-Marquardt algorithms (also known as trust region methods) (see Fine, 1999 for an explanation of conjugate gradient and Levenberg Marquardt algorithms). Two specific algorithms commonly recommended and used in the literature are presented here: (i) the steepest descent algorithm and (ii) the Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-Newton algorithm. To present these two algorithms, a single-hidden layer FFBANN with a single node in the output layer is assumed to be trained using the *MSE* fitting criterion:

$$E = \frac{1}{N}\sum_{i=1}^{N}\left(R_i - Y_i\right)^2 = \frac{1}{N}\sum_{i=1}^{N} e_i^{\,2} \,, \tag{41}$$

where e_i is the error as a result of feeding the i^{th} input pattern through the network.

3.2.3.1 Steepest Descent Algorithm

The method of steepest descent for minimizing a function amounts to performing line searches along the directions of steepest descent or the negative gradient of the fitting criterion (Bazaraa, Sherali and Shetty, 1993). For online training, this algorithm is based upon the following weight update scheme:

$$\Delta w_{m_{l-1},m_l}^{(l)}\left(i\right) = -h \cdot \frac{\partial e_i^{\,2}}{\partial w_{m_{l-1},m_l}^{(l)}} + a \cdot \Delta w_{m_{l-1},m_l}^{(l)}\left(i-1\right), \tag{42}$$

where h is the learning rate or step size, and a is the rate of momentum. When $a = 0$, the update scheme given by equation (42) is referred to as the delta rule. Assuming training is done online and using equations (36) and (42), by the chain rule:

$$\frac{\partial e_i^2}{\partial w_{k,m}^{(1)}} = -2 \cdot e_i \cdot F_2'(net) \cdot w_m^{(2)} \cdot F_1'(net_m) \cdot X_{k,i}\,, \tag{43}$$

$$\frac{\partial e_i^2}{\partial w_{0,m}^{(1)}} = -2 \cdot e_i \cdot F_2'(net) \cdot w_m^{(2)} \cdot F_1'(net_m)\,, \tag{44}$$

$$\frac{\partial e_i^2}{\partial w_m^{(2)}} = -2 \cdot e_i \cdot F_2'(net) \cdot y_m, \text{ and} \tag{45}$$

$$\frac{\partial e_i^2}{\partial w_0^{(2)}} = -2 \cdot e_i \cdot F_2'(net)\,, \tag{46}$$

where net_m and y_m are given by equation (32) and $net = \sum_{m=1}^{M} w_m^{(2)} y_m$. If batch training is used instead, then $\frac{\partial e_i^2}{\partial w_{m_{l-1},m_l}^{(l)}}$ in equation (42) is replaced with $\frac{\partial E}{\partial w_{m_{l-1},m_l}^{(l)}} = \sum_{i=1}^{N} \frac{\partial e_i^2}{\partial w_{m_{l-1},m_l}^{(l)}}$ (Mehrotra, Mohan and Ranka, 1997) (see Fine, 1999 for MATLAB code for the steepest descent algorithm).

The learning rate, h, represents the rate at which the connection weights are updated during the training of the neural network. In an algorithmic sense, h is the step size taken along the direction of the negative gradient of the fitting criterion with respect to the connection weights for a given iteration in the connection weight space. The learning rate has been traditionally set by the modeler. If the learning rate is set too large, the algorithm can overshoot the minimum and diverge. On the other hand if the learning rate is too small, then the algorithm may converge, but only after a large number of epochs has been performed (Fine, 1999). To avoid such undesirable behavior, it is suggested that the learning rate be made large at the beginning of the training process and then be gradually decreased as training progresses, in order to obtain accurate final connection weight values. This process of controlling the learning rate is known as annealing (see Principe, Euliano and Lefebvre, 2000 for annealing methods). Another method is to conduct an approximate line search along the direction of the negative gradient, such as a golden-section or quadratic interpolation line search, in which the learning rate is determined directly by the algorithm itself (see Bazaraa, Sherali and Shetty, 1993; Fine, 1999).

The momentum rate, a , is used to speed up and stabilize the convergence of the algorithm. In a sense adding momentum to the update scheme is like applying exponential smoothing to the delta rule. The momentum rate can be set (and adjusted) by the modeler (usually between 0.5 and 0.9) or determined by the algorithm along with h using some type of grid search procedure (Fine, 1999; Principe, Euiliano and Lefebvre, 2000).

A disadvantage of using the steepest descent algorithm is that the algorithm tends to experience a zigzagging phenomenon as the algorithm approaches a stationary point or (local) optima. This phenomena results in the algorithm taking small steps in almost orthogonal directions toward the minimum, resulting in poor convergence. There are methods to counteract this behavior, such as Armijo’s inexact line search, that can be used to deflect the gradient toward the stationary point (Bazaraa, Sherali and Shetty, 1993). Another option is to use a more advanced technique, such as the BFGS quasi-Newton algorithm.

3.2.3.2 Broyden–Fletcher-Goldfarb-Shanno (BFGS) Quasi-Newton Algorithm

If the fitting criterion was quadratic, then the modeler could find the weight vector that minimizes the fitting criterion in one step using Newton's method, i.e. $w^* = w_0 - H^{-1}\nabla E$, where w_0 is the initial weight vector, H^{-1} is the inverse of the Hessian of the fitting criterion with respect to the connection weights, and ∇E is the gradient of the fitting criterion with respect to the connection weights. Since the surface of the fitting criterion does not tend to be globally quadratic and the evaluation of the true Hessian is computationally expensive, Netwon's method is not practical. Thus, quasi-Netwon methods are adopted that approximate H^{-1} using an update scheme to track the inverse of the Hessian (Fine, 1999).

Assuming the modeler trains online, the weight update scheme for the BFGS quasi-Newton algorithm is given by:

$$\Delta w(i) = -h \cdot \mathrm{M}_i \cdot \nabla e_i^2, \tag{47}$$

where M_i is the approximation of the inverse of the Hessian of $e_i = (R_i - Y_i)$, ∇e_i^2 is the gradient of $e_i^2 = (R_i - Y_i)'(R_i - Y_i)$ with respect to the weight vector and i refers to the i^{th} pattern being introduced. Once a new weight vector is found using equation (47), M_i is updated using the following update procedure:

$$\mathrm{M}_{i+1} = \mathrm{M}_i + \frac{p_i p_i'}{p_i' q_i} \cdot \left[1 + \frac{q_i' \mathrm{M}_i q_i}{p_i' q_i}\right] - \frac{(\mathrm{M}_i q_i p_i' + p_i q_i' \mathrm{M}_i)}{p_i' q_i}, \tag{48}$$

where $p_i = h \cdot \mathrm{M}_i \cdot \nabla e_i^2$ and $q_i = \nabla e_{i+1}^2 - \nabla e_i^2$. It should be noted that equation (48) arises due to the relationship that $q_i = H_i p_i$ (see Fine, 1999). If batch training is performed, then M_i and ∇e_i^2 in equation (47) are replaced with $\mathrm{M} = \sum_{i=1}^{N} \mathrm{M}_i$ and $\nabla E = \sum_{i=1}^{N} \nabla e_i^2$ respectively. The update scheme for M_i is performed for each input pattern and summed to get M.

The learning rate in quasi-Newton methods is determined using an approximate line search, since the convergence properties of these types of algorithms depends on the line search procedures used (see Bazaraa, Sherali and Shetty, 1993 and Fine, 1999 for a further discussion). Thus, the determination of h is no longer an issue for the modeler. Furthermore, no momentum rate is used since the information about the gradient from the previous iteration is captured by M_i. This algorithm is based upon a second-order Taylor Series approximation of the fitting criterion, which incorporates changes in the gradient of the fitting criterion with respect to the weight vector via the approximation of the Hessian matrix.

Fletcher (1987) notes that the BFGS quasi-Newton algorithm enjoys superlinear convergence for general functions, which includes non-quadratic functions. Thus, asymptotically, the BFGS quasi-Newton algorithm should converge more rapidly than the steepest descent algorithm (Fine, 1999; Fletcher, 1987).

3.2.3.3 Additional Issues Concerning Training

The successful use of both of the algorithms presented in the previous two sub-sections, as well as the others mentioned previously all depend on a number of issues that need to be decided prior to training the network. These issues include (i) initialization of the algorithm, (ii) choice of the training data set, and (iii) choice of stopping rule(s), which concerns a primary issue known as generalization.

The initialization of the algorithm requires choosing the starting values for the connection weights and preparing the training data set. The initial starting value for the weights is usually chosen by randomly generating the weight values using a uniform random number generator for a suitable interval $[a, b]$ on the real line. The choice of initial weight values can affect whether the algorithm converges to a global or local minimum of the fitting criterion during training. Too small or too large initial weight values can result in longer training times. It is recommended that several initial vectors of starting values be used to train the neural network, and then the network with the minimum error be chosen as the best fit (Fausett, 1994; Fine, 1999). If the hyperbolic tangent activation function is used in the first hidden layer of a neural network, then the modeler can use a weight initialization procedure developed by Nguyen and Widrow. This procedure chooses weight values so that the hidden nodes in the network will be able to learn the input-output mapping more readily, thus decreasing the time it takes to train the network (see Fausett, 1999).

The data set used to train a neural network should be constructed so that it is representative of the population it is purporting to describe. This choice ensures that the training of the neural network is optimal, in the sense that the neural network will properly learn to distinguish between the different classes of objects of interest to the researcher. Furthermore, the training data should be appropriately scaled to the range $[a, b]$ of the activation functions used in the hidden layer(s) of the neural network, in order to avoid the time it takes the algorithm to scale the data to achieve the correct output scale. (Fine, 1999).[27]

The stopping rule(s) used to terminate training (the search for a global (or local) minimum) vary depending upon the objective of the modeler. Four common stopping rules used are: for epoch $j \in T$

(S1) $\|w(j+1) - w(j)\| < e$ for $e > 0$ but small,

(S2) $\|\nabla E(j)\| < e$ for $e > 0$ but small,

(S3) $E \leq E_{\min}$, a pre-specified lower bound for the training error, or

(S4) $j > MAX$, where MAX is the maximum number of epochs that the algorithm is allowed to perform.

The training rule given by (S4) is usually used in combination with (S1), (S2), or (S3), and any of the four can be used in combination with each other. The rule given by (S3) is probably the least desirable due to the fact that a reasonable lower bound for the training error can be difficult to determine. The stopping rule (S1) can be used to replace (S2), since they both imply that the gradient is close to (within a small e-neighborhood of) a local (or global) optimum, but empirically a large number of epochs are usually required for either rule to be effective (Fine, 1999).

A key issue during training is the question of how well the network performs in classifying input patterns that were not used to train the network or generalization. This issue arises due to the fear that the network will be overtrained or overfitted. Fine (1999) states that over-fitting is

[27] If the data varies by different orders of magnitude, then the training algorithm will first adjust the connection weights between the input and first hidden layer so that the magnitudes of the input variables are of the same scale.

> *"a condition in which the network overfits the training set and fails to generalize well. One explanation for the failure of generalization when overtraining occurs is that overtraining renders accessible the more complex members of the excessively flexible family of neural networks being deployed. Hence, we may end up fitting the data with a more complex function than the true relationship (e.g. a higher degree polynomial can fit the same points as a lower degree polynomial). A more common explanation observes that the target variables often contain noise as well as signal – there is usually only a stochastic relationship between feature vectors $\underline{x}$ and target t, with repetitions of the same feature vector often corresponding to different target values. Fitting too closely to the training set means fitting to the noise as well and thereby doing less well on new inputs that will have noise independent of that found in the training set.* (p. 155)."

Thus, underfitting can be desirable in the sense that the model is capturing only the systematic information in the observed data (e.g. $E(R_i \mid X_i = x_i)$) and not nonsystematic noise.

To avoid overfitting a network, a validation data set that is independent of the training data set is constructed or set aside from the original sample (Principe, Euliano and Lefebvre, 2000). The validation data set is then used in conjunction with a stopping rule based on an out-of-sample performance criterion. Two such criteria proposed in the literature are:

(S5) $E_{Val}(j) \le E_{val}(j+1) \le ... \le E_{val}(j+v)$ for some $v = 1,2,....$ chosen by the modeler, where $E_{val}(j)$ is the value of the fitting criterion at the j^{th} epoch evaluated using the validation data set; or

(S6) $PR(j) \le PR(j+1) \le ... \le PR(j+v)$ for some $v = 1,2,....$ chosen by the modeler, where $PR = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(R_i - Y_i = 0)$ and $\mathbf{1}(R_i - Y_i = 0)$ is an indicator function that equals 1 when the output of the neural network is equal to the output target and 0 otherwise.

Both rules terminate training after a particular criterion fails to be improved upon after v epochs. Rule (S5) states that training should be terminated when the mean square error of the fitting criterion evaluated using the validation data set does not decrease after v epochs, while rule (S6) states that training should be terminated when the number of input patterns correctly specified begins to decrease after v epochs (Fine, 1999; Kastens and Featherstone, 1996). Using stopping rules (S5) and (S6) will result in estimating the connection weights less precisely and thereby not iterating to a convergent solution for the training set. Such a situation is desirable if the modeler wants to achieve better generalization and avoid overfitting (Kastens and Featherstone, 1996).

Principe, Euliano and Lefebvre (2000) present another approach to generalization known as regularization that does not require the use of a validation data set, but instead introduces a penalty into the fitting criterion. Based on the work by Tikhonov and Arsenin (1977), regularization tries to control the complexity of the neural network being constructed by introducing a priori information to help stabilize the problem by imposing limits on the variability of the solution. Such information is usually introduced by adding terms proportional to H, the Hessian of the fitting criterion with respect to the connection weights. One example of such a penalty is:

$$E_R = E + l \sum_{w_j \in w} w_j^2 \,, \qquad (49)$$

where l is a regularization constant chosen experimentally by the modeler (Mehrotra, Mohan and Ranka, 1997; Principe, Euliano and Lefebvre, 2000). Regularization has the effect of smoothing out the fitting criterion, so that it is easier for the training algorithm to converge to a global (or local) minimum (Demuth and Beale, 2001).

3.3 A Statistical Perspective of Neural Networks as Dichotomous Choice Models

FFBANNs provide an alternative specification to the more traditional logit and probit specifications for dichotomous choice contingent valuation models. Given that FFBANNs can be viewed as universal approximators, under certain conditions they provide a flexible functional form for modeling dichotomous choice data. The focus in this section of the paper is to expand on that point from a statistical perspective. The proceeding subsections begin this venture by examining the link between FFBANNs and the logit and probit models.

3.3.1 The Link between the Binary Logit and Probit Models and FFBANNs

The conditional Bernoulli distribution (under certain conditions) gives rise to the statistical generating mechanism (or regression function) given by equation (21) in section 2.2. The traditional logit and probit specifications are represented using equation (21) by letting $E(R_i \mid X_i = x_i) = F(a_0 + a'g(x_i))$, where a is a $(J \times 1)$ vector of coefficients, $g(x_i)$ is a $(J \times 1)$ vector of suitable transformations of the $(K \times 1)$ vector x_i, and F is the logistic cdf in case of the logit model and the standard normal cdf in the case of the probit model. If the researcher is concerned about potential functional form misspecifications of equation (21), then the modeler may wish to approximate $E(R_i \mid X_i = x_i)$. By viewing a FFBANN as an approximation (a flexible functional form) to $E(R_i \mid X_i = x_i)$, gives rise to an alternative statistical model to the traditional logit and probit specifications. For a single-hidden layer FFBANN the regression function would take the following functional form (with bias):

$$R_i = F_2\left(w_0^{(2)} + \sum_{m=1}^{M} w_m^{(2)} \cdot F_1\left(w_{0,m}^{(1)} + \sum_{k=1}^{K} w_{k,m}^{(1)} \cdot x_{k,i} \right)\right) + u_i\,, \tag{50}$$

where $u_i = \text{bin}(1{,}0)$ with variance $p_i(1-p_i)$. A FFBANN with two or more hidden layers could also be used in place of the single-hidden layer FFBANN in equation (49). In general, such a model would be represented as (with bias):

$$R_i = F_L\left(w_{0,m_L}^{(L)} + \sum_{m_{L-1}=1}^{M_{L-1}} w_{m_{L-1},m_L}^{(L)} \cdot F_{L-1}\left(w_{0,m_{L-1}}^{(L-1)} + \sum_{m_{L-2}=1}^{M_{L-2}} w_{m_{L-2},m_{L-1}}^{(L-1)} \cdot F_{L-2}\left(\cdots F_1\left(w_{0,m_1}^{(1)} + \sum_{k=1}^{K} w_{k,m_1}^{(1)} \cdot x_k\right)\cdots\right)\right)\right) + u_i, \text{ (51}$$

where $u_i = \text{bin}(1{,}0)$ with variance $p_i(1-p_i)$ and there are $l = 0,1,2,...L$ layers to the network.

By eliminating the hidden layer in equation (50), replacing the inputs with $g_j(x_i)$ for $j = 1,...,J$, where $g_j(.)$, is a suitable transformation of the elements of x_i (e.g. $x_{k,i}$ and $x_{k,i}x_{j,i}$ for $k = 1,...,K$ and $j = 1,...,K$), and making $F_2(.)$ the logistic or standard normal cdf, the traditional logit and probit models can be obtained. Furthermore, any other parametric specification using a CDF can be represented in the same manner, by replacing the activation function, $F_2(.)$, at the output node of the FFBANN with another appropriate cdf (see section 2.2). It should be noted that the traditional logit and probit specifications are not nested in equations (50) or (51).

The above discussion is enlightening since it provides the basis for viewing a FFBANN as a proper statistical model. Furthermore, such a viewing angle provides the needed basis for using a FFBANN for statistical inference, but with the caveat that any such inference will tend to be less reliable than using a properly specified parametric model. These topics are discussed further in the following sub-sections.

3.3.2 FFBANNs as a flexible functional form for $E(R_i \mid X_i = x_i)$

Recall, that the interest in using FFBANNs lies in its ability to approximate square integrable real valued measurable functions, in particular $E(R_i \mid X_i = x_i)$**.** Since

$E\left(|R|^2\right) < \infty$, R is square integrable and is a member of $L_2(\mathcal{X}, m)$, the set of all square integrable real-valued functions (with range $\mathcal{X}$) with respect to a finite nonnegative measure m, which is absolutely continuous with respect to Lebesque measure (or counting measure in the case of a discrete random variable). Assuming that $E\left(|X_k|^2\right) < \infty$ for $k = 1,...,K$, then $X = (X_1,...,X_K) \in L_2(\mathcal{X}, m)$ and is treated as a random vector or set of random variables.[28] Furthermore, it should be noted that any Borel function of the elements of X (e.g. $g_j(X)$) is a member of $L_2(\mathcal{X}, m)$ as well. Thus, by the classical projection theorem, $E(R \mid X = x)$ is also a member of $L_2(\mathcal{X}, m)$. In this sense $E(R \mid X = x)$ is the projection of R onto the subspace spanned by X (and/or the Borel functions of X) (Luenberger, 1969; Small and McLeish, 1994). This result implies that a FFBANN can be used to approximate $E(R_i \mid X_i = x_i)$ in $L_2(\mathcal{X}, m)$ space (see section 3.1). Leshno et. al. (1993) states that if m is given as above, $\mathcal{X}$ is compact, then the set of all single-hidden layer feed-forward neural networks with linear activation function in the output node is dense in $L_2(\mathcal{X}, m)$ with respect to the L_2 metric, as long as the activation functions in the hidden layer are not almost everywhere polynomials, locally bounded, and are discontinuous only on a set of measure zero (which includes sigmoidal activation functions) (see Fine, 1999 as well). Ripley (1994) noted that this can be extended to a single-hidden layer FFBANN with a logistic activation function (or

[28] First, note that if m is absolutely continuous with respect to Lebesque measure, then it implies that m has a density function by the Radon-Nikodym Theorem (Billingsley, 1995). Second, note that R_i can be represented by a vector of zeros and ones even when R_i takes on more than one value, i.e. for $j = 1,2,3,4$, $R_i = (R_{i,1}, R_{i,2}, R_{i,3}, R_{i,4})$, and when the event of interest corresponds to class 3 say, $R_{i,3} = 1$ and the rest zeros, so that $R_i = (0,0,0,1)$. Thus, one can view the range of R as being $\mathcal{Y}$, which is a subset of $\mathcal{X}$. The subscript *i* is removed to emphasize that R_i and X_i are random variables and thus measurable functions.

potentially any other appropriate cdf) in the output layer as long as the values taken by the network are bounded away from 0 and 1. The essence of this discussion is that a (single-layer) FFBANN can act as a flexible functional form for $E(R_i \mid X_i = x_i)$.

Kuan and White (1994) point out that another "issue of both theoretical and practical importance is the "degree of approximation problem": how rapidly does the approximation to an arbitrary function improve as the number of hidden units [or layers] increases (p. 10)?"[29] Barron (1991) partially answers this question by considering single-hidden layer FFBANNs with a linear activation function in the output layer and continuously differentiable sigmoid activation functions at each hidden node in the hidden layer. He states that the degree of approximation by a single-hidden layer FFBANN with the above properties, based upon the L_2 norm is $O(1/q^{1/2})$ when the function being approximated "belongs to a certain class of smooth functions satisfying a summability condition on its Fourier transform (Kuan and White (1994), p. 10)."

A guideline used in the neural network literature for constructing neural networks is that "*any learning machine should be sufficiently large to solve the problem, but not larger* (Principe, Euliano and Lefebvre, 2000; p. 208)."[30] In this sense, the neural network should be large enough to provide a good approximation, but not so large that it results in overfitting to the training data set.

Given that a FFBANN represents a flexible functional form, how should we view these FFBANN approximations to $E(R_i \mid X_i = x_i)$? First consider a single-hidden layer FFBANN with a single output node with logistic activation function. In this case the regression function given by equation (50) becomes:

[29] The prose in square brackets was added by the author for further clarification.

[30] This principle is based upon the principle of Occam's Razor.

$$R_i = \left[1 + \exp\left\{ - \left[w_0^{(2)} + \sum_{m=1}^{M} w_m^{(2)} \cdot F_1 \left(w_{0,m}^{(1)} + \sum_{k=1}^{K} w_{k,m}^{(1)} \cdot x_{k,i} \right) \right] \right\} \right] + u_i \, . \qquad (52)$$

In this case, $h(x_i ; q) = w_0^{(2)} + \sum_{m=1}^{M} w_m^{(2)} \cdot F_1 \left(w_{0,m}^{(1)} + \sum_{k=1}^{K} w_{k,m}^{(1)} \cdot x_{k,i} \right)$ is a single-hidden layer FFBANN with a single output node with linear activation function. According to White, Hornik and Stinchcombe (1992) such a network can approximate any continuous function uniformly. Thus, one could interpret equation (52) as a semi-nonparametric model. The transformation function in this case is assumed to be logistic at the outset, while the index function is approximated by using a single-hidden layer FFBANN with a single output node with linear activation function. This interpretation could be extended to any other specification of $F_2(.)$ as long as it has the properties of a cdf. The interpretation of a two-hidden-layer FFBANN could be interpreted in the same manner, but could also be seen as approximating both the transformation and index functions simultaneously (see section 2.2).

3.3.3 Choice of Fitting Criterion and Model Estimation

When training (estimating) a FFBANN the modeler has a choice of different fitting criteria. The two most commonly used are the *MSE* fitting criterion:

$$E(.) = \frac{1}{N} \sum_{i=1}^{N} \left\| R_i - Y_i \right\|^2 \, ,$$

and the Kullback-Leibler fitting criterion:

$$E(.) = \sum_{i=1}^{N} \sum_{l=1}^{L} \left[R_{l,i} \cdot \log\left(\frac{R_{l,i}}{Y_{l,i}} \right) + \left(1 - R_{l,i}\right) \cdot \log\left(\frac{1 - R_{l,i}}{1 - Y_{i,i}} \right) \right] .$$

Estimating the parameters of the FFBANN using the *MSE* fitting criterion amounts to performing the method of nonlinear least squares. If the Kullback-Leibler fitting criterion is used instead, then the parameters are essentially being estimated via the method of maximum likelihood. Thus, the choice of fitting criterion has direct implications for the statistical properties of the parameters of interest in the model, the connection weights.

3.3.3.1 *MSE* Fitting Criterion

Minimizing the *MSE* fitting criterion to estimate the parameters (connection weights) of a FFBANN, while viewing the FFBANN as an approximation to $E(R_i \mid X_i = x_i)$, is a nonlinear least squares (NLS) problem of the general form, i.e,

$$\min_{q \in \Theta} N^{-1} \sum_{i=1}^{N} [R_i - f(X_i; q)]^2 , \tag{53}$$

where q is a vector of parameters and Θ the parameter space (Kuan and White, 1994). The solution to problem (53), with $f(X_i; q)$ representing a (single-hidden layer) FFBANN approximation to $E(R_i \mid X_i = x_i)$ gives rise to the NLS estimator:

$$\hat{q}_N = \arg\min_{q \in \Theta} \left[E_n(q) = N^{-1} \sum_{i=1}^{N} [R_i - f(X_i; q)]^2 \right], \tag{54}$$

where $q \equiv \left(w_{0,1}^{(1)}, \ldots, w_{0,K}^{(1)}, w_{1,1}^{(1)}, \ldots, w_{k,m}^{(1)}, \ldots, w_{K,M}^{(1)}, w_0^{(2)}, \ldots, w_m^{(2)}, \ldots, w_M^{(2)}\right) \in \Theta$. The estimator $\hat{q}_N$ is generally consistent and is distributed asymptotically normal, allowing for asymptotic inference (White, 1989).

White (1989) establishes consistency for a single-hidden layer FFBANN of the form discussed in this paper, under the assumptions that: (i) the data set $\{Z_i = (R_i, X_i)\}_{i=1}^{N}$ represent a sequence of IID real-valued and bounded random vectors, and (ii) $\hat{q}_N$ is locally identifiable. The second assumption ensures that during training the backpropagation algorithm will converge or diverge to a local minimum, thus ruling out "flat" areas of the fitting criterion (White, 1989). Kuan and White (1994) establish consistency of $\hat{q}_N$ in single-hidden layer FFBANN with linear activation functions in the output layer under the assumption that $\{Z_i = (R_i, X_i)\}_{i=1}^{N}$ is an asymptotically mean stationary stochastic process near epoch dependent on an underlying mixingale process (a broad class of dependent heterogenous processes). These results need to be extended further. White suggests, that in order to relax the assumption of asymptotic stationarity, one would need to adopt a tracking algorithm to ensure consistent estimation of $\hat{q}_N$,and recommends that results by Gerencsér (1986) may be of help.

Asymptotic normality of $\hat{q}_N$ is established by White (1989) and Kuan and White (1994) for similar conditions as those stated for consistency. In addition to the assumptions stated earlier, White (1989) assumes that the derivatives of the activation functions are bounded at all layers of the network and Kuan and White (1994) replace the assumption of asymptotic mean stationarity of the sequence $\{Z_i = (R_i, X_i)\}_{i=1}^{N}$ with strict stationarity.

The above asymptotic results provide the basis for asymptotic inferences using single-hidden layer FFBANNs. White (1989) states that using the steepest-descent gradient algorithm (i.e. delta rule) to train a FFBANN leads to less reliable inferences and suggests performing one Newton-Raphson step at the end of training. A quasi-Newton or conjugate gradient algorithm should be used instead for training, due to its superior convergence properties. The results by White (1989) suggest that this might be a superior approach to gain more reliable asymptotic inferences.

3.3.3.2 Kullback-Leibler Fitting Criterion

Minimizing the Kullback-Leibler fitting criterion in order to estimate the parameters of a FFBANN being used to approximate $E\left(R_i \,|X_i = x_i\right)$, amounts to minimizing the negative of (or maximizing) the log-likelihood function of a conditional Bernoulli distribution (i.e. the method of maximum likelihood). To see the connection, the Kullback-Leibler fitting criterion can be rewritten in the following manner, letting $Y_i = f\left(X_i ;q\right)$ be the output of the FFBANN being trained:

$$E(q) = \sum_{i=1}^{N}\left[\log\left(R_i^{\,R_i}\right) - \log\left(Y_i^{\,R_i}\right) + \log\left((1-R_i)^{1-R_i}\right) - \log\left((1-Y_i)^{1-R_i}\right)\right]. \qquad (55)$$

Since R_i takes only on a value of 0 or 1, $\log\left(R_i^{\,R_i}\right) = \log\left((1-R_i)^{1-R_i}\right) = 0$, making

$$E(q) = -\sum_{i=1}^{N}\left[R_i \log(Y_i) + (1-R_i)\log(1-Y_i)\right], \qquad (56)$$

which is the negative of the log-likelihood function for a dichotomous choice model with a conditional Bernoulli distribution. This result is dependent upon the existence of the conditional Bernoulli distribution, which can be ensured if the FFBANN being used has a logistic activation function in the output layer (see section 3.3.2). Thus, if some regularity conditions are satisfied, then the properties of the maximum likelihood estimator (MLE):

$$\hat{q}_{MLE} = \arg\min_{q\in\Theta}\left[E(q) = -\sum_{i=1}^{N}\left[R_i \log(f\left(X_i;q\right)) + (1-R_i)\log(1- f\left(X_i;q\right))\right]\right], \quad (57)$$

of consistency, asymptotic normality and asymptotic efficiency follow (see Spanos, 1999).

To implement the method of maximum likelihood in certain computer packages (e.g. MATLAB), the algorithms used to train a FFBANN require that the derivative of the fitting criterion be taken with respect to the error , $e_i = R_i - Y_i$. Instead of using the

Kullback-Leibler fitting criterion, the modeler could choose to use equation (56) instead. In this case,

$$\frac{\partial E(.)}{\partial e_i} = \frac{\partial E(.)}{\partial (R_i - Y_i)} = \left[\frac{\partial R_i - \partial Y_i}{\partial E(.)}\right]^{-1} = \left[\left(\frac{\partial E(.)}{\partial R_i}\right)^{-1} - \left(\frac{\partial E(.)}{\partial Y_i}\right)^{-1}\right]^{-1}, \qquad (58)$$

where $\frac{\partial E(.)}{\partial R_i} = \log(1 - Y_i) - \log(Y_i)$ and $\frac{\partial E(.)}{\partial Y_i} = \frac{(1 - R_i)}{(1 - Y_i)} - \frac{R_i}{Y_i}$. The derivative given by equation (27) would then be used in the algorithms discussed in section 3.2.3.[31]

To proceed to perform asymptotic inference the modeler needs to ensure that an identifiable local minimum is obtained and that the other conditions mentioned in this and the previous subsection are satisfied. White (1989) stresses the point that one of the drawbacks of using the back-propagation algorithms is that it can get stuck at an unidentifiable local minimum, saddle points or diverge. One practical method to increase the likelihood that one has obtained an identifiable local minimum is to vary the starting points randomly and/or the choice of training data set, and then pick the FFBANN that gives you the smallest value for $E(q)$. If enough training runs are conducted and all networks that obtain the same local minima have approximately similar weight values, then the modeler is provided with an indication that a locally identifiable local minimum has been found This procedure applies to both fitting criteria presented (White, 1989), and should be preformed before any inference is undertaken.

3.3.4 Hypothesis Testing and Interpretation of the Connection Weights

[31] MATLAB procedures for an MLE (Kullback-Leibler) fitting criterion and its derivative with respect to e_i that work in conjunction with the procedures in the Neural Networks Toolbox are provided in Appendix A. Simulation results conducted by the author suggest that when using this fitting criterion and associated procedures, the modeler should use the Levenberg-Marquardt algorithm due to its convergence properties. Second, if cross-validation is used, *v* should be greater than or equal to 15.

There are two sets of hypotheses of particular interest when using FFBANNs. The first set of hypotheses deal with the significance of particular variables used in the regression, so-called “siginifcance tests” (White, 1989). The second set of hypotheses deals with the significance of particular hidden units within a FFBANN. Other linear and nonlinear restrictions are beyond the scope of this paper and are an area for future research. Furthermore, all models estimated in the calculation of test statistics discussed below should follow the same guidelines as discussed earlier in the paper.

Recall, a single-hidden layer FFBANN takes the following functional form:

$$Y_i = F_2\left(w_0^{(2)} + \sum_{m=1}^{M} w_m^{(2)} \cdot F_1\left(w_{0,m}^{(1)} + \sum_{k=1}^{K} w_{k,m}^{(1)} \cdot X_{k,i} \right)\right).$$

If the modeler is interested in testing the significance of a particular variable X_k, then the modeler would want to test the following set of hypotheses:

$$H_0 : w_{k,m}^{(1)} = 0 \ \forall m \text{ against } H_1 : w_{k,m}^{(1)} \neq 0 \text{ for at least one } m. \qquad (59)$$

Assuming the modeler uses the *MSE* fitting criterion and that $\hat{q}_N$ is both consistent and asymptotically normal, the hypotheses in (59) can be tested using the pseudo-F statistic presented by Davidson and MacKinnon (1993):

$$F(X_i) = \frac{\overline{T}}{s^2}\left(E_R(\tilde{q}) - E_U(\hat{q})\right) \overset{H_0}{\underset{\propto}{\sim}}` c^2(m), \qquad (60)$$

where $\overline{T}$ is the number of observations in the training data set, $E(.)$ is the *MSE* fitting criterion, $E_R(\tilde{q})$ is the mean square (training) error from the restricted model (i.e. where the restrictions under H_0 are applied), $E_U(\hat{q})$ is the mean square (training) error from the unrestricted model, *m* is the number of nodes in the hidden layer,

$s^2 = \frac{\hat{e}'\hat{e}}{\bar{T}-K} = \frac{E_U\left(\hat{q}\right)\cdot\bar{T}}{\bar{T}-K}$, $\hat{e} \equiv \left(R-Y\right)$, and *R* and *Y* are the corresponding column vectors of the output targets and predicted outputs of the unrestricted FFBANN (from the training data set), respectively. Other methods to test the above hypotheses have been proposed by White (1989) when the *MSE* fitting criterion is utilized. Practical procedures for implementing these tests are discussed in section 5.3.

When the Kullback-Leibler (or MLE) fitting criterion is used the modeler can test the hypotheses in (59) by using the asymptotic likelihood ratio test statistic:

$$LR(X_i) = 2\left(E_U\left(\hat{q}\right) - E_R\left(\tilde{q}\right)\right) \overset{H_0}{\underset{\propto}{\sim}} c^2(m), \tag{61}$$

where: $E(.)$ is the Kullback-Leibler fitting criterion, $E_R\left(\tilde{q}\right)$ is the negative of the log likelihood of the restricted model, and $E_U\left(\hat{q}\right)$ is the negative of the log likelihood of the unrestricted model (Spanos, 1999).

The second set of hypotheses of interest is of the form:

$$H_0 : w_m^{(2)} = 0 \text{ against } H_1 : w_m^{(2)} \neq 0. \tag{62}$$

In essence, the modeler is interested in testing if one (or more) hidden nodes in the hidden layer of a single-hidden layer FFBANN help in obtaining the achieved local minimum (White, 1989). The primary difficulty with this specification test is that under the null hypotheses, when $w_m^{(2)} = 0$, the parameters $w_{k,m}^{(1)}$ are not identified for all *k*. Thus, it becomes difficult to determine the distribution of any corresponding test statistic under the null hypotheses, making inference more difficult. One potential solution, might be to test the hypotheses:

$$H_0 : w_m^{(2)} = 0 \text{ and } w_{k,m}^{(1)} = 0 \ \forall k \text{ against } H_1 : w_m^{(2)} \neq 0 \text{ and/or } w_{k,m}^{(1)} \text{ for only one } k \text{ ,(63)}$$

using the test statistics discussed earlier. Another method discussed by Kuan and White (1994) for testing the hypotheses in equation (62) is based upon a test procedure developed by Bierens (1990), which seeks to "choose [$w_m^{(2)}$] values that optimize the direction in which nonlinearity is sought (p. 25)" (see Kuan and White (1994) for further details).

4. Model Construction and Estimation

The second objective of this study was to provide modeling guidelines for the construction and estimation of FFBANNs using (CV) dichotomous choice data. This section examines the modeling decisions made during the model construction and estimation process. The first sub-section presents the empirical data used to estimate all the models examined in the study. Sub-section two presents the statistical (performance) measures used to determine out-of-sample predictive accuracy upon which the models will be compared and evaluated. The third sub-section examines the different decisions modelers encounter in constructing FFBANNs to determine if any general guidelines can be learned for constructing models using dichotomous choice (survey) data. The design and estimation decisions examined in section 3 will be expanded on in this sub-section. Finally, the "optimal" FFBANNs obtained from the simulations, as well as, the estimated logit and probit models used in the comparative analyses in section 5 will be presented in subsections four and five, respectively.

4.1 Data

The survey data set (denoted E-K from here on) used to estimate all the models in the paper was collected by Eisen-Hecht and Kramer (2002). They contracted with a third party to conduct a CV survey, interviewing 1,085 individuals in the Catawba River Basin of North and South Carolina. Of the 1,085 respondents, 915 were usable for further analysis due to discrepancies during collection of the data.[32] All of the respondents were asked questions concerning water quality, use of aquatic resources in the Catawba River Basin, demographics and financial status. Of primary interest was a referendum style CV question, which asked respondents if they would be willing to adopt a water quality management plan that maintained water quality at current levels over time. “The management plan was offered to respondents at one of eight different price levels ranging from $5 to $250 [randomly assigned] per year for five years (Eisen-Hecht and Kramer, 2002; p. 6).”[33] Table 1 provides a description and descriptive statistics for the variables in the E-K data set.

The objective of Eisen-Hecht and Kramer’s (2002) study was to perform a cost-benefit analysis of maintaining the current level of water quality in the Catawba River Basin. Of particular interest is that Eisen-Hecht and Kramter estimated a probit model to examine the potential support for the proposed management plan referred to in the referendum style CV question. Estimation results for the re-estimated probit (and logit) model are presented in section 4.5.

4.2 Measurement Statistics for Evaluating Predictive Accuracy

Using the data provided by Eisen-Hecht and Kramer (2002), FFBANNs as well as traditional binary logit and probit models will all be evaluated using a number of performance meansures of out-of-sample predictive accuracy and fit found in the literature. The first measure used is the mean square forecast error (*MSFE*), which is

[32] See Eise-Hecht and Kramer (2002) for further explanation.
[33] The prose in square brackets was added by the author for further clarification.

Table 1: Variables and Descriptive Statistics (915 observations)

Name	Description	Scale[1]	Mean	Standard Error[2]
VOTE[3]	Vote on the proposed management plan.	Binary	0.6776	---
WTPAMT	Stated price for the management plan presented to the respondent.	Discrete/ Ordered	---	---
TAX	Respondent's view on the importance of reducing taxes.	Binary	0.7071	---
WPCONTROL	Indicates if the respondent has heard of efforts to control water quality in the Catawba Basin.	Binary	0.7148	---
USE	Respondent's use of the Catawba River and if it affected their vote for the management plan.	Binary	0.5498	---
DRQUAL	Respondent's view on drinking water quality and if it affected their vote on the management plan.	Binary	0.9333	---
OTHERUSE	Respondent's view on family's use of the Catawba River and if it affected their vote on the management plan.	Binary	0.6022	---
EXIST	Respondent's knowledge of water quality in the Catawba basin and if it affected their vote on the management plan.	Binary	0.7738	---
LIKELY	Respondent's view on whether or not the management plan will succeed.	Binary	0.7781	---
ITEM	Does the respondent own an item used for outdoor water-based recreation?	Binary	0.7093	---
ENVORG	Does the respondent belong to an environmental organization?	Binary	0.1235	---
QUALWORS	Respondent's view on water quality and if it has gotten worse over the past five years.	Binary	0.4951	---
TAPGOOD	Respondent's view on quality of tap water and if it has gotten worse over the last five years.	Binary	0.4667	---
AGE	Age of Respondent	Continuous	49.90	14.73
NEWAREA	Has the respondent lived in the Catawba Basin for less than five years?	Binary	0.1202	---

UNIV	Respondent’s viewpoint on the trustworthiness of universities.	Binary	0.7071	---
EDU	Education Level – College or Higher.	Binary	0.6153	---
SEX	Male (1) or Female (0)	Binary	0.5443	---
INCOME	Household Income of Respondent ($)	Continuous	55,929	39,334
STATE	North Carolina (1), South Carolina (0)	Binary	0.8022	---
DATELAG	Lag in days between receiving information and phone interview.	Continuous	25.04	20.89

Source: Eisen-Hecht and Kramer (2002)

[1] A value of ‘1’ indicates an affirmative/yes answer and a value of ‘0’ indicates a negative/no answer by the respondent.

[2] Only reported if applicable.

[3] Dependent Variable in FFBANN, logit and probit models.

equivalent to *MSE* and is calculated using a test data set independent of the training and validation data sets, as follows:

$$MSFE = \frac{1}{N}\sum_{i=1}^{N}\left(R_i - Y_i\right)^2 , \qquad (64)$$

where N is the number of input/output patterns in the test data set (Qi, 2001). As with *MSE*, this measure provides an idea of the “closeness” of the predictions (Kastens and Featherstone, 1996). The second measure used is the percent of input patterns out of the test data set that are correctly classified by the model being examined (see Kastens and Featherstone, 1996; West, Brockett and Golden, 1997). This measure is denoted as PR and is explicitly given in section 3.2 of the paper. In addition, two error measures similar to *PR* are used, which analyze the type-I and type-II prediction errors. The Type-I error statistic examines the percentage of times the model predicts that a respondent does not vote for the proposed water quality management plan, when the respondent did. The Type-II error statistic examines the percentage of times the model predicts that a respondent does vote for the proposed water quality management plan, when the respondent did not (see West, Brocket and Golden, 1997). These last two measures help to illustrate the predictive patterns that are of interest to the modeler, by providing a measure of how the models are erring when predicting incorrectly (Principe, Euliano and Lefebvre, 2000).

4.3 Feed-Forward Artificial Neural Network Construction and Simulation

This sub-section concerns the construction and estimation of the FFBANNs that will be used for comparative analyses, as well as, statistical inference in the next section of the paper. The first sub-section examines the methods used to perform a number of simulations to inspect various design and implementation decisions. Further subsections then consider a number of these modeling decisions individually to determine if any general guidelines (i.e. "rules of thumb") can be established. Kastens and Featherstone (1996) highlight the difficulty of this process by stating that:

> "*Because* [FFBANN] *choice models are not supported by a large body of empirical research* [primarily in the field of qualitative choice statistical models], *there is little a priori basis for choosing a particular* [FFBANN] *(given the explanatory variables) for effecting out-of-sample prediction.* (p. 406)."[34]

4.3.1 Model Construction

FFBANNs are usually seen as a "black box", since in general the individual connection weights are not readily interpretable economically (Peng and Wen, 1999). Furthermore, the choice of net architecture and training style tend to be problem dependent. Recall from section three that there are a number of decisions that have to be made by the modeler when constructing and training a FFBANN. These decisions (or issues) include: (i) choice of training algorithm, (ii) number of hidden layers in the neural

[34] The prose in square brackets was added by the author for further clarification.

network, (iii) number of hidden nodes in each hidden layer, (iv) type of activation functions for the hidden and output layers, (v) choice of fitting criterion, and (vi) choice of stopping rule. Decisions (ii) thru (iv) are concerned with net architecture, while decisions (i), (v) and (vi) are concerned with the training of a neural network. Other decisions (not explicitly examined via simulations) include weight initialization procedures, scaling of the data, and choice of training, validation and test data sets. Standard methods from the literature were applied in order to attend to these latter decisions and are discussed further below (see Fausett, 1994; Mehrotra, Mohan and Ranka, 1997; Principe, Euliano and Lefebvre, 2000 for further discussion).

Any one choice for a particular design/training decision is likely to be dependent upon the choices made about the remaining design/training decisions. For example, the choice of training algorithm may depend on the size and structure of the training data set. Thus, the optimal approach to constructing a model is to perform a grid search over all possible combinations of potential choices for each design/training decision, but such an approach is not practical due to the potentially large number of models that would have to be examined. For the design/training decisions examined in this study, if all combinations were analyzed, 57,600 unique FFBANNs would have to have been estimated and compared. To make this number more manageable, guidelines from the literature were used to help decrease the dimensionality of this task so only 580 unique FFBANNs were analyzed.

To guide the model construction process and to help develop potential guidelines for constructing FFBANNs, simulations using the E-K data set were conducted. From these simulations, the "optimal" FFBANNs were chosen for the comparative analyses conducted in section five. "Optimal" here is defined as those networks that provide the best out-of-sample predictive capabilities, measured using the *MSFE*, *PR*, Type-I Error and Type-II Error measurement statistics on the entire E-K and test data sets.

For the purpose of obtaining a FFBANN that is generalizable to new data, independent training, validation and test data sets are constructed from the empirical data set. The validation data set is used to prevent over-fitting of the FFBANN to the training data set during training, and is not used to examine out-of-sample performance between various net architectures (West, Brockett and Golden, 1997). In this context, performance of a FFBANN on the validation data set provides another indication of within-sample predictive performance, given termination of the training algorithm is based upon the performance of the FFBANN on this data set. Thus, in order to compare the out-of-sample predictive capabilities of all the models estimated, a separate independent test data set was used.[35] Following West, Brockett and Golden (1997) the E-K data set was subdivided in the following manner: sixty percent of the data is used for training, twenty percent for validation and twenty percent for testing. A larger training data set is used to control the level over-fitting to the training data set (Ferret, 1993).

All input data vectors where scaled to between $[-1,1]$ when the hyperbolic tangent activation function was used in the hidden layer(s) of a network, and to between $[0,1]$

[35] The validation data set can be used to choose between models that are tested using the same training and validation data sets and that have the same net architecture. Such a case arises when the modeler uses a re-initialization procedure, which trains the network using a number of different starting points, and for generalization purposes the modeler chooses the network with the minimum mean square error on the validation data set. An alternative would be to use the test data set, and choose the network with the lowest *MSFE* using the test data set. This alternative provides the same benefit as using the validation data set, while at the same time ensuring that the best models are being chosen for comparisons with other (statistical) models. The modeler should keep in mind that it may be the case that a local minimum on the training data set has not been obtained in either case. Thus, future research still needs to be conducted to find a termination criteria that preserves within and out-of-sample properties of the neural network.

when the logistic activation function was used. The scaling was completed by a linear transformation using the minimum and maximum values of the explanatory variables from the entire E-K data set (see Demuthe and Beale, 2001).

To prevent starting bias in the weights during initialization, connection weights were initialized using the procedure developed by Nguyen and Widrow (see Fausett, 1994). In addition, batch training was used for the estimation of all network weights. Due to its flexibility as a matrix based programming language, MATLAB along with the Neural Networks Toolbox was used to train all FFBANNs examined in this study.

To examine the design and training decisions (i) thru (vii), simulations using the E-K data set were conducted. For each simulation a sample re-use procedure was utilized. This procedure randomly partitioned the data into training, validation and test data sets as described above (West, Brockett and Golden, 1997).[36] For each simulation, 500 independent training, validation and test data sets (partitions) were randomly generated using replacement, and a separate neural network was trained on each sample partition of the E-K data set generated. Golden, West and Brockett (1997) note that the above sample re-use procedure "provides information about the distribution of outcomes (i.e. mean, range, and variance) and the expected accuracy and reliability of the modeling procedure (p. 378)." Thus, for each run the performance statistics discussed in section 4.2 are collected for each network estimated. Then for each simulation the mean, standard deviation and minimum and maximum of these measures for each distinct network type

[36] West, Brocket and Golden (1997) note that such a procedure minimizes the risk of obtaining a local minimum on the surface of the fitting criterion when used for finding an "optimal" network. The procedure is similar to using multiple starting points for the parameter vector when conducting maximum likelihood or some other off-line optimization procedure in order to obtain a global optima. The latter procedure of generating multiple starting points and finding the optimal solution using each starting point to initialize the training algorithm can also be used and is recommended by the author.

and training style were used to examine the design/training decisions previously mentioned. Tables for all summary statistics are provided in Appendix B.

Each of the net architectures and training decisions, (i) – (vii), are examined individually in the following sub-sections. The specific simulations conducted concerning each are discussed there. The order of the issues examined is reflective of how the model construction process was approached.

After all the simulations had been conducted, the networks with the overall best out-of-sample predictive capabilities, based on the performance measures using the test and entire E-K data sets, were chosen to be used in the comparison with the logit and probit models. This decision was based on the premise that:

> "*a researcher interested in developing a predictive model would only be interested in finding the "best" training session. This can be accomplished by identifying the data partitioning that produced the largest percentage of correctly classified data when applied to its own testing subsample … (i.e., the best out-of-sample predictive accuracy). In essence, by taking the best performing model one achieves a more optimal result than that obtained using a single sample partitioning* (West, Brockett and Golden, 1997: p. 378-9)."

The process used to identify these "optimal" FFBANNs is further examined in section 4.4.

4.3.2 Choice of Training Algorithm

A significant amount of empirical evidence has shown that the traditional steepest descent algorithm (the delta rule) used to train neural networks tends to be a poor choice due to the slow progression during training toward an optimal solution (see Bazarra, Sherali and Shetty, 1993; Ripley, 1996). Thus, the question arises as to what (type of) algorithm should be used in its place? Demuth and Beale (2001) mention that the performance of any particular algorithm is dependent upon a number of factors, which include the nature of the problem, the size of the training set, the size of the network (number of connection weights), the choice of fitting criterion and choice of stopping rule. Some of these factors are further discussed in following sub-sections.

For classification problems, Demuth and Beale (2001) find an alternative to the steepest descent algorithm, known as the resilient backpropagation (RP) algorithm, which tends to be the fastest algorithm (in terms of time) in converging to an optimal solution. This algorithm uses the signs of the elements of the gradient of the fitting criterion to determine the direction of change, and replaces the derivative $\frac{\partial e_i^2}{\partial w_{m_{l-1},m_l}^{(l)}}$ in equation (42) with $\operatorname{sgn}\left(\frac{\partial e_i^2}{\partial w_{m_{l-1},m_l}^{(l)}}\right)\cdot r$, where r is a parameter initially set by the modeler and incrementally changed by the algorithm depending on the sign of $\frac{\partial e_i^2}{\partial w_{m_{l-1},m_l}^{(l)}}$ as i changes (see Reidmiller and Braun, 1993 for full a description of this algorithm).

For a reasonable number of weights (up to 1,000), Ripley (1996) recommends the use of quasi-Newton methods. One such algorithm, the BFGS quasi-Newton algorithm, was presented in section 3.2.3. For larger problems, Ripley suggests using conjugate gradient methods or the limited-memory BFGS quasi-Newton algorithm (see Bazaraa, Sherali and Shetty, 1993 for a description of the latter). The benefit of these types of algorithms is that they have super-linear convergence and in practice tend to converge quickly once they are in a local neighborhood of an optimal solution. The quasi-Newton algorithms are especially effective in this respect (Ripley, 1996).

In order to examine the choice of algorithm for training a FFBANN (using the E-K data), a simulation was conducted comparing six different algorithms in the MATLAB

Neural Networks Toolbox. The algorithms examined include: (i) the BFGS quasi-Newton (BFG) Algorithm, (ii) a conjugate gradient (CGF) algorithm with Fletcher-Reeves update (see Fine, 1999), (iii) steepest descent (GDX) algorithm with adaptive learning and momentum rates (see Demuth and Beale, 2001), (iv) the Levenberg-Marquardt (LM) algorithm (see Fine, 1999), (v) the Resilient Backpropagation (RP) algorithm, and (vi) a scaled conjugate gradient (SCG) algorithm, which combines the traditional conjugate gradient and Levenberg-Marquardt algorithms (see Moller, 1993). For each algorithm, five hundred simulation runs were conducted using the sample re-use procedure described in the previous subsection. For each run and algorithm, twenty single-hidden layer FFBANNs with one to twenty nodes in the hidden layer were trained. Each network used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, the stopping rule (S5) was used with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1 \times 10^{-15}$, $E_{\min} = 1 \times 10^{-5}$ and $MAX = 1000$. These settings were used for all further simulations as well, unless otherwise noted. The results of the simulation runs are shown in Figures 4 thru 7, and the summary statistics are provided in Appendix B. The measures reported in each figure are means of the performance measures described in section 4.2 over the individual neural networks estimated for each net architecture and training algorithm examined.

According to Figure 4, the FFBANNs trained using the LM algorithm provided the “best” fit to the training data set (i.e. the mean *MSE* was the lowest across all the different size FFBANNs trained). In contrast, Figures 5 and 6 provide evidence that increased performance on the training data set for these models resulted in over-fitting

Figure 4: Simulation Results - Mean *MSE* using the Training Data Set

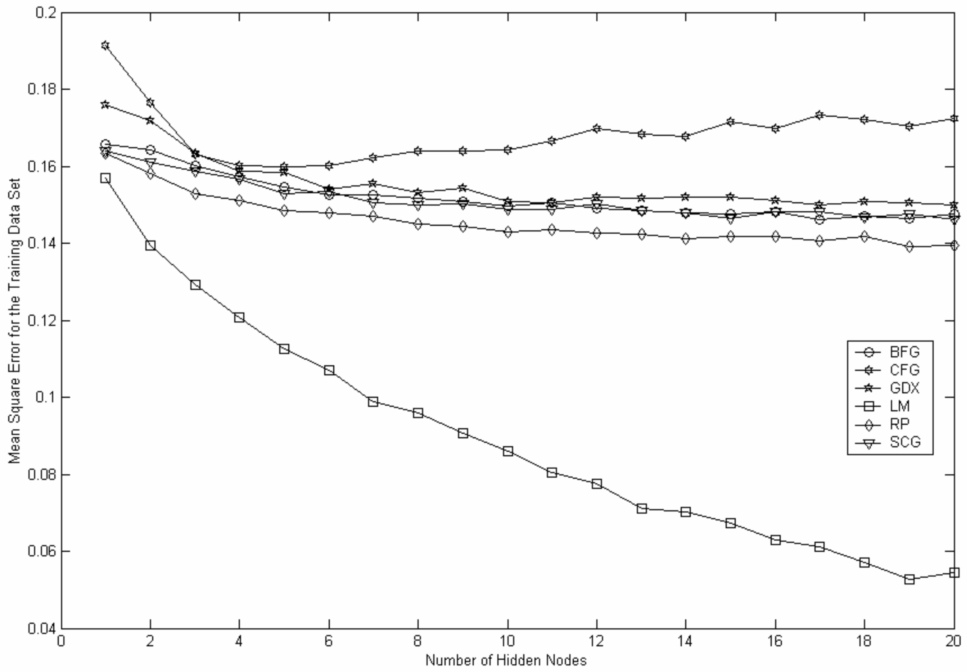


Figure 5: Simulation Results - Mean *MSFE* using the Test Data Set

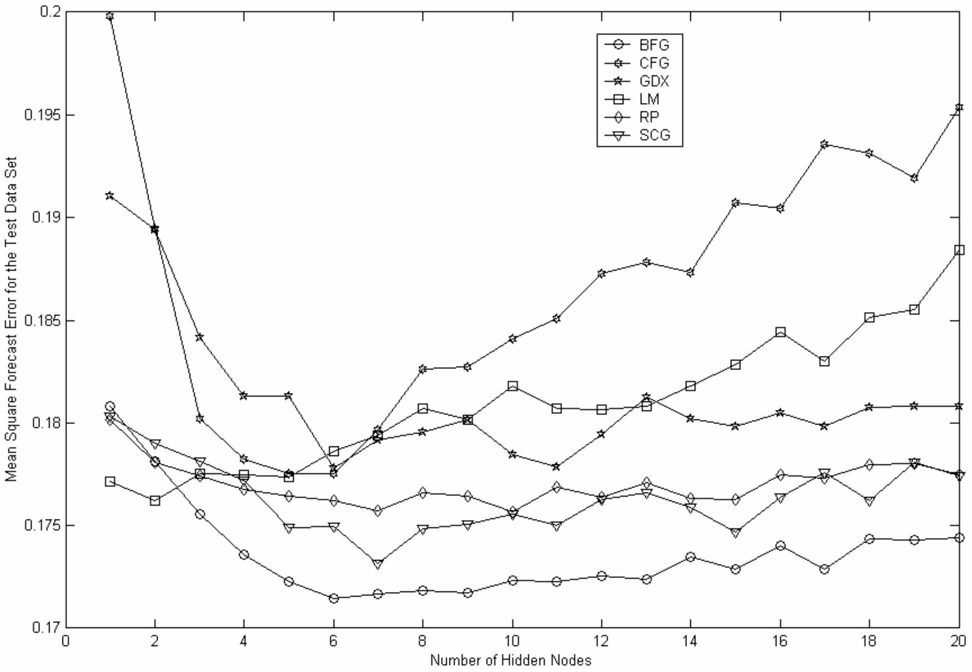


Figure 6: Simulation Results: Mean *PR* using Test Data Set

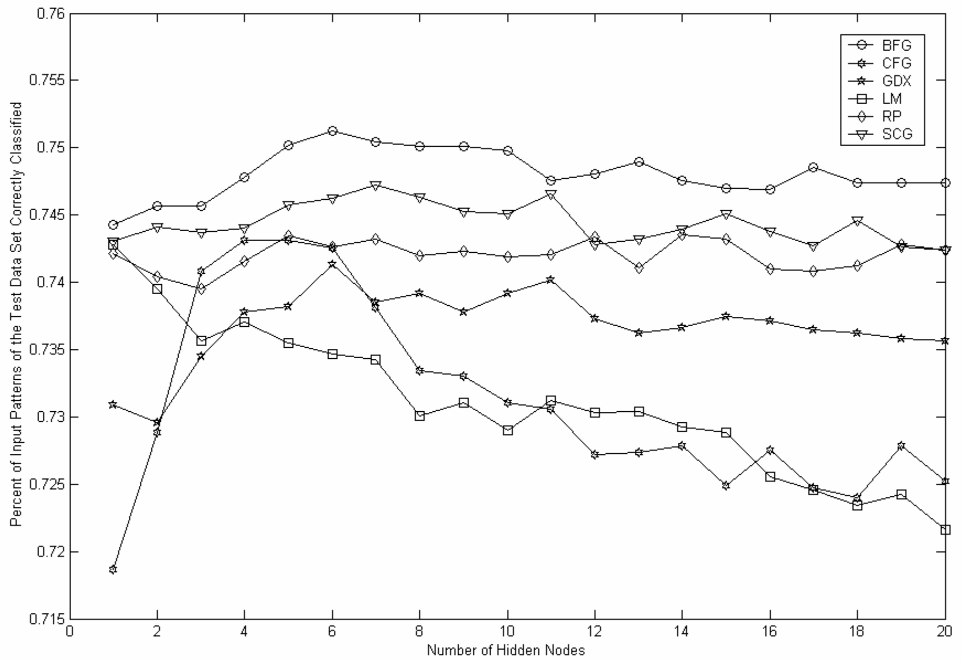
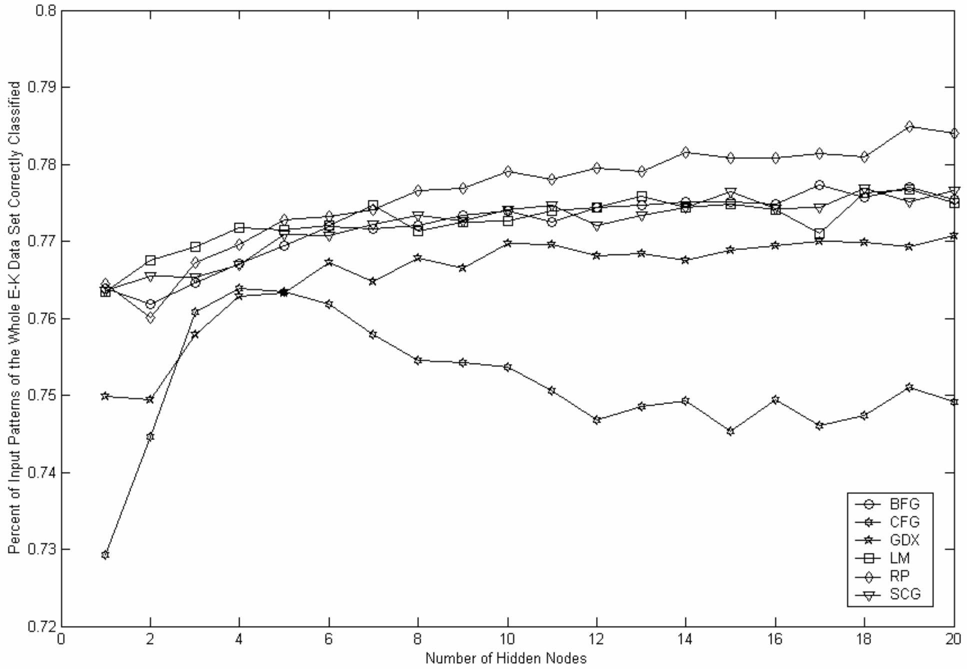


Figure 7: Simulation Results - Mean *PR* using the Entire E-K Data Set



and decreased performance on the test data set (i.e. a lower *MSFE*). In contrast, the BFG algorithm provided the “best” fit (with the lowest *MSFE* as network size increased) to the test data set, and was comparable to the other algorithms’ performance (except the LM algorithm) on the training data set. If the modeler is primarily interested in generalizability and the predictive performance of the neural network being trained then Figures 6 and 7 indicate that the BFG algorithm should be used during training. A close second choice would be the SCG algorithm. For statistical inference purposes the LM algorithm will converge to a local optimum the fastest, but at the cost of potentially overfitting to the training data set and poor generalization. These results emphasize the tradeoff between within and out-of-sample performance (i.e. obtaining a local minimum on the training data set versus generalizability). This tradeoff is important to keep in mind, because any statistical inference is dependent upon the consistency and asymptotic normality of the connection weights, which are achieved by finding a local minimum using the training data set.

The BFG algorithm was used to train the rest of the FFBANNs examined in this paper. Simulation results show that this algorithm did not tend to result in overfitting to the training data set during training (potentially due to the slower convergence experienced by this algorithm when compared to the others), and provided the best overall generalization capabilities in terms of out-of-sample predictive accuracy. The reader should keep in mind that the choice of algorithm is problem dependent, and the above results should be analyzed in that context.

4.3.3 Number of Hidden Layers

We know from the approximation theorems in section 3.2.2 that a single-hidden layer FFBANN can approximate any measurable function up to a set of points of measure zero. These results are easily extended to FFBANNs with more than one hidden layer (see White, Hornick and Stinchcombe, 1992). Fine (1999) states that while single-hidden layer FFBANNs are useful for approximating particular families of functions, such as the family of continuous functions, p^{th}-order integrable functions, and r^{th}-differentiable functions, multiple hidden layer FFBANNs are useful for approximating composite functions. Recall, that the conditional mean function for the logit and probit models are of this form, i.e. $p = F[h(x_i ;q)]$. Furthermore, multiple hidden layer FFBANNs can exactly approximate functions with discontinuities, even though single-hidden layer FFBANNs can achieve relatively close approximations (Fine, 1999). Depending on the nature of the problem and function being approximated, the modeler must choose the number of hidden layers to include in the net architecture.

Given that one of our primary goals is a network that can generalize well to new data, Ferrett (1993) recommends that the smaller (in size) the network the better the network will be able to generalize. This follows from the principle of Occam’s Razor discussed in section 3.3.2. Thus, for practical purposes a single-hidden layer FFBANN would be suggested over a two-hidden layer FFBANN. To examine this choice, a simulation was conducted examining the capabilities of a single-hidden layer FFBANN when compared to a two-hidden layer FFBANN using the E-K data set. For both the single and two-hidden layer FFBANNs the number of nodes in the each hidden layer was varied from one to twenty nodes. The hyperbolic tangent activation function was used in each hidden layer and the logistic activation function was used in the output layer.

Finally, each network was batch trained using the BFG algorithm with the stopping rules specified in subsection 4.3.2.

Figures 8 thru 11 provide the mean (training) *MSE*, *MSFE* and *PR* measures for the neural networks trained using the 500 randomly generated data partitions using the E-K data set. The results for the two-hidden layer networks in each figure represent that two-layer network that performed the “best” for the particular measure being examined in order to obtain the most reliable comparisons. Full summary statistics for this simulation are provided in Appendix B.

The results in Figures 8 thru 11 support Ferret’s (1993) recommendation of using a single-hidden layer FFBANN over a two-hidden layer FFBANN if the purpose of the model is out-of-sample prediction. The single-hidden layer FFBANN clearly provided a “better” fit to the test data set and predicted the most out-of-sample responses correctly when compared to the “best” performing two-hidden layer FFBANN. In contrast, if the model is to be utilized for statistical inference, then a two-hidden layer FFBANN may be a more suitable choice. Given that obtaining a local minimum on the training data set is of higher importance for inferential purposes, a second hidden layer is not that costly in terms of generalization (see section 3.3.2). In Figures 8 thru 11, the performance measures tended to differ by less than 0.005, indicating that the tradeoff between within and out-of-sample performance for a one or two-hidden layer FFBANN is not that significant. In this case, the modeler may rely on the principle of Occam’s Razor and use a single-hidden-layer FFBANN.

4.3.4 Number of Hidden Nodes

To motivate how the number of nodes can affect the approximation results of

Figure 8: Simulation Results - Mean *MSE* using the Training Data Set

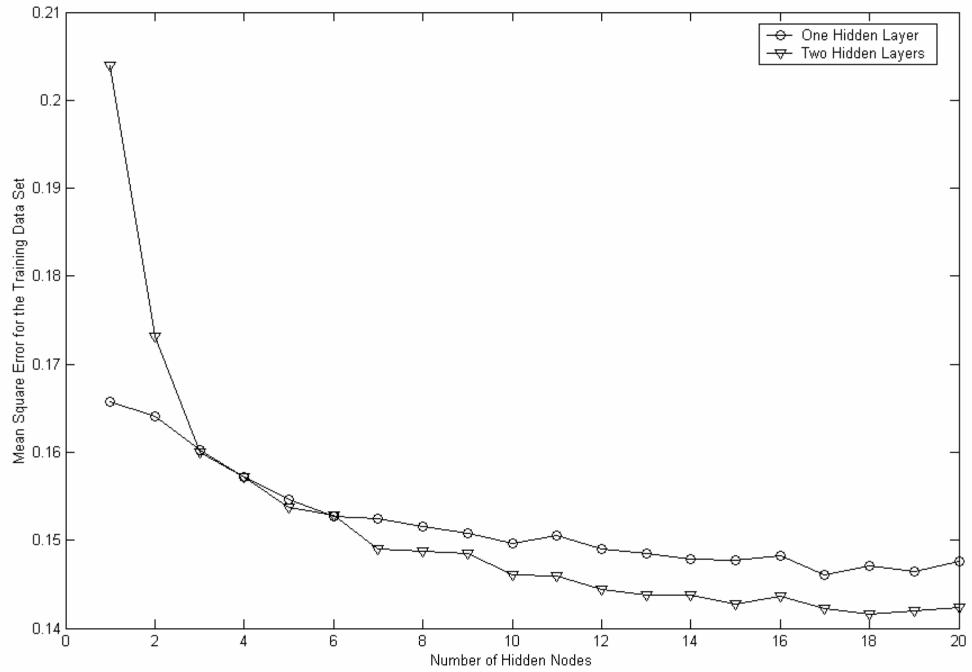


Figure 9: Simulation Results - Mean *MSFE* using the Test Data Set

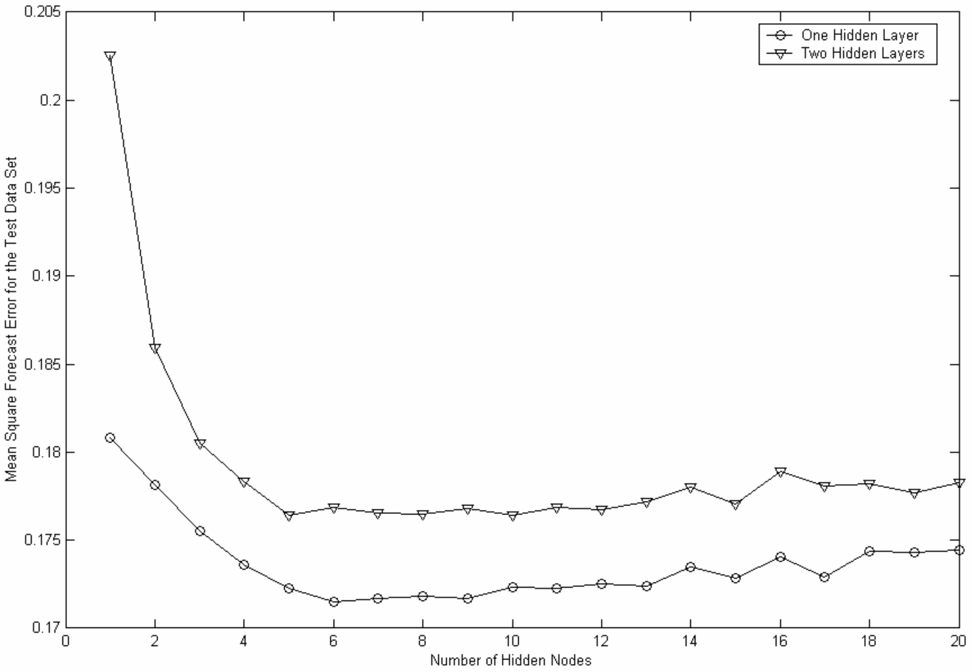


Figure 10: Simulation Results: Mean *PR* using Test Data Set

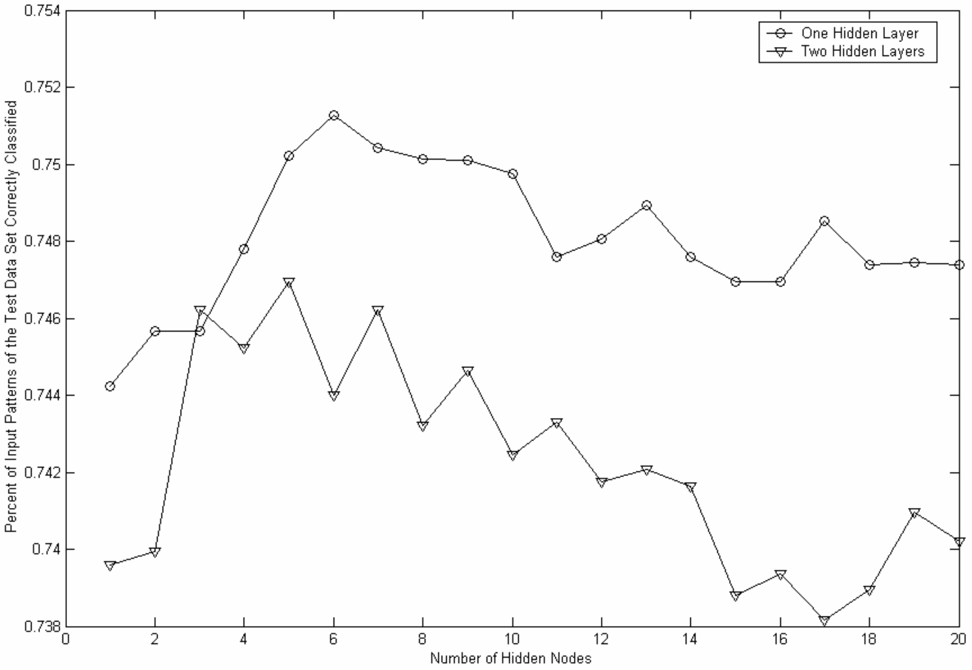
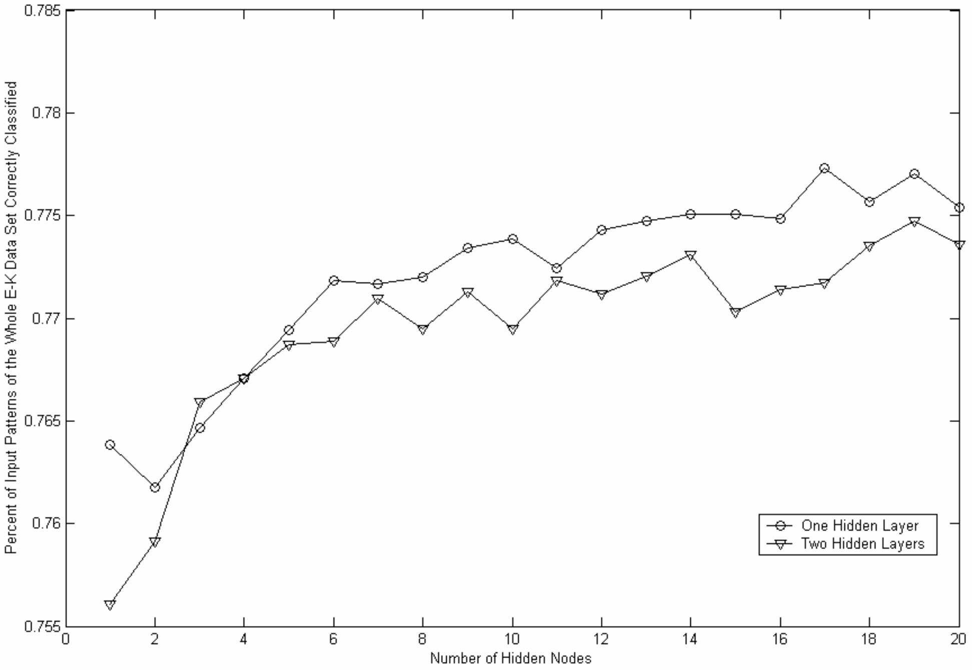


Figure 11: Simulation Results - Mean *PR* using the Entire E-K Data Set



a FFBANN, consider the following two cases: (i) a network with too few nodes and (ii) a network with too many nodes. The neural network with two few nodes does not have enough degrees of freedom (i.e. independent parameters) to correctly classify all the input data patterns of the training data set, so it will adjust the weights to minimize the fitting criterion, thereby hopefully correctly classifying the majority of input patterns. The neural network with too many nodes, has more than enough degrees of freedom, and perfectly classifies all the input training patterns, but performs poorly when used to examine the predictive accuracy of the network on an independent test data set due to over-fitting on the training data set (Principe, Euliano and Lefebvre, 1999). These two cases illustrate the problem faced by the modeler when choosing the number of hidden nodes in each layer of the neural network. In essence, each additional node in the hidden layer increases the capacity of the neural network to fit the data.

How many hidden nodes are too many? West. Brockett and Golden (1997) state that the answer to that question is problem dependent. They suggest that in practice, the modeler start with the simplest net architecture, no neurons in the hidden layer, and successively increase the number of neurons in the hidden layer as long as the *MSFE* for the validation data set decreases, which amounts to finding a peak of generalizability. Following this approach, Fahlman and Lebiere (1990) developed an iterative construction algorithm for FFBANNs, sometimes referred to as cascade correlation, based on the idea of adding additional nodes to the hidden layer, while only training the new weights associated with the new node at each iteration. Once a pre-specified measure of fit is achieved, the algorithm ceases and outputs the current net architecture as optimal, which is used to retrain the neural network to update all the weights. The new weights are selected “to maximize the absolute value of the covariance (over training-set examples) between the output value of the unit [node] and the prediction error before that unit is added (Ripley, 1996; p. 172).”[37]

To determine how many training input patterns are needed to achieve a pre-specified percentage of predictive accuracy for a given size network, Baum and Haussler (1997) proposed using the following inequality:

$$P > \frac{|w|}{1 - E(PR)}, \tag{65}$$

where P is the number of input patterns (observations) in the training data set, $|w|$ is the number of connection weights, and $E(PR)$ is the expected prediction accuracy (as a decimal percentage) of the neural network on an independent test data set. Equation (65) can also be used as a guideline to determine the size of network that should be trained for a given level of predictive accuracy on the test data set. For example, if the training data set has 549 input patterns with 20 input variables and the desired predictive accuracy on the test data set is 0.75, then the network should have no more than 137 weights, which means that the modeler should start with a FFBANN with no more than 6 nodes in the hidden layer. The above method is not recommended as a determining factor in obtaining an “optimal” net architecture, but as a starting point to find one.

During training, it is expected that as the number of hidden nodes increases, *MSE* should decrease due to an increasing number of degrees of freedom. Figure 4 supports this conclusion. For all of the algorithms, as the number of hidden nodes increases, *MSE* on the training data set decreases. As seen above, this usually implies that

[37] This option is not explored in this paper, but more information can be found in Fahlman and Lebiere (1990). The prose in square brackets was added by the author for further clarification.

generalization by the network becomes poorer as the number of hidden nodes increases, which is partially supported by Figures 5 and 6. A useful “rule of thumb” becomes apparent when examining these two figures. The number of hidden nodes in the first hidden layer should be less than (or equal to) the number of inputs in the input layer. For a second hidden layer, this implies that the number of hidden nodes in the second hidden layer should be less than (or equal to) the number of hidden nodes in the first hidden layer. Recall, the addition of a hidden layer is aimed at decreasing the dimensionality of the problem; thereby increasing the approximation capabilities of the FFBANN.[38] While the above principle usually holds true for single-hidden layer FFBANN, the tables in Appendix B indicate that this might not be the case for two-hidden layer FFBANNs. For example, examining Table B5, the *MSFE* for a two-hidden layer FFBANN with 9 hidden nodes in the first hidden layer and 13 hidden nodes in the second layer did better on average then other net architectures with the same number of hidden nodes in the first layer.

Following the above recommendations of using the BFG algorithm and a single-hidden layer, Figures 4 – 7 indicate that a single-hidden layer FFBANN with 5 to 8 nodes in the hidden layer can provide an “optimal” net architecture (using the E-K data set). In addition, the previous simulations seem to suggest that the modeler should use the rule suggested by Baum and Haussler (1997) to initially choose network size and then use the method of determining network size by West, Brockett and Golden (1997) to help prune unnecessary hidden nodes. Again, it must be emphasized that the determination of the number of hidden nodes is dependent upon the other issues discussed in section 4.3.

[38] The better approximation capabilities from adding an additional hidden layer arise due to an increase in the number of degrees of freedom as the number of nodes in the neural network increases. Furthermore, the additional hidden layers allow more nonlinear and disjoint functions to be approximated.

4.3.5 Type of Activation Function in the Hidden Layer

The activation functions in the input and output layers are readily determined. The identity function is used in the input layer by construction, and the logistic tends to be used in the output layer for neural networks modeling dichotomous choice data with a single output node. The question of what activation function to utilize in the hidden layer of a FFBANN is not readily evident. Given the discussion in section 3.2, the only guidance given by FFBANN approximation theorems is that the activation function should have the properties of a squashing function, i.e. the function should be nondecreasing, $\lim_{l \to \infty} F_l(l) = 1$ and $\lim_{l \to -\infty} F_l(l) = 0$, for $l = 1,2,\ldots,L-1$. West, Brockett and Golden (1997) note that the logistic activation function is usually chosen due to the fact that it has desirable mathematical and computational properties, which is also true of the hyperbolic tangent activation function.

Two activation functions were examined in this study, the logistic and hyperbolic tangent. A simulation was conducted examining the out-of-sample predictive capabilities of single-hidden layer FFBANNs with logistic and hyperbolic tangent activation functions in the hidden layer of each network and a logistic activation function in the output layer. Each network was batch trained using the BFG algorithm using the stopping rules specified in section 4.3.2. All data was scaled appropriately (see section 4.3.1). The within and out- of-sample prediction results for the simulations are presented in Figures 12 thru 15. A summary of the all the performance measures calculated for each network trained during the simulation is provided in Appendix B.

The results in Figures 12 thru 15 indicate that a single-hidden layer FFBANN with a hyperbolic-tangent activation function in the hidden layer provided better within

Figure 12: Simulation Results - Mean *MSE* using the Training Data Set

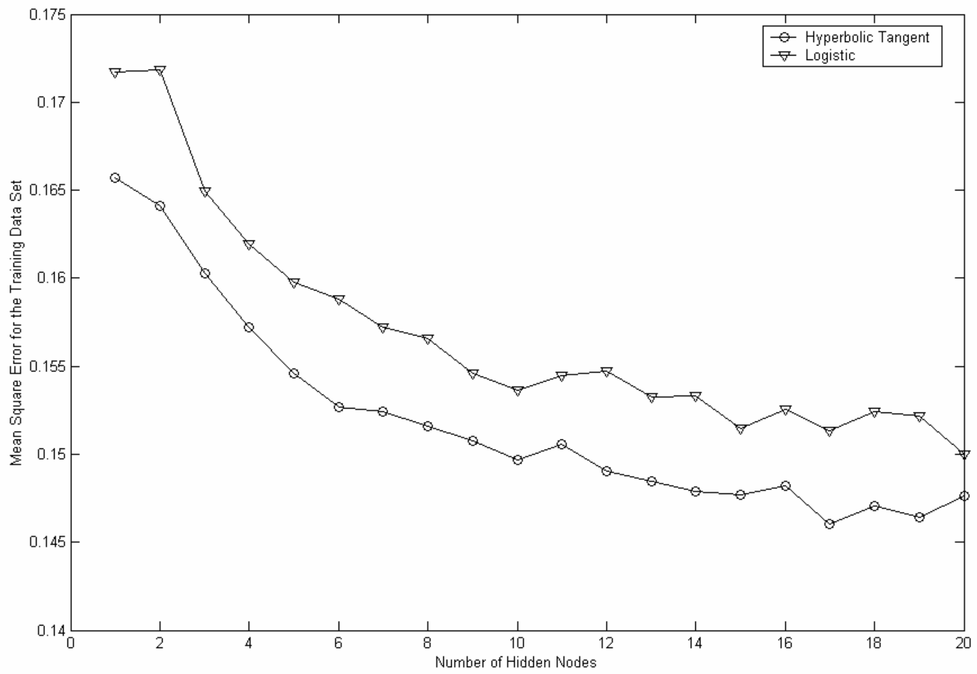


Figure 13: Simulation Results - Mean *MSFE* using the Test Data Set

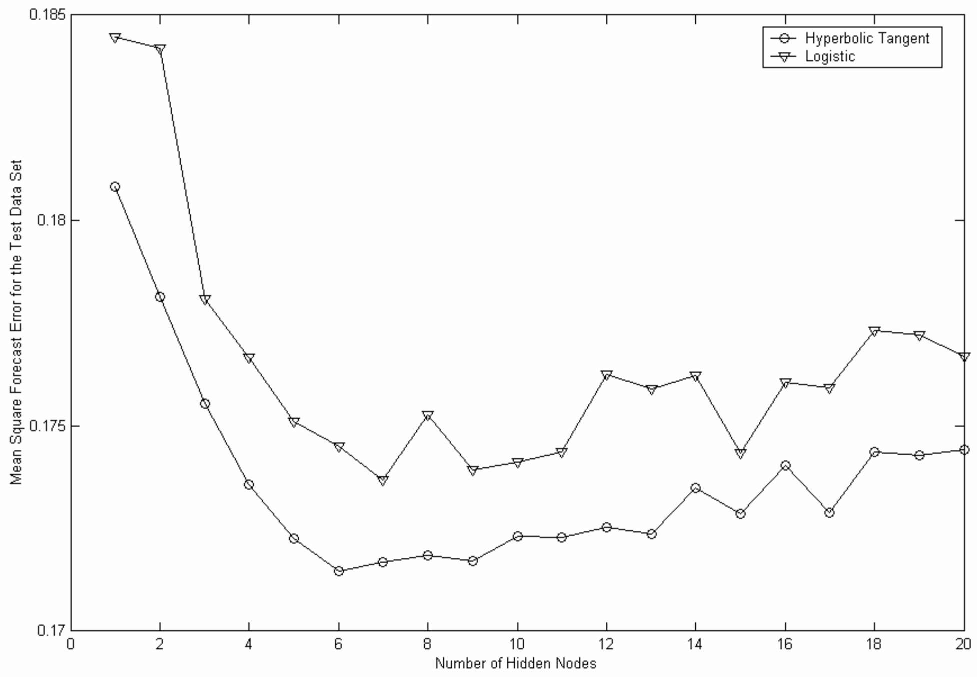


Figure 14: Simulation Results: Mean *PR* using Test Data Set

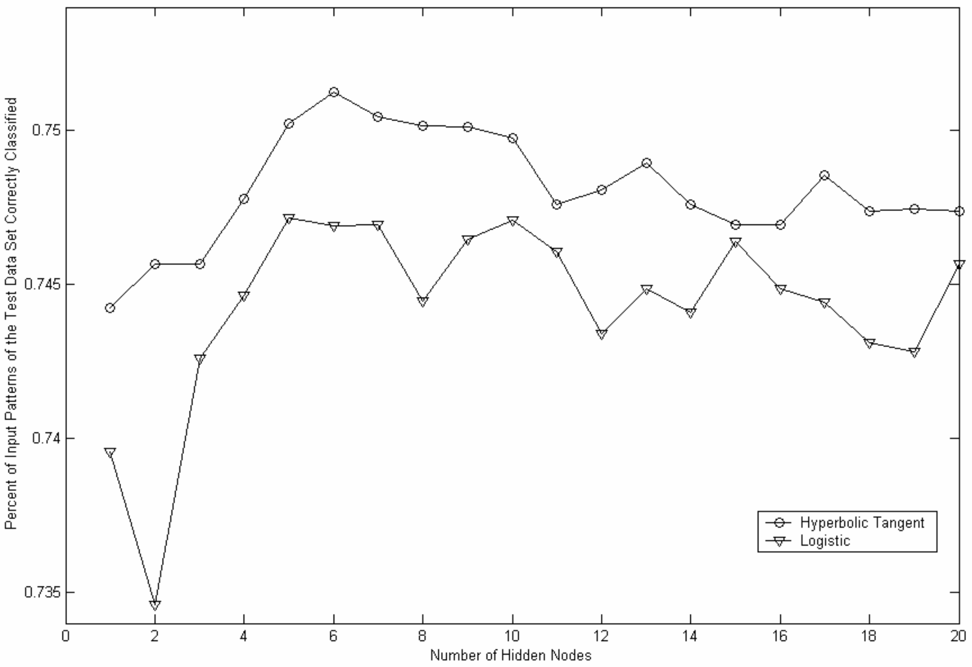
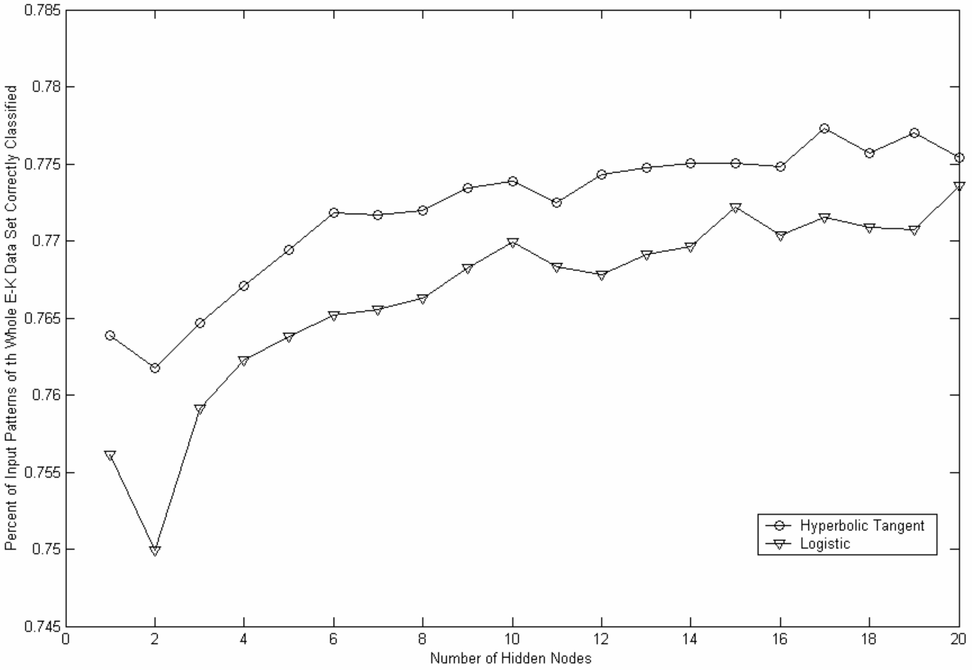


Figure 15: Simulation Results - Mean *PR* using the Entire E-K Data Set



and out-of-sample performance than a single-hidden layer FFBANN with logistic activation function in the hidden layer. It should be mentioned that the differences in the performance measures examined were not that significant. For example, the number of samples correctly classified of the test data set for each net architecture examined tended to only differ by as few as one or two input patterns out of 183 (the size of the test data set). This suggests that either activation function could be used, keeping in mind that the modeler should scale the data depending on their choice. For the remainder of the paper, the hyperbolic activation function was used in all net architectures.

4.3.6 Choice of Fitting Criterion

The fitting or loss criterion provides a measure of how close a neural network fits the data. The two fitting criterions, the *MSE* and Kullback-Leibler criterions, examined in section 3.2.3, are two of an infinite number of fitting criterions that could have been chosen (Fine, 1999). To narrow the field of choices, two desirable properties of any fitting criterion should be that (i) the fitting criterion's range should be restricted to $[0, \infty)$ (Mehrotra, Moran and Ranka, 1997) and (ii) the fitting criterion should be a second-order differentiable function. The first property ensures that a (global) minimum to the fitting criterion exists. The second allows for the development of the algorithms examined in section 4.3.2 (Ripley, 1996). Two common choices are the *MSE* and Kullback-Leibler fitting criteria given by equations (39) and (40).

The appeal of the *MSE* criterion is that it leads to a linear optimization problem when the neural network is linear, provides for a probabilistic interpretation of the output of the neural network, and the criterion is relatively easy to implement, since the instantaneous error is used for backpropagation (Principe, Euliano and Lefebvre, 1999).

The Kullback-Leibler criterion is of interest in that training amounts to estimating the weights using maximum likelihood, which provides a direct link for statistical interpretation of the network, even though this is not ruled out by the *MSE* criterion (see section 3.3.3). In this subsection, the *MSE* criterion is used.

If the objective of training the network is generalizability to new data, then a second related issue that the modeler must contend with once the fitting criterion is chosen is whether to perform regularization or strictly use cross-validation. Demuth and Beale (2001) found that regularization tended to provide better results for function approximation problems, but not necessarily for classification problems. Furthermore, it is not usually recommended in the literature that both be done at the same time, since regularization usually requires a larger number of epochs before converging to an optimal solution, whereas cross-validation techniques, which use stopping rules (S5) and/or (S6), tend to converge much faster (Demuth and Beale, 2001). Thus, these results would suggest that generalization tends to be poorer when both methods are used simultaneously since the training algorithm could prematurely terminate at a potentially non-optimal solution.

The primary advantage of only using regularization is that it tends to smooth out the fitting criterion, making it easier to find an optimal solution by restricting the variability of the weight changes via the addition of a penalty term in the fitting criterion. The two primary disadvantages of regularization are the increased training times required to converge to an optimal solution and the difficulty in determining the regularization constant. A method based on viewing FFBANNs from a Bayesian perspective proposes treating the regularization constant as a random variable and then using its prior distribution in maximizing the log likelihood function (or Kullback-Leibler criterion). Since this method automatically treats the regularization constant as a variable, it is determined by the algorithm used for training the neural network (MacKay, 1992; Ripley, 1996). Due to the implications of such an interpretation of the network using this approach and the frequentist approach toward probability adopted in this paper, a fixed value for the regularization constant is used for the fitting criterions incorporating regularization.

A simulation was conducted comparing which method, regularization, cross-validation (using stopping rule (S5)) or using both simultaneously, would provide better out-of-sample predictive accuracy using the test data set. The regularization constant in MATLAB was determined by g , a performance ratio, which essentially acts the same as the regularization constants usually found in the literature. The fitting criterion used with regularization was:

$$E = gE_{MSE} + (1-g)\frac{1}{|w|}\sum_{w_i \in w} w_i^2 \ , \tag{66}$$

where g is set to 0.5 and $|w|$ is the cardinality of the connection weight vector of the FFBANN being examined (Demuth and Beale, 2001). To perform the simulation, single-hidden layer FFBANNs with one to twenty nodes in the hidden layer were trained using regularization, cross-validation and both simultaneously for 500 randomly generated data partitions using the E-K data set. Each network was batch trained using the BFG algorithm with the hyperbolic tangent activation function in the hidden layer using the stopping rules specified in section 4.3.2. For the runs using only regularization, the stopping rule (S5) is excluded (i.e. cross-validation is not performed in these cases). The results of the simulations are presented in Figures 16 thru 19 and in Appendix B.

Figure 16 shows the fit of the FFBANNs based on the mean *MSE* after training was terminated. It is clear from the figure that when regularization is used instead of cross-validation, the mean square error after training is lower, meaning that the FFBANNs using equation (66) for a fitting criterion provided a superior "fit" to the training data set. In addition, according to Figures 17 and 18, the FFBANNs using regularization tended to generalize well, in terms of the mean *MSFE* and *PR* measures for the test data set when there were 5 or less neurons in the hidden layer. This result was expected, given that more complex networks are penalized for having a large number of connection weights, thereby increasing training time and decreasing generalization.

Figure 16: Simulation Results - Mean *MSE* using the Training Data Set

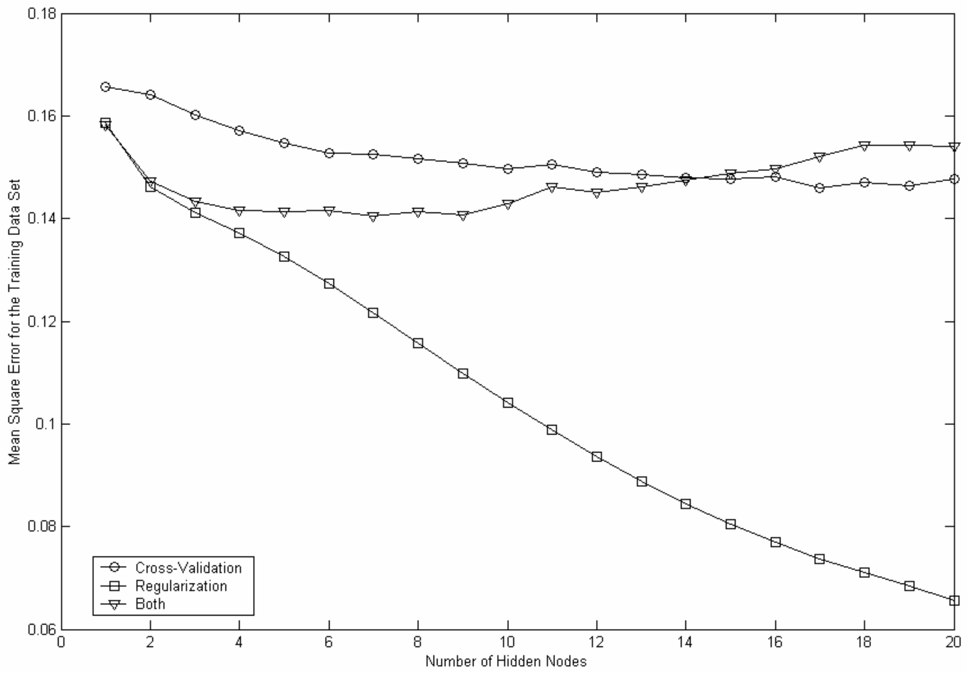


Figure 17: Simulation Results - Mean *MSFE* using the Test Data Set

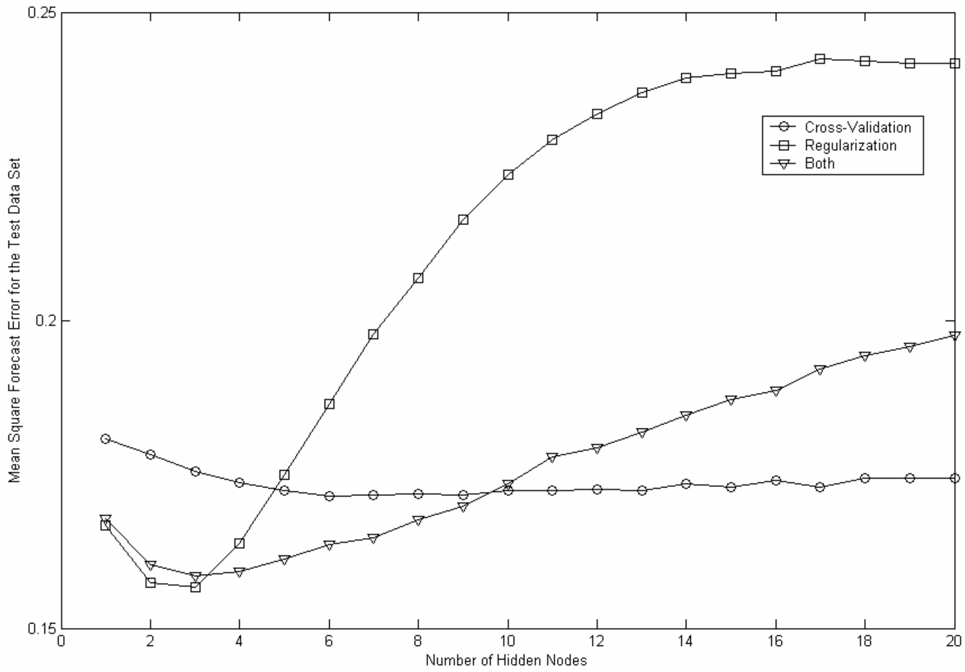


Figure 18: Simulation Results: Mean *PR* using Test Data Set

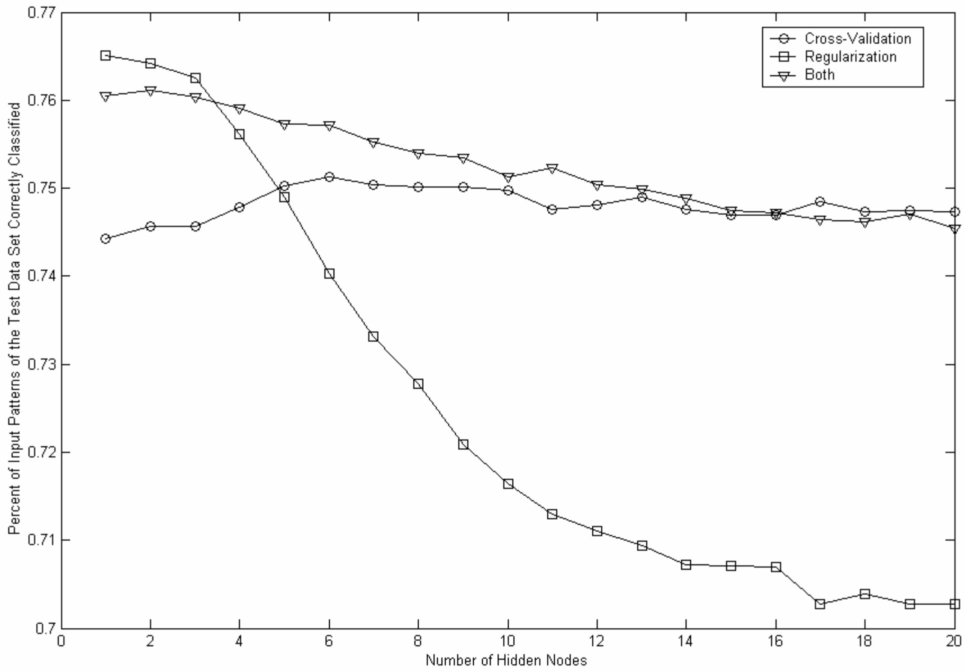
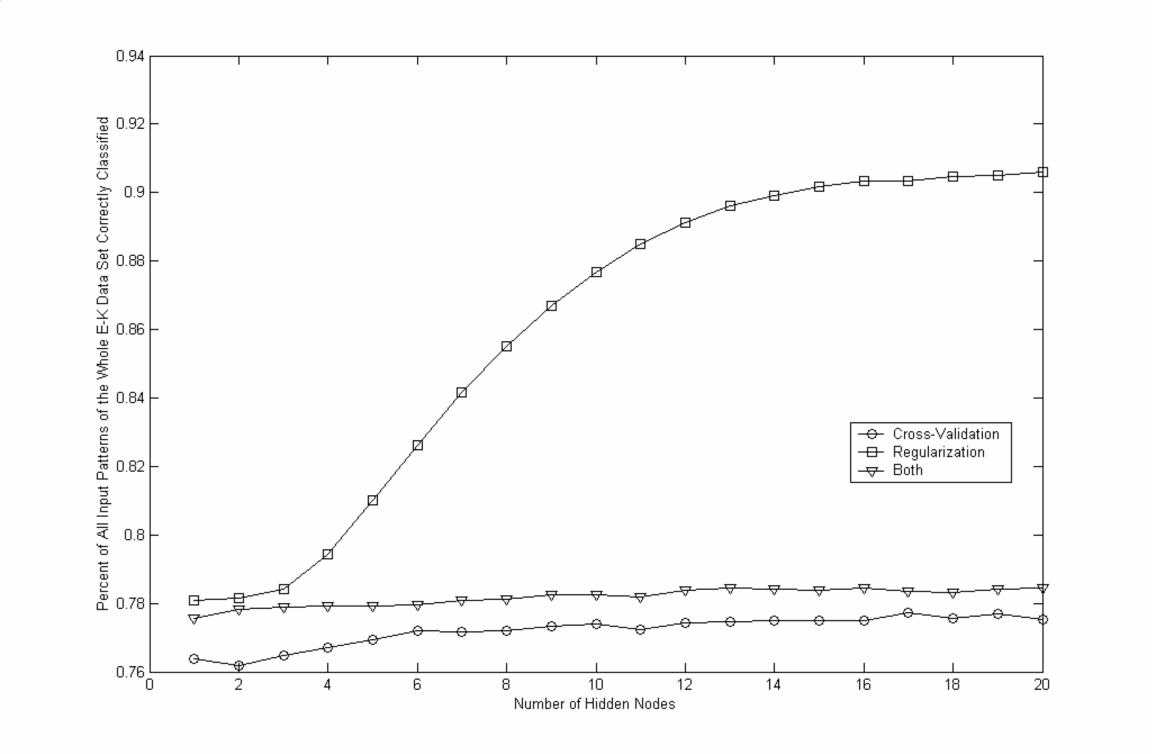


Figure 19: Simulation Results - Mean *PR* using the Entire E-K Data Set



Thus, do not let the mean *PR* results for the whole E-K data set in Figure 19 fool you, since the percent of training input patterns correctly classified is close to 100 percent for the larger sized networks trained using regularization.

When both regularization and cross-validation are used (which is not recommended in the literature) Figure 17 suggests that networks with less than nine hidden nodes in the hidden layer performed as well as or better than networks using regularization. This conjuecture is further supported by Figure 18 for the FFBANNs using both regularization and cross validation with more than 9 nodes in the hidden layer. Thus, contrary to recommendations in the literature, the results of this simulation seem to imply that the use of regularization will prune unnecessary hidden nodes (due to the penalty added in equation (4-3)) and in conjunction with a cross-validation stopping rule will help sustain peak generalizability. If the modeler is unsure about using both, then it is recommended that the modeler choose cross-validation before attempting to use regularization, based on the results in Figures 17 and 18.

Ripley (1994) states that cross-validation can sometimes stop too soon, thereby not obtaining a local (or global) minimum. The modeler has two avenues to help them avoid this situation. First the modeler can increase v in stopping rules (S5) and (S6) to help ensure that an optimal solution has been obtained. Second, the modeler might consider using regularization, in which case the modeler needs to make sure to set the maximum number of epochs allowed for training high enough to achieve convergence to a (local) minimum. In addition, the modeler should also be aware of the increase in training time that will be required by a training algorithm with regularization.[39] For example, when using cross-validation and the E-K data set about 5,000 FFBANNs could be trained in a 24 hour period (1 day), compared to about 100 FFBANNs using regularization.

4.3.7 Choice of Stopping Rule

The stopping rules (S2), (S3) and (S4) are standard rules used in a number of computer packages for terminating training and all were used when training the FFBANNs examined in this paper. The primary interest of this section of the paper is to examine the stopping rules (S5) and (S6). The cross-validation stopping rule used by MATLAB is (S5) with $v = 5$.

[39] In this study, the maximum number of epochs allowed during training when regularization was used was 1,000. Very few networks achieved any of the other termination criteria used before hitting the iteration limit when training with regularization. To ensure a (local) minimum is obtained, it is recommended that the iteration limit be set at least at 5,000, which would only allow about 10 FFBANNs to be trained using regularization in a given day (depending upon the speed of the computer – a 450 MHz machine was used to train all networks in this study).

No formal simulation was conducted comparing these stopping rules, but experience suggests that rule (S5) provides for better generalization than rule (S6). A separate simulation examining the optimal choice for v when using stopping rule (S5) found that when v was increased above 5, within and out-of-sample performance of the FFBANNs trained did not improve. When stopping rule (S6) was used, training tended to end too prematurely for many neural networks, resulting in a very poor fit to the training data set and lower *PR* scores for the whole E-K data set. To alleviate this problem, it is suggested that $v \geq 10$ when stopping rule (S6) is used.

4.3.8 Rules of Thumb for Constructing and Training a FFBANN.

From the simulations conducted in the preceding six subsections, a number of guidelines or "rules of thumb" can be learned for constructing FFBANNs using dichotomous choice data. These guidelines are only that, "rules of thumb." It may be the case that when constructing and training a FFBANN, these guidelines do not hold at all. It will take a significant number of studies in a large number of fields before some "hard" guidelines for the construction of a FFBANN are obtained. Thus, the author encourages further exploration of using FFBANNs in the field of economics, especially for modeling dichotomous choice data.

The guidelines or "rules of thumb" obtained from the simulations performed here are as follows:

1. When training FFBANNs with a reasonable number of weights (around 1,000) use the BFGS Quasi-Newton algorithm for training. For larger networks use a conjugate gradient algorithm or a variant of the Levenberg-Marquardt algorithm (such as the SCG algorithm discussed in section 4.2.2).

2. Try to observe the principle of Occam’s Razor and keep the net architecture of the FFBANN to as few layers and nodes as is needed to obtain “optimal” results. Use single-hidden layer FFBANNs over two-hidden layer FFBANN when possible. Make sure the number of hidden nodes in any given layer is less than or equal to the number of nodes in the preceding layer.
3. Use the hyperbolic tangent activation function over the logistic activation function (unless all of your input variables are binary), and be sure to scale the values of all input patterns to the ranges of the respective activation functions in the first hidden layer of the neural network.
4. For small networks, regularization may be performed, but use one of the cross-validation stopping rules (rules (S5) or (S6)) presented in the paper in order to ensure that the network generalizes to new data. For larger networks it is recommended that only one of the cross-validation stopping rules be used. In either case, make sure that *v* is set high enough for a local minimum to be obtained.
5. The modeler should always train a number of (different) neural networks during the construction process to obtain the best FFBANN for the performance criteria being considered. Thus, it is recommended that a sampling reuse procedure, as well as, a re-initialization procedure (that trains the network using different initial connection weight values) be used to train all neural networks being considered. For the best results it is recommended that the sample reuse procedure be applied first and then using the “optimal” data partitions, perform the re-initialization

procedure to find the FFBANN that provides the best results for the desired performance criteria (see subsection 4.4).

Ferret (1993) provides a good summary of some of the other guidelines found in the literature. All of the above guidelines where considered when determining the nine FFBANNs that would be used in the comparative analyses found in the next section of the paper.

4.4 Optimal Networks Used for Comparative Analyses

Based on the commentary in section 4.3.1, the FFBANNs chosen to be compared with the binary logit and probit models were those with the best out-of-sample predictive accuracy, as measured by the performance measures described in section 4.2. Of the 288,000 FFBANNs trained during simulations, the nine net architectures choosen (some with different training schemes) were those with the minimum *MSFE* or maximum *PR* using the test and/or entire E-K data sets over all the random data partitions examined. These nine FFBANNs are presented in Table 2. All of the FFBANNS in the table used the logistic activation function in the output layer and stopping rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-15}$ and $MAX = 1{,}000$ during training.

As stated by West. Brockett and Golden (1997),

> "*a researcher interested in developing a predictive model would only be interested in finding the "best" training session. This can be accomplished by identifying the data partitioning that produced the largest percentage of correctly classified data when applied to its own testing subsample* (p. 378-9)."

Thus, the data partition that provided the best results on the largest number of networks trained during the simulations was used to retrain all of the networks specified in Table 2 using the criteria specified above. This optimal partition allows not only the neural

Table 2: FFBANNs Chosen For Comparison with Logit and Probit Models

FFBANN[1]	Algorithm used for training	# of Hidden Layers	# of Nodes in each Hidden Layer[2]	Activation Function used in Hidden Layers	Regularization, Cross-Validation or Both[3]
BFG 20-7-1	BFGS Quasi-Newton	1	7	Hyperbolic Tangent	Cross Validation
CGF 20-19-1	Conjugate Gradient with Fletcher – Reeves update	1	19	Hyperbolic Tangent	Cross Validation
LM 20-1-1	Levenberg-Marquardt	1	1	Hyperbolic Tangent	Cross Validation
SCG 20-12-1	Scaled Conjugate Gradient	1	12	Hyperbolic Tangent	Cross Validation
BFG/LOG 20-17 -1	BFGS Quasi-Newton	1	17	Logistic	Cross Validation
BFG/REG 20-20-1	BFGS Quasi-Newton	1	20	Hyperbolic Tangent	Regularization
BFG/REG/ VLD 20-6-1	BFGS Quasi-Newton	1	6	Hyperbolic Tangent	Both
BFG 20-18-17-1	BFGS Quasi-Newton	2	18/17	Hyperbolic Tangent	Cross Validation
BFG 20-11-19-1	BFGS Quasi-Newton	2	11/19	Hyperbolic Tangent	Cross Validation

[1] In the literature, the net architecture of a FFBANN is represented as (# of imput variables/nodes)- (# of nodes in hidden layer(s))-(# of nodes in the output layer).
[2] (Number of nodes in first hidden layer)/(Number of nodes in second hidden layer)
[3] Cross-validation is performed using stopping rule (S5).

networks to be compared to the estimated logit and probit models, but to each other as well. In addition, following the guidelines presented in section 4.3.8, a re-initialization procedure was used to ensure that consistent estimates for the connection weights were obtained for some local minimum (White, 1989). The re-initialization procedure amounted to re-training the networks specified in Table 2, 100 times (using the criterion specified earlier), each time re-initializing the weights using the procedure developed by

Nguyen and Widrow for each layer of the network. The connection weight values from the best training sessions (i.e. the sessions with the lowest *MSFE* using the test data set) for each network type specified in Table 2 (chosen based upon the measures presented in section 4.2) were used for the comparative analyses in section 5. The training results for the connection weights are not reported due to the large amount of space required to report them (for example, the model BFG 20-18-17-1 would require the reporting of 683 connection weight values), but these estimates are available from the author upon request.

4.5 Estimated Logit and Probit Models

For the comparative analyses conducted in section 5 of the paper, the two primary dichotomous CVMs used in the literature, the binary logit and probit, were estimated using the E-K data set. In order to obtain a valid comparison using the test data set, the E-K data set was partitioned into an estimation (or training) data set and a test data set based upon the “best” data partition discussion in subsection 4.3. The estimation data set accounted for 80 percent of the data, while the test data set contained the remaining twenty percent, (i.e. the estimation data set included the training and validation data sets used to train the FFBANNs).

To obtain a comparison between the traditional econometric approaches to modeling dichotomous choice CV data, the index functions (i.e. $h(x_i;q)$ of the logit and probit models were assumed to be linear. The reader should keep in mind that this assumption could be invalid and should be tested before any statistical inference using the model is made. A procedure written by the author in MATLAB was used to estimate the models. The procedure uses the traditional Newton-Raphson algorithm, which corresponds with logit and probit procedures in standard statistical packages (e.g. SAS).

Table 3: Estimation Results for the Logit and Probit Models using the E-K Data Set

	Logit Model		**Probit Model**	
Variable	**Coefficient**	**Standard Error**	**Coefficient**	**Standard Error**
Intercept	-0.5216	0.6733	-0.3181	0.3993
TAX	-0.6596	0.2224	-0.3772	0.1270
WPCONTROL	-0.1182	0.2131	-0.0694	0.1240
WTPAMT	-0.0111	0.0012	-0.0065	0.0007
USE	-0.2670	0.2326	-0.1625	0.1364
DRQUAL	0.6009	0.3627	0.3596	0.2126
OTHERUSE	0.6515	0.2341	0.3858	0.1368
EXIST	0.7670	0.2356	0.4459	0.1390
LIKELY	1.0121	0.2250	0.6018	0.1306
ITEM	0.3954	0.2186	0.2278	0.1282
ENVORG	0.9000	0.3421	0.5140	0.1951
QUALWORS	0.1738	0.1943	0.1147	0.1131
TAPGOOD	-0.3491	0.1940	-0.2013	0.1130
AGE	-0.0055	0.0066	-0.0033	0.0039
NEWAREA	0.4022	0.3222	0.2085	0.1855
UNIV	0.6893	0.2046	0.4190	0.1204
EDU	0.5249	0.2057	0.3124	0.1201
SEX	0.1189	0.2010	0.0817	0.1166
INCOME	0.0000	0.0000	0.0000	0.0000
STATE	-0.5580	0.2505	-0.3360	0.1461
DATELAG	0.0015	0.0045	0.0010	0.0027
Other Statistics				
Number of Iterations	7		6	
Log-Likelihood	-348.8254		-348.4396	
Likelihood	3.2139×10^{-152}		4.7270×10^{-152}	
Convergence[1]	5.5213×10^{-12}		1.3023×10^{-9}	

[1] The convergence criteria used terminated the estimation procedure when the maximum element of the direction of change (the inverse of the Hessian times the gradient of the objective function) was less than a specified amount, in this case 0.000001.

[2] The estimates here differ somewhat with those estimated by Eisenhecht and Kramer due to the use of a smaller training data set.

The procedure is available upon request from the author. The estimation results for the two models are presented in Table 3. The probit model with linear index function was the functional form estimated by Eisen-Hecht and Kramer (2002).

5 Comparative Analyses

The first sub-section sets-up the comparative analyses for the FFBANNs and binary logit and probit models presented in sections 4.4 and 4.5, while the second sub-section examines the results of the estimation of each of the models

and the calculated performance measures for each. Finally, some statistical inference results are reported in the third sub-section of the paper using the estimated FFBANNs.

5.1 Set-Up

The nine "best" FFBANNs presented in section 4.4 were compared to the dichotomous choice (or binary) logit and probit models estimated in section 4.5 using the E-K data set. This comparison is based on the out-of-sample and overall performance of the models using the *MSFE*, *PR* and Type-I and Type-II Error measures presented in section 4.2. For completeness, these statistics are presented for the training, validation, test and entire E-K data sets (when applicable).[40]

Given that these measures alone may not provide a clear comparison, the values of the *PR*, Type-I Error and Type-II Error measures for the FFBANN are compared to those for the logit and probit models, to see if the results are statistically different. Thus, the following sets of hypotheses are tested: (Since the measures are percentages, the corresponding measure for the FFBANN is indicated as p_i and for the logit/probit model as p_j.)

$$H_0 : p_i = p_j \text{ vs. } H_1 : p_i \neq p_j \ \forall \ i, j \ . \tag{67}$$

The *PR*, Type-I Error and Type-II Error measures are generated by comparing the predicted model response to the actual response made by the respondents. Assuming that

[40] For the BFG/REG 20-20-1 FFBANN and logit and probit models there are no measurement statistics to report for the validation data set, because the data set used for estimation of the parameters of the models included the validation data set as additional training patterns.

the observations of the E-K data set are independent, the vector of comparisons (i.e. the i^{th} element of the comparison vector takes a value of ‘1’ if the predicted response is the same as the respondent’s and ‘0’ otherwise) used to calculate each of the above measures is distributed binomial (n, p), where *n* is the size of the data set being examined and *p* is the value of the measure (Spanos, 1999). Thus, the set of hypotheses given by (67) is tested using a squared version of McNemar’s Binomial Test (Hollander and Wolfe, 1999). This test statistic was used due to its ability to take account of the dependence between the two measures being examined (i.e. the same data set is used to calculate the measure for both models being compared). The test statistic used takes the following functional form:

$$d^2 = \frac{(x_{12} - x_{21})^2}{x_{12} + x_{21}} \overset{H_0}{\underset{\propto}{\sim}} c^2(1), \qquad (68)$$

where x_{12} is the number of times the FFBANN being examined predicted correctly and the logit/probit model did not and x_{21} is the number of times the logit/probit model predicted correctly and the FFBANN being examined did not. One would reject the null hypothesis in (67) if $d^2 \geq c^2_{a,1}$, where a is the level of significance chosen by the modeler (Hollander and Wolfe, 1999). The test statistic given by equation (68) provides a way to compare the statistical accuracy of the FFBANNs and dichotomous choice models using the test and whole E-K data sets.

5.2 Results and Discussion

Within-sample and out-of-sample measures for the “optimal” FFBANNs, as well as, the logit and probit models using the E-K data set (and partitions) are provided in Table 4. Even though within-sample performance using the training and validation data sets is not used to compare out-of-sample performance, it is interesting to note that the logit and probit models do not over-fit the training data used for estimation. This observation is attributed to the fact that the index function is linear, smoothing out nonlinearities in the discriminant produced by these models. Furthermore, for statistical inference purposes a modeler wants to choose a FFBANN that performs well not only on the test data set, but on the training and validation data sets as well. This fact highlights the tradeoff between within-sample and out-of-sample performance that every modeler must consider when choosing models. Given that the primary focus of this paper is on out-of-sample predictive accuracy (or generalization), this sub-section of the paper focuses upon the out-of sample predictive measures. In the next sub-section, we will step back from this basis to consider using FFBANN for statistical inference purposes.

The measures in Table 4 indicate that on average the FFBANNs perform relatively better than the binary logit and probit models on the test and whole E-K data, but the differences are not that significant. The BFG 20-18-17-1 FFBANN provided the best out-of-sample predictive accuracy (87.98 percent), while the BFG/REG 20-20-1 FFBANN provided the overall best predictive accuracy (95.41 percent) on the entire E-K data set, but the latter result is somewhat misleading. As seen in section 4.3.6, the out-of sample predictive accuracy for the BFG/REG 20-20-1 FFBANN is considerably lower than the other FFBANNs examined, due to the fact that during training this FFBANN was overfit to the training data set (evidenced by the high within-sample *PR* measure of 0.9945).

Table 4: Within-Sample and Out-of-Sample Performance of the FFBANNs and Binary Logit and Probit Models: *MSE*, *MSFE* and *PR*

Measure	**Model**[1]										
	BFG 20-7-1	CGF 20-19-1	LM 20-1-1	SCG 20-12-1	BFG/LOG 20-17-1	BFG/REG 20-20-1	BFG/REG VLD 20-6-1	BFG 20-18-17-1	BFG 20-11-19-1	Logit Model	Probit Model
Training Data set											
PR[2]	0.7723	0.8015	0.7705	0.7887	0.7668	0.9945	0.7760	0.7741	0.8069	0.7650	0.7609
Type-I Error[3]	0.0747	0.0674	0.0820	0.0729	0.0856	0.0014	0.0729	0.0638	0.0601	0.0902	0.0915
Type-II Error[4]	0.1530	0.1311	0.1475	0.1384	0.1475	0.0041	0.1512	0.1621	0.1330	0.1448	0.1475
MSE[5]	0.1534	0.1364	0.1575	0.1551	0.1561	0.0702	0.1465	0.1505	0.1390	0.2350	0.2391
Validation Data Set[6]											
PR	0.7541	0.7705	0.7760	0.7432	0.7650	N/A	0.7705	0.7432	0.7705	N/A	N/A
Type-I Error	0.0984	0.1038	0.0929	0.1202	0.0984	N/A	0.0929	0.1148	0.0929	N/A	N/A
Type-II Error	0.1475	0.1257	0.1311	0.1366	0.1366	N/A	0.1366	0.1421	0.1366	N/A	N/A
MSFE	0.1625	0.1663	0.1616	0.1711	0.1644	N/A	0.1522	0.1800	0.1867	N/A	N/A
Test Data Set											
PR	0.8579	0.8689	0.8689	0.8579	0.8579	0.7923	0.8634	0.8798	0.8525	0.8470	0.8470
Type-I Error	0.0601	0.0656	0.0601	0.0656	0.0656	0.1257	0.0601	0.0492	0.0820	0.0710	0.0710
Type-II Error	0.820	0.0656	0.0710	0.0765	0.0765	0.0820	0.0765	0.0710	0.0656	0.0820	0.0820
MSFE	0.1213	0.1311	0.1236	0.1192	0.1262	0.2076	0.1198	0.1193	0.1343	0.1530	0.1530
Entire E-K Data Set											
PR	0.7858	0.8087	0.7913	0.7934	0.7847	0.9541	0.7923	0.7891	0.8087	0.7814	0.7781
Type-I	0.0765	0.0743	0.0798	0.0809	0.0842	0.0262	0.0743	0.0710	0.0710	0.0863	0.0874

Error										
Type-II Error	0.1377	0.1169	0.1290	0.1257	0.1311	.0197	0.1333	0.1399	0.1202	0.1322	0.1344
MSFE	0.1488	0.1413	0.1515	0.1511	0.1518	0.0977	0.1423	0.1502	0.1476	0.2186	0.2219

[1] The definitions of each model are provided in Table 2.

[2] Percent Correctly Classified

[3] A Type-I Error occurs when a respondent is classified as not voting for the management plan when they actually did.

[4] A Type-II Error occurs when a respondent is classified as voting for the management plane when they actually did not.

[5] Mean Square (Forecast) Error

[6] For the BFG/REG 20-20-1 FFBANN and logit and probit models there are no measurement statistics to report for the validation data set, because the training data set include the validation data set as additional training patters

Comparing the *MSFE* for the test and entire E-K data sets reveals that overall the FFBANNs tend to fit the data better than the traditional logit and probit models with a linear index function. This result is explained by the fact that a FFBANN can formulate highly nonlinear, and even disjoint discriminants in the input variable space (Principe, Euliano, Lefebvre, 2000). Furthermore, a FFBANN provides a flexible function form for dichotomous choice models. The improved fit from using a FFBANN is even more apparent when the *MSE* and *MSFE* are examined for the training and validation sets in Table 4. In contrast, the *PR*, Type-I Error and Type-II Error measures in Table 4 indicate otherwise. These other measures suggest that the *MSFE* may not be an independent and completely reliable measure for comparing out-of-sample predictive accuracy. It could be the case that the low *MSE* and *MSFE* measures for the FFBANNs indicate that such networks are overfitted to the training data set, thereby resulting in lower values for the other performance measures examined on the test data set. Thus, how can the *MSFE* be useful in these comparisons? The measure provides an idea of how well the FFBANN acts as a discriminant for classifying the data. The lower the *MSFE* the "tighter" fitting the discriminant is to the data. If the "true" discriminant needs to be identified, then the *MSFE* provides a mechanism for determining which model provides the closest approximation to the "true" discriminant. In this paper, the FFBANNs provide closer approximations.

To further validate the above results, consider a naïve model, where the responses for the 183 respondents to whether or not they would vote for the management plan proposed by Eisen-Hecht and Kramer (2002) are predicted by using the responses (output targets) from the remainder of the data set (training and validation data sets). Each prediction is performed by randomly drawing a response from the remainder of the data set using a uniform random number generator and sampling by replacement. This can be thought of as a forecaster trying to naively predict the response of a non-respondent by randomly selecting the response of one of the 915 responses from the E-K data set (Kastens and Featherstone, 1996). Using the test data set for the naïve model, the *PR* measure was 0.5355, the Type-I Error measure was 0.2842 and Type-II Error measure was 0.1803 respectively. Thus, all of the above models perform better than the naïve model, providing justification for using the models used in the above comparisons.[41]

The results of the binomial tests for the sets of hypotheses given by (67) are reported in Table 5. For out-of-sample predictive accuracy on the test data set the BFG 20-18-17-1 FFBANN performed better statistically (at a 0.10 level of significance) than the more traditional logit and probit models. Even though the difference between the *PR* statistics for the BFG/REG 20-20-1 FFBANN and logit and probit models was statistically different, the tests indicate that the BFG/REG 20-20-1 FFBANN performed worse statistically than the logit and probit models on the test data set (given the performance measures in Table 4), due to overfitting to the training data set. In contrast, when examining predictive performance on the entire E-K data set, the BFG/REG 20-20-1 FFBANN performed statistically better (at a 0.01 level of significance) than the logit and probit models. The CGF 20-19-1 and BFG 20-11-19-1 FFBANNs performed better statistically (at a 0.01 level of significance) on the entire E-K data set as well.

The test statistics comparing the Type-I Error and Type-II Error measures suggest that the FFBANNs in Table 2, except BFG/REG 20-20-1, are not better classifiers

[41] In essence, all the models pass a “common sense” test.

Table 5: McNemar’s (Chi-Square) Binomial Test Results for Comparing the Predictive Accuracy of FFBANNs to the Logit/Probit Models

Test		BFG	CGF	LM	SCG	BFG/LOG	BFG/REG	BFG/REG/VLD	BFG	BFG
$H_0 : p_i = p_j$		20-7-1	20-19-1	20-1-1	20-12-1	20-17-1	20-20-1	20-6-1	20-18-17-1	20-11-19-1
						Test Statistics				
Comparing:						***Test Data Set***				
PR	Logit	0.6667	2.0000	2.6667	0.4000	0.5000	3.3333	1.2857	3.0000	0.0667
	Probit	0.6667	2.0000	2.6667	0.4000	0.5000	3.3333	1.2857	3.0000	0.0667
Type-I	Logit	2.0000	0.2000	2.0000	0.1429	0.2000	5.0000	1.0000	2.6667	0.4000
Error	Probit	2.0000	0.2000	2.0000	0.1429	0.2000	5.0000	1.0000	2.6667	0.4000
Type-II	Logit	0.0000	3.0000	1.0000	0.3333	0.3333	0.0000	0.3333	0.6667	1.8000
Error	Probit	0.0000	3.0000	1.0000	0.3333	0.3333	0.0000	0.3333	0.6667	1.8000
Comparing:						***Entire E-K Data Set***				
PR	Logit	0.4211	8.5616	2.4545	1.6575	0.1200	124.8200	2.5000	0.5506	6.8681
	Probit	1.2564	10.8889	4.5000	2.5789	0.4737	127.6897	3.9302	1.1364	8.9091
Type-I	Logit	1.0000	0.6667	1.0000	0.6667	0.6667	12.0000	2.0000	0.1111	0.0000
Error	Probit	0.0000	1.8000	0.0000	0.1429	0.1429	13.0000	0.3333	0.5000	0.0769
Type-II	Logit	0.3333	0.5000	0.3333	0.0909	0.5000	32.0000	0.2000	0.5000	0.1111
Error	Probit	0.0000	1.2857	1.0000	0.3333	1.0000	33.0000	0.0000	0.1429	0.0000
						Critical Values				
Significance Level			**0.01**			**0.05**			**0.10**	
$H_1 : p_i \neq p_j$			6.6349			3.8415			2.7055	

statistically than the binary logit and probit models. The BFG/REG 20-20-1 FFBANN was superior to the logit and probit models when considering classification abilities over the entire E-K data set, but this result is somewhat misleading. This neural network was overfitted to a larger training data set, consisting of the training and validation data sets used for the other FFBANNs in Table 2, giving rise to higher performance statistics on the entire E-K data set. When regularization is used during training of a FFBANN, cross-validation techniques should be used to ensure that the out-of-sample predictive properties of the network remain desirable.

Overall, the FFBANNs provide marginally better out-of-sample predictive capabilities than the more traditional logit and probit models with linear index functions, but on average the differences between the performance measures used in the study were not statistically different. Furthermore, the networks chosen for these comparative analyses further strengthen the observation that the guidelines set forth in section 4.3.8 for constructing and training FFBANNs are not absolute. The fact that the BFG 20-11-19-1 FFBANN performed statistically better than the estimated logit and probit models on a number of the performance measures reinforces this point. For example, the BFG 20-11-19-1 FFBANN has two-hidden layers, with more hidden nodes in the second layer than in the first, which goes against the second guideline suggested in section 4.3.8. The above results underpin the problem-dependent nature of using FFBANNs for modeling dichotomous choice data.

Given that any statistical inferences made using the logit and probit models is dependent upon the statistical validity of the functional form assumptions, a semi-nonparametric approach, such as a FFBANN, maybe more desirable than traditional statistical techniques when the functional form assumption of the model is in question. In particular, statistical inferences concerning statistical significance and WTP are dependent upon the functional form assumption of the model, and using a flexible functional form can help provide a means of potentially avoiding model misspecifications, especially given the statistical interpretation of FFBANNs in section 3.3. The next subsection begins to explore the differences in the statistical inferences provided by a FFBANN and the binary logit and probit models.

5.3 Statistical Significance Hypothesis Tests Using A FFBANN

Theoretically, the statistical procedures for hypothesis testing presented in section 3.3.4 can be used to perform tests of statistical significance using a FFBANN. For practical purposes, a number of issues arise that complicate the use of these procedures. This sub-section is a first attempt at addressing some of these problems using one of the FFBANNs examined in the previous subsection.

Keep in mind, when training a FFBANN the weights are randomly initialized to provide a starting point for the training algorithm. If a bad starting point is generated, it can affect the training process by resulting in poor generalization and/or termination before obtaining a local minimum of the fitting criterion. In addition, the choice of the training data set used can result in similar problems.[42] Finally, Wang (1995) notes that for a given “training data set, the classification results generated by individual neural networks may differ in thousands of ways for no obvious reason (p. 556).” Thus, when considering the tests for the hypotheses:

$$H_0 : w_{k,m}^{(1)} = 0 \ \forall m \text{ against } H_1 : w_{k,m}^{(1)} \neq 0 \text{ for at least one } m, \qquad (69)$$

[42] This is the reason the author recommends using a sample reuse and reinitialization procedure for model construction and training.

the pseudo-F test and the Likelihood Ratio test given in section 3.3.4 could provide negative test statistics and/or wide-ranging values when different networks are trained using different data sets (partitions) and/or starting points.

For example, consider the BFG 20-7-1 FFBANN examined in the previous section. Imposing the restrictions $w_{1,m}^{(1)} = 0 \quad \forall m$ amounts to testing the significance of the first input in the neural network. Calculating a pseudo-F test statistic requires that the modeler re-train the BFG 20-7-1 neural network imposing the above restrictions. In accomplishing this task, the modeler trains a BFG 19-7-1 FFBANN (by dropping the first variable) using a randomly generated starting point and obtains a *MSE* for the training data set lower than that obtained for the BFG 20-7-1 FFBANN, thereby giving rise to a negative pseudo-F statistic.

For statistical inference purposes, the modeler wants to ensure two things: (i) that a local minimum of the fitting criterion is obtained and (ii) that the model is balanced, i.e. the FFBANN trained has desirable within-sample and out-of-sample performance capabilities. At times the modeler is not able to meet both criteria satisfactorily due to time and computer memory costs.[43] The procedure presented here is designed to help meet both of the above goals, while at the same time taking into consideration the costs born by the modeler for conducting such inferences. The procedure uses the sampling re-use and reinitialization procedures used for the simulations in sections 4.3 and 4.4.

The following test procedure is a bootstrapping procedure designed to take account of the variability in the training results of FFBANNs and to find a practical way to achieve the goals presented in the preceding paragraph. The procedure is presented for

[43] These costs include the time it takes to conduct the tests, which could be one plus days, and the amount of computer memory required to conduct the tests, i.e. the modeler needs to ensure that the computer running the testing program is adequate for the task being asked of it.

obtaining a pseudo-F test statistic for the hypotheses given by (69) and is easily adapted for obtaining a Likelihood Ratio test statistic. The procedure is as follows:

Step 1: Using the sample reuse and reinitialization procedures presented in sections 4.3 and 4.4, re-train the FFBANN being examined $N \cdot S$ times, where *N* is the number of randomly generated sample partitions used and S is the number of random starting points used for each training session with a different sample partition.[44] Using the *MSE* on the training data sets for the *N* “best” (i.e. over *S*) training sessions (one for each sample partition generated), based upon criteria set by the modeler (see section 4.4 for the criteria used in this study), calculate the mean *MSE,* call it $\overline{MSEU}$.

Step 2: Impose the restrictions given in hypothesis (69) for the input to be examined and repeat the same procedure given in Step 1 with the same randomly generated sample partitions using the restricted FFBANN. Again calculate the mean *MSE* on the training data set for the “best” *N* training sessions, call it $\overline{MSER}$.

Step 3: Using $\overline{MSEU}$ and $\overline{MSER}$ calculate the pseudo-F statistic via:

$$F(X_i) = \left(\overline{T} - |w|\right)\frac{\overline{MSER} - \overline{MSEU}}{\overline{MSEU}} \overset{H_0}{\underset{\propto}{\sim}} c^2(m), \qquad (70)$$

where $|w|$ is the number of weights estimated in the FFBANNs trained in Step 1.

Letting $N = 1{,}000$, the above procedure was utilized to conduct significance tests on the inputs of the BFG 20-7-1 FFBANN. This network was chosen for statistical inference purposes based upon the guidelines given in section 4. The results for the pseudo-F statistics and corresponding Likelihood Ratio statistics for the logit and probit models are provided in Table 6. The pseudo-F test statistics for the variables EXIST and

[44] Based upon simulation results, for the reinitialization procedure it is recommended that at least five or more randomly generated starting points be used for each network trained. Five were used in this study.

Table 6: Test Results for the Statistical Significance of the Input Variables

Variable[1]		**Model**	
	BFG 20-7-1	**Logit Model**	**Probit Model**
	Psuedo-F Statistic[2]	**Likelihood Ratio**[2]	**Likelihood Ratio**[2]
Intercept	---[3]	7.6886	8.2408
		[0.0056]	[0.0041]
TAX	8.9262	9.1273	9.0180
	[0.2580]	[0.0025]	[0.0027]
WPCONTROL	1.1818	0.3087	0.3140
	[0.9913]	[0.5785]	[0.5752]
WTPAMT	7.6691	108.9037	108.4253
	[0.3627]	[0.0000]	[0.0000]
USE	0.3537	1.3299	1.4267
	[0.9998]	[0.2488]	[0.2323]
DRQUAL	102.7782	2.7160	2.8552
	[0.0000]	[0.1402]	[0.0911]
OTHERUSE	0.6066	7.8098	7.9818
	[0.9990]	[0.0052]	[0.0047]
EXIST	-0.7896	10.6231	10.2956
	[1.0000]	[0.0011]	[0.0013]
LIKELY	5.5457	20.5312	21.3844
	[0.5937]	[0.0000]	[0.0000]
ITEM	13.4043	3.2612	3.1502
	[0.0628]	[0.0709]	[0.0759]
ENVORG	20.1238	7.5550	7.3607
	[0.0053]	[0.0060]	[0.0067]
QUALWORS	0.4517	0.8004	1.0297
	[0.9996]	[0.3710]	[0.3102]
TAPGOOD	4.0197	3.2511	3.1773
	[0.7775]	[0.0714]	[0.0749]
AGE	1.0416	0.6792	0.7391
	[0.9941]	[0.4099]	[0.3899]
NEWAREA	2.9623	1.5923	1.2778
	[0.8885]	[0.2070]	[0.2583]
UNIV	0.6297	11.3573	12.1199
	[0.9988]	[0.0007]	[0.0005]
EDU	0.9010	6.5697	6.8094
	[0.9963]	[0.0104]	[0.0091]
SEX	6.9701	0.3499	0.4903
	[0.4320]	[0.5542]	[0.4838]
INCOME	1.9555	6.7133	6.4720
	[0.9623]	[0.0096]	[0.0110]
STATE	-0.7746	5.1390	5.416
	[1.0000]	[0.0234]	[0.0199]
DATELAG	12.4279	0.1110	0.1324
	[0.0873]	[0.7390]	[0.7160]

[1] Variable definitions are provided in Table 1.
[2] The number in square brackets is the associated p-value.
[3] The significance of the bias term in the BFG 20-7-1 FFBANN was not examined due to the inability to easily eliminate the bias from the network architecture between the input and first hidden layer of the network using the procedures provided in the Neural Networks Toolbox in MATLAB.

STATE were negative. As mentioned above this type of behavior would exist if the FFBANN provided a better fit when these variables were excluded from the model. Thus, the modeler can view a negative pseudo-F test statistic as verification that the contribution of the excluded variable in helping to determine the “true” discriminant is relatively insignificant. Any testing results of this kind should be approached with some caution, given the somewhat unpredictable nature of network results during training, i.e. the testing results for a FFBANN are dependent upon the starting points and data partitions used in the proposed testing procedure.

The test results for the BFG 20-7-1 FFBANN are quite different from the test results for the binary logit and probit models. As seen in Table 6, the only statistically significant variables for the BFG 20-7-1 FFBANN were DRQUAL, ITEM, ENVORG and DATELAG. Furthermore, according to the test results for the BFG 20-7-1 FFBANN, WTP was not a decisive factor in the determination of a respondent’s decision to vote for the water quality management plan in the Catawba River basin. This result is in stark contrast to the result obtained using the binary logit and probit models, which indicate that WTP was a decisive factor in determining how a respondent would vote for the management plan. Again, it should be stressed that the above procedure provides a mechanism to help determine the statistical contribution of each input variable in predicting a respondent’s voting behavior and that the statistical properties of the testing procedure need to be explored further.

6. Estimating Median Willingness to Pay\Willingness to Accept using FFBANNs

This section of the paper examines how one can estimate median WTP (or WTA) measures using a FFBANN. The focus here is on median WTP, but the method examined is applicable for estimating median WTA as well, with just a few minor adjustments to the algorithm presented in section 6.2. The first sub-section provides the theoretical background for the algorithm used to estimate median WTP, which is then presented in the second sub-section. The final sub-section will compare the median WTP estimates using the “optimal” FFBANNs presented in section 4.4 and to the mean WTP estimates obtained using the binary logit and probit models estimated in section 4.5.

6.1 Theoretical Background

One of three approaches presented by Hanemann (1984) for estimating an individual’s minimum WTP for an environmental amenity is to determine the amount of money the individual would have to pay to just make them indifferent between sustaining or not sustaining that amenity. In the case of Eisen-Hecht and Kramer (2002), this amounts to finding the minimum amount a respondent would pay that would make them indifferent between voting for or against the proposed management plan to sustain the current level of water quality in the Catawba River Basin. Using the framework presented in section 2.1, this indifference can be interpreted as the level of *C* that makes:

$$p = \mathbf{P}\big(V_1(q_1, y - C, s, e_1) \geq V_0(q_0, y, s, e_0)\big) = 0.5\,. \qquad (71)$$

The respondent is indifferent when there is a 50:50 chance that the respondent will vote for sustaining the environmental amenity. Solving equation (71) for C, gives C_p , the median WTP for the environmental amenity (Hanemann, 1984).

When using the logit and probit models, finding C_p can be done by setting:

$$\Delta v = v_1(q_1, y - C, s) - v_0(q_0, y, s) = 0, \tag{72}$$

and solving for C, since $F(0) = 0.5$ when $F(.)$ is either the logistic or standard normal cdf (Cooper, 2002). If ΔV is linear, then the mean and median WTP are equal (Hanemann, 1984). Finding C_p using a FFBANN is not so straightforward, due to the highly nonlinear and interconnected nature of FFBANNs. Thus, the problem must be solved numerically (Cooper, 2002). The modeler has two options, (i) use equation (71) to find C_p or (ii) if the FFBANN is only being used to approximate the index function, switch the output node to a linear output node and use equation (72). Option (i) amounts to performing a grid search using the FFBANN for that level of C that will make Y_i , the output of the FFBANN, equal to 0.50. Option (ii) requires that the activation function in the output node of the FFBANN satisfy the requirement of a cdf, and that $F(0) = 0.5$. A logistic or standard normal activation function in the output node would satisfy this requirement. In this case, the activation function in the output node is replaced with a linear output node and the resulting FFBANN is interpreted as an approximation to the index function. Since equation (72) amounts to setting the index function equal to zero, the modeler using option (2) would find the level of C that would make the altered FFBANN equal to zero. The next section presents an algorithm for determining the level of C using option (i). The algorithm can be readily altered for application to option (ii).

6.2 An Algorithm for Determining Median WTP

The following algorithm determines the median WTP for a group of individuals by first determining the median WTP for each individual and then by taking the mean over the members of the group. Thus, for each individual the only variable that is not fixed is C. In the case of the Eisen-Hecht and Kramer (2002) study, C represents the price of the management plan or the yearly amount of tax each respondent would be willing to pay to support the management plan.

The algorithm is a grid search along a closed interval of the real line. The search method is based on the golden section method presented by Bazaraa, Sherali and Shetty (1993), a line search procedure that does not use derivatives. The closed interval on the real line represents the search area for the algorithm or the upper and lower bound of WTP. For the E-K data, the upper and lower bounds were set at $0 and $250 respectively, since if a respondent doesn’t vote for the management plan their WTP is assumed to be $0, and if they do vote the highest bid amount offered was $250. The algorithm is summarized as follows:

Initialization: Determine the initial WTP interval $[a,b]$ on the real line. Choose a tolerance level, $g > 0$, which represents how close the algorithm needs to converge to the median level of WTP to terminate. Set $l = a + (1 - a)(b - a)$ and $m = a + a(b - a)$, where $a = 0.618$. For the i^{th} individual, fix the remaining explanatory variables or input values to their current level and calculate $Y_i(a)$, $Y_i(b)$, $Y_i(l)$ and $Y_i(m)$, where $Y_i(x)$ is the output of the FFBANN with $C = x$.

Main Step: For the i^{th} individual

1. If $Y_i(a) \leq 0.5$ or $Y_i(b) \geq 0.5$, then stop. The median WTP is $C_p = a$ or $C_p = b$ respectively. Otherwise, go to step 2.
2. If $|b-a| < g$, then stop. The optimal solution lies in the interval $[a,b]$. In this case let $C_p = 0.5 \cdot (b-a)$. Otherwise, if $Y_i(b) < 0.5$ and $Y_i(l) > 0.5$ go to step 3, if $Y_i(l) < 0.5$ and $Y_i(m) > 0.5$ go to step 4, or if $Y_i(m) < 0.5$ and $Y_i(a) > 0.5$ go to step 5.
3. Let $a = l$ and $Y_i(a) = Y_i(l)$. Recalculate $l, m, Y_i(l), Y_i(m)$ using the new interval $[a,b]$ with the formulas presented above and then return to step 2.
4. Let $a = m$, $b = l$, $Y_i(a) = Y_i(m)$ and $Y_i(b) = Y_i(l)$. Recalculate $l, m, Y_i(l), Y_i(m)$ using the new interval $[a,b]$ with the formulas presented above and then return to step 2.
5. Let $b = m$ and $Y_i(b) = Y_i(m)$. Recalculate $l, m, Y_i(l), Y_i(m)$ using the new interval $[a,b]$ with the formulas presented above and then return to step 2.

Do this for all the individuals to obtain a vector of median WTP values for all individuals. Once this vector has been obtained calculate the mean and standard deviation of the vector using standard statistical techniques to obtain the median WTP for the group.

The reader should note that when progressing through steps 2 thru 5 of the main step of the algorithm, the interval $[a,b]$ is updated after each iteration. For the median WTP estimates found in the next subsection, $g = 0.000001$.

6.3 Application and Results

Using the algorithm developed in the previous sub-section, the (mean) median WTP using the FFBANNs presented in section 4.4 were calculated. These results are presented in Table 7 along with the median WTP estimates obtained using the binary the logit and probit models. The median WTP found using the FFBANNs tends to be lower than the estimates obtained from the traditional binary logit and probit models using Hanemann's (1984) approach. The lower estimates provided by the FFBANNs could be the result of a number of different factors. First, the lower estimates could have resulted from the highly nonlinear nature of the FFBANN, i.e. the index function and therefore the utility difference, ΔV, are nonlinear functions. Remember, that a FFBANN can be interpreted as a flexible functional form for $E\left(R_i \mid X_i = x_i\right)$, allowing for a better approximation to the "true" conditional mean.[45] This result underscores the importance the functional form, ΔV, has in determining the median WTP, as stressed by Hanemann

Table 7: Median WTP Estimates

Model	Median WTP ($)	Standard Error ($)
BFG 20-7-1	164.84	70.69
CGF 20-19-1	155.65	64.48
LM 20-1-1	157.17	69.14
SCG 20-12-1	159.72	70.00
BFG/LOG 20-17-1	159.75	74.82
BFG/REG 20-20-1	163.25	80.55
BFG/REG/VLD 20-6-1	161.91	68.62
BFG 20-18-17-1	164.56	70.84
BFG 20-11-19-1	162.84	69.48
Logit Model[1]	186.98	110.01
Probit Model[1]	194.68	112.98

[1] The median WTP for the logit and probit model was found by solving equation (72) (following Hanemann (1984)) for each respondent and then taking the mean across the WTP estimates obtained for all of the respondents.

[45] This result holds true of course only in the case that the functional form assumption about the transformation function (i.e. the activation function in the output layer of the network) is correct.

(1984). Second, the parameters of the logit and probit models with linear index functions could be being influenced by a number of large values (i.e. outliers), which the FFBANNs are able to capture. Thus, one of the potential benefits of FFBANNs is that these models allow researchers to classify objects that may have a low probability of occurrence. Finally, the logit and probit models could be misspecified.[46] The modeler should make sure that the underlying functional form assumptions of the model are valid given the observed data. If the logit and probit models are misspecified, then the median WTP estimates provided by these models potentially biased upwards by as much as 15 percent.[47]

Eisen-Hecht and Kramer (2002) estimated the median WTP for their probit model, using the method suggested by Hanneman (1984), to be $198. The estimates for the logit and probit models obtained in Table 7 differ from this result, since the data set used to estimate these models consisted of eighty percent of the original E-K data set. In addition, they also estimated a lower bound for the mean WTP using a method developed by Turnbull (1976), which turned out to be $139. Following NOAA guidelines, they use the more conservative estimate in their study (Arrow et al., 1993). Applying the method developed by Turnball to a FFBANN is beyond the scope of this study and is suggested as an avenue for future research.

7. Concluding Remarks

The dichotomous choice CVM model as proposed by Hanemann (1984) is subject to potential functional form misspecification due to the imposition of an a priori theoretical interpretation of the model, without consideration of the statistical adequacy

[46] This fact does not rule out the FFBANNs being misspecified as well, but that is the topic of a future paper.
[47] Assuming the FFBANN are properly specified.

of the underlying probabilistic assumptions of the model. Kay and Little (1987) show that the traditional logit model with linear (in the parameters) index function arises under somewhat stringent conditions, putting into question many of the dichotomous choice CVMs used in the literature. Feed-forward backpropagation artificial neural networks (FFBANNs) provide a semi-nonparametric alternative to the traditional logit and probit models. These types of neural networks can act as universal approximators and provide a robust classification tool for out-of-sample prediction, allowing the modeler to potentially avoid problems arising from an incorrect functional form.

The statistical properties underlying FFBANNs need to be further developed. While the estimates (under certain regularity conditions) are found to be consistent and asymptotically normal, the statistical foundations for conducting statistical inference via hypothesis tests using FFBANN is at times uncertain and requires a considerable amount of future research. Furthermore, the construction and training of a FFBANN is problem dependent. For every guideline that exists in the literature, the unpredictability during training of a FFBANN ensures that an "optimal" neural network will be found that violates the given guidelines. Thus, all these guidelines act as "rules of thumb," requiring the modeler to perform some type of model search procedure to obtain an "optimal network" based upon the performance measures of interest to the modeler. As stated in the literature, to help direct these model search processes future research using FFBANN should be conducted in a wide range of fields, disciplines, and problem applications.

This paper showed that FFBANNs do provide an alternative to the more traditional logit and probit models with linear index functions. Direct comparisons between the models showed that the FFBANN performed marginally better than the logit and probit models for a number of within-sample and out-of-sample performance measures on a contingent valuation data set, but in the majority of cases these differences were not statistically significant. A significant issue that arose in the comparison, was the tradeoff between within-sample and out-of-sample performance. Regularization provided the best results when compared to the logit and probit models using the entire data sample (not just the test set), but the results were somewhat misleading due to the tendency of a FFBANN using regularization to overfit the training data set, resulting in poor generalization. Thus, the modeler must weigh the importance of both of these issues during model construction, and especially when considering conducting statistical inference using a FFBANN.

Future research concerning the use of FFBANN in the area of dichotomous choice models, should consider the modeling of nested logit and probit models. Such an approach would involve potentially nesting two or more FFBANNs, which brings in another set of additional modeling concerns that need to be considered, such as model interpretation, how to effectively nest FFBANNs and training methods.

References

1. Amemiya, T. “Qualitative Response Models: A Survey.” *Journal of Economic Literature*. 19(December 1981): 1483 – 1536.
2. An, M.Y. “A Semiparametric Distribution For Willingness to Pay and Statistical Inference with Dichotomous Choice Contingent Valuation Data.” *American Journal of Agricultural Economics*. 82(August 2000):487 – 500.
3. Arana, E., P. Delicado and L. Marti-Bonmati. “Validation Procedures in Radiological Diagnostic Models: Neural Networks and Logistic Regression.” *Investigative Radiology* 34(Oct. 1999): 636-642.
4. Arnold, B.C., E. Castillo and J.M. Sarabia. *Conditional Specification of Statistical Models*. New York: Springer Verlag, 1999.
5. Arnold, B.C. and S.J. Press. “Compatible Conditional Distributions.” *Journal of the American Statistical Association*. 84(March 1989): 152 – 156.
6. Arrow, K.R., R. Solow, P.R. Portney, E.F. Leamer, R. Radner and H. Schuman. “Report of the NOAA Panel on Contingent Valuation.” *Federal Register*. 58(January 15, 1993): 4601 – 4614.
7. Barron, A. “Universal Approximation Bounds for Superpositions of a Sigmoidal Function.” University of Illinois at Urbana – Champaign Department of Statistics Technical Report 58, 1991.
8. Baum, E. and D. Haussler. “What Size Net Gives Valid Generalization?” *Advances in Neural Information Processing Systems 1*. D. Touretzky editor. San Mateo, CA: Morgan Kaufman Pub., 1989. p. 81 – 90.

9. Bazaraa, M.S., H.D. Sherali and C.M. Shetty. *Nonlinear Programming: Theory and Algorithms*. 2nd Edition. New York: John Wiley and Sons, Inc., 1993. 638p.

10. Bierens, H. “A Consistent Conditional Moment Test of Functional Form.” *Econometrica*. 58(1990): 1443 – 1458.

11. Billingsley, P. *Probability and Measure*. 3rd edition. New York: J. Wiley and Sons. 1995.

12. Cooper, J.C. “Flexible Functional Form Estimation of Willingness to Pay Using Dichotomous Choice Data.” *Journal of Environmental Economics and Management*. 43(2002): 267-279.

13. Cosslett, S.R. “Distribution-Free Maximum Likelihood Estimator of the Binary Choice Model.” *Econometrica*. 51(May 1983): 765-782.

14. Cox, D.R. “The Regression Analysis of Binary Sequences.” *Journal of the Royal Statistical Society, Series B (Methodological).* 20(1958): 215-242.

15. Dasgupta, C.G., G.S. Dispensa and S. Ghose. “Comparing the Predictive Performance of a Neural Network Model with some Traditional Market Response Models.” *International Journal of Forecasting*. 10(1994): 235 – 244.

16. Davidson, R. and J.G. MacKinnon. *Estimation and Inference in Econometrics*. Oxford: Oxford University Press, 1993. 875p.

17. Demuth, H. and M. Beale. *Neural Network Toolbox User’s Guide (For Use with MATLAB).* Version 4. Natick, MA: The Mathworks Inc., 2001.

18. Eisen-Hecht, J.I. and R.A. Kramer. “A Cost-Benefit Analysis of Water Quality Protection in the Catawba River Basin.” *Journal of Water Resources Association*, in press, 2002.

19. Fahlman, S. and C. Lebiere. “The Cascade-Correlation Learning Architecture.” *Advances in Neural Information Processing Systems 2*. D.S. Touretzky editor. San Mateo, CA: Morgan Kaufmann Pub., 1990. p. 524 – 532.

20. Fahrmeir, L. and G. Tutz. *Mutlivariate Statistical Modelling Based on Generalized Linear Models*. New York: Springer-Verlag, 1994.

21. Fausett, L. *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*. Upper Saddle River, NJ: Prentice Hall, 1994. 461p.

22. Ferret, E. “Improving the Neural Network Testing Process”, *Adaptive Intelligent Systems, Proceedings of the 3rd BANKAI Workshop, Brussels, Belgium, 12-14 October 1992*, Society for Worldwide Interbank Financial Telecommunications SC editors. New York: Elsevier Science Publishers, 1993.

23. Fine, T.L. *Feedforward Neural Network Methodology*. New York: Springer-Verlag, 1999.

24. Finney, D.J. *Statistical Method in Biological Assay*. London: Charles Griffin and Company, Ltd., 1978.

25. Fletcher, R. *Practical Methods of Optimization*. New York: John Wiley and Sons, Inc., 1987.

26. Gabler, S., F. Laisney and M. Lechner. “Seminonparametric Estimation of Binary-Choice Models With an Application to Labor-Force Participation.” *Journal of Business and Economic Statistics*. 11(January, 1993): 61 – 80.

27. Gallant, A.R. and D. Nychka “Semi-Nonparametric Maximum Likelihood Estimation.” *Econometrica*. 55(March 1987): 363-390.

28. Gallant, A.R. “Unbiased determination of production technologies. *Journal of Econometrics*. 20(1982): 285 – 323.

29. Gerencsér, L. “Parameter Tracking of Time-Varying Continuous-Time Linear Stochastic Systems,” *Modelling, Identification and Robust Control*. C.I. Byrnes and A. Lindquist editors. Amsterdam: North-Holland, 1986. pp. 581-594.

30. Goss, E.P. and H. Ramchandani. “Survival Prediction in the Intensive Care Unit: A Comparison of Neural Networks and Binary Logit Regression.” *Socio-Economic Planning Science*. 32(1998): 189 – 198.

31. Hand, D.J. and W.E. Henley. “Statistical Classification Methods in Consumer Credit Scoring: A Review.” *Journal of the Royal Statistical Society. Series A (Statistics in Society)*. 160(1997): 523 – 541.

32. Hanemann, W.M. “Welfare Evaluations in Contingent Valuation Experiments with Discrete Responses.” *American Journal of Agricultural Economics*. 66(1984): 332 – 341.

33. Hanemann, W.M. “Willingness to Pay and Willingness to Accept: How Much Can They Differ?” *The American Economic Review*. 81(June 1991): 635-647.

34. Heng, A.C. and A. Randall. “Semi-nonparametric Estimation of Binary Response Models With an Application to Natural Resource Valuation.” *Journal of Econometrics*. 76(1997): 323 -340.

35. Hollander, M. and D.A. Wolfe. *Nonparametric Statistical Methods. Second Edition.* New York: John Wiley & Sons, Inc., 1999.

36. Horrowitz, J.L. “Semiparametric and nonparametric estimation of quantal response models.” *Handbook of Statistics*. Volume 11. G.S. Maddala, C.R. Rao and H.D. Vinod editors. New York: North-Holland, 1993.

37. Jeng, J.M. and D.R. Fesenmaier. “A Neural Network Approach to Discrete Choice Modeling.” *Recent Advances in Tourism Marketing Research*. D.R. Fesenmaier, J.T. O’Leary and M. Uysal editors. New York: The Haworth Press, Inc., 1996.

38. Kastens, T.L. and A.M. Featherstone. “Feedforward Backpropagation Neural Networks in Prediction of Farmer Risk Preferences.” *American Journal of Agricultural Economics*. 78(May 1996): 400 – 415.

39. Kay, R. and S. Little. “Transformations of the Explanatory Variables in the Logistic Regression Model for Binary Data.” *Biometrika*. 74(September 1987): 495 – 501.

40. Kuan, C. and H. White. “Artificial Neural Networks: An Econometric Perspective.” *Econometric Reviews*. 13(1994): 1 – 91.

41. Leshno, M., V. Lin, A. Pinkus, S. Schocken. “Multilayer Feed-Forward Networks with a Nonpolynomial Activation Function Can Approximate Any Function.” *Neural Networks*. 6(1993): 861 – 867.

42. Luenberger, D.C. *Optimization by Vector Space Methods*. New York: John Wiley and Sons, Inc., 1969.

43. MacKay, D.J.C. “Bayesian Interpolation.” *Neural Computation*. 4(1992): 415 – 447.

44. Maddala, G.S. *Limited Dependent and Qualitative Variables in Econometrics*. Cambridge, UK: Cambridge University Press, 1983.

45. McFadden, D.L. “Econometric Analysis of Qualitative Response Models.” *Handbook of Econometrics Volume II.*. Griliches, Z and M.D. Intriligator editors. New York: North-Holland, 1984.

46. Mehrotra, K., C.K. Mohan and S. Ranka. *Elements of Artificial Neural Networks*. Cambridge, MA: MIT Press, 1997.

47. Moller, M. “A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning.” *Neural Networks*. 6(1993): 525 -533.

48. Peng, C. and X. Wen. “Recent Applications of Artificial Neural Networks in Forest Resource Management: An Overview.” *Environmental Decision Support Systems and Artificial Intelligence*. Technical Report WS-99-07. U. Corté and M. Sánchez-Marré editors. Menlo Park, CA: AAAI Press, 1999: 15 – 22.

49. Powers, D.A. and Y. Xie. *Statisical Methods for Categorical Data Analysis*. San Diego, CA: Academic Press, 2000.

50. Principe, J.C., N.R. Euliano and W.C. Lefebvre. *Neural and Adaptive Systems: Fundamentals Through Simulations*. New York: John Wiley and Sons, Inc., 2000. 656p.

51. Qi, M. “Predicting U.S. Recessions with Leading Indicators via Neural Network Models.” *International Journal of Forecasting*. 17(2001): 383 – 401.

52. Ranasinghe, M., G. Bee Hua and T. Barathithsasan. “A Comparative Study of Artificial Neural Networks and Multiple Regression Analysis in Estimating Willingess to Pay for Urban Water Supply.” Selected Paper presented at the 2nd International Conference on Construction Industry Development and 1st Conference

of CIB TG 29 on Construction in Developing Countries. Pan Pacific, Singapore. 27-29, October 1999.

53. Reidmiller, M. and H. Braun. “A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm.” *Proceedings of the IEEE International Conference on Neural Networks*, 1993.

54. Ripley, B.D. “Neural Networks and Related Methods for Classification.” *Journal of the Royal Statistical Society, Series B (Methodological)*. 56(1994): 409-456.

55. Ripley, B.D. *Pattern Recognition and Neural Networks*. Cambridge, UK: Cambridge University Press, 1996.

56. Small, C.G. and D.L. McLeish. *Hilbert Space Methods in Probability and Statistical Inference*. New York: John Wiley and Sons, Inc., 1994.

57. Spanos, A. “Chapter 20: Regression Like Models.” Department of Economics, Virginia Polytechnic Institute and State University, mimeo. 2000.

58. Spanos, A. *Probability Theory and Statistical Inference: Econometric Modeling with Observational Data.* Cambridge: Cambridge University Press, 1999.

59. Spanos, A. *Statistical Foundations of Econometric Modeling*. Cambridge: Cambridge University Press, 1986.

60. Tam, K.Y. and M.Y. Kiang. “Managerial Applications of Neural Networks: The Case of Bank Failure Predictions.” *Management Science*. 38(Jul. 1992): 926-947.

61. Tikhonov, A. and V. Arsenin. *Solution of Ill-Posed Problems*. Washington, D.C.: Winston, 1977.

62. Train, K.E. *Discrete Choice Models with Simulation*. Cambridge, UK: Cambridge University Press, 2003.

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

64. Wang, S. “The Unpredictability of Standard Back Propagation Neural Networks in Classification Problems.” *Management Science*. 41(March 1995): 555 – 559.

65. Weisstein, E.W. “Normal Distribution Function.” Eric Weisstein’s World of Mathematics. Wolfram Research. 1999. http://mathworld.wolfram.com/NormalDistributionFunction.html.

66. West, P.M., P.L. Brockett and L.L Golden. “A Comparitive Analysis of Neural Networks and Statistical Models for Predicting Consumer Choice.” *Marketing Science*. 16(1997): 370 – 391.

67. White, H. “Some Asymptotic Results for Learning in Single-hidden-Layer Feedforward Network Models.” *Journal of the American Statistical Association*. 84(Dec. 1989): 1003 – 1013.

68. White, H., K. Hornik and M. Stinchcombe. “Multilayer Feedforward Networks Are Universal Approximators.” *Artificial Neural Networks: Approximationa and Learning Theory*. H. White editor. Oxford, UK: Blackwell Publishers, 1992.

69. Zeng, L. “Prediction and Classification with Neural Network Models.” Paper presented at the American Political Science Association annual meeting, San Francisco, CA., August, 1996.

70. Zurada, J.M., B.P. Foster, T.J. Ward and R.M. Barker. “Neural Networks Versus Logistic Regression Models For Predicting Financial Distress Response Variables.” *Journal of Applied Business Research*. 15(1999): 21-29.

Appendix A: MATLAB Procedures for the Maximum Likelihood (Kullback-Leibler) Fitting Criterion and its Gradient (With Respect to the Training Errors)

Maximum Likelihood Fitting Criterion Procedure

```
function perf = mleperf(e,x,pp)

%'mleperf' - Kullback-Leibler Information performance function for
%            dichotomous choice models.
%
%     Syntax
%
%       perf = mleperf(e,x,pp)
%       perf = mleperf(e,net,pp)
%       info = mleperf(code)
%
%     Description
%
%       'mleperf' is a network performance function.  It measures the
%       network's performance according to Kullback-Leibler Information
%       Criterion. The performance criterion is only usable with
%       networks that have an activation function in the output layer
%       that is restricted in range to [0,1], and output targets that
%       take only values of 0 or 1.
%
%       mleperf(E,X,PP) takes from one to three arguments,
%         E  - Matrix or cell array of error vector(s).
%         X  - Vector of all weight and bias values (ignored).
%         PP - Performance parameters (ignored).
%       and returns the negative of the maximum likelihood function for
%       a network with a activation in the output layer with range
%       equal to [0,1].
%
%       mleperf(CODE) returns useful information for each CODE string:
%         'deriv'     - Returns name of derivative function.
%         'name'      - Returns full name.
%         'pnames'    - Returns names of training parameters.
%         'pdefaults' - Returns default training parameters.
%
%     Examples
%
%       Here a two layer feed-forward network is created with a 1
%       -element input ranging from -10 to 10, four hidden TANSIG
%       neurons, and one PURELIN output neuron.
%
%         net = newff([-10 10],[4 1],{'tansig','logsig'});
%
%       Here the network is given a batch of inputs P.  The error
%       is calculated by subtracting the output A from target T.
%       Then the negative maximum likelihood is calculated.
%
%         p = [-10 -5 0 5 10];
%         t = [0 0 1 1 1];
%         y = sim(net,p)
%         e = t-y
%         perf = mleperf(e)
```

```
%
%   Note that 'mleperf' can be called with only one argument
%   because the other arguments are ignored.  'mleperf' supports
%   those ignored arguments to conform to the standard performance
%   function argument list.
%
%  Network Use
%
%   You can create a standard network that uses 'mleperf' with
%   NEWFF, NEWCF, or NEWELM.
%
%   To prepare a custom network to be trained with 'mleperf' set
%   NET.performFcn to 'mleperf'.  This will automatically set
%   NET.performParam to the empty matrix [], as MSE has no
%   performance parameters.
%
%   In either case, calling TRAIN will result
%   in 'mleperf' being used to calculate performance.
%
%   See NEWFF or NEWCF for examples.
%
%  See also MSE, MSEREG, MAE, dmleperf

% Jason Bergtold - Virginia Tech 2003

if nargin < 1, error('Not enough input arguments.'), end

% FUNCTION INFO
% =============

if isstr(e)
 switch lower(e)
  case 'deriv',
    perf = 'dmleperf';
  case 'name',
    perf = 'Kullback-Leibler Information Criterion for Dichotomous …
    Choice Models';
  case 'pnames',
    perf = {};
  case 'pdefaults',
    perf = [];
  otherwise,
    error('Unrecognized code.')
 end
 return
end

% CALCULATION
% ===========

if isa(e,'cell');
  e = cell2mat(e);
end

d = [];
t = [];
for i = 1:1:length(e);
```

```
    if e(i) < 0;
        d(i) = - e(i);
        t(i) = 0;
    else
        d(i) = 1 - e(i);
        t(i) = 1;
    end
end

[T,k] = size(d);
if T == 1;
    d = d';
    t = t';
end

c = ones(T,1);
cd = c - d;
lcd = [];
ld = [];
for i = 1:1:length(d);
    if cd(i) == 0;
        lcd(i) = log(1e-323);
    else
        lcd(i) = log(cd(i));
    end
    if d(i) == 0;
        ld(i) = log(1e-323);
    else
        ld(i) = log(d(i));
    end
end
temp = (c - t)'*lcd' + t'*ld';
perf = - temp;
```

Gradient of the Maximum Likelihood Fitting Criterion with respect to the Training Errors Procedure

```
function d = dmleperf(code,e,x,perf,pp)

%'dmleperf' Kullback Leibler Information derivatives function for
% dichotomous choice models.
%
%     Syntax
%
%       dPerf_dE = dmleperf('e',e,x,perf,pp)
%       dPerf_dX = dmleperf('x',e,x,perf,pp)
%
%     Description
%
%       'dmleperf' is the derivative function for 'mleperf'.
%
%       dmleperf('e',e,x,perf,pp) takes these arguments,
%         e    - Matrix or cell array of error vector(s).
%         x    - Vector of all weight and bias values.
%         perf - Network performance (ignored).
%         pp   - Performance parameters (ignored).
%       and returns the derivative dPerf/dE.
%
%       dmleperf('x',e,x,perf,pp) returns the derivative dPerf/dX.
%
%     Examples
%
%       Here we define E and X for a network with one
%       3-element output and six weight and bias values.
%
%         E = {[1; -2; 0.5]};
%         X = [0; 0.2; -2.2; 4.1; 0.1; -0.2];
%
%       Here we calculate the network's performance and
%       derivatives of performance.
%
%         perf = mleperf(e)
%         dPerf_dE = dmleperf('e',e,x)
%         dPerf_dX = dmleperf('x',e,x)
%
%       Note that 'mleperf' can be called with only one argument and
%       'dmleperf' with only three arguments because the other
%       arguments are ignored.  The other arguments exist so that
%       'mleperf' and 'dmleperf' conform to standard performance
%       function argument lists.
%
%     See also mleperf.

% Jason Bergtold, Virginia Tech 2003

if nargin < 3, error('Not enough input arguments.'),end

doubleForm = 0;
if isa(e,'double'), e = {e}; doubleForm = 1; end

if isa(e,'cell');
    em = cell2mat(e);
end

y = [];
t = [];
for i = 1:1:length(em);
    if em(i) < 0;
        y(i) = - em(i);
        t(i) = 0;
    else
        y(i) = 1 - em(i);
        t(i) = 1;
    end
end

[T,k] = size(y);
if T == 1;
    y = y';
    t = t';
end

switch lower(code)
```

```
case 'e',
    [r,c] = size(e);
    d = cell(r,c);
    for i = 1:1:r;
        dPdt = -log(y(i)) + log(1-y(i));
        dPdy = (-t(i)/y(i)) + ((1 - t(i))/(1 - y(i)));
        d{i,1} = inv(inv(dPdt) - inv(dPdy));
    end
    if doubleForm, d = cell2mat(d); end

case 'x',
    d = zeros(size(x));

otherwise,
   error(['Unrecognized code.'])
end
```

Appendix B: Simulation Results

The tables in this Appendix provide summary statistics for the simulations conducted in section 4.3. The tables present the means and standard deviations of the performance measures used to examine the simulation runs for the training, test and entire E-K data sets. Each simulation run is associated with a different sample partition generated using the sample reuse procedure in section 4.3. Table B1 and B2 present the mean and standard deviation of the *MSE* using the training data set over all 500 simulation runs for each simulation conducted. Tables B3 and B4 present the mean and standard deviation of the *MSFE* using the test data set over all 500 simulation runs for each simulation conducted. Tables B5 and B6 present the mean and standard deviation of the *PR* measure using the test data set over all 500 simulation runs for each simulation conducted. Tables B7 and B8 present the mean and standard deviation of the *PR* measure using the entire E-K data set over all 500 simulation runs for each simulation conducted.

The first six rows of each table represent the simulations examining the six algorithms analyzed in the MATLAB Neural Networks Toolbox in section 4.3.2. For most of the simulations conducted in section 4.3, row one represents the base case. The seventh row replaced the hyperbolic activation function with the logistic activation function to compare the performance between the two for the analysis in section 4.3.5. Rows eight and nine are the simulations for analyzing regularization and both regularization and cross validation simultaneously in section 4.3.6. Rows ten thru twenty-nine were used to examine the potential effectiveness of two-hidden layer FFBANNs. The columns represent the number of hidden nodes in the first hidden layer of the FFBANN architecture being examined.

Table B1: Mean *MSE* using the Training Data Sets from the 500 Simulation Runs[1]

Network Architecture[2]/ # of Hidden Nodes	1	2	3	4	5	6	7	8	9	10
1. BFG 20-?-1	0.1657	0.1641	0.1603	0.1572	0.1546	0.1527	0.1524	0.1516	0.1508	0.1497
2. CGF 20-?-1	0.1914	0.1766	0.1632	0.1600	0.1597	0.1602	0.1621	0.1638	0.1640	0.1644
3. GDX 20-?-1	0.1760	0.1718	0.1635	0.1587	0.1583	0.1539	0.1555	0.1531	0.1543	0.1508
4. LM 20-?-1	0.1569	0.1393	0.1292	0.1207	0.1127	0.1069	0.0989	0.0958	0.0907	0.0860
5. RP 20-?-1	0.1633	0.1580	0.1529	0.1511	0.1486	0.1478	0.1470	0.1451	0.1444	0.1428
6. SCG 20-?-1	0.1639	0.1609	0.1588	0.1565	0.1529	0.1533	0.1506	0.1499	0.1501	0.1487
7. BFG/LOG 20-?-1	0.1717	0.1718	0.1649	0.1619	0.1598	0.1588	0.1572	0.1566	0.1546	0.1536
8. BFG/REG 20-?-1	0.1587	0.1462	0.1412	0.1372	0.1327	0.1273	0.1216	0.1158	0.1099	0.1041
9. BFG/REG/VLD 20-?-1	0.1581	0.1472	0.1432	0.1417	0.1413	0.1416	0.1404	0.1413	0.1408	0.1430
10. BFG 20-?-1-1	0.1714	0.1672	0.1619	0.1600	0.1589	0.1579	0.1562	0.1571	0.1562	0.1566
11. BFG 20-?-2-1	0.1744	0.1714	0.1702	0.1651	0.1656	0.1627	0.1606	0.1594	0.1600	0.1588
12. BFG 20-?-3-1	0.1816	0.1744	0.1668	0.1626	0.1606	0.1592	0.1569	0.1566	0.1561	0.1558
13. BFG 20-?-4-1	0.1868	0.1726	0.1659	0.1619	0.1607	0.1565	0.1567	0.1555	0.1550	0.1544
14. BFG 20-?-5-1	0.1880	0.1736	0.1634	0.1610	0.1572	0.1578	0.1553	0.1529	0.1531	0.1519
15. BFG 20-?-6-1	0.1895	0.1705	0.1629	0.1608	0.1574	0.1559	0.1544	0.1524	0.1519	0.1507
16. BFG 20-?-7-1	0.1921	0.1740	0.1635	0.1594	0.1564	0.1539	0.1537	0.1518	0.1507	0.1501
17. BFG 20-?-8-1	0.1955	0.1731	0.1638	0.1589	0.1568	0.1546	0.1521	0.1514	0.1508	0.1504
18. BFG 20-?-9-1	0.1944	0.1732	0.1650	0.1593	0.1564	0.1540	0.1523	0.1515	0.1516	0.1501
19. BFG 20-?-10-1	0.1949	0.1740	0.1629	0.1582	0.1561	0.1536	0.1537	0.1508	0.1496	0.1489
20. BFG 20-?-11-1	0.2000	0.1751	0.1621	0.1584	0.1557	0.1537	0.1515	0.1507	0.1493	0.1493
21. BFG 20-?-12-1	0.1985	0.1717	0.1632	0.1577	0.1563	0.1544	0.1517	0.1494	0.1486	0.1472
22. BFG 20-?-13-1	0.1985	0.1739	0.1621	0.1584	0.1564	0.1525	0.1511	0.1503	0.1502	0.1486
23. BFG 20-?-14-1	0.2008	0.1758	0.1606	0.1588	0.1544	0.1534	0.1513	0.1487	0.1481	0.1487
24. BFG 20-?-15-1	0.2001	0.1743	0.1609	0.1566	0.1531	0.1517	0.1506	0.1490	0.1481	0.1469
25. BFG 20-?-16-1	0.2015	0.1745	0.1632	0.1579	0.1554	0.1526	0.1504	0.1493	0.1491	0.1475
26. BFG 20-?-17-1	0.2001	0.1747	0.1610	0.1564	0.1545	0.1522	0.1499	0.1495	0.1487	0.1467
27. BFG 20-?-18-1	0.2014	0.1760	0.1608	0.1570	0.1543	0.1520	0.1508	0.1489	0.1480	0.1467
28. BFG 20-?-19-1	0.2036	0.1748	0.1618	0.1561	0.1545	0.1513	0.1502	0.1484	0.1485	0.1461
29. BFG 20-?-20-1	0.2039	0.1731	0.1600	0.1572	0.1537	0.1529	0.1490	0.1488	0.1485	0.1460

Table B1 continued

Network Architecture[2]/										
# of Hidden Nodes	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1. BFG 20-?-1	0.1506	0.1491	0.1485	0.1479	0.1477	0.1482	0.1460	0.1471	0.1464	0.1476
2. CGF 20-?-1	0.1665	0.1697	0.1683	0.1676	0.1715	0.1699	0.1733	0.1721	0.1702	0.1724
3. GDX 20-?-1	0.1506	0.1521	0.1518	0.1520	0.1521	0.1512	0.1498	0.1507	0.1505	0.1498
4. LM 20-?-1	0.0805	0.0775	0.0711	0.0702	0.0672	0.0628	0.0611	0.0570	0.0527	0.0544
5. RP 20-?-1	0.1434	0.1425	0.1424	0.1412	0.1417	0.1417	0.1407	0.1418	0.1390	0.1394
6. SCG 20-?-1	0.1487	0.1501	0.1484	0.1480	0.1464	0.1481	0.1483	0.1466	0.1476	0.1463
7. BFG/LOG 20-?-1	0.1544	0.1547	0.1532	0.1533	0.1515	0.1525	0.1513	0.1524	0.1522	0.1500
8. BFG/REG 20-?-1	0.0988	0.0936	0.0889	0.0845	0.0806	0.0770	0.0737	0.0710	0.0683	0.0657
9. BFG/REG/VLD 20-?-1	0.1462	0.1450	0.1461	0.1474	0.1487	0.1496	0.1520	0.1543	0.1542	0.1540
10. BFG 20-?-1-1	0.1551	0.1553	0.1544	0.1540	0.1552	0.1550	0.1542	0.1529	0.1525	0.1536
11. BFG 20-?-2-1	0.1577	0.1573	0.1582	0.1580	0.1573	0.1564	0.1564	0.1564	0.1545	0.1546
12. BFG 20-?-3-1	0.1545	0.1556	0.1539	0.1543	0.1529	0.1519	0.1525	0.1519	0.1521	0.1517
13. BFG 20-?-4-1	0.1533	0.1521	0.1512	0.1513	0.1506	0.1521	0.1514	0.1503	0.1515	0.1496
14. BFG 20-?-5-1	0.1511	0.1512	0.1510	0.1497	0.1489	0.1498	0.1501	0.1485	0.1487	0.1479
15. BFG 20-?-6-1	0.1506	0.1500	0.1502	0.1487	0.1494	0.1499	0.1486	0.1484	0.1478	0.1467
16. BFG 20-?-7-1	0.1504	0.1488	0.1485	0.1485	0.1480	0.1473	0.1485	0.1475	0.1470	0.1458
17. BFG 20-?-8-1	0.1490	0.1481	0.1483	0.1479	0.1481	0.1475	0.1468	0.1463	0.1455	0.1456
18. BFG 20-?-9-1	0.1493	0.1494	0.1483	0.1467	0.1455	0.1463	0.1470	0.1462	0.1457	0.1444
19. BFG 20-?-10-1	0.1478	0.1474	0.1475	0.1479	0.1466	0.1450	0.1458	0.1449	0.1443	0.1443
20. BFG 20-?-11-1	0.1478	0.1464	0.1468	0.1462	0.1453	0.1453	0.1443	0.1453	0.1443	0.1433
21. BFG 20-?-12-1	0.1469	0.1469	0.1467	0.1468	0.1452	0.1449	0.1449	0.1448	0.1451	0.1427
22. BFG 20-?-13-1	0.1467	0.1457	0.1458	0.1443	0.1445	0.1454	0.1446	0.1432	0.1435	0.1434
23. BFG 20-?-14-1	0.1465	0.1460	0.1453	0.1451	0.1445	0.1436	0.1432	0.1438	0.1438	0.1430
24. BFG 20-?-15-1	0.1468	0.1462	0.1448	0.1458	0.1451	0.1436	0.1435	0.1438	0.1424	0.1437
25. BFG 20-?-16-1	0.1469	0.1471	0.1451	0.1459	0.1438	0.1437	0.1445	0.1435	0.1433	0.1433
26. BFG 20-?-17-1	0.1462	0.1461	0.1444	0.1441	0.1437	0.1421	0.1429	0.1422	0.1421	0.1443
27. BFG 20-?-18-1	0.1464	0.1446	0.1453	0.1442	0.1437	0.1426	0.1424	0.1421	0.1429	0.1425
28. BFG 20-?-19-1	0.1448	0.1441	0.1455	0.1438	0.1421	0.1425	0.1433	0.1427	0.1407	0.1423
29. BFG 20-?-20-1	0.1460	0.1444	0.1437	0.1438	0.1428	0.1437	0.1422	0.1415	0.1419	0.1423

[1] For each simulation run, a sample partition of the E-K data set was randomly generated using the sample reuse procedure discussed in section 4.3.

[2] Each row represents the different network architectures and training schemes used in the simulations conducted in section 4.3. Each FFBANN used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, each FFBANN was batch trained using stopping rule (S5) with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-5}$ and $MAX = 1000$. These settings were used for all the simulations, unless otherwise noted. The first acronym used to describe the network architecture (e.g. BFG) indicates the algorithm used to train the network. The acronyms are: (i) BFG – BFGS quasi-Newton algorithm, (ii) CGF – conjugate gradient algorithm with Fletcher-Reeves update, (iii) GDX – steepest descent algorithm with adaptive learning and momentum rates, (iv) LM – Levenberg-Marquardt algorithm, (v) RP – resilient backpropagation algorithm, and (vi) SCG – scaled conjugate gradient algorithm (see section 3.2.3 and 4.3.2). The other acronyms indicate other specific changes. LOG is used when the hyperbolic tangent activation function in the hidden layer is replaced with the logistic activation function (see section 4.3.5). REG is used when regularization is built into the fitting criterion (see sections 3.2.3.3 and 4.3.6). VLD indicates that stopping rule (S5) is used along with REG (see section 4.3.6). The number of hidden layers and nodes in each layer is given by the string of numbers following the set of acronyms in each row. This is represented as (# of input variables) – (# of nodes in the first hidden layer) – (# of hidden nodes in the second hidden layer if applicable) – (# of output variables). The (# of nodes in the first hidden layer) is replaced by '?', to indicate that this number is determined by the column of the table you are in. The columns numbered 1 – 20, indicate the number of hidden nodes in the first hidden layer of the network.

Table B2 Standard Deviation of the *MSE* using the Training Data Sets from the 500 Simulation Runs[1]

Network Architecture[2]/ # of Hidden Nodes	1	2	3	4	5	6	7	8	9	10
1. BFG 20-?-1	0.0175	0.0169	0.0130	0.0117	0.0121	0.0105	0.0105	0.0119	0.0113	0.0112
2. CGF 20-?-1	0.0251	0.0258	0.0204	0.0197	0.0220	0.0223	0.0256	0.0277	0.0289	0.0307
3. GDX 20-?-1	0.0407	0.0359	0.0342	0.0304	0.0326	0.0265	0.0315	0.0290	0.0321	0.0263
4. LM 20-?-1	0.0402	0.0243	0.0243	0.0240	0.0206	0.0272	0.0247	0.0267	0.0234	0.0273
5. RP 20-?-1	0.0109	0.0132	0.0112	0.0131	0.0127	0.0130	0.0137	0.0139	0.0140	0.0137
6. SCG 20-?-1	0.0224	0.0229	0.0253	0.0247	0.0195	0.0233	0.0162	0.0188	0.0224	0.0224
7. BFG/LOG 20-?-1	0.0187	0.0210	0.0149	0.0142	0.0140	0.0138	0.0138	0.0138	0.0126	0.0131
8. BFG/REG 20-?-1	0.0032	0.0035	0.0036	0.0036	0.0036	0.0036	0.0035	0.0035	0.0033	0.0033
9. BFG/REG/VLD 20-?-1	0.0040	0.0049	0.0061	0.0077	0.0090	0.0111	0.0113	0.0129	0.0134	0.0145
10. BFG 20-?-1-1	0.0186	0.0170	0.0131	0.0131	0.0129	0.0136	0.0125	0.0153	0.0140	0.0144
11. BFG 20-?-2-1	0.0229	0.0218	0.0224	0.0199	0.0219	0.0205	0.0206	0.0184	0.0175	0.0194
12. BFG 20-?-3-1	0.0267	0.0238	0.0191	0.0170	0.0165	0.0176	0.0155	0.0161	0.0152	0.0172
13. BFG 20-?-4-1	0.0267	0.0229	0.0186	0.0165	0.0150	0.0142	0.0150	0.0151	0.0157	0.0150
14. BFG 20-?-5-1	0.0268	0.0229	0.0173	0.0159	0.0142	0.0157	0.0132	0.0135	0.0136	0.0138
15. BFG 20-?-6-1	0.0265	0.0222	0.0181	0.0158	0.0132	0.0132	0.0136	0.0133	0.0142	0.0130
16. BFG 20-?-7-1	0.0263	0.0242	0.0175	0.0153	0.0136	0.0127	0.0142	0.0126	0.0134	0.0136
17. BFG 20-?-8-1	0.0256	0.0250	0.0188	0.0175	0.0145	0.0147	0.0132	0.0134	0.0130	0.0139
18. BFG 20-?-9-1	0.0259	0.0238	0.0198	0.0152	0.0151	0.0154	0.0136	0.0135	0.0144	0.0143
19. BFG 20-?-10-1	0.0247	0.0254	0.0180	0.0149	0.0147	0.0134	0.0165	0.0136	0.0133	0.0140
20. BFG 20-?-11-1	0.0251	0.0253	0.0180	0.0162	0.0151	0.0141	0.0134	0.0136	0.0135	0.0148
21. BFG 20-?-12-1	0.0233	0.0236	0.0207	0.0149	0.0149	0.0145	0.0128	0.0134	0.0141	0.0123
22. BFG 20-?-13-1	0.0238	0.0246	0.0191	0.0157	0.0150	0.0141	0.0122	0.0141	0.0135	0.0154
23. BFG 20-?-14-1	0.0235	0.0257	0.0177	0.0160	0.0143	0.0137	0.0135	0.0126	0.0138	0.0145
24. BFG 20-?-15-1	0.0248	0.0258	0.0175	0.0156	0.0151	0.0126	0.0130	0.0135	0.0140	0.0137
25. BFG 20-?-16-1	0.0238	0.0248	0.0189	0.0157	0.0148	0.0140	0.0131	0.0135	0.0152	0.0147
26. BFG 20-?-17-1	0.0230	0.0248	0.0185	0.0142	0.0151	0.0145	0.0137	0.0132	0.0142	0.0143
27. BFG 20-?-18-1	0.0237	0.0262	0.0182	0.0164	0.0147	0.0148	0.0135	0.0137	0.0147	0.0140
28. BFG 20-?-19-1	0.0239	0.0248	0.0197	0.0157	0.0160	0.0141	0.0145	0.0139	0.0156	0.0147
29. BFG 20-?-20-1	0.0231	0.0263	0.0189	0.0170	0.0144	0.0163	0.0144	0.0143	0.0149	0.0156

Table B2 continued

Network Architecture[2]/										
# of Hidden Nodes	11	12	13	14	15	16	17	18	19	20
1. BFG 20-?-1	0.0118	0.0119	0.0122	0.0115	0.0113	0.0131	0.0118	0.0127	0.0129	0.0134
2. CGF 20-?-1	0.0321	0.0327	0.0331	0.0337	0.0368	0.0370	0.0398	0.0369	0.0368	0.0395
3. GDX 20-?-1	0.0264	0.0289	0.0303	0.0303	0.0308	0.0306	0.0294	0.0289	0.0307	0.0301
4. LM 20-?-1	0.0274	0.0240	0.0219	0.0297	0.0306	0.0315	0.0291	0.0299	0.0308	0.0401
5. RP 20-?-1	0.0152	0.0145	0.0156	0.0160	0.0165	0.0171	0.0166	0.0180	0.0188	0.0184
6. SCG 20-?-1	0.0213	0.0280	0.0270	0.0248	0.0188	0.0252	0.0281	0.0207	0.0287	0.0235
7. BFG/LOG 20-?-1	0.0143	0.0165	0.0133	0.0163	0.0151	0.0142	0.0152	0.0154	0.0152	0.0157
8. BFG/REG 20-?-1	0.0033	0.0032	0.0031	0.0030	0.0029	0.0027	0.0026	0.0026	0.0025	0.0025
9. BFG/REG/VLD 20-?-1	0.0170	0.0172	0.0179	0.0182	0.0183	0.0188	0.0190	0.0192	0.0190	0.0189
10. BFG 20-?-1-1	0.0150	0.0147	0.0143	0.0142	0.0156	0.0147	0.0150	0.0144	0.0143	0.0155
11. BFG 20-?-2-1	0.0185	0.0184	0.0195	0.0187	0.0191	0.0187	0.0187	0.0189	0.0173	0.0181
12. BFG 20-?-3-1	0.0168	0.0161	0.0160	0.0177	0.0165	0.0158	0.0168	0.0170	0.0156	0.0155
13. BFG 20-?-4-1	0.0149	0.0134	0.0149	0.0141	0.0142	0.0171	0.0142	0.0151	0.0167	0.0162
14. BFG 20-?-5-1	0.0148	0.0144	0.0138	0.0139	0.0139	0.0148	0.0158	0.0144	0.0155	0.0139
15. BFG 20-?-6-1	0.0131	0.0133	0.0144	0.0151	0.0142	0.0154	0.0144	0.0149	0.0141	0.0142
16. BFG 20-?-7-1	0.0140	0.0141	0.0136	0.0148	0.0132	0.0134	0.0145	0.0146	0.0141	0.0147
17. BFG 20-?-8-1	0.0138	0.0136	0.0142	0.0133	0.0139	0.0148	0.0143	0.0148	0.0150	0.0148
18. BFG 20-?-9-1	0.0146	0.0153	0.0146	0.0147	0.0126	0.0146	0.0152	0.0144	0.0152	0.0148
19. BFG 20-?-10-1	0.0137	0.0135	0.0154	0.0149	0.0146	0.0146	0.0147	0.0143	0.0141	0.0150
20. BFG 20-?-11-1	0.0137	0.0130	0.0135	0.0137	0.0141	0.0146	0.0138	0.0163	0.0155	0.0152
21. BFG 20-?-12-1	0.0134	0.0134	0.0146	0.0146	0.0153	0.0151	0.0157	0.0150	0.0160	0.0147
22. BFG 20-?-13-1	0.0150	0.0135	0.0154	0.0126	0.0149	0.0156	0.0148	0.0152	0.0157	0.0158
23. BFG 20-?-14-1	0.0151	0.0150	0.0136	0.0151	0.0154	0.0140	0.0151	0.0154	0.0154	0.0153
24. BFG 20-?-15-1	0.0139	0.0135	0.0148	0.0148	0.0145	0.0149	0.0143	0.0153	0.0146	0.0158
25. BFG 20-?-16-1	0.0151	0.0163	0.0145	0.0154	0.0157	0.0145	0.0150	0.0145	0.0158	0.0154
26. BFG 20-?-17-1	0.0140	0.0151	0.0147	0.0147	0.0158	0.0141	0.0146	0.0152	0.0151	0.0158
27. BFG 20-?-18-1	0.0147	0.0148	0.0158	0.0151	0.0145	0.0143	0.0141	0.0157	0.0159	0.0165
28. BFG 20-?-19-1	0.0131	0.0142	0.0150	0.0147	0.0150	0.0148	0.0158	0.0163	0.0151	0.0143
29. BFG 20-?-20-1	0.0152	0.0147	0.0154	0.0154	0.0147	0.0170	0.0151	0.0158	0.0151	0.0161

[1] For each simulation run, a sample partition of the E-K data set was randomly generated using the sample reuse procedure discussed in section 4.3.

[2] Each row represents the different network architectures and training schemes used in the simulations conducted in section 4.3. Each FFBANN used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, each FFBANN was batch trained using stopping rule (S5) with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-5}$ and $MAX = 1000$. These settings were used for all the simulations, unless otherwise noted. The first acronym used to describe the network architecture (e.g. BFG) indicates the algorithm used to train the network. The acronyms are: (i) BFG – BFGS quasi-Newton algorithm, (ii) CGF – conjugate gradient algorithm with Fletcher-Reeves update, (iii) GDX – steepest descent algorithm with adaptive learning and momentum rates, (iv) LM – Levenberg-Marquardt algorithm, (v) RP – resilient backpropagation algorithm, and (vi) SCG – scaled conjugate gradient algorithm (see section 3.2.3 and 4.3.2). The other acronyms indicate other specific changes. LOG is used when the hyperbolic tangent activation function in the hidden layer is replaced with the logistic activation function (see section 4.3.5). REG is used when regularization is built into the fitting criterion (see sections 3.2.3.3 and 4.3.6). VLD indicates that stopping rule (S5) is used along with REG (see section 4.3.6). The number of hidden layers and nodes in each layer is given by the string of numbers following the set of acronyms in each row. This is represented as (# of input variables) – (# of nodes in the first hidden layer) – (# of hidden nodes in the second hidden layer if applicable) – (# of output variables). The (# of nodes in the first hidden layer) is replaced by ‘?’, to indicate that this number is determined by the column of the table you are in. The columns numbered 1 – 20, indicate the number of hidden nodes in the first hidden layer of the network.

Table B3: Mean *MSFE* using the Test Data Sets from the 500 Simulation Runs[1]

Network Architecture[2]**/** **# of Hidden Nodes**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1. BFG 20-?-1	0.1808	0.1781	0.1755	0.1736	0.1723	0.1715	0.1717	0.1718	0.1717	0.1723
2. CGF 20-?-1	0.1998	0.1895	0.1802	0.1782	0.1775	0.1775	0.1796	0.1826	0.1827	0.1841
3. GDX 20-?-1	0.1910	0.1894	0.1842	0.1813	0.1813	0.1778	0.1792	0.1796	0.1802	0.1784
4. LM 20-?-1	0.1772	0.1762	0.1775	0.1775	0.1773	0.1786	0.1794	0.1807	0.1802	0.1818
5. RP 20-?-1	0.1801	0.1781	0.1774	0.1768	0.1764	0.1762	0.1757	0.1766	0.1764	0.1756
6. SCG 20-?-1	0.1803	0.1790	0.1781	0.1772	0.1749	0.1750	0.1732	0.1749	0.1750	0.1755
7. BFG/LOG 20-?-1	0.1845	0.1842	0.1781	0.1767	0.1751	0.1745	0.1737	0.1753	0.1739	0.1741
8. BFG/REG 20-?-1	0.1667	0.1573	0.1567	0.1638	0.1748	0.1865	0.1977	0.2068	0.2164	0.2236
9. BFG/REG/VLD 20-?-1	0.1677	0.1603	0.1586	0.1592	0.1612	0.1637	0.1646	0.1676	0.1698	0.1735
10. BFG 20-?-1-1	0.1834	0.1815	0.1780	0.1771	0.1771	0.1775	0.1773	0.1780	0.1778	0.1794
11. BFG 20-?-2-1	0.1863	0.1837	0.1844	0.1813	0.1828	0.1812	0.1806	0.1806	0.1809	0.1807
12. BFG 20-?-3-1	0.1932	0.1874	0.1812	0.1801	0.1780	0.1782	0.1776	0.1787	0.1782	0.1788
13. BFG 20-?-4-1	0.1978	0.1865	0.1816	0.1790	0.1782	0.1771	0.1773	0.1778	0.1783	0.1773
14. BFG 20-?-5-1	0.2013	0.1860	0.1798	0.1783	0.1768	0.1777	0.1766	0.1769	0.1765	0.1766
15. BFG 20-?-6-1	0.2025	0.1859	0.1805	0.1783	0.1764	0.1768	0.1765	0.1765	0.1768	0.1764
16. BFG 20-?-7-1	0.2066	0.1876	0.1800	0.1776	0.1767	0.1758	0.1766	0.1767	0.1765	0.1765
17. BFG 20-?-8-1	0.2095	0.1884	0.1810	0.1786	0.1778	0.1768	0.1761	0.1770	0.1766	0.1765
18. BFG 20-?-9-1	0.2092	0.1879	0.1830	0.1780	0.1769	0.1770	0.1766	0.1766	0.1765	0.1764
19. BFG 20-?-10-1	0.2097	0.1891	0.1810	0.1778	0.1774	0.1767	0.1775	0.1771	0.1774	0.1776
20. BFG 20-?-11-1	0.2143	0.1902	0.1799	0.1790	0.1768	0.1769	0.1767	0.1776	0.1770	0.1770
21. BFG 20-?-12-1	0.2135	0.1881	0.1821	0.1786	0.1779	0.1780	0.1774	0.1774	0.1770	0.1765
22. BFG 20-?-13-1	0.2133	0.1906	0.1810	0.1782	0.1775	0.1760	0.1754	0.1771	0.1760	0.1767
23. BFG 20-?-14-1	0.2157	0.1908	0.1810	0.1789	0.1790	0.1778	0.1775	0.1775	0.1773	0.1769
24. BFG 20-?-15-1	0.2144	0.1905	0.1811	0.1788	0.1780	0.1768	0.1763	0.1769	0.1776	0.1778
25. BFG 20-?-16-1	0.2163	0.1908	0.1824	0.1786	0.1768	0.1766	0.1766	0.1763	0.1767	0.1770
26. BFG 20-?-17-1	0.2156	0.1913	0.1828	0.1788	0.1785	0.1785	0.1772	0.1783	0.1774	0.1770
27. BFG 20-?-18-1	0.2175	0.1928	0.1825	0.1791	0.1785	0.1788	0.1781	0.1777	0.1774	0.1773
28. BFG 20-?-19-1	0.2192	0.1924	0.1825	0.1797	0.1793	0.1787	0.1790	0.1779	0.1786	0.1785
29. BFG 20-?-20-1	0.2183	0.1913	0.1818	0.1802	0.1786	0.1794	0.1790	0.1779	0.1785	0.1787

Table B3 continued

Network Architecture[2]/ # of Hidden Nodes	11	12	13	14	15	16	17	18	19	20
1. BFG 20-?-1	0.1723	0.1725	0.1724	0.1735	0.1729	0.1740	0.1729	0.1744	0.1743	0.1744
2. CGF 20-?-1	0.1851	0.1872	0.1878	0.1873	0.1907	0.1905	0.1936	0.1931	0.1919	0.1954
3. GDX 20-?-1	0.1778	0.1794	0.1813	0.1802	0.1798	0.1805	0.1799	0.1808	0.1808	0.1808
4. LM 20-?-1	0.1807	0.1807	0.1808	0.1818	0.1829	0.1844	0.1830	0.1851	0.1855	0.1884
5. RP 20-?-1	0.1769	0.1764	0.1771	0.1763	0.1762	0.1774	0.1773	0.1780	0.1780	0.1775
6. SCG 20-?-1	0.1750	0.1763	0.1766	0.1759	0.1747	0.1764	0.1776	0.1762	0.1781	0.1774
7. BFG/LOG 20-?-1	0.1744	0.1762	0.1759	0.1762	0.1743	0.1761	0.1759	0.1773	0.1772	0.1767
8. BFG/REG 20-?-1	0.2293	0.2335	0.2370	0.2393	0.2400	0.2404	0.2424	0.2420	0.2417	0.2418
9. BFG/REG/VLD 20-?-1	0.1779	0.1793	0.1819	0.1845	0.1871	0.1886	0.1920	0.1942	0.1956	0.1975
10. BFG 20-?-1-1	0.1793	0.1796	0.1796	0.1803	0.1819	0.1815	0.1820	0.1816	0.1808	0.1817
11. BFG 20-?-2-1	0.1811	0.1814	0.1816	0.1820	0.1823	0.1826	0.1828	0.1838	0.1816	0.1817
12. BFG 20-?-3-1	0.1792	0.1793	0.1796	0.1811	0.1799	0.1793	0.1805	0.1798	0.1805	0.1801
13. BFG 20-?-4-1	0.1778	0.1767	0.1781	0.1780	0.1777	0.1785	0.1783	0.1786	0.1800	0.1786
14. BFG 20-?-5-1	0.1773	0.1781	0.1774	0.1774	0.1778	0.1781	0.1780	0.1782	0.1792	0.1782
15. BFG 20-?-6-1	0.1768	0.1767	0.1772	0.1780	0.1770	0.1789	0.1781	0.1782	0.1777	0.1783
16. BFG 20-?-7-1	0.1774	0.1772	0.1771	0.1770	0.1773	0.1769	0.1777	0.1777	0.1779	0.1781
17. BFG 20-?-8-1	0.1769	0.1769	0.1778	0.1771	0.1778	0.1772	0.1775	0.1778	0.1782	0.1784
18. BFG 20-?-9-1	0.1762	0.1781	0.1769	0.1768	0.1768	0.1769	0.1771	0.1778	0.1776	0.1775
19. BFG 20-?-10-1	0.1772	0.1769	0.1780	0.1779	0.1778	0.1774	0.1783	0.1787	0.1770	0.1781
20. BFG 20-?-11-1	0.1765	0.1770	0.1776	0.1762	0.1777	0.1776	0.1777	0.1782	0.1779	0.1783
21. BFG 20-?-12-1	0.1768	0.1773	0.1782	0.1788	0.1776	0.1778	0.1774	0.1776	0.1784	0.1783
22. BFG 20-?-13-1	0.1772	0.1759	0.1767	0.1759	0.1779	0.1778	0.1771	0.1771	0.1765	0.1780
23. BFG 20-?-14-1	0.1766	0.1774	0.1776	0.1779	0.1786	0.1777	0.1786	0.1788	0.1785	0.1788
24. BFG 20-?-15-1	0.1782	0.1768	0.1782	0.1792	0.1785	0.1774	0.1780	0.1781	0.1782	0.1784
25. BFG 20-?-16-1	0.1775	0.1776	0.1772	0.1767	0.1774	0.1775	0.1773	0.1773	0.1779	0.1769
26. BFG 20-?-17-1	0.1774	0.1783	0.1778	0.1781	0.1788	0.1789	0.1784	0.1786	0.1783	0.1796
27. BFG 20-?-18-1	0.1779	0.1782	0.1788	0.1786	0.1776	0.1782	0.1787	0.1790	0.1787	0.1789
28. BFG 20-?-19-1	0.1775	0.1782	0.1785	0.1790	0.1785	0.1788	0.1781	0.1788	0.1784	0.1785
29. BFG 20-?-20-1	0.1784	0.1773	0.1777	0.1784	0.1789	0.1795	0.1785	0.1786	0.1794	0.1788

[1] For each simulation run, a sample partition of the E-K data set was randomly generated using the sample reuse procedure discussed in section 4.3.
[2] Each row represents the different network architectures and training schemes used in the simulations conducted in section 4.3. Each FFBANN used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, each FFBANN was batch trained using stopping rule (S5) with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-5}$ and $MAX = 1000$. These settings were used for all the simulations, unless otherwise noted. The first acronym used to describe the network architecture (e.g. BFG) indicates the algorithm used to train the network. The acronyms are: (i) BFG – BFGS quasi-Newton algorithm, (ii) CGF – conjugate gradient algorithm with Fletcher-Reeves update, (iii) GDX – steepest descent algorithm with adaptive learning and momentum rates, (iv) LM – Levenberg-Marquardt algorithm, (v) RP – resilient backpropagation algorithm, and (vi) SCG – scaled conjugate gradient algorithm (see section 3.2.3 and 4.3.2). The other acronyms indicate other specific changes. LOG is used when the hyperbolic tangent activation function in the hidden layer is replaced with the logistic activation function (see section 4.3.5). REG is used when regularization is built into the fitting criterion (see sections 3.2.3.3 and 4.3.6). VLD indicates that stopping rule (S5) is used along with REG (see section 4.3.6). The number of hidden layers and nodes in each layer is given by the string of numbers following the set of acronyms in each row. This is represented as (# of input variables) – (# of nodes in the first hidden layer) – (# of hidden nodes in the second hidden layer if applicable) – (# of output variables). The (# of nodes in the first hidden layer) is replaced by '?', to indicate that this number is determined by the column of the table you are in. The columns numbered 1 – 20, indicate the number of hidden nodes in the first hidden layer of the network.

Table B4: Standard Deviation of the *MSFE* using the Test Data Sets from the 500 Simulation Runs[1]

Network Architecture[2]/ # of Hidden Nodes	1	2	3	4	5	6	7	8	9	10
1. BFG 20-?-1	0.0198	0.0184	0.0172	0.0165	0.0167	0.0163	0.0163	0.0161	0.0171	0.0168
2. CGF 20-?-1	0.0221	0.0237	0.0225	0.0214	0.0216	0.0212	0.0247	0.0251	0.0253	0.0258
3. GDX 20-?-1	0.0372	0.0325	0.0307	0.0304	0.0305	0.0259	0.0304	0.0273	0.0286	0.0254
4. LM 20-?-1	0.0401	0.0261	0.0265	0.0240	0.0230	0.0231	0.0238	0.0233	0.0204	0.0237
5. RP 20-?-1	0.0182	0.0190	0.0179	0.0178	0.0179	0.0179	0.0171	0.0173	0.0176	0.0171
6. SCG 20-?-1	0.0218	0.0242	0.0264	0.0262	0.0214	0.0244	0.0206	0.0197	0.0233	0.0241
7. BFG/LOG 20-?-1	0.0220	0.0218	0.0200	0.0200	0.0195	0.0181	0.0181	0.0182	0.0182	0.0184
8. BFG/REG 20-?-1	0.0077	0.0088	0.0091	0.0096	0.0102	0.0110	0.0117	0.0117	0.0121	0.0121
9. BFG/REG/VLD 20-?-1	0.0112	0.0129	0.0138	0.0148	0.0159	0.0171	0.0164	0.0174	0.0179	0.0189
10. BFG 20-?-1-1	0.0210	0.0202	0.0190	0.0191	0.0191	0.0184	0.0178	0.0186	0.0186	0.0186
11. BFG 20-?-2-1	0.0232	0.0223	0.0227	0.0214	0.0217	0.0209	0.0214	0.0194	0.0197	0.0203
12. BFG 20-?-3-1	0.0242	0.0231	0.0206	0.0202	0.0187	0.0198	0.0190	0.0194	0.0192	0.0190
13. BFG 20-?-4-1	0.0258	0.0217	0.0202	0.0187	0.0184	0.0185	0.0185	0.0179	0.0192	0.0179
14. BFG 20-?-5-1	0.0251	0.0222	0.0198	0.0198	0.0181	0.0184	0.0190	0.0189	0.0181	0.0188
15. BFG 20-?-6-1	0.0257	0.0217	0.0210	0.0194	0.0178	0.0184	0.0184	0.0178	0.0186	0.0177
16. BFG 20-?-7-1	0.0271	0.0236	0.0208	0.0190	0.0191	0.0183	0.0185	0.0178	0.0176	0.0180
17. BFG 20-?-8-1	0.0252	0.0233	0.0211	0.0192	0.0185	0.0182	0.0187	0.0190	0.0183	0.0180
18. BFG 20-?-9-1	0.0252	0.0236	0.0211	0.0189	0.0166	0.0172	0.0177	0.0175	0.0182	0.0176
19. BFG 20-?-10-1	0.0247	0.0239	0.0212	0.0186	0.0191	0.0184	0.0193	0.0179	0.0184	0.0185
20. BFG 20-?-11-1	0.0251	0.0246	0.0204	0.0195	0.0192	0.0179	0.0176	0.0184	0.0184	0.0185
21. BFG 20-?-12-1	0.0241	0.0231	0.0225	0.0191	0.0199	0.0187	0.0187	0.0185	0.0182	0.0177
22. BFG 20-?-13-1	0.0252	0.0243	0.0206	0.0192	0.0185	0.0182	0.0184	0.0179	0.0184	0.0185
23. BFG 20-?-14-1	0.0245	0.0239	0.0216	0.0188	0.0191	0.0189	0.0189	0.0182	0.0192	0.0184
24. BFG 20-?-15-1	0.0243	0.0253	0.0198	0.0192	0.0193	0.0187	0.0180	0.0182	0.0177	0.0184
25. BFG 20-?-16-1	0.0243	0.0248	0.0211	0.0193	0.0183	0.0189	0.0187	0.0177	0.0179	0.0179
26. BFG 20-?-17-1	0.0231	0.0247	0.0219	0.0196	0.0183	0.0181	0.0179	0.0192	0.0182	0.0190
27. BFG 20-?-18-1	0.0244	0.0239	0.0208	0.0197	0.0182	0.0191	0.0183	0.0183	0.0186	0.0180
28. BFG 20-?-19-1	0.0229	0.0243	0.0210	0.0197	0.0193	0.0188	0.0182	0.0181	0.0193	0.0187
29. BFG 20-?-20-1	0.0244	0.0242	0.0218	0.0196	0.0189	0.0198	0.0195	0.0190	0.0194	0.0184

Table B4 continued

Network Architecture[2]/ # of Hidden Nodes	11	12	13	14	15	16	17	18	19	20
1. BFG 20-?-1	0.0169	0.0167	0.0169	0.0177	0.0169	0.0174	0.0164	0.0164	0.0164	0.0165
2. CGF 20-?-1	0.0273	0.0291	0.0285	0.0293	0.0310	0.0300	0.0340	0.0291	0.0314	0.0346
3. GDX 20-?-1	0.0242	0.0270	0.0294	0.0265	0.0277	0.0292	0.0265	0.0285	0.0268	0.0285
4. LM 20-?-1	0.0227	0.0197	0.0184	0.0226	0.0224	0.0265	0.0222	0.0232	0.0237	0.0290
5. RP 20-?-1	0.0170	0.0175	0.0176	0.0176	0.0179	0.0181	0.0174	0.0183	0.0184	0.0182
6. SCG 20-?-1	0.0218	0.0277	0.0285	0.0263	0.0220	0.0257	0.0290	0.0223	0.0291	0.0240
7. BFG/LOG 20-?-1	0.0183	0.0207	0.0187	0.0203	0.0188	0.0186	0.0187	0.0195	0.0194	0.0199
8. BFG/REG 20-?-1	0.0130	0.0123	0.0129	0.0125	0.0131	0.0137	0.0126	0.0138	0.0130	0.0134
9. BFG/REG/VLD 20-?-1	0.0191	0.0187	0.0186	0.0192	0.0187	0.0186	0.0185	0.0181	0.0177	0.0176
10. BFG 20-?-1-1	0.0189	0.0186	0.0188	0.0190	0.0188	0.0187	0.0183	0.0184	0.0189	0.0189
11. BFG 20-?-2-1	0.0192	0.0195	0.0203	0.0188	0.0200	0.0200	0.0201	0.0194	0.0189	0.0191
12. BFG 20-?-3-1	0.0191	0.0191	0.0196	0.0196	0.0194	0.0188	0.0184	0.0178	0.0178	0.0187
13. BFG 20-?-4-1	0.0186	0.0176	0.0174	0.0176	0.0178	0.0184	0.0182	0.0176	0.0180	0.0173
14. BFG 20-?-5-1	0.0188	0.0182	0.0179	0.0187	0.0179	0.0191	0.0188	0.0177	0.0186	0.0183
15. BFG 20-?-6-1	0.0186	0.0185	0.0181	0.0188	0.0171	0.0179	0.0176	0.0185	0.0175	0.0187
16. BFG 20-?-7-1	0.0179	0.0177	0.0178	0.0185	0.0189	0.0178	0.0182	0.0179	0.0180	0.0175
17. BFG 20-?-8-1	0.0172	0.0183	0.0193	0.0178	0.0181	0.0181	0.0183	0.0175	0.0175	0.0176
18. BFG 20-?-9-1	0.0177	0.0181	0.0176	0.0162	0.0171	0.0170	0.0176	0.0165	0.0173	0.0168
19. BFG 20-?-10-1	0.0178	0.0180	0.0181	0.0190	0.0179	0.0173	0.0186	0.0185	0.0185	0.0188
20. BFG 20-?-11-1	0.0175	0.0183	0.0181	0.0179	0.0178	0.0169	0.0176	0.0185	0.0187	0.0184
21. BFG 20-?-12-1	0.0185	0.0178	0.0186	0.0182	0.0181	0.0181	0.0185	0.0182	0.0181	0.0183
22. BFG 20-?-13-1	0.0182	0.0181	0.0182	0.0169	0.0173	0.0180	0.0184	0.0177	0.0175	0.0183
23. BFG 20-?-14-1	0.0183	0.0186	0.0193	0.0184	0.0187	0.0190	0.0175	0.0183	0.0183	0.0176
24. BFG 20-?-15-1	0.0194	0.0180	0.0178	0.0184	0.0181	0.0185	0.0177	0.0185	0.0188	0.0174
25. BFG 20-?-16-1	0.0177	0.0179	0.0177	0.0173	0.0178	0.0180	0.0183	0.0167	0.0171	0.0167
26. BFG 20-?-17-1	0.0184	0.0181	0.0179	0.0182	0.0188	0.0185	0.0178	0.0179	0.0176	0.0180
27. BFG 20-?-18-1	0.0186	0.0196	0.0190	0.0189	0.0180	0.0187	0.0179	0.0182	0.0187	0.0174
28. BFG 20-?-19-1	0.0182	0.0184	0.0186	0.0185	0.0182	0.0179	0.0184	0.0182	0.0181	0.0180
29. BFG 20-?-20-1	0.0184	0.0183	0.0183	0.0181	0.0183	0.0185	0.0187	0.0178	0.0179	0.0171

[1] For each simulation run, a sample partition of the E-K data set was randomly generated using the sample reuse procedure discussed in section 4.3.

[2] Each row represents the different network architectures and training schemes used in the simulations conducted in section 4.3. Each FFBANN used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, each FFBANN was batch trained using stopping rule (S5) with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-5}$ and $MAX = 1000$. These settings were used for all the simulations, unless otherwise noted. The first acronym used to describe the network architecture (e.g. BFG) indicates the algorithm used to train the network. The acronyms are: (i) BFG – BFGS quasi-Newton algorithm, (ii) CGF – conjugate gradient algorithm with Fletcher-Reeves update, (iii) GDX – steepest descent algorithm with adaptive learning and momentum rates, (iv) LM – Levenberg-Marquardt algorithm, (v) RP – resilient backpropagation algorithm, and (vi) SCG – scaled conjugate gradient algorithm (see section 3.2.3 and 4.3.2). The other acronyms indicate other specific changes. LOG is used when the hyperbolic tangent activation function in the hidden layer is replaced with the logistic activation function (see section 4.3.5). REG is used when regularization is built into the fitting criterion (see sections 3.2.3.3 and 4.3.6). VLD indicates that stopping rule (S5) is used along with REG (see section 4.3.6). The number of hidden layers and nodes in each layer is given by the string of numbers following the set of acronyms in each row. This is represented as (# of input variables) – (# of nodes in the first hidden layer) – (# of hidden nodes in the second hidden layer if applicable) – (# of output variables). The (# of nodes in the first hidden layer) is replaced by '?', to indicate that this number is determined by the column of the table you are in. The columns numbered 1 – 20, indicate the number of hidden nodes in the first hidden layer of the network.

Table B5: Mean Percent Correctly Classified using the Test Data Sets from the 500 Simulation Runs[1]

Network Architecture[2]/ # of Hidden Nodes	1	2	3	4	5	6	7	8	9	10
1. BFG 20-?-1	0.7442	0.7457	0.7457	0.7478	0.7502	0.7513	0.7504	0.7501	0.7501	0.7498
2. CGF 20-?-1	0.7187	0.7288	0.7409	0.7431	0.7431	0.7426	0.7381	0.7334	0.7330	0.7310
3. GDX 20-?-1	0.7309	0.7296	0.7345	0.7378	0.7382	0.7413	0.7385	0.7392	0.7377	0.7392
4. LM 20-?-1	0.7428	0.7395	0.7356	0.7371	0.7355	0.7347	0.7343	0.7301	0.7310	0.7290
5. RP 20-?-1	0.7421	0.7404	0.7395	0.7416	0.7434	0.7427	0.7432	0.7420	0.7423	0.7419
6. SCG 20-?-1	0.7430	0.7441	0.7437	0.7441	0.7457	0.7463	0.7472	0.7463	0.7452	0.7451
7. BFG/LOG 20-?-1	0.7396	0.7346	0.7426	0.7446	0.7471	0.7469	0.7470	0.7445	0.7464	0.7471
8. BFG/REG 20-?-1	0.7650	0.7642	0.7625	0.7561	0.7490	0.7403	0.7331	0.7278	0.7209	0.7164
9. BFG/REG/VLD 20-?-1	0.7605	0.7611	0.7604	0.7591	0.7572	0.7572	0.7553	0.7540	0.7535	0.7513
10. BFG 20-?-1-1	0.7396	0.7399	0.7462	0.7452	0.7469	0.7440	0.7462	0.7432	0.7446	0.7424
11. BFG 20-?-2-1	0.7320	0.7356	0.7343	0.7389	0.7356	0.7372	0.7391	0.7395	0.7381	0.7378
12. BFG 20-?-3-1	0.7236	0.7328	0.7381	0.7401	0.7424	0.7414	0.7436	0.7412	0.7407	0.7398
13. BFG 20-?-4-1	0.7162	0.7303	0.7371	0.7404	0.7418	0.7433	0.7418	0.7412	0.7395	0.7423
14. BFG 20-?-5-1	0.7090	0.7327	0.7419	0.7414	0.7429	0.7428	0.7439	0.7437	0.7433	0.7455
15. BFG 20-?-6-1	0.7073	0.7320	0.7404	0.7437	0.7441	0.7442	0.7431	0.7445	0.7425	0.7445
16. BFG 20-?-7-1	0.7001	0.7280	0.7418	0.7438	0.7438	0.7460	0.7428	0.7439	0.7427	0.7447
17. BFG 20-?-8-1	0.6960	0.7280	0.7391	0.7410	0.7416	0.7436	0.7446	0.7444	0.7432	0.7427
18. BFG 20-?-9-1	0.6948	0.7281	0.7355	0.7427	0.7435	0.7444	0.7439	0.7440	0.7432	0.7439
19. BFG 20-?-10-1	0.6945	0.7279	0.7392	0.7429	0.7438	0.7439	0.7427	0.7431	0.7417	0.7418
20. BFG 20-?-11-1	0.6875	0.7257	0.7408	0.7402	0.7430	0.7435	0.7436	0.7412	0.7435	0.7433
21. BFG 20-?-12-1	0.6891	0.7282	0.7380	0.7423	0.7434	0.7431	0.7426	0.7426	0.7431	0.7426
22. BFG 20-?-13-1	0.6894	0.7235	0.7390	0.7430	0.7440	0.7443	0.7456	0.7429	0.7447	0.7420
23. BFG 20-?-14-1	0.6853	0.7257	0.7388	0.7418	0.7417	0.7413	0.7427	0.7420	0.7432	0.7431
24. BFG 20-?-15-1	0.6869	0.7237	0.7401	0.7425	0.7427	0.7450	0.7434	0.7443	0.7423	0.7403
25. BFG 20-?-16-1	0.6843	0.7257	0.7372	0.7435	0.7443	0.7453	0.7435	0.7438	0.7432	0.7415
26. BFG 20-?-17-1	0.6846	0.7249	0.7373	0.7421	0.7418	0.7405	0.7436	0.7407	0.7403	0.7434
27. BFG 20-?-18-1	0.6826	0.7208	0.7371	0.7413	0.7419	0.7411	0.7417	0.7427	0.7439	0.7410
28. BFG 20-?-19-1	0.6795	0.7207	0.7384	0.7411	0.7410	0.7419	0.7410	0.7410	0.7418	0.7399
29. BFG 20-?-20-1	0.6800	0.7237	0.7384	0.7399	0.7423	0.7403	0.7403	0.7416	0.7410	0.7412

Table B5 continued

Network Architecture[2]/ # of Hidden Nodes	11	12	13	14	15	16	17	18	19	20
1. BFG 20-?-1	0.7476	0.7481	0.7489	0.7476	0.7470	0.7469	0.7485	0.7474	0.7474	0.7474
2. CGF 20-?-1	0.7306	0.7272	0.7274	0.7278	0.7249	0.7276	0.7247	0.7240	0.7278	0.7252
3. GDX 20-?-1	0.7402	0.7373	0.7362	0.7366	0.7374	0.7371	0.7365	0.7362	0.7358	0.7356
4. LM 20-?-1	0.7312	0.7303	0.7304	0.7293	0.7289	0.7255	0.7246	0.7234	0.7243	0.7216
5. RP 20-?-1	0.7421	0.7434	0.7410	0.7436	0.7432	0.7410	0.7408	0.7412	0.7428	0.7424
6. SCG 20-?-1	0.7466	0.7428	0.7432	0.7440	0.7451	0.7438	0.7427	0.7446	0.7427	0.7424
7. BFG/LOG 20-?-1	0.7461	0.7434	0.7449	0.7441	0.7464	0.7449	0.7444	0.7431	0.7428	0.7457
8. BFG/REG 20-?-1	0.7130	0.7111	0.7094	0.7072	0.7071	0.7070	0.7027	0.7039	0.7028	0.7027
9. BFG/REG/VLD 20-?-1	0.7523	0.7504	0.7499	0.7489	0.7475	0.7472	0.7464	0.7462	0.7471	0.7454
10. BFG 20-?-1-1	0.7433	0.7417	0.7421	0.7416	0.7388	0.7394	0.7382	0.7390	0.7410	0.7402
11. BFG 20-?-2-1	0.7371	0.7371	0.7375	0.7367	0.7352	0.7358	0.7350	0.7329	0.7370	0.7371
12. BFG 20-?-3-1	0.7405	0.7402	0.7395	0.7355	0.7392	0.7397	0.7375	0.7394	0.7384	0.7395
13. BFG 20-?-4-1	0.7415	0.7443	0.7407	0.7422	0.7407	0.7398	0.7408	0.7394	0.7364	0.7404
14. BFG 20-?-5-1	0.7420	0.7407	0.7423	0.7429	0.7413	0.7417	0.7393	0.7409	0.7399	0.7398
15. BFG 20-?-6-1	0.7430	0.7420	0.7416	0.7408	0.7419	0.7389	0.7405	0.7405	0.7408	0.7395
16. BFG 20-?-7-1	0.7411	0.7416	0.7420	0.7422	0.7416	0.7434	0.7425	0.7417	0.7400	0.7407
17. BFG 20-?-8-1	0.7432	0.7424	0.7409	0.7427	0.7408	0.7423	0.7403	0.7403	0.7401	0.7400
18. BFG 20-?-9-1	0.7439	0.7407	0.7420	0.7437	0.7428	0.7431	0.7418	0.7404	0.7417	0.7411
19. BFG 20-?-10-1	0.7402	0.7424	0.7397	0.7413	0.7399	0.7418	0.7397	0.7377	0.7420	0.7399
20. BFG 20-?-11-1	0.7430	0.7420	0.7407	0.7436	0.7402	0.7406	0.7412	0.7404	0.7395	0.7410
21. BFG 20-?-12-1	0.7421	0.7412	0.7388	0.7396	0.7409	0.7414	0.7427	0.7397	0.7378	0.7408
22. BFG 20-?-13-1	0.7428	0.7448	0.7425	0.7445	0.7402	0.7414	0.7421	0.7428	0.7430	0.7393
23. BFG 20-?-14-1	0.7433	0.7416	0.7416	0.7421	0.7405	0.7422	0.7398	0.7393	0.7395	0.7397
24. BFG 20-?-15-1	0.7406	0.7431	0.7386	0.7392	0.7393	0.7415	0.7419	0.7408	0.7417	0.7382
25. BFG 20-?-16-1	0.7425	0.7409	0.7405	0.7428	0.7415	0.7410	0.7416	0.7416	0.7411	0.7417
26. BFG 20-?-17-1	0.7412	0.7415	0.7411	0.7411	0.7406	0.7401	0.7397	0.7393	0.7409	0.7380
27. BFG 20-?-18-1	0.7400	0.7401	0.7404	0.7402	0.7414	0.7410	0.7391	0.7399	0.7397	0.7384
28. BFG 20-?-19-1	0.7408	0.7411	0.7397	0.7387	0.7399	0.7390	0.7418	0.7399	0.7398	0.7395
29. BFG 20-?-20-1	0.7410	0.7433	0.7420	0.7403	0.7401	0.7387	0.7392	0.7404	0.7377	0.7383

[1] For each simulation run, a sample partition of the E-K data set was randomly generated using the sample reuse procedure discussed in section 4.3.

[2] Each row represents the different network architectures and training schemes used in the simulations conducted in section 4.3. Each FFBANN used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, each FFBANN was batch trained using stopping rule (S5) with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-5}$ and $MAX = 1000$. These settings were used for all the simulations, unless otherwise noted. The first acronym used to describe the network architecture (e.g. BFG) indicates the algorithm used to train the network. The acronyms are: (i) BFG – BFGS quasi-Newton algorithm, (ii) CGF – conjugate gradient algorithm with Fletcher-Reeves update, (iii) GDX – steepest descent algorithm with adaptive learning and momentum rates, (iv) LM – Levenberg-Marquardt algorithm, (v) RP – resilient backpropagation algorithm, and (vi) SCG – scaled conjugate gradient algorithm (see section 3.2.3 and 4.3.2). The other acronyms indicate other specific changes. LOG is used when the hyperbolic tangent activation function in the hidden layer is replaced with the logistic activation function (see section 4.3.5). REG is used when regularization is built into the fitting criterion (see sections 3.2.3.3 and 4.3.6). VLD indicates that stopping rule (S5) is used along with REG (see section 4.3.6). The number of hidden layers and nodes in each layer is given by the string of numbers following the set of acronyms in each row. This is represented as (# of input variables) – (# of nodes in the first hidden layer) – (# of hidden nodes in the second hidden layer if applicable) – (# of output variables). The (# of nodes in the first hidden layer) is replaced by ‘?’, to indicate that this number is determined by the column of the table you are in. The columns numbered 1 – 20, indicate the number of hidden nodes in the first hidden layer of the network.

Table B6: Standard Deviation of the Percent Correctly Classified using the Test Data Sets from the 500 Simulation Runs[1]

Network Architecture[2]/ # of Hidden Nodes	1	2	3	4	5	6	7	8	9	10
1. BFG 20-?-1	0.0398	0.0325	0.0316	0.0304	0.0303	0.0297	0.0305	0.0311	0.0297	0.0304
2. CGF 20-?-1	0.0428	0.0431	0.0393	0.0368	0.0386	0.0361	0.0430	0.0444	0.0436	0.0451
3. GDX 20-?-1	0.0487	0.0458	0.0448	0.0445	0.0427	0.0391	0.0434	0.0416	0.0413	0.0387
4. LM 20-?-1	0.0411	0.0371	0.0337	0.0339	0.0337	0.0328	0.0330	0.0338	0.0350	0.0350
5. RP 20-?-1	0.0334	0.0350	0.0323	0.0324	0.0321	0.0316	0.0315	0.0319	0.0312	0.0313
6. SCG 20-?-1	0.0371	0.0363	0.0416	0.0389	0.0356	0.0338	0.0345	0.0318	0.0360	0.0361
7. BFG/LOG 20-?-1	0.0388	0.0398	0.0364	0.0351	0.0346	0.0327	0.0325	0.0324	0.0306	0.0315
8. BFG/REG 20-?-1	0.0204	0.0198	0.0207	0.0220	0.0229	0.0226	0.0235	0.0225	0.0242	0.0222
9. BFG/REG/VLD 20-?-1	0.0305	0.0305	0.0304	0.0307	0.0300	0.0304	0.0295	0.0304	0.0303	0.0291
10. BFG 20-?-1-1	0.0381	0.0376	0.0358	0.0337	0.0345	0.0338	0.0318	0.0336	0.0324	0.0320
11. BFG 20-?-2-1	0.0407	0.0398	0.0410	0.0383	0.0387	0.0371	0.0371	0.0340	0.0361	0.0356
12. BFG 20-?-3-1	0.0436	0.0402	0.0376	0.0349	0.0334	0.0350	0.0345	0.0352	0.0340	0.0330
13. BFG 20-?-4-1	0.0478	0.0398	0.0368	0.0338	0.0339	0.0333	0.0338	0.0330	0.0342	0.0329
14. BFG 20-?-5-1	0.0453	0.0400	0.0343	0.0363	0.0321	0.0331	0.0339	0.0327	0.0320	0.0332
15. BFG 20-?-6-1	0.0481	0.0401	0.0363	0.0350	0.0334	0.0326	0.0321	0.0311	0.0331	0.0313
16. BFG 20-?-7-1	0.0480	0.0419	0.0371	0.0343	0.0361	0.0326	0.0326	0.0324	0.0314	0.0318
17. BFG 20-?-8-1	0.0451	0.0427	0.0365	0.0343	0.0330	0.0333	0.0334	0.0345	0.0314	0.0327
18. BFG 20-?-9-1	0.0468	0.0412	0.0364	0.0339	0.0312	0.0312	0.0316	0.0319	0.0325	0.0308
19. BFG 20-?-10-1	0.0469	0.0428	0.0368	0.0334	0.0348	0.0319	0.0353	0.0319	0.0329	0.0317
20. BFG 20-?-11-1	0.0455	0.0430	0.0364	0.0335	0.0344	0.0317	0.0308	0.0326	0.0322	0.0327
21. BFG 20-?-12-1	0.0439	0.0414	0.0383	0.0340	0.0341	0.0319	0.0329	0.0325	0.0349	0.0304
22. BFG 20-?-13-1	0.0461	0.0448	0.0357	0.0351	0.0330	0.0305	0.0314	0.0307	0.0328	0.0323
23. BFG 20-?-14-1	0.0447	0.0419	0.0371	0.0340	0.0331	0.0345	0.0327	0.0331	0.0335	0.0329
24. BFG 20-?-15-1	0.0434	0.0455	0.0336	0.0346	0.0327	0.0322	0.0326	0.0315	0.0308	0.0337
25. BFG 20-?-16-1	0.0436	0.0430	0.0384	0.0333	0.0325	0.0341	0.0341	0.0322	0.0320	0.0328
26. BFG 20-?-17-1	0.0428	0.0440	0.0378	0.0346	0.0320	0.0318	0.0321	0.0347	0.0332	0.0336
27. BFG 20-?-18-1	0.0456	0.0421	0.0363	0.0353	0.0317	0.0330	0.0328	0.0324	0.0331	0.0323
28. BFG 20-?-19-1	0.0418	0.0443	0.0368	0.0351	0.0335	0.0335	0.0326	0.0315	0.0330	0.0335
29. BFG 20-?-20-1	0.0435	0.0430	0.0376	0.0350	0.0336	0.0352	0.0342	0.0334	0.0339	0.0327

Table B6 continued

Network Architecture[2]/										
# of Hidden Nodes	11	12	13	14	15	16	17	18	19	20
1. BFG 20-?-1	0.0309	0.0307	0.0292	0.0309	0.0292	0.0312	0.0279	0.0296	0.0280	0.0286
2. CGF 20-?-1	0.0468	0.0482	0.0450	0.0491	0.0474	0.0440	0.0446	0.0435	0.0448	0.0442
3. GDX 20-?-1	0.0369	0.0408	0.0427	0.0390	0.0401	0.0420	0.0395	0.0404	0.0398	0.0411
4. LM 20-?-1	0.0329	0.0330	0.0338	0.0348	0.0330	0.0348	0.0353	0.0365	0.0348	0.0366
5. RP 20-?-1	0.0310	0.0315	0.0308	0.0305	0.0311	0.0317	0.0317	0.0311	0.0316	0.0315
6. SCG 20-?-1	0.0335	0.0427	0.0453	0.0395	0.0356	0.0390	0.0396	0.0331	0.0400	0.0361
7. BFG/LOG 20-?-1	0.0326	0.0333	0.0316	0.0331	0.0324	0.0312	0.0293	0.0322	0.0334	0.0322
8. BFG/REG 20-?-1	0.0235	0.0224	0.0235	0.0231	0.0239	0.0253	0.0233	0.0237	0.0217	0.0243
9. BFG/REG/VLD 20-?-1	0.0300	0.0312	0.0307	0.0313	0.0315	0.0303	0.0321	0.0307	0.0319	0.0312
10. BFG 20-?-1-1	0.0323	0.0329	0.0334	0.0330	0.0324	0.0327	0.0315	0.0327	0.0322	0.0324
11. BFG 20-?-2-1	0.0339	0.0342	0.0377	0.0342	0.0346	0.0341	0.0360	0.0343	0.0335	0.0336
12. BFG 20-?-3-1	0.0343	0.0344	0.0335	0.0350	0.0342	0.0333	0.0343	0.0321	0.0311	0.0330
13. BFG 20-?-4-1	0.0334	0.0323	0.0318	0.0320	0.0333	0.0322	0.0323	0.0326	0.0330	0.0296
14. BFG 20-?-5-1	0.0333	0.0328	0.0318	0.0324	0.0320	0.0337	0.0336	0.0319	0.0328	0.0336
15. BFG 20-?-6-1	0.0339	0.0334	0.0319	0.0339	0.0307	0.0318	0.0322	0.0326	0.0316	0.0336
16. BFG 20-?-7-1	0.0327	0.0313	0.0302	0.0328	0.0337	0.0329	0.0316	0.0317	0.0334	0.0309
17. BFG 20-?-8-1	0.0304	0.0331	0.0343	0.0336	0.0309	0.0333	0.0330	0.0333	0.0315	0.0310
18. BFG 20-?-9-1	0.0325	0.0319	0.0322	0.0293	0.0306	0.0306	0.0323	0.0308	0.0305	0.0312
19. BFG 20-?-10-1	0.0315	0.0323	0.0312	0.0333	0.0328	0.0307	0.0338	0.0318	0.0325	0.0319
20. BFG 20-?-11-1	0.0318	0.0328	0.0323	0.0327	0.0323	0.0295	0.0318	0.0327	0.0325	0.0327
21. BFG 20-?-12-1	0.0317	0.0318	0.0331	0.0332	0.0309	0.0313	0.0320	0.0322	0.0324	0.0312
22. BFG 20-?-13-1	0.0317	0.0326	0.0330	0.0305	0.0309	0.0320	0.0322	0.0315	0.0315	0.0324
23. BFG 20-?-14-1	0.0306	0.0335	0.0337	0.0325	0.0332	0.0341	0.0323	0.0327	0.0331	0.0322
24. BFG 20-?-15-1	0.0339	0.0314	0.0322	0.0322	0.0322	0.0323	0.0312	0.0335	0.0326	0.0312
25. BFG 20-?-16-1	0.0312	0.0325	0.0322	0.0323	0.0317	0.0320	0.0321	0.0304	0.0307	0.0296
26. BFG 20-?-17-1	0.0341	0.0319	0.0322	0.0327	0.0332	0.0329	0.0317	0.0318	0.0302	0.0328
27. BFG 20-?-18-1	0.0325	0.0330	0.0327	0.0333	0.0315	0.0325	0.0327	0.0325	0.0329	0.0319
28. BFG 20-?-19-1	0.0323	0.0334	0.0331	0.0324	0.0324	0.0325	0.0325	0.0315	0.0324	0.0326
29. BFG 20-?-20-1	0.0326	0.0314	0.0329	0.0310	0.0301	0.0326	0.0334	0.0318	0.0317	0.0300

[1] For each simulation run, a sample partition of the E-K data set was randomly generated using the sample reuse procedure discussed in section 4.3.

[2] Each row represents the different network architectures and training schemes used in the simulations conducted in section 4.3. Each FFBANN used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, each FFBANN was batch trained using stopping rule (S5) with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-5}$ and $MAX = 1000$. These settings were used for all the simulations, unless otherwise noted. The first acronym used to describe the network architecture (e.g. BFG) indicates the algorithm used to train the network. The acronyms are: (i) BFG – BFGS quasi-Newton algorithm, (ii) CGF – conjugate gradient algorithm with Fletcher-Reeves update, (iii) GDX – steepest descent algorithm with adaptive learning and momentum rates, (iv) LM – Levenberg-Marquardt algorithm, (v) RP – resilient backpropagation algorithm, and (vi) SCG – scaled conjugate gradient algorithm (see section 3.2.3 and 4.3.2). The other acronyms indicate other specific changes. LOG is used when the hyperbolic tangent activation function in the hidden layer is replaced with the logistic activation function (see section 4.3.5). REG is used when regularization is built into the fitting criterion (see sections 3.2.3.3 and 4.3.6). VLD indicates that stopping rule (S5) is used along with REG (see section 4.3.6). The number of hidden layers and nodes in each layer is given by the string of numbers following the set of acronyms in each row. This is represented as (# of input variables) – (# of nodes in the first hidden layer) – (# of hidden nodes in the second hidden layer if applicable) – (# of output variables). The (# of nodes in the first hidden layer) is replaced by '?', to indicate that this number is determined by the column of the table you are in. The columns numbered 1 – 20, indicate the number of hidden nodes in the first hidden layer of the network.

Table B7: Mean Percent Correctly Classified using the Entire E-K Data Set for all 500 Simulation Runs[1]

Network Architecture[2]/ # of Hidden Nodes	1	2	3	4	5	6	7	8	9	10
1. BFG 20-?-1	0.7638	0.7618	0.7646	0.7671	0.7694	0.7718	0.7717	0.7720	0.7734	0.7739
2. CGF 20-?-1	0.7292	0.7446	0.7608	0.7639	0.7635	0.7618	0.7578	0.7545	0.7542	0.7537
3. GDX 20-?-1	0.7499	0.7494	0.7578	0.7629	0.7634	0.7672	0.7647	0.7678	0.7665	0.7697
4. LM 20-?-1	0.7634	0.7675	0.7693	0.7718	0.7715	0.7720	0.7747	0.7713	0.7725	0.7727
5. RP 20-?-1	0.7645	0.7601	0.7672	0.7695	0.7728	0.7733	0.7741	0.7766	0.7769	0.7791
6. SCG 20-?-1	0.7635	0.7655	0.7654	0.7670	0.7709	0.7707	0.7722	0.7733	0.7726	0.7740
7. BFG/LOG 20-?-1	0.7562	0.7500	0.7592	0.7623	0.7638	0.7652	0.7656	0.7662	0.7683	0.7699
8. BFG/REG 20-?-1	0.7807	0.7815	0.7843	0.7943	0.8101	0.8262	0.8415	0.8550	0.8668	0.8768
9. BFG/REG/VLD 20-?-1	0.7755	0.7783	0.7789	0.7793	0.7792	0.7797	0.7810	0.7813	0.7826	0.7826
10. BFG 20-?-1-1	0.7561	0.7591	0.7659	0.7671	0.7687	0.7689	0.7710	0.7695	0.7713	0.7695
11. BFG 20-?-2-1	0.7469	0.7507	0.7516	0.7579	0.7563	0.7594	0.7622	0.7641	0.7626	0.7633
12. BFG 20-?-3-1	0.7361	0.7480	0.7564	0.7616	0.7634	0.7629	0.7669	0.7661	0.7669	0.7659
13. BFG 20-?-4-1	0.7292	0.7483	0.7572	0.7615	0.7623	0.7670	0.7665	0.7669	0.7672	0.7684
14. BFG 20-?-5-1	0.7255	0.7471	0.7611	0.7619	0.7659	0.7654	0.7679	0.7706	0.7690	0.7714
15. BFG 20-?-6-1	0.7219	0.7499	0.7610	0.7633	0.7659	0.7674	0.7680	0.7703	0.7704	0.7714
16. BFG 20-?-7-1	0.7170	0.7450	0.7601	0.7646	0.7669	0.7697	0.7689	0.7706	0.7714	0.7721
17. BFG 20-?-8-1	0.7121	0.7450	0.7594	0.7641	0.7660	0.7684	0.7707	0.7709	0.7714	0.7711
18. BFG 20-?-9-1	0.7127	0.7456	0.7569	0.7647	0.7668	0.7691	0.7708	0.7710	0.7700	0.7720
19. BFG 20-?-10-1	0.7129	0.7452	0.7598	0.7653	0.7669	0.7695	0.7687	0.7712	0.7722	0.7728
20. BFG 20-?-11-1	0.7046	0.7428	0.7605	0.7642	0.7665	0.7691	0.7712	0.7705	0.7731	0.7729
21. BFG 20-?-12-1	0.7057	0.7466	0.7592	0.7661	0.7666	0.7678	0.7703	0.7728	0.7733	0.7746
22. BFG 20-?-13-1	0.7063	0.7430	0.7605	0.7647	0.7660	0.7701	0.7714	0.7715	0.7720	0.7730
23. BFG 20-?-14-1	0.7020	0.7425	0.7621	0.7637	0.7688	0.7684	0.7715	0.7731	0.7741	0.7727
24. BFG 20-?-15-1	0.7041	0.7426	0.7615	0.7663	0.7693	0.7710	0.7717	0.7740	0.7733	0.7744
25. BFG 20-?-16-1	0.7018	0.7439	0.7587	0.7652	0.7667	0.7700	0.7718	0.7725	0.7723	0.7743
26. BFG 20-?-17-1	0.7036	0.7422	0.7621	0.7666	0.7678	0.7700	0.7722	0.7720	0.7719	0.7752
27. BFG 20-?-18-1	0.7006	0.7406	0.7612	0.7657	0.7679	0.7698	0.7709	0.7724	0.7741	0.7751
28. BFG 20-?-19-1	0.6977	0.7408	0.7604	0.7660	0.7676	0.7711	0.7718	0.7731	0.7732	0.7748
29. BFG 20-?-20-1	0.6974	0.7438	0.7623	0.7650	0.7686	0.7687	0.7726	0.7727	0.7727	0.7752

Table B7 continued

Network Architecture[2]/										
# of Hidden Nodes	11	12	13	14	15	16	17	18	19	20
1. BFG 20-?-1	0.7725	0.7743	0.7747	0.7751	0.7751	0.7748	0.7773	0.7757	0.7770	0.7754
2. CGF 20-?-1	0.7505	0.7468	0.7486	0.7493	0.7453	0.7494	0.7460	0.7474	0.7510	0.7491
3. GDX 20-?-1	0.7695	0.7682	0.7685	0.7675	0.7689	0.7695	0.7699	0.7698	0.7692	0.7707
4. LM 20-?-1	0.7740	0.7743	0.7758	0.7743	0.7748	0.7741	0.7710	0.7763	0.7768	0.7749
5. RP 20-?-1	0.7780	0.7795	0.7791	0.7816	0.7809	0.7808	0.7814	0.7809	0.7848	0.7841
6. SCG 20-?-1	0.7746	0.7720	0.7734	0.7744	0.7764	0.7743	0.7744	0.7768	0.7751	0.7765
7. BFG/LOG 20-?-1	0.7683	0.7678	0.7692	0.7696	0.7722	0.7704	0.7715	0.7709	0.7707	0.7735
8. BFG/REG 20-?-1	0.8849	0.8913	0.8962	0.8992	0.9016	0.9034	0.9035	0.9046	0.9050	0.9059
9. BFG/REG/VLD 20-?-1	0.7820	0.7837	0.7843	0.7840	0.7839	0.7843	0.7836	0.7833	0.7843	0.7846
10. BFG 20-?-1-1	0.7718	0.7712	0.7720	0.7731	0.7703	0.7714	0.7717	0.7735	0.7748	0.7736
11. BFG 20-?-2-1	0.7640	0.7644	0.7635	0.7637	0.7641	0.7650	0.7642	0.7637	0.7671	0.7669
12. BFG 20-?-3-1	0.7675	0.7671	0.7676	0.7664	0.7687	0.7699	0.7684	0.7695	0.7690	0.7694
13. BFG 20-?-4-1	0.7691	0.7708	0.7712	0.7713	0.7712	0.7693	0.7702	0.7718	0.7692	0.7711
14. BFG 20-?-5-1	0.7706	0.7704	0.7708	0.7725	0.7724	0.7718	0.7704	0.7727	0.7726	0.7735
15. BFG 20-?-6-1	0.7711	0.7712	0.7714	0.7726	0.7722	0.7709	0.7727	0.7727	0.7735	0.7743
16. BFG 20-?-7-1	0.7718	0.7726	0.7732	0.7732	0.7736	0.7745	0.7724	0.7732	0.7740	0.7754
17. BFG 20-?-8-1	0.7728	0.7731	0.7725	0.7740	0.7731	0.7738	0.7738	0.7744	0.7754	0.7748
18. BFG 20-?-9-1	0.7727	0.7716	0.7731	0.7749	0.7760	0.7762	0.7742	0.7749	0.7750	0.7767
19. BFG 20-?-10-1	0.7730	0.7740	0.7732	0.7735	0.7740	0.7756	0.7748	0.7749	0.7768	0.7760
20. BFG 20-?-11-1	0.7742	0.7747	0.7740	0.7756	0.7754	0.7757	0.7767	0.7756	0.7760	0.7774
21. BFG 20-?-12-1	0.7742	0.7745	0.7739	0.7737	0.7753	0.7760	0.7762	0.7754	0.7743	0.7776
22. BFG 20-?-13-1	0.7750	0.7765	0.7757	0.7769	0.7761	0.7752	0.7767	0.7780	0.7773	0.7767
23. BFG 20-?-14-1	0.7751	0.7755	0.7763	0.7766	0.7765	0.7774	0.7773	0.7766	0.7768	0.7771
24. BFG 20-?-15-1	0.7743	0.7753	0.7749	0.7747	0.7752	0.7773	0.7772	0.7764	0.7779	0.7763
25. BFG 20-?-16-1	0.7745	0.7739	0.7757	0.7750	0.7766	0.7769	0.7766	0.7774	0.7770	0.7774
26. BFG 20-?-17-1	0.7746	0.7750	0.7765	0.7766	0.7775	0.7785	0.7781	0.7781	0.7785	0.7761
27. BFG 20-?-18-1	0.7741	0.7759	0.7753	0.7762	0.7769	0.7776	0.7779	0.7780	0.7776	0.7778
28. BFG 20-?-19-1	0.7763	0.7768	0.7749	0.7769	0.7782	0.7778	0.7778	0.7777	0.7796	0.7777
29. BFG 20-?-20-1	0.7750	0.7765	0.7775	0.7766	0.7773	0.7764	0.7781	0.7789	0.7780	0.7774

[1] For each simulation run, a sample partition of the E-K data set was randomly generated using the sample reuse procedure discussed in section 4.3.

[2] Each row represents the different network architectures and training schemes used in the simulations conducted in section 4.3. Each FFBANN used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, each FFBANN was batch trained using stopping rule (S5) with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-5}$ and $MAX = 1000$. These settings were used for all the simulations, unless otherwise noted. The first acronym used to describe the network architecture (e.g. BFG) indicates the algorithm used to train the network. The acronyms are: (i) BFG – BFGS quasi-Newton algorithm, (ii) CGF – conjugate gradient algorithm with Fletcher-Reeves update, (iii) GDX – steepest descent algorithm with adaptive learning and momentum rates, (iv) LM – Levenberg-Marquardt algorithm, (v) RP – resilient backpropagation algorithm, and (vi) SCG – scaled conjugate gradient algorithm (see section 3.2.3 and 4.3.2). The other acronyms indicate other specific changes. LOG is used when the hyperbolic tangent activation function in the hidden layer is replaced with the logistic activation function (see section 4.3.5). REG is used when regularization is built into the fitting criterion (see sections 3.2.3.3 and 4.3.6). VLD indicates that stopping rule (S5) is used along with REG (see section 4.3.6). The number of hidden layers and nodes in each layer is given by the string of numbers following the set of acronyms in each row. This is represented as (# of input variables) – (# of nodes in the first hidden layer) – (# of hidden nodes in the second hidden layer if applicable) – (# of output variables). The (# of nodes in the first hidden layer) is replaced by '?', to indicate that this number is determined by the column of the table you are in. The columns numbered 1 – 20, indicate the number of hidden nodes in the first hidden layer of the network.

Table B8: Standard Deviation of the Percent Correctly Classified using the Entire E-K Data Set for all 500 Simulation Runs[1]

Network Architecture[2]/ # of Hidden Nodes	1	2	3	4	5	6	7	8	9	10
1. BFG 20-?-1	0.0323	0.0245	0.0183	0.0147	0.0157	0.0129	0.0142	0.0149	0.0140	0.0137
2. CGF 20-?-1	0.0392	0.0394	0.0290	0.0272	0.0302	0.0300	0.0377	0.0394	0.0402	0.0429
3. GDX 20-?-1	0.0474	0.0438	0.0416	0.0374	0.0389	0.0332	0.0377	0.0357	0.0384	0.0332
4. LM 20-?-1	0.0317	0.0227	0.0206	0.0186	0.0188	0.0197	0.0185	0.0201	0.0230	0.0211
5. RP 20-?-1	0.0159	0.0186	0.0150	0.0172	0.0165	0.0159	0.0167	0.0170	0.0165	0.0162
6. SCG 20-?-1	0.0324	0.0266	0.0328	0.0294	0.0249	0.0239	0.0198	0.0186	0.0260	0.0246
7. BFG/LOG 20-?-1	0.0288	0.0319	0.0219	0.0200	0.0193	0.0179	0.0179	0.0171	0.0159	0.0156
8. BFG/REG 20-?-1	0.0047	0.0050	0.0053	0.0063	0.0079	0.0080	0.0085	0.0082	0.0085	0.0074
9. BFG/REG/VLD 20-?-1	0.0085	0.0068	0.0068	0.0074	0.0084	0.0092	0.0094	0.0101	0.0105	0.0109
10. BFG 20-?-1-1	0.0291	0.0263	0.0186	0.0187	0.0187	0.0197	0.0165	0.0206	0.0188	0.0191
11. BFG 20-?-2-1	0.0360	0.0329	0.0336	0.0294	0.0312	0.0288	0.0283	0.0255	0.0258	0.0273
12. BFG 20-?-3-1	0.0402	0.0359	0.0288	0.0242	0.0223	0.0243	0.0223	0.0218	0.0211	0.0243
13. BFG 20-?-4-1	0.0421	0.0341	0.0277	0.0237	0.0216	0.0203	0.0210	0.0205	0.0217	0.0207
14. BFG 20-?-5-1	0.0414	0.0349	0.0243	0.0221	0.0194	0.0212	0.0183	0.0178	0.0178	0.0188
15. BFG 20-?-6-1	0.0406	0.0336	0.0263	0.0218	0.0175	0.0175	0.0173	0.0170	0.0180	0.0172
16. BFG 20-?-7-1	0.0408	0.0370	0.0258	0.0210	0.0174	0.0158	0.0192	0.0168	0.0170	0.0169
17. BFG 20-?-8-1	0.0385	0.0366	0.0262	0.0233	0.0198	0.0190	0.0175	0.0178	0.0173	0.0191
18. BFG 20-?-9-1	0.0395	0.0356	0.0288	0.0209	0.0194	0.0186	0.0173	0.0181	0.0202	0.0184
19. BFG 20-?-10-1	0.0387	0.0367	0.0261	0.0202	0.0193	0.0164	0.0216	0.0176	0.0174	0.0169
20. BFG 20-?-11-1	0.0361	0.0374	0.0255	0.0209	0.0200	0.0188	0.0161	0.0176	0.0180	0.0189
21. BFG 20-?-12-1	0.0350	0.0352	0.0287	0.0201	0.0193	0.0179	0.0171	0.0172	0.0181	0.0156
22. BFG 20-?-13-1	0.0372	0.0385	0.0272	0.0217	0.0205	0.0183	0.0169	0.0175	0.0179	0.0202
23. BFG 20-?-14-1	0.0354	0.0371	0.0249	0.0232	0.0193	0.0192	0.0162	0.0167	0.0185	0.0192
24. BFG 20-?-15-1	0.0361	0.0383	0.0245	0.0216	0.0203	0.0161	0.0162	0.0171	0.0175	0.0176
25. BFG 20-?-16-1	0.0352	0.0361	0.0276	0.0205	0.0204	0.0190	0.0167	0.0176	0.0193	0.0189
26. BFG 20-?-17-1	0.0346	0.0376	0.0255	0.0194	0.0198	0.0183	0.0176	0.0185	0.0188	0.0174
27. BFG 20-?-18-1	0.0348	0.0378	0.0258	0.0220	0.0192	0.0194	0.0183	0.0170	0.0180	0.0178
28. BFG 20-?-19-1	0.0340	0.0368	0.0274	0.0204	0.0197	0.0182	0.0179	0.0171	0.0200	0.0190
29. BFG 20-?-20-1	0.0331	0.0383	0.0269	0.0228	0.0201	0.0222	0.0184	0.0192	0.0192	0.0191

Table B8 continued

Network Architecture[2]/										
# of Hidden Nodes	11	12	13	14	15	16	17	18	19	20
1. BFG 20-?-1	0.0150	0.0145	0.0144	0.0145	0.0138	0.0161	0.0129	0.0150	0.0153	0.0148
2. CGF 20-?-1	0.0445	0.0453	0.0439	0.0461	0.0478	0.0450	0.0455	0.0455	0.0415	0.0443
3. GDX 20-?-1	0.0330	0.0362	0.0374	0.0370	0.0364	0.0371	0.0357	0.0357	0.0366	0.0367
4. LM 20-?-1	0.0221	0.0202	0.0194	0.0248	0.0214	0.0250	0.0235	0.0241	0.0241	0.0273
5. RP 20-?-1	0.0178	0.0174	0.0186	0.0177	0.0193	0.0190	0.0183	0.0194	0.0204	0.0209
6. SCG 20-?-1	0.0226	0.0359	0.0369	0.0281	0.0225	0.0318	0.0306	0.0204	0.0313	0.0264
7. BFG/LOG 20-?-1	0.0170	0.0183	0.0170	0.0181	0.0185	0.0173	0.0173	0.0177	0.0180	0.0175
8. BFG/REG 20-?-1	0.0075	0.0075	0.0074	0.0074	0.0074	0.0078	0.0074	0.0072	0.0067	0.0073
9. BFG/REG/VLD 20-?-1	0.0127	0.0130	0.0138	0.0136	0.0148	0.0151	0.0162	0.0160	0.0166	0.0156
10. BFG 20-?-1-1	0.0198	0.0201	0.0199	0.0185	0.0198	0.0195	0.0193	0.0191	0.0190	0.0202
11. BFG 20-?-2-1	0.0260	0.0267	0.0280	0.0269	0.0259	0.0257	0.0267	0.0259	0.0240	0.0246
12. BFG 20-?-3-1	0.0238	0.0217	0.0218	0.0245	0.0224	0.0221	0.0230	0.0224	0.0211	0.0216
13. BFG 20-?-4-1	0.0208	0.0179	0.0195	0.0186	0.0188	0.0227	0.0187	0.0201	0.0209	0.0213
14. BFG 20-?-5-1	0.0200	0.0195	0.0183	0.0188	0.0179	0.0188	0.0209	0.0193	0.0202	0.0184
15. BFG 20-?-6-1	0.0172	0.0182	0.0190	0.0195	0.0185	0.0195	0.0183	0.0188	0.0182	0.0182
16. BFG 20-?-7-1	0.0179	0.0176	0.0178	0.0192	0.0172	0.0171	0.0193	0.0183	0.0185	0.0176
17. BFG 20-?-8-1	0.0173	0.0178	0.0190	0.0175	0.0187	0.0183	0.0184	0.0195	0.0186	0.0188
18. BFG 20-?-9-1	0.0188	0.0191	0.0194	0.0189	0.0162	0.0185	0.0199	0.0184	0.0194	0.0185
19. BFG 20-?-10-1	0.0179	0.0184	0.0192	0.0189	0.0189	0.0186	0.0185	0.0188	0.0183	0.0185
20. BFG 20-?-11-1	0.0171	0.0165	0.0172	0.0169	0.0179	0.0184	0.0181	0.0202	0.0186	0.0183
21. BFG 20-?-12-1	0.0175	0.0167	0.0185	0.0203	0.0183	0.0191	0.0197	0.0183	0.0201	0.0181
22. BFG 20-?-13-1	0.0193	0.0176	0.0190	0.0161	0.0188	0.0203	0.0185	0.0190	0.0200	0.0195
23. BFG 20-?-14-1	0.0177	0.0186	0.0170	0.0183	0.0190	0.0178	0.0190	0.0189	0.0195	0.0182
24. BFG 20-?-15-1	0.0189	0.0166	0.0187	0.0195	0.0183	0.0188	0.0177	0.0188	0.0183	0.0195
25. BFG 20-?-16-1	0.0192	0.0199	0.0181	0.0191	0.0197	0.0188	0.0181	0.0185	0.0196	0.0191
26. BFG 20-?-17-1	0.0171	0.0188	0.0183	0.0188	0.0192	0.0180	0.0183	0.0188	0.0191	0.0189
27. BFG 20-?-18-1	0.0189	0.0182	0.0201	0.0189	0.0181	0.0191	0.0182	0.0198	0.0195	0.0206
28. BFG 20-?-19-1	0.0170	0.0181	0.0185	0.0187	0.0185	0.0194	0.0194	0.0184	0.0183	0.0175
29. BFG 20-?-20-1	0.0191	0.0183	0.0197	0.0197	0.0178	0.0208	0.0190	0.0200	0.0191	0.0191

[1] For each simulation run, a sample partition of the E-K data set was randomly generated using the sample reuse procedure discussed in section 4.3.

[2] Each row represents the different network architectures and training schemes used in the simulations conducted in section 4.3. Each FFBANN used the hyperbolic tangent activation function in the hidden layer and logistic activation function in the output layer. Furthermore, each FFBANN was batch trained using stopping rule (S5) with $v = 5$, as well as rules (S2), (S3) and (S4) with $e = 1\times10^{-15}$, $E_{\min} = 1\times10^{-5}$ and $MAX = 1000$. These settings were used for all the simulations, unless otherwise noted. The first acronym used to describe the network architecture (e.g. BFG) indicates the algorithm used to train the network. The acronyms are: (i) BFG – BFGS quasi-Newton algorithm, (ii) CGF – conjugate gradient algorithm with Fletcher-Reeves update, (iii) GDX – steepest descent algorithm with adaptive learning and momentum rates, (iv) LM – Levenberg-Marquardt algorithm, (v) RP – resilient backpropagation algorithm, and (vi) SCG – scaled conjugate gradient algorithm (see section 3.2.3 and 4.3.2). The other acronyms indicate other specific changes. LOG is used when the hyperbolic tangent activation function in the hidden layer is replaced with the logistic activation function (see section 4.3.5). REG is used when regularization is built into the fitting criterion (see sections 3.2.3.3 and 4.3.6). VLD indicates that stopping rule (S5) is used along with REG (see section 4.3.6). The number of hidden layers and nodes in each layer is given by the string of numbers following the set of acronyms in each row. This is represented as (# of input variables) – (# of nodes in the first hidden layer) – (# of hidden nodes in the second hidden layer if applicable) – (# of output variables). The (# of nodes in the first hidden layer) is replaced by '?', to indicate that this number is determined by the column of the table you are in. The columns numbered 1 – 20, indicate the number of hidden nodes in the first hidden layer of the network.

.

Chapter 3

The FAST Method: Estimating Unconditional Demand Elasticities for Processed Foods in the Presence of Fixed Effects[48]

Abstract

This study estimated a set of unconditional own-price and expenditure elasticities across time for 49 processed food categories using scanner data and the FAST multistage demand system with fixed effects across time. Estimated own-price elasticities are generally much larger, in absolute terms, than previous estimates, while our expenditure elasticities are generally much lower. The use of disaggregated product groupings, scanner data, and the estimation of unconditional elasticities likely accounts for these differences. Half of the own-price elasticities are larger in absolute value than 1.0, and more than half of the expenditure elasticities are less than zero. Providing more disaggregate product level demand elasticities could aid in the economic analysis of issues relating to industry competitiveness or the impact of public policy.

[48] A version of this paper has been accepted for publication and will be forthcoming in the *Journal of Agricultural and Resource Economics*.

1. Introduction

Economic analyses of issues relating to firm or industry competitiveness and the impact of public policy upon the performance of the food system depend critically upon the existence of reliable and disaggregate elasticity of demand estimates. For example, recently developed methods to estimate welfare loss, based on a variety of oligopoly models, require product or market level demand elasticity estimates (Bhuyan and Lopez, 1998; Clarke and Davies, 1982; Gisser, 1986; Peterson and Connor, 1995; Willner, 1989). Furthermore, demand elasticities are crucial in defining relevant product markets and measuring market power in antitrust enforcement activities (Cotterill, 1994; Levy and Reitzes, 1992; Starek and Stockum, 1995). Disaggregated, product level demand elasticities allow for more meaningful benefit-cost analyses of proposed regulations for the food processing industries.[49] The increasing importance of food processing and marketing activities in the U.S. and global food systems requires that future analyses of domestic and international agricultural commodity policies focus on the demand for processed food products rather than raw agricultural commodities (Peterson, Hertel, and Stout, 1994). To facilitate these analyses analysts will need a set of unconditional and disaggregated price and expenditure elasticities for a large group of processed food products.

There have been several barriers to empirically estimating a set of demand elasticities for processed food products. The most obvious is the difficulty in obtaining price and quantity data for a disaggregate set of processed food products. To illustrate this problem, consider the study by Huang (1993), who estimated a complete demand system for 39 food categories and one non-

[49] For example, consider the case of analyzing a new HAACP regulation for a processed food product. The amount of the production cost increase that is passed on to consumers is determined by the relative magnitude of the supply and demand elasticities. The more elastic consumer demand is for any given product, any change in policy will then result in a smaller price change at the retail level, compared to a more inelastic demand response. So in the case of a production cost increasing policy or regulation, the loss in consumer surplus from a price increase will be lower when consumer demand is more elastic.

food category. He developed a time series of food quantity indices based on disappearance data and food price indices based on components of the consumer price index. Because the quantity indices are not direct estimates of actual purchases (or consumption) at the retail level, the correspondence between the price and quantity indices is not perfect because they are different data series. Furthermore, time-series data can create a problem if the number of time periods observed is not sufficient to estimate a large demand system. In this case, the analyst may be required to aggregate across food products.

One solution to the above data problems is the use of scanner data. This type of data provides an exact correspondence between price and quantity and provides detailed information on prices and quantities across both time and regions (or cross-sections), which can help solve the degrees of freedom problem. However, because scanner data are costly to obtain, a small, but growing number of demand analyses have been conducted using scanner data (Capps, 1989; Capps and Lambregts, 1991; Cotterill, 1994; Cotterill and Haller, 1994; Green and Park; 1998; Kinoshita *et al*, 2001.; Maynard, 2000; Nayga and Capps, 1994; Schmit *et al.*, 2000; Wessells and Wallström, 1999). These analyses have focused on small groups of processed food products, such as meat products, beverages, canned salmon and tuna, and dairy products. To date, scanner data has not been used to estimate a set of demand elasticities for a more encompassing group of processed food products.

The main objective of this article is to estimate a set of unconditional price and expenditure elasticities for 49 different processed food categories and one composite good. To alleviate the parametric burden of a large disaggregated demand system, Moschini's (2001) flexible and separable translog (FAST) multistage demand system is used to obtain unconditional own-price and expenditure demand elasticities. Due to the use of panel data in the model, the FAST model is extended to include fixed-effects across time.

2. Data

The data used in this study are from Information Resources, Incorporated (IRI) InfoScan® retail data.[50] The IRI data includes prices and total sales for 140 different processed food products for 42 US metropolitan areas from the first quarter of 1988 to the fourth quarter of 1992.[51] Total sales for each metropolitan area is the total amount sold each quarter by all supermarkets in the metropolitan area and the price is a weighted average price per unit of that particular product. Because it is not possible to estimate a demand system with 140 processed food products, these food products are aggregated into 49 processed food categories based on IRI category definitions. These product categories are shown in figure 1.

One limitation of the IRI data is that it does not include information on several key food categories, such as fresh meats and fresh fruits and vegetables. This is because supermarkets either did not give bar codes to these items or the codes assigned were not uniform during the sample time period. Thus, it is not possible for IRI to provide information on these food product categories. As such, they are included in an "all other goods" composite good in our model. Another limitation is that IRI collects data from only supermarkets, so food purchases from other retailers, such as convenience stores, are not included in the data.

The IRI price and total sales data are also supplemented with information on median household income and total number of households from the IRI InfoScan® market profiles for each metropolitan area. Given that the data on median household income are on an annual basis

[50] The data was made available to us by an arrangement with Professor Ron Cotterill at the Food Marketing Policy Center, University of Connecticut.

[51] Cotterill and Haller (1994) as well as Wessells and Wallström (1999) have used a subset of this data. The selected metropolitan areas where chosen based on completeness of data.

and the remaining data are on a quarterly basis, the level of median household income is allocated across the four quarters for each year to provide median household income on a quarterly basis. This allocation is accomplished through the use of quarterly data series on Disposable Personal Income and Personal Outlays from the Bureau of Economic Analysis (BEA) for the years 1988 to 1992 (US Department of Commerce, 2002). Using these series, annual savings rates are calculated that are used to adjust the annual data on median household income (by subtracting out household savings). The adjusted annual levels of household income are then allocated across quarters using the quarterly percentages calculated from the BEA's series on Disposable Personal Income.

The "all other goods" composite is a residual good that represents expenditure on all goods and services not included in the 49 processed food categories examined in the paper. Total sales for the "all other goods" composite good is calculated by subtracting household sales (total sales divided by number of households) for all 49 processed food categories from the estimated quarterly median household income. A price index for this residual good is computed using regional consumer price indices (US Department of Labor, 2002) for "All Urban Consumers." This consumer price index series is chosen because of the relatively large consumption share of the "all other goods" category. Regional data are available for New York, Philadelphia, Boston, Chicago, Cleveland, Detroit, St. Louis, Dallas/Ft. Worth, Houston, Miami/Ft. Lauderdale, Los Angeles, and San Francisco. For all other regions, the composite consumer price index for the Northeast, Midwest, South, and West were utilized.

Descriptive statistics for total sales and average prices for each of the 49 processed food categories are provided in table 1. To illustrate the variability in the panel data set across markets

Table 1: Sample Means and Standard Deviations

	Average Price ($/Unit)				**Total Sales ($)**			
	Pooled Mean	**Standard Deviation**	**Standard Deviation of Means (across)**		**Pooled Mean**	**Standard Deviation**	**Standard Deviation of Means (across)**	
Product Grouping			**Markets**	**Time**			**Markets**	**Time**
Coffee	0.8457	0.1616	0.1324	0.0882	11135549	11795926	11829689	1028573
Coffee Creamer and Flavorings	1.7328	0.1783	0.1522	0.0757	661324	547440	544408	72117
Tea	8.0076	2.1091	1.9446	0.6803	2081575	2407966	2390600	218878
Bottled Water	0.9983	0.0600	0.0069	0.0271	4636961	7127086	7152379	485462
Low-Calorie Soft Drinks	3.7292	0.2987	0.2440	0.0871	10376526	9485445	9484894	933990
Regular Soft Drinks	0.9006	0.1591	0.1428	0.0387	19604021	17015937	16975890	1797670
Refrigerated Juices	0.9964	0.0618	0.0030	0.0494	10423344	15491568	15566054	1153337
Shelf-Stable Juices	0.9991	0.0563	0.0008	0.485	14425856	16913652	16968070	1386250
Frozen Juices	0.9960	0.0627	0.0028	0.0475	6642821	5519396	5548842	334281
Refrig. Pickles and Relish	0.9988	0.1017	0.0020	0.0993	485566	570803	570532	45809
Shelf-Stable Pickles and Relish	0.9960	0.0607	0.0017	0.0547	3493288	3157395	3147582	381052
Pourable Salad Dressing	1.0000	0.0803	0.0002	0.0783	3294211	3084270	3035592	542243
Dry Dressings and Toppings	0.8665	0.2757	0.0475	0.2546	675399	543808	540443	61123
Mayonnaise	0.6630	0.0944	0.0388	0.0848	3282010	3063168	3070609	284257
Ketchup	0.7201	0.0737	0.0664	0.0237	1364697	1206257	1203572	121675
Sauces and Marinades	0.8529	0.1200	0.0238	0.1125	2513323	2158855	2087289	494755
Sour Cream	1.1519	0.1277	0.1196	0.0290	1501821	1690018	1694568	153101
Whole Milk	0.2992	0.0314	0.0223	0.0191	10141940	11263201	11250633	629306
Skim/Low-Fat Milk	0.2756	0.0361	0.0288	0.0191	16506631	14093536	13878606	2531526
Powdered/Condensed Milk	0.9948	0.0946	0.0046	0.0832	1240180	1363586	1304845	345549
Cocoa Mix and Flavored Milks	0.9957	0.0829	0.0041	0.0776	2008606	1938968	1674860	787423
Ice Cream/Yogurt	0.9869	0.0416	0.0102	0.0187	10708396	10864883	10778967	1573946
Cheese (non-shredded)	0.9992	0.0812	0.0009	0.0795	11382555	12407589	12362212	1372988
Shredded Cheese	3.5737	0.4102	0.2819	0.2885	2177247	1761291	1628914	579716

Table 1 cont’d.

	Average Price ($/Unit)				**Total Sales ($)**			
	Pooled Mean	**Standard Deviation**	**Standard Deviation of Means (across)**		**Pooled Mean**	**Standard Deviation**	**Standard Deviation of Means (across)**	
Product Grouping			**Markets**	**Time**			**Markets**	**Time**
Imitation Cheese	2.5971	0.5474	0.4062	0.2036	91898	119944	99784	26535
Cheese Spreads	0.9975	0.0504	0.0016	0.424	3418286	3530889	3489768	585262
Dried Fruits	0.9988	0.0543	0.0019	0.0464	1614831	1826564	1793823	289980
Shelf-Stable Fruits	0.9307	0.0585	0.0246	0.0485	4985008	4692063	4635957	720924
Baked Beans	0.5424	0.0757	0.0653	0.0321	1107529	828919	777465	230705
Shelf-Stable Vegetables	0.8783	0.2333	0.0386	0.2311	7811451	8082719	7782702	1774772
Frozen Vegetables	0.9992	0.0424	0.0005	0.0367	6119662	7375367	7379023	723566
Frozen Fries and Onion Rings	0.7000	0.0785	0.0420	0.0635	2283763	2007918	2003372	226308
Bread	1.0000	0.0639	0.0000	0.0560	15012560	15085067	15162453	1127889
Muffins and Rolls	0.9981	0.0680	0.0014	0.0577	5464413	6303311	6290252	637305
Rice	1.2599	0.2986	0.2949	0.0300	3430886	4097155	4102936	361768
Pasta	0.9321	0.1103	0.1031	0.0287	4088161	5047888	5072262	359868
Spaghetti Sauce	0.9314	0.0906	0.0843	0.0258	3513022	3761892	3748880	383305
Peanut Butter	1.8257	0.2054	0.1307	0.1541	2980863	2459919	2463054	241661
Jams and Jellies	0.9994	0.0651	0.0007	0.0590	2754568	2895557	2908443	232433
Mixes	0.9983	0.0436	0.0011	0.0329	3328600	2552430	2509950	407723
Seasonings and Preservatives	0.9983	0.0802	0.0043	0.0700	4558537	4685780	4626443	718888
Syrups	0.9991	0.0580	0.0008	0.0528	1746297	1613675	1598382	240637
Flour	0.9884	0.1734	0.0060	0.1639	1278964	1089218	1036151	1127889
Canned Soup	0.9127	0.1063	0.0598	0.0874	6301093	5808765	5444426	1611846
Dry Soup	0.9906	0.0587	0.0080	0.0200	2978058	3822765	3704676	685760
Gelatin/Pudding Mix	0.9978	0.0686	0.0011	0.0643	1823759	1621042	1617356	175556
Popcorn	0.9989	0.0407	0.0023	0.0178	1877550	1539600	1525218	227107
Snack Nuts	1.0402	0.2019	0.0623	0.1845	2199366	2462671	2424247	370042
Candy and Mints	1.0017	0.0466	0.0033	0.0414	8017229	7583452	7289926	1730964

and time, sample standard deviations are calculated across markets and time for each of the processed food categories.

2. 1 Nonparametric Tests of Utility Maximization

An underlying premise in the estimation of a demand system is that consumer behavior is consistent with the maintained hypothesis of utility maximization. Before estimating a demand system parametrically, it is important to determine if the data to be used is consistent with this hypothesis. Varian (1982) proposed a nonparametric procedure for evaluating whether a set of observed data is consistent with the utility maximization hypothesis by directly testing whether the data satisfy the Weak Axiom of Revealed Preference (WARP) and/or the General Axiom of Revealed Preference (GARP). Diaye, Gardes, and Starzec (2002) show that if WARP is satisfied then one can claim that there exists a utility function that rationalizes that data, but no other conclusions can be drawn about the nature of the utility function. If GARP is also satisfied, then there exists a nonsatiated utility function and a demand correspondence that rationalizes the data.

WARP and GARP are tested simultaneously for each metropolitan area using a variant of Warshall's algorithm (as presented by Varian (1982)) using the normalized prices (in order to lessen the effects of seasonality) and quantities (calculated by dividing total sales by price) from each of the 49 processed food categories. Strict adherence to the rules of nonparametric testing would lead one to reject the hypothesis that WARP or GARP are satisfied by the data if just one violation is found. Thus, Varian (1982) proposed that one should reject an axiom if the violation rate is greater than five percent, where the violation rate is defined as the number of violations of the revealed preference relation being examined divided by the total number of pairs belonging to the revealed preference relation (Diaye, Gardes, and Starzec, 2002). Testing for both WARP and GARP using this rule, the data for three of the original 42 metropolitan areas, Columbus (6.2

percent violation rate for both tests), Kansas City (8.6 percent violation rate for both tests), and Portland (8.6 percent violation rate for both tests), failed the nonparametric tests and were summarily excluded from the sample, leaving 780 observations in the panel data set.[52]

3. Model Specification

Following Moschini (2001), the empirical demand model utilized in this study is based on the notion of indirect separability. Preferences are indirectly weakly separable in the partition $\hat{I} = \left\{ I^1, ..., I^N \right\}$ if the indirect utility function $V\left(p/y\right)$ can be written as

$$V(p/y) = V^0\left[V^1(p^1/y), ..., V^N(p^N/y)\right] \tag{1}$$

where p^r is the vector of prices in the r^{th} group, $r = 1, ..., N$ and $V^r\left(p^r/y\right)$ are indices dependent only on p^r and total expenditure (*y*). It is assumed that $V^0(.)$ is continuous, nonincreasing, and quasiconvex, and $V^r(.)$ is continuous, nondecreasing, and quasiconcave, such that $V(p/y)$ retains the general properties of an indirect utility function.

The advantage of indirect separability, compared to direct separability, is that it allows a consistent specification of the unconditional demand functions and conditional demand functions of a weakly separable preference structure. Using Roy's identity, the unconditional (Marshallian) demand functions $q_i\left(p/y\right)$ and the conditional demand functions $c_i\left(p^r/y\right)$ are defined as:

[52] The metropolitan areas included in the sample are Albany, Atlanta, Baltimore/Washington, Birmingham, Boston, Buffalo/Rochester, Chicago, Cincinnati/Dayton, Cleveland, Dallas/Ft. Worth, Denver, Detroit, Grand Rapids, Hartford/Springfield, Houston, Indianapolis, Los Angeles, Louisville, Memphis, Miami/Ft. Lauderdale, Milwaukee, Minneapolis/St. Paul, Nashville, New York, Oklahoma City, Omaha, Philadelphia, Phoenix/Tucson, Pittsburgh, Raleigh/Greensboro, Sacramento, Salt Lake City, San Antonio, San Diego, San Francisco/Oakland, Seattle/Tacoma, St. Louis, Tampa/St. Pete, and Wichita.

$$q_i\left(p/y\right)=-\frac{\frac{\partial V^0}{\partial V^r}\frac{\partial V^r\left(p^r/y\right)}{\partial p_i}}{\sum_{s=1}^{N}\frac{\partial V^0}{\partial V^s}\frac{\partial V^s\left(p^s/y\right)}{\partial y}},\ i\in I^r \text{ and} \tag{2}$$

$$c_i\left(p^r/y\right)=-\frac{\frac{\partial V^r\left(p^r/y\right)}{\partial p_i}}{\frac{\partial V^r\left(p^r/y\right)}{\partial y}},\ i\in I^r. \tag{3}$$

Explicit forms for equations (2) and (3) can be obtained once functional forms are specified for $V^0(.)$ and $V^r(.)$. Moschini (2001) derives the first-stage group share equations and second-stage conditional share equations using equations (2) and (3) via the following relationship:

$$q_i(p/y)=\frac{y_r(p/y)}{y}c_i(p^r/y),\ i\in I^r$$

where $y_r(p/y)=\sum_{i\in I^r}p_iq_i$, the within-group expenditure allocation for partition I^r.

Moschini (2001) adopts the translog specification of Christensen, Jorgenson, and Lau for $V^0(.)$ and $V^r(.)$. Specifically:

$$V^0\left(\cdot\right)=-\left[g_0+\sum_{r=1}^{N}g_r\log V^r\left(\cdot\right)+\frac{1}{2}\sum_{r=1}^{N}\sum_{s=1}^{N}g_{rs}\log V^r\left(\cdot\right)\log V^s\left(\cdot\right)\right]\text{, and} \tag{4}$$

$$\log V^r(p^r/y)=b_0^r+\sum_{i\in I^r}b_i\log\left(p_i/y\right)+\frac{1}{2}\sum_{i\in I^r}\sum_{j\in I^r}b_{ij}\log\left(p_i/y\right)\log\left(p_j/y\right) \quad (5).$$

Homogeneity is satisfied by construction and symmetry is imposed by setting $b_{ij}=b_{ji}\ \forall i,j$ and $g_{rs}=g_{sr}\ \forall r,s$. To ensure that the indirect utility function based on equations (4) and (5) is a flexible functional form and satisfies the properties of indirect weak separability, Moschini shows that the following parametric restrictions are also applicable:

$$b_0^r = 0 \text{ for } r = 1,...,N,$$

$$g_0 = 0,$$

$$\sum_{i \in I^r} b_i = 1 \text{ for } r = 1,...,N,$$

$$\sum_{r=1}^{N} g_r = 1, \text{ and}$$

$$\sum_{i \in I^r} \sum_{j \in I^r} b_{ij} = 0 \text{ for } r = 1,...,N.$$

The last restriction allows for the case of asymmetric separability, where the r^{th} group has only one price.

For estimation purposes, Moschini (2001) suggests that it may be convenient to estimate the conditional share equations and the group share equations using a two-step process. First, the conditional share equations, expressed as:

$$w_i^r = \frac{b_i + \sum_{j \in I^r} b_{ij} \log\left(p_j / y\right)}{1 + \sum_{k \in I^r} \sum_{j \in I^r} b_{kj} \log\left(p_j / y\right)} \quad \forall\, i \in I^r, \tag{6}$$

where $w_i^r = (p_i q_i)/y_r$ and y_r is the within-group expenditure, are estimated. Then, the group share equations, expressed as:

$$w^r = \frac{B^r\left(p^r / y\right)\left(g_r + \sum_{s=1}^{N} g_{rs} \log V^s\left(p^s / y\right)\right)}{\sum_{g=1}^{N} B^g\left(p^g / y\right)\left(g_g + \sum_{s=1}^{N} g_{gs} \log V^s\left(p^s / y\right)\right)} \text{ for } r = 1,....,N, \tag{7}$$

where $w^r = y^r / y$ and $B^g\left(p^g / y\right) = 1 + \sum_{j \in I^g} \sum_{i \in I^g} b_{ij} \log\left(p_i / y\right)$ for $g = 1,...,N$, are estimated. The indices $\log V^r$ and B^g are computed using the estimated parameters of the conditional share equations in the first step.

3. 1 Incorporating Fixed Effects

Given the nature of panel data and the presence of heterogeneity in pooled models, one should consider the use of fixed or random effects in the model to account for any heterogeneity bias (Hsiao, 1986). In order to capture this heterogeneity, fixed effects across markets and time should be incorporated into the conditional and group share equations. Due to the size of the demand system estimated in this paper, only quarterly fixed effects are included in the empirical model, in order to leave enough degrees of freedom for estimation.

If prices and expenditure are both normalized by their respective means, then the sample mean of $\log\left(p_i/y\right)=\log\left(1/1\right)=0$. This implies that the intercept term in equation (6) is b_i. To incorporate fixed effects across time, redefine b_i in equation (6) to equal:

$$b_i = \sum_{s\in S} t_{is} D_s \text{ for } i \in I^r,\ r = 1,...,N, \tag{8}$$

where t_{is} is the time-specific fixed effect, D_s is a dummy variable equal to one if the observation being examined occurred in time interval s, and S is the set of time periods. The set *S* can represent individual time periods, quarters, years, etc. For this paper, *S* includes the four standard quarters of the calendar year, which makes D_s quarterly dummies. Substituting equation (8) into equation (6) gives the revised conditional share equations.

To take account of heterogeneity at the top level of the two-stage demand system, fixed effects can be incorporated into the group share equations as well. Again, the sample mean of $\log\left(p_i/y\right)=0$, implies that $B^g\left(p^g/y\right)=1$ for $g=1,...,N$ and $\log V^r\left(p^r/y\right)=0$ for $r=1,...,N$. Thus, at the sample means, g_r is the intercept term in equation (7). To incorporate fixed effects redefine g_r as:

$$g_r = \sum_{s \in S} r_{rs} D_s \text{ for } r = 1,\ldots,N \,, \tag{9}$$

where r_{rs} is analogous to t_{is} in equation (8). Substituting equation (9) into equation (7) yields the revised group share equations.

3.2 Indirectly Weakly Separable Structure

The 49 processed food categories and one composite good are partitioned into six weakly separable partitions as shown in figure 1. Thus, the underlying indirect utility function can be expressed as:

$$V^0 = \left[V^1\left(p^1 / y\right),\ldots,V^5\left(p^5 / y\right), p^6 / y\right]$$

where the indices 1 through 5 refer to the product groupings: (1) beverages, (2) dairy products, (3) milled grain and pasta, (4) fruits and vegetables, and (5) baking, condiments and deserts. Group 6 is an asymmetric group with only one good, the "all other goods" composite good.[53] A few goods do not lend themselves easily to classification according to our system. These products have been placed in the group that appears to be the most reasonable from the point of view of the consumer who is constrained to allocate her expenditure budget amongst these particular product groups. For example, consider the milled grain and pasta product group. Given their hypothesized complementary relationship, pasta and spaghetti sauce are placed in the same group (partition). Similarly, peanut butter and jellies and jams are placed in the milled grain product group because of their hypothesized complementary relationships with bread and muffins and rolls.[54] The fifth product group, baking, condiments and deserts comprises the largest number of goods, sixteen, due to the fact that many of these goods are compliments and substitutes for each other and/or where not easily placed into another group.

[53] The six partitions used in this paper are assumed to be weakly separable.
[54] In results not reported, these goods were found to be complements.

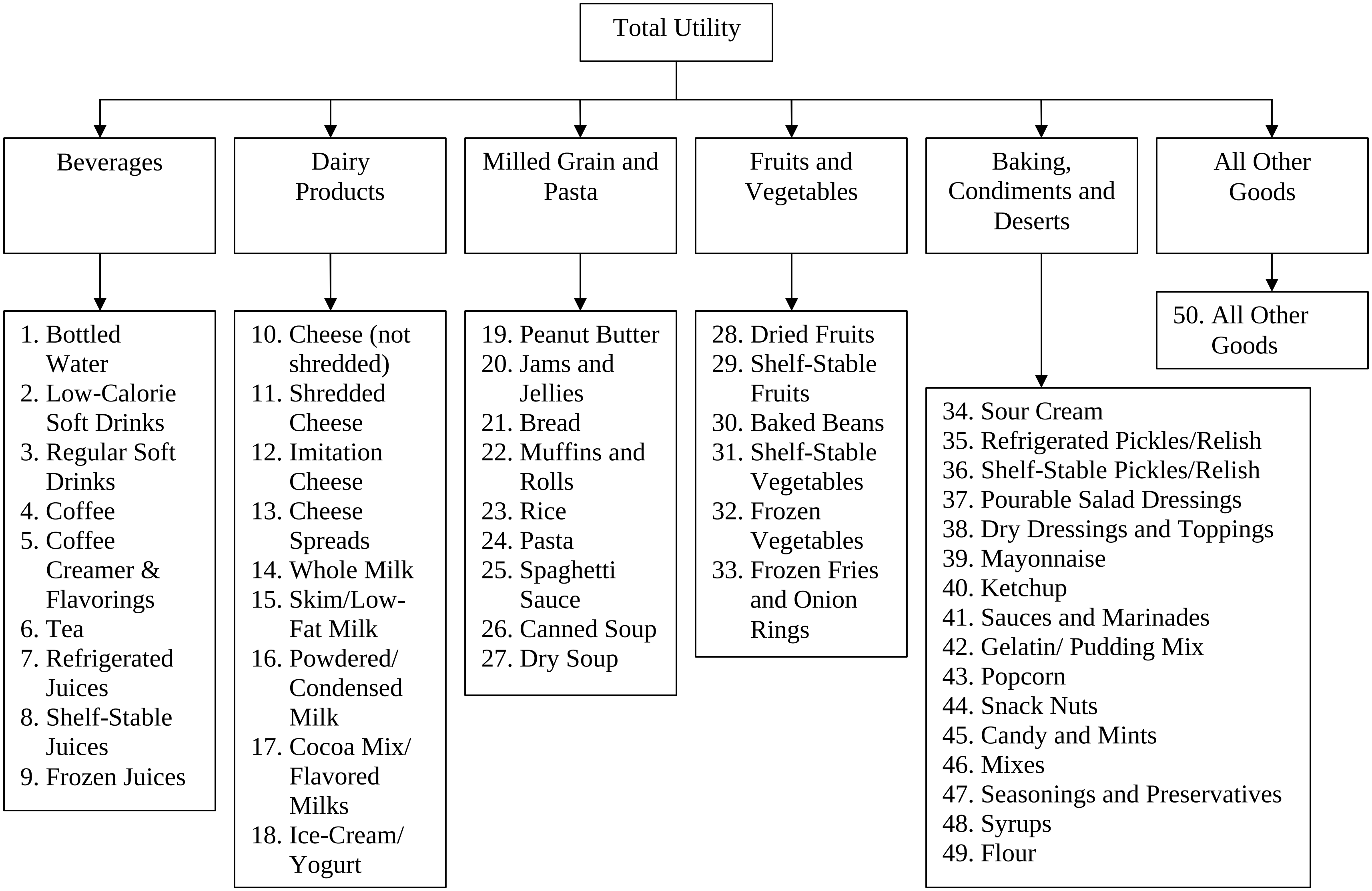


Figure 1: Separable Preference Structure of Representative Consumer

4. Results

Equations (6) and (7), modified by equations (8) and (9) to allow for fixed effects across quarters, are estimated to obtain unconditional price and expenditure elasticities across quarters using the two-step process as presented by Moschini (2001). In the first step, five systems of conditional within-group share equations are estimated. The "all other goods" group is a trivial estimation because it only contains one composite good. In the second step, a system of five group share equations is estimated. Due to the adding-up conditions, one share equation is dropped in each system during estimation to avoid singularity of the variance/covariance matrix of the residuals.[55] In total, with the fixed effects across time, there are 488 parameters to estimate for all the systems of equations at both stages.[56] With 780 observations, this leaves a 1.6:1 ratio between the number of parameters and the total number of degrees of freedom.

Due to the highly nonlinear nature of the share equations, the Full Information Maximum Likelihood (FIML) procedure in SAS is utilized to estimate each system of equations at both stages of the estimation process. This iterative procedure was found to be superior to the iterative seemingly unrelated regression (ITSUR) estimation procedure in SAS in terms of the convergence properties of the algorithm. The specific estimation results for each system are not presented in detail, but are available from the authors upon request. The majority of the fixed effects across time where found to be statistically significant, indicating that temporal heterogeneity is present in the data.

[55] In the system of group share equations, the "all other goods" group share equation is dropped. In addition, it should be noted that the use of maximum likelihood estimation is invariant to the share equation being dropped (Moschini, 2001).

[56] The "beverages" group, "dairy products" group and "milled grain and pasta" group each had 76 parameters to estimate. The "fruits and vegetables" group had 40 parameters to estimate. The "baking, condiments and deserts" group had 179 parameters to estimate. The second stage system of group share equations had 41 parameters to estimate.

Misspecification tests were conducted for each individual share equation and system of share equations at each stage of estimation of the model. The misspecification tests conducted and results are presented in Appendix C. The misspecification tests indicate that at a system level, the model is misspecified, which is not surprising given the highly nonlinear nature of the demand system. The misspecification tests do indicate that the quarterly dummies were adequate for capturing the temporal heterogeneity in the data, but market dummies should have also been included. Given the large number of markets and limited degrees of freedom available, the inclusion of market dummies was not a feasible option for capturing all the market heterogeneity present in the panel data set. Furthermore, the estimation of a random effects model for such a highly nonlinear system of equations is beyond the scope of this paper. The misspecification tests indicate the presence of temporal and spatial dependence as well. The inclusion of lagged variables and spatial variables needs to be considered, but the incorporation of such effects into the model is not clear. One possibility may be the inclusion of habit formation in the model (see Deaton and Muellbauer, 1999, p. 373-7), but again the limited number of degrees of freedom makes this problematic.

The ultimate goal of the estimation of the demand system was to obtain unconditional expenditure and price elasticities for all of the product categories examined in the empirical model. Thus, the majority of the remaining discussion here pertains to this goal.

4. 1 Derivation of Unconditional Demand Elasticities

Moschini (2001) points out that the main payoff to using the FAST multistage demand model is the derivation of a complete matrix of unconditional Marshallian expenditure and price elasticities. If the data are normalized so that $p_i = y = 1 \ (\forall \ i)$, the unconditional expenditure (h) and price (e) elasticities for good i are:

$$e_{ijs} = \frac{b_{ij}}{b_i(s)} + \frac{g_{rr} b_j(s)}{g_r(s)} - g_r(s)\left(\sum_{q\in I^r} b_{iq}\right) - b_j\left(\sum_{s=1}^{N} g_{rs}\right) - d_{ij} \text{ for } (i, j) \in I^r, \qquad (10)$$

$$e_{ijs} = \frac{g_{rs} b_j(s)}{g_r(s)} - g_s(s)\left(\sum_{q\in I^s} b_{qj}\right) - b_j(s)\left(\sum_{p=1}^{N} g_{rp}\right) \text{ for } i \in I^r \text{ and } j \in I^s\text{, and} \qquad (11)$$

$$h_{is} = 1 - \frac{\sum_{j\in I^r} b_{ij}}{b_i(s)} - \frac{\sum_{s=1}^{N} g_{rs}}{g_r(s)} + \sum_{r=1}^{N}\sum_{s=1}^{N} g_{rs} \text{ for } i \in I^r, \qquad (12)$$

where d_{ij} is the Kronecker delta ($d_{ij} = 1$ if $i = j$ and 0 otherwise), $s \in S$, $b_i(s)$ is given by equation (8), and $g_r(s)$ is given by equation (9).[57] The elasticities given by equations (10) through (12) not only vary over goods, but vary over time as well. This last result is due to the fact that the elasticities are functions of the fixed effects across time. Given the nonlinear nature of the model, no estimator of the elasticity can be derived that is not a function of the fixed effects included in the model (Heckman and MaCurdy, 1980).

To decrease the dimensionality of each of these elasticity estimates, one could take pooled means over time, but such elasticity estimates would fail to take account of the heterogeneity present in the panel data used for estimation. Thus, elasticity estimates are reported across quarters (or temporally), to draw attention to the variation in the demand elasticities across time. Due to the fact that the scanner data used in this paper is aggregated over households, the elasticity estimates derived from the FAST multi-stage demand model should be interpreted in aggregate terms (i.e. at a macro level) (Edgerton, 1997). This interpretation is more desirable for policymakers, given policy-oriented studies tend to be focused at the aggregate (i.e. market), not household level.

[57] The formulas for the elasticities are different than those derived by Moschini (2001), because Moschini does not completely take account of the different restrictions imposed when using asymmetric groups.

4. 2 Unconditional Price and Expenditure Elasticity Estimates

The own-price and expenditure elasticity estimates for the product groupings examined are presented in table 2.[58] All elasticity estimates are reported temporally (i.e. across quarters) with their respective standard errors given in parentheses. Standard errors were calculated using a Monte Carlo method, assuming the parameters were asymptotically distributed multivariate normal for each system of equations estimated. Standard errors were obtained by randomly generating 5,000 sets of parameters and calculating standard errors of the elasticity values obtained using the generated parameters for each product category for all four quarters.

Overall, the pooled mean own-price elasticity estimates tend to be fairly large. Out of the fifty estimated own-price elasticities, twenty-three are less than or equal to -1.00 and thirty are less than or equal to -0.85 across all four quarters. Only six of the fifty estimated own-price elasticities have a value greater than -0.6 across all four quarters. As will be discussed later, the price elasticity estimates tend to be much larger in absolute terms than those reported in previous studies.

The processed food goods that have the most inelastic own-price elasticities tend to be products that are likely used as condiments or ingredients in a prepared meal, such as sour cream, syrup and ketchup, or meet some perceived need, such as coffee (and coffee creamer and flavorings) for the caffeine or bottled water for consumers who may feel it is superior to tap water. The processed food products that are more price elastic tend to have good substitutes available. For example, examining cross-price elasticities within product groups indicates that

[58] Due to space limitations, it is not possible to list the complete 50x50 matrices of unconditional price elasticities for all quarters in this article. A complete set of price and expenditure elasticities is available from the authors upon request.

Table 2: Unconditional Own-Price and Expenditure Elasticity Estimates

	Uncompensated Price Elasticities				**Uncompensated Expenditure Elasticities**			
	Quarter							
Product Groupings	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
Bottled Water	-0.28	-0.34	-0.39	-0.29	-0.21	-0.20	-0.19	-0.21
	(0.30)[1]	(0.27)	(0.25)	(0.29)	(0.16)	(0.15)	(0.14)	(0.16)
Low-Calorie Soft Drinks	-1.26	-1.24	-1.25	-1.28	-0.09	-0.09	-0.09	-0.09
	(0.20)	(0.19)	(0.19)	(0.21)	(0.10)	(0.09)	(0.10)	(0.10)
Regular Soft Drinks	-1.06	-1.04	-1.03	-1.05	0.03	0.02	0.02	0.03
	(0.16)	(0.15)	(0.15)	(0.16)	(0.09)	(0.09)	(0.09)	(0.09)
Coffee	-0.05	0.07	0.09	-0.07	-0.37*	-0.41*	-0.41*	-0.36*
	(0.12)	(0.15)	(0.15)	(0.l3)	(0.10)	(0.10)	(0.10)	(0.10)
Coffee Creamer and Flavorings	-0.34	-0.24	-0.22	-0.34	-0.24*	-0.27*	-0.27*	-0.24*
	(0.15)	(0.18)	(0.19)	(0.16)	(0.12)	(0.13)	(0.13)	(0.12)
Tea	-1.07	-1.08	-1.08	-1.07	-0.14	-0.15	-0.15	-0.15
	(0.09)	(0.10)	(0.09)	(0.09)	(0.12)	(0.12)	(0.13)	(0.12)
Refrigerated Juices	-0.53	-0.52	-0.53	-0.53	-0.09	-0.10	-0.09	-0.09
	(0.22)	(0.23)	(0.22)	(0.22)	(0.11)	(0.11)	(0.11)	(0.11)
Shelf-Stable Juices	-0.78	-0.78	-0.78	-0.78	-0.06	-0.06	-0.06	-0.06
	(0.12)	(0.12)	(0.12)	(0.12)	(0.09)	(0.09)	(0.09)	(0.09)
Frozen Juices	-0.70	-0.71	-0.70	-0.70	-0.14	-0.14	-0.14	-0.14
	(0.07)	(0.07)	(0.07)	(0.07)	(0.08)	(0.08)	(0.08)	(0.08)
Cheese (not-shredded)	-0.70	-0.70	-0.70	-0.70	-0.17*	-0.17	-0.17	-0.18*
	(0.13)	(0.13)	(0.14)	(0.13)	(0.09)	(0.09)	(0.09)	(0.08)
Shredded Cheese	-0.95	-0.95	-0.95	-0.95	0.47*	0.53*	0.53*	0.48*
	(0.40)	(0.44)	(0.44)	(0.41)	(0.13)	(0.14)	(0.14)	(0.13)
Imitation Cheese	-1.84	-1.89	-1.96	-1.90	-0.39*	-0.41*	-0.44*	-0.42*
	(0.16)	(0.17)	(0.18)	(0.17)	(0.18)	(0.19)	(0.20)	(0.19)
Sour Cream	-0.58	-0.59	-0.60	-0.62	0.02	0.02	0.04	-0.01
	(0.32)	(0.31)	(0.31)	(0.29)	(0.14)	(0.14)	(0.14)	(0.13)
Whole Milk	-0.91	-0.91	-0.91	-0.92	-0.28*	-0.28*	-0.28*	-0.30*
	(0.47)	(0.48)	(0.48)	(0.50)	(0.12)	(0.12)	(0.12)	(0.11)
Skim/Low-Fat Milk	-0.69	-0.69	-0.69	-0.71	0.01	0.01	0.01	0.01
	(0.28)	(0.29)	(0.29)	(0.29)	(0.10)	(0.10)	(0.10)	(0.09)
Powdered/ Condensed Milk	-0.80	-0.79	-0.79	-0.86	-0.16	-0.16	-0.16	-0.13
	(0.17)	(0.18)	(0.18)	(0.11)	().11)	(0.11)	(0.11)	(0.09)
Cocoa Mix and Flavored Milks	-0.86	-0.81	-0.82	-0.88	-0.11	-0.14	-0.14	-0.10
	(0.19)	(0.29)	(0.27)	(0.13)	(0.10)	(0.12)	(0.11)	(0.08)
Ice Cream/ Yogurt	-0.89	-0.85	-0.85	-0.91	0.04	0.03	0.03	0.05
	(0.09)	(0.08)	(0.07)	(0.10)	(0.09)	(0.09)	(0.09)	(0.08)
Cheese Spreads	-1.88	-1.87	-1.87	-1.90	1.40*	1.41*	1.41*	1.40*
	(0.08)	(0.08)	(0.08)	(0.08)	(0.08)	(0.08)	(0.08)	(0.08)
Peanut Butter	-0.61	-0.63	-0.62	-0.61	-0.52*	-0.50*	-0.50*	-0.52*
	(0.37)	(0.34)	(0.35)	(0.37)	(0.13)	(0.14)	(0.14)	(0.13)
Jams and Jellies	-0.96	-0.95	-0.95	-0.96	0.18	0.18	0.19	0.20
	(0.30)	(0.29)	(0.30)	(0.31)	(0.11)	(0.12)	(0.13)	(0.11)
Refrigerated Pickles and Relish	-0.82	-0.84	-0.84	-0.79	-0.12	-0.11	-0.10	-0.18
	(0.40)	(0.36)	(0.37)	(0.47)	(0.20)	(0.19)	(0.19)	(0.23)

Table 2 cont’d.

	Uncompensated Price Elasticities				Uncompensated Expenditure Elasticities			
	Quarter							
Product Groupings	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
Shelf-Stable Pickles and Relish	-1.02	-1.02	-1.02	-1.02	0.02	0.02	0.04	-0.01
	(0.15)	(0.13)	(0.14)	(0.14)	(0.09)	(0.09)	(0.10)	(0.07)
Pourable Salad Dressing	-0.59	-0.66	-0.63	-0.44	0.06	0.06	0.07	0.05
	(0.15)	(0.12)	(0.13)	(0.20)	(0.10)	(0.09)	(0.10)	(0.10)
Dry Dressings and Toppings	-1.01	-1.00	-1.01	-1.01	0.08	0.07	0.09	0.05
	(0.02)	(0.02)	(0.02)	(0.02)	(0.10)	(0.09)	(0.10)	(0.08)
Mayonnaise	-1.14	-1.12	-1.12	-1.15	0.05	0.05	0.06	0.03
	(0.09)	(0.08)	(0.07)	(0.10)	(0.10)	(0.09)	(0.10)	(0.09)
Ketchup	-0.67	-0.67	-0.64	-0.57	-0.01	-0.01	0.00	-0.04
	(0.13)	(0.13)	(0.14)	(0.17)	(0.10)	(0.10)	(0.11)	(0.10)
Sauces and Marinades	-1.94	-1.75	-1.86	-2.15	0.05	0.04	0.06	0.03
	(0.31)	(0.26)	(0.30)	(0.38)	(0.12)	(0.11)	(0.12)	(0.13)
Bread	-0.82	-0.78	-0.77	-0.82	-0.21*	-0.21*	-0.21*	-0.21*
	(0.05)	(0.06)	(0.06)	(0.05)	(0.08)	(0.10)	(0.10)	(0.08)
Muffins and Rolls	-1.05	-1.00	-1.01	-1.04	-0.60*	-0.50*	-0.52*	-0.58*
	(0.83)	(0.62)	(0.67)	(0.78)	(0.17)	(0.15)	(0.15)	(0.16)
Rice	-0.83	-0.83	-0.83	-0.83	0.10	0.11	0.11	0.12
	(1.23)	(1.25)	(1.23)	(1.30)	(0.25)	(0.26)	(0.26)	(0.27)
Pasta	-0.91	-0.90	-0.90	-0.91	-0.23*	-0.23*	-0.23*	-0.23*
	(0.04)	(0.04)	(0.04)	(0.04)	(0.08)	(0.10)	(0.10)	(0.08)
Spaghetti Sauce	-0.90	-0.89	-0.90	-0.90	-0.24*	-0.24*	-0.23*	-0.24*
	(0.10)	(0.10)	(0.10)	(0.11)	(0.08)	(0.10)	(0.10)	(0.08)
Gelatin/Pudding Mix	-1.15	-1.16	-1.18	-1.17	0.01	0.01	0.02	-0.02
	(0.14)	(0.15)	(0.17)	(0.17)	(0.10)	(0.10)	(0.12)	(0.10)
Popcorn	-1.13	-1.15	-1.15	-1.16	0.02	0.02	0.03	-0.01
	(0.11)	(0.14)	(0.14)	(0.14)	(0.10)	(0.11)	(0.11)	(0.10)
Snack Nuts	-1.21	-1.20	-1.19	-1.16	-0.23	-0.22	-0.19	-0.20*
	(0.26)	(0.26)	(0.25)	(0.21)	(0.13)	(0.12)	(0.13)	(0.10)
Candy and Mints	-0.76	-0.73	-0.74	-0.79	0.05	0.06	0.07	0.02
	(0.18)	(0.20)	(0.19)	(0.16)	(0.10)	(0.11)	(0.11)	(0.08)
Dried Fruits	-1.03	-1.03	-1.03	-1.03	-0.30*	-0.30*	-0.31*	-0.29*
	(0.20)	(0.21)	(0.22)	(0.18)	(0.09)	(0.11)	(0.13)	(0.09)
Shelf-Stable Fruits	-1.04	-1.04	-1.04	-1.04	-0.30*	-0.31*	-0.31*	-0.30*
	(0.15)	(0.15)	(0.16)	(0.14)	(0.08)	(0.10)	(0.13)	(0.08)
Baked Beans	-1.16	-1.10	-1.11	-1.17	-0.39*	-0.33*	-0.34*	-0.39*
	(0.29)	(0.17)	(0.18)	(0.31)	(0.13)	(0.11)	(0.14)	(0.13)
Shelf-Stable Vegetables	-1.78	-1.85	-1.80	-1.79	-0.12	-0.11	-0.12	-0.12
	(0.16)	(0.17)	(0.17)	(0.16)	(0.08)	(0.10)	(0.12)	(0.08)
Frozen Vegetables	-1.00	-1.00	-1.00	-1.00	-0.36*	-0.36*	-0.37*	-0.36*
	(0.18)	(0.19)	(0.19)	(0.19)	(0.09)	(0.11)	(0.13)	(0.09)
Frozen Fries and Onion Rings	-1.07	-1.09	-1.10	-1.07	-0.27*	-0.27*	-0.27*	-0.27*
	(0.14)	(0.16)	(0.18)	(0.14)	(0.07)	(0.10)	(0.12)	(0.08)
Mixes	-0.72	-0.69	-0.69	-0.71	0.12	0.14	0.15	0.10
	(0.13)	(0.15)	(0.15)	(0.14)	(0.20)	(0.22)	(0.23)	(0.20)

Table 2 cont'd.

Product Groupings	**Uncompensated Price Elasticities**				**Uncompensated Expenditure Elasticities**			
	Quarter							
	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**
Seasonings/Preservatives	-1.01	-1.01	-1.01	-1.01	0.09	0.09	0.10	0.05
	(0.14)	(0.15)	(0.13)	(0.12)	(0.18)	(0.19)	(0.18)	(0.15)
Syrups	-0.31	-0.22	-0.26	-0.26	-0.67*	-0.76*	-0.71*	-0.75*
	(0.09)	(0.10)	(0.10)	(0.10)	(0.15)	(0.17)	(0.17)	(0.16)
Flour	-1.02	-1.02	-1.02	-1.01	-0.01	-0.01	0.00	-0.04
	(0.06)	(0.06)	(0.06)	(0.06)	(0.08)	(0.08)	(0.09)	(0.07)
Canned Soup	-0.94	-0.97	-0.97	-0.93	-0.11	-0.07	-0.07	-0.11
	(0.09)	(0.12)	(0.12)	(0.09)	(0.09)	(0.11)	(0.11)	(0.09)
Dry Soup	-2.08	-2.04	-2.04	-2.08	1.26*	1.26*	1.26*	1.26*
	(0.11)	(0.12)	(0.12)	(0.11)	(0.08)	(0.10)	(0.10)	(0.08)
All Other Goods	-1.00	-1.00	-1.00	-1.00	0.99*	0.99*	0.99*	0.99*
	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)

* Indicates that the specified expenditure elasticity estimate is statistically different from zero at a 0.05 level of significance.

[1] Standard Errors are reported in parentheses.

consumers who could easily substitute regular cheese for shredded cheese, shredded cheese for imitation cheese, refrigerated juice for shelf-stable or frozen juices, and canned soup for dry soup. The imitation cheese, cheese spreads, sauces and marinades, and dry soup food categories have the largest own-price elasticities in absolute value (less than -1.78) across all four quarters. These high elasticity estimates could be reflective of the nature of these products, i.e. that consumers primarily purchase these products in bulk when prices are significantly reduced due to price discounts.

Many of the own-price elasticities in table 2 exhibit temporal fluctuations across quarters. For the majority of the processed food categories, the variation in the own-price elasticity estimates across time is relatively small, changing by an amount less than or equal to 0.1. There is substantial variation (0.12 to 0.40) in the estimated own-price elasticities across time for the coffee creamer and flavorings, pourable salad dressings, and sauces and marinades processed food categories. These differences likely reflect temporal consumption behavior of consumers.

For example, sales and the own-price elasticities for coffee creamer and flavorings are much larger during the colder months (first and fourth quarters). In contrast, consumption of ice cream and yogurt increases during the warmer months (second and third quarters), resulting in lower own-price elasticities. These different temporal patterns likely reflect the different characteristics of the product categories being examined. In addition, the temporal variations in the own-price elasticities could be attributed to the availability of compliments and substitutes. For example, the lower own-price elasticities in the first and fourth quarters for shelf-stable vegetables could be due in part to the decreased availability of substitutes, such as fresh vegetables and fruits during the winter season (Feng and Chern, 2000).

The expenditure elasticity estimates vary from -0.75 to 1.41 across products and quarters. Twenty-one of the processed food categories have statistically significant expenditure elasticities, with 17 of them with estimates less than or equal to 0. The remaining expenditure elasticities are not significantly different from 0. The relatively large number of non-significant expenditure elasticities may be due to consumers not changing their consumption of these processed food products as income increases and/or the introduction of measurement error in the allocation of median household income across quarters. Of the product categories with expenditure elasticities less than or equal to zero, certain product categories, such as imitation cheese, powdered/condensed milk, whole milk, peanut butter, bread, muffins and rolls, pasta, spaghetti sauce, shelf-stable fruit, frozen and shelf-stable vegetables, baked beans and syrups could be considered basic food categories. For example, as income increases, households substitute fresh vegetables for frozen or shelf-stable vegetables. For product categories such as coffee and coffee creamer and flavorings, the negative expenditure elasticities likely arise due to the perceived need for caffeine on a daily basis.

The variability of the estimated expenditure elasticities is much less than for the estimated own-price elasticities. Variation across quarters by more than 0.06 occurs in only five of the fifty product categories in the study. These categories are: shredded cheese, refrigerated pickles and relish, muffins and rolls, baked beans and syrups. Again, these variations are likely due to changes in temporal consumption, but could have arisen due to the potential introduction of measurement error.

4.3 Comparison with Elasticity Estimates in Literature

As mentioned earlier, on average the unconditional own-price elasticity estimates obtained in this study are higher than elasticity estimates reported in a number of studies found in the literature, whereas the expenditure elasticities tend to be lower. This result may reflect the three factors that differentiate this study from previous studies (e.g. Feng and Chern, 2000; Huang, 1993; Huan and Lin, 2000; Lamm, 1982). First, the product groupings examined in this study are much more disaggregated than groupings utilized in similar studies. Thus, we expect that the own-price elasticity estimates should be of larger magnitude then those obtained using more aggregated groupings. In addition, Maynard (2000) mentions that temporal disaggregation can also lead to higher estimates. The second difference is the use of different data sets, i.e. scanner data was used in this study compared to disappearance data or household data from the Nationwide Food Consumption Survey (NCFS). Scanner data provides a more accurate picture of consumer purchases because it measures actual quantity purchased and provides an exact correspondence between quantity and price levels. This higher level of accuracy will result in elasticity estimates being more reflective of actual purchasing behavior providing a potential explanation of the differing estimates obtained here (Maynard, 2000). The third difference has to do with the use of unconditional rather than conditional demand functions as the basis for deriving the own-price and expenditure elasticities. The unconditional elasticity estimates are based on total rather than on group expenditure. Thus, they allow for more interaction between separable groups as the consumer reallocates consumption in response to price and expenditure changes. Not surprisingly, the unconditional own-price elasticity estimates will tend to be higher, by taking account of the effect of price changes on the allocation of expenditures between groups. In contrast, because expenditure elasticities have to satisfy Engel aggregation, conditional (or within-group) expenditure elasticities will be larger than unconditional expenditure elasticities because they apply to a smaller set of products (i.e. on group expenditure).

The study by Huang (1993) is probably the most comprehensive disaggregate estimated demand system, in terms of number of goods, available in the literature. But the majority of goods included in the model were meats, fresh fruits and vegetables (20 of the 39 food product categories). The first two columns of table 3 provide a summary of Huang's elasticity estimates for a select set of comparable product groups. Our elasticity estimates for these same product groups are presented in the last two columns of table 3. Except for juices and coffee, our estimated own-price elasticities are greater in absolute value than those reported by Huang. For a number of the product categories these differences may reflect the use of more aggregate product categories by Huang. Our own-price elasticity estimates for frozen juices are fairly close to those reported in Huang's aggregate juice category. However, for the remaining product categories, the differences are substantial. For example, Huang's estimate of the own-price elasticity for fluid milk is -0.04, which is more than an order of magnitude smaller than our estimate. The same is true for the flour and rice categories. Furthermore, except for fruits and vegetables, our estimated expenditure elasticities are lower than those reported by Huang.

Table 3: Elasticity Estimate Comparisons

Product Categories	Huang (1993)[a]		Feng and Chern (2000)		Huang and Lin (2000)[b]		Lamm (1982)[c]		Park et al.(1996)[d]		Our Estimates[a]	
	e_{ii} [e]	h_i	e_{ii}	h_i	e_{ii}	h_i	e_{ii}	h_i	e_{ii}	h_i	e_{ii}	h_i
Juices	-0.56	0.37			-1.01	1.04	-0.82	2.37			-0.52 to -0.78	-0.06 to -0.14
Milk/Dairy	-0.04 to -0.28	0.12 to 0.51			-0.79	0.67	-0.17 to 0.08	-0.19 to 0.47	-0.47 to -0.53	0.60	-0.69 to -0.91	-0.28 to 0.01
Cheese	-0.25	0.42					-0.21	0.57	-0.01 to -0.24	0.50	-0.70 to -1.96	-0.44 to 1.41
Bread					-0.35	0.58			-0.17 to -0.21	0.38 to 0.52	-0.77 to -0.82	-0.21
Flour	-0.08	0.13					-0.06	0.15			-1.01 to -1.02	-0.04 to 0.0
Rice	0.07	0.15					0.00	0.00			-0.83	0.10 to 0.12
Processed Fruit			-0.27	0.83	-0.72	1.16	-0.93	2.68			-1.03 to -1.04	-0.29 to -0.31
Processed Vegetables	-0.17 to -0.53	0.68 to 0.87	-0.56	0.62	-0.72	0.98	-0.12 to -0.09	0.27 to 0.33			-1.07 to -1.85	-0.11 to -0.39
Fruits and Vegetables	-0.09 to -1.18	-0.49 to 1.29							-0.32 to -0.52	0.56 to 0.69		

Table 3 cont'd.

Product Categories	Huang (1993)[a]		Feng and Chern (2000)		Huang and Lin (2000)[b]		Lamm (1982)[c]		Park et al. (1996)[d]		Our Estimates[a]	
	e_{ii}[e]	h_i	e_{ii}	h_i	e_{ii}	h_i	e_{ii}	h_i	h_i	e_{ii}	h_i	e_{ii}
Baking Goods			-0.48	0.64	-0.40	0.82					-0.22 to -1.01	-0.76 to 0.10
Coffee	-0.18	0.82									-0.05 to 0.09	-0.36 to -0.41

[a] The elasticity ranges are across products within the product group.
[b] The expenditure elasticity estimates are those that have been adjusted for quality.
[c] All elasticities are from a dynamic linear expenditure system.
[d] The elasticity ranges are across income groups.
[e] e_{ii} is the own-price elasticity estimate and h_i is the expenditure elasticity estimate.

Lamm (1982) estimated a dynamic demand model consisting of thirty-one disaggregated groups, in some cases more disaggregated then Huang (1993). As a result, Lamm's study tends to not fully characterize some food groups. For example, Lamm only examines fluid whole milk, but not fluid low-fat or skim milk. His estimated own-price and expenditure elasticities for the relevant processed food categories are provided in the seventh and eighth columns of table 3. Comparing these elasticities to our own, the difference in estimates is substantial for all the products compared in table 3, except rice. Comparing his estimates to other static studies, Lamm claims that his estimates are more price inelastic due to the inclusion of habit formation in his empirical model, resulting in a negative specification bias if lagged consumption is omitted. In view of the fact that Lamm examines a dataset that spans three decades, one could expect that habit formation would be a significant phenomenon, but might not be as significant for shorter time periods, as in this study. However, this phenomenon could partially explain the lower expenditure elasticities obtained here. In contrast to our results, Lamm's expenditure elasticity estimates for juices and processed fruits are significantly greater than ours. In both of these categories, only one specific commodity was examined: frozen orange juice concentrate and canned fruit cocktail respectively. Of further interest is the fact that Lamm obtained a negative expenditure elasticity for fluid whole milk (-0.19) close to ours (-0.28).

The other three sets of elasticity estimates given in table 3, also differ from the estimates obtained in this study. Again, our own-price elasticities tend to be substantially higher than those obtained by Feng and Chern (2000), Huang and Lin (2000), and Park et al. (1996), while our expenditure elasticities tend to be substantially lower.

Maynard (2000) estimates a double log model of seven demand equations for chunk, sliced, grated, shredded, snack food, cubed, and other cheese products using weekly scanner data. The own-price and expenditure demand elasticities estimated are equal to or greater than the range of estimates found in this study. These higher estimates provide evidence that disaggregated scanner data, both temporally and by product, gives rise to elasticity estimates greater in absolute value when compared to elasticity estimates in studies using more aggregated groupings. Maynard's own-price and expenditure elasticities for cheese products ranged from -0.154 to -3.965 and from -0.747 to -0.782, respectively. Comparing the ranges of these estimates to those found in table 2, provides some evidence of the above discussion.

Other studies have found negative expenditure elasticities for similar product categories examined in this study, providing support for the estimates obtained in this paper. Edgerton (1997) obtained a negative expenditure elasticity for potatoes equal to -0.05, providing evidence for why the expenditure elasticity for the frozen potatoes and onions product group is negative. You, Epperson and Huang (1996) found negative expenditure elasticities for a number of fresh fruits, suggesting that the negative expenditure elasticities obtained for dry and shelf-stable fruits is plausible. Brown, Lee and Seale, Jr. (1994) obtained similar expenditure elasticity estimates (between -0.10 and 0.10) using the CBS model developed by the Netherlands Central Bureau of Statistics for the juice categories.[59]

5. Summary and Conclusion

This study estimated a set of unconditional own-price and expenditure elasticities for 49 processed food categories using scanner data and Moschini's FAST multistage demand system. Because of the richness of the scanner data and the availability of a consistent specification of the unconditional demand functions and conditional demand functions of a weakly separable preference structure, this study overcomes previous barriers to estimating large, disaggregate

[59] Even though these other studies provide support for the findings of this paper, the estimates obtained here are not invariant to the use of other functional forms like the AIDS or Rotterdam.

demand systems. In addition, the FAST model formulation was expanded to incorporate fixed effects across time to take account of temporal heterogeneity present in the data and to provide more reliable elasticity estimates.

Overall, our estimated own-price elasticities are generally much larger, in absolute terms, than previous estimates, while our expenditure elasticities tend to be significantly lower than previous estimates. Over forty percent of the own-price elasticities where larger, on an absolute basis, than 1.0, which tended to be greater than the estimates obtained in the other studies examined. In contrast, sixty percent of the expenditure elasticities were less than or equal to zero across all four quarters examined, substantially lower than the expenditure elasticities in many of the studies examined. In part, this is due to estimating unconditional elasticities for a more disaggregate set of processed food products using scanner data. The implications for policy analysis of this result could be significant. First of all, having elasticity estimates available for more disaggregate products across time may help analysts select more appropriate elasticity values. This would aid in estimating more accurately the changes in consumer surplus from any proposed policy change; and allow policy analysts to take account of temporal fluctuations in the elasticity estimates when examining products that are subject to temporal consumption and pricing fluctuations. Second, the estimation of unconditional demand elasticities is of greater use to policy analysts for general market studies. Moschini (2001) states that: “It is clear that such conditional demand functions cannot provide the parameters (i.e. elasticities) that are typically of interest for policy questions. This is because the optimal allocation of expenditure to the goods in any one partition depends on the all prices and total expenditure (p.24).” In essence, if one wants to say something meaningful about a consumer’s response to a change in the price of a particular good, then one needs to determine what the unconditional elasticities are (Moschini, 2001). The FAST multi-stage demand system allows one to accomplish this very task.

References

1. Brown, M.G., J. Lee and J.L. Seale Jr. "Demand Relationships Among Juice Beverages: A Differential Demand System Approach." *Journal of Agriculture and Applied Economics*. 26(December 1994): 417 – 429.

2. Bhuyan, S. and R.A. Lopez. "Oligopoly Power and Allocative Efficiency in US Food and Tobacco Industries." *Journal of Agricultural Economics* 49 No. 3 (September 1998): 434-442.

3. Capps, O., Jr. "Utilizing Scanner Data to Estimate Retail Demand Functions for Meat Products." *American Journal of Agricultural Economics* 71(1989): 750-760.

4. Capps, O., Jr. and J. Lambregts. "Assessing Effects of Prices and Advertising on Purchases of Finfish and Shellfish in a Local Market in Texas." *Southern Journal of Agricultural Economics* 23 (1991): 191-194.

5. Christensen, L.R., D.W. Jorgenson and L.J. Lau. "Transcendental Logarithmic Utility Functions." *American Economic Review*. 65(1975): 367 – 383.

6. Cotterill, R.W. "Scanner Data: New Opportunities for Demand and Competitive Analysis." *Agricultural and Resource Economic Review* (October 1994): 125-139.

7. Cotterill, R.W., L.E. Haller. "Market Strategies in Branded Dairy Product Markets." In R.W. Cotterill (Ed.) *Competitive Strategy Analysis for Agricultural Marketing Cooperatives,* Boulder, CO: Westview Press, pp. 99-144, 1994.

8. Clarke, R. and S.W. Davies. "Market Structure and Price-Cost Margins." *Economica* 49(August 1982): 277-287.

9. D'Agostino, R.B. and Stephens, M.A. *Goodness of Fit Techniques*. New York: Marcel Dekker, Inc., 1986.

10. Deaton, A. and J. Muellbauer. *Economics and Consumer Behavior*. Cambridge: Cambridge University Press, 1999.

11. Diaye, M., F. Gardes, and C. Starzec. "The World According to GARP: Non-parametric Tests of Demand Theory and Rational Behavior." Working Paper, Institut National De La Statistique Et Des Etudes Economiques. September 2001. Available at http://www.crest.fr/doctravail/documents.html. March 2002.

12. Edgerton, D.L. "Weak Separability and the Estimation of Elasticities in Multistage Demand Systems." *American Journal of Agricultural Economics.* 79(February 1997): 62-79.

13. Feng, X. and W.S. Chern. "Demand for Healthy Foods in the United States." Selected paper from annual meetings of the American Agricultural Economics Association, Tampa, FL. July 30 – August 2, 2000.

14. Gisser, M. "Price Leadership and Welfare Losses in U.S. Manufacturing." *American Economic Review* 76(September 1986): 756-767.

15. Green, G.M. and J.L. Park. "Retail Demand for Whole vs. Low-Fat Milk: New Perspectives on Loss Leader Pricing." Selected Paper from annual meetings of the American Agricultural Economics Association, Salt Lake City, UT. August 2 – 5, 1998.

16. Heckman, J. and T.E. MaCurdy. " A Life Cycle Model of Female Labour Supply." *Review of Economic Studies*. 47(1980): 47 – 74.

17. Huang, K.S. "A Complete System of U.S. Demand for Food." Washington D.C.: U.S. Department of Agriculture, ERS Technical Bulletin No. 1821, September, 1993.

18. Huang, K.S. and B.Lin. "Estimation of Food Demand and Nutrient Elasticities from Household Survey Data." Washington D.C.: US Department of Agriculture, Economic Research Service. Technical Bulletin. Number 1887. August 2000.

19. Hsiao, C. *Analysis of Panel Data*. Cambridge, UK: Cambridge University Press, 1986.

20. Kinoshita, J., N. Suzuki, T. Kawamura, Y. Watanabe, H.M. Kaiser. "Estimating Own and Cross Brand Price Elasticities and Price-Cost Margin Ratios Using Store-Level Daily Scanner Data." *Agribusiness* 17 No. 4 (Autumn 2001): 515-525.

21. Lamm Jr., R.M. "A System of Dynamic Demand Functions For Food." *Applied Economics*. 14 (1982): 375 - 389.

22. Levy, D. and J. Reitzes. "Anticompetitive Effects of Mergers in Markets with Localized Competition.: *Journal of Law, Economics and Organization* 8(1992): 427-437.

23. Maynard, L.J. "Sources of Irreversible Consumer Demand in U.S. Dairy Products." Selected paper from the annual meetings of the American Agricultural Economics Association, Tampa, FL. July 30 – August 2, 2000.

24. McGuirk, A.M., P. Driscoll and J. Alwang. "Misspecification Testing: A Comprehensive Approach." *American Journal of Agricultural Eoconomics*. 75 (November 1993): 1044 – 1055.

25. McGuirk, A.M., P. Driscoll, J. Alwang and H. Huang. "System Misspecification Testing and Structural Change in the Demand for Meats." *Journal of Agricultural and Resource Economics*, 20 (1995):1-21.

26. Moschini, G. "A Flexible Multistage Demand System Based on Indirect Separability." *Southern Economic Journal* 68 No. 1 (2001): 22-41

27. Nayga, R. and O. Capps. "Tests of Weak Separability in Disaggregated Meat Products." *American Journal of Agricultural Economics* 76(November 1994): 800-808.

28. Park, J.L., R.B. Holcomb, K.C. Raper, and O. Capps, Jr. "A Demand System Analysis of Food Commodities by U.S. Households Segmented by Income." *American Journal of*

Agricultural Economics. 78(2): 290-300. (1996)

29. Peterson, E.B. and J.M. Connor. "A Comparison of Oligopoly Welfare Loss Estimates for U.S. Food Manufacturing." *American Journal of Agricultural Economics* 77(May 1995): 300-308.
30. Peterson, E.B., T.W. Hertel, and J.V. Stout. "A Critical Assessment of Supply-Demand Models of Agricultural Trade." *American Journal of Agricultural Economics* 76(November 1994): 709-721.
31. Ramsey, J.H. "Tests for Specification Errors in Classical Linear Least-Squares Regression Analysis." *Journal of the Royal Statistical Society: Series B (Methodological)*. 31 (1969): 350 – 371.
32. Schmit, T.M., C. Chung, D. Dong, H.M. Kaiser and B. Gould. "The Effects of Generic Dairy Advertising on the Household Demand for Milk and Cheese." Selected paper from annual meetings of the American Agricultural Economics Association, Tampa, FL. July 30 – August 2, 2000.
33. Spanos, A. *Statistical Foundations of Econometrics Modelling*. Cambridge: Cambridge University Press, 1986.
34. Spanos, A. *Probability Theory and Statistical Inference: Econometric Modeling with Observational Data*. Cambridge: Cambridge University Press, 1999.
35. Starek, R. and S. Stockum. "What Makes Mergers Anticompetitive? 'Unilateral Effects' Analysis Under the 1992 Merger Guidelines." *Antitrust Law Journal* 63(1995): 801-821.
36. Varian, H.R. "The Nonparametric Approach to Demand Analysis." *Econometrica* 50 No. 4 (July 1982): 945-974.
37. Wessells, C.R. and P. Wallström. "Modeling Demand Structure Using Scanner Data:

Implications for Salmon Enhancement Policies." *Agribusiness* 15 No. 4 (Autumn 1999): 449-461.

38. Willner, J. "Price Leadership and Welfare Losses in U.S. Food Manufacturing: Comment." *American Economic Review* 79(June 1989): 604-609.

39. US Department of Commerce, Bureau of Economic Analysis. "NIPA Table 2.2. Personal Consumption Expenditures by Major Type of Product." April 2002.Available at http://www.bea.doc.gov/bea/dn/nipaweb/TableViewFixed.asp#Mid.

40. US Department of Labor, Bureau of Labor Statistics. "Consumer Price Index: All Urban Consumers." June 2002 Available at http://data.bls.gov/cgi-bin/dsrv.

41. You, Z., J.E. Epperson and C.L. Huang. "A Composite System Demand Analysis for Fresh Fruits and Vegetables in the United States." *Journal of Food Distribution Research*. (October 1996): 11 – 2

Appendix C: Misspecification Tests

Misspecification tests were conducted for each individual equation and system of equations at each stage of estimation of the model. System level, individual and joint misspecification tests are presented along with the results of the tests in tables C1 and C2. Due to the number of tests conducted, only the p-values obtained from each tests are provided in the tables. The test statistics are available from the authors upon request. For misspecification tests using auxiliary regressions, F-tests were used when testing individual equations and Likelihood Ratio tests (see Spanos, 1986, p. 592) were used when testing systems of equations.

C.1 Normality Tests

For individual equations, the D'Agostino-Stephans (1986) Skewness and Kurtosis Tests were used to test for normality. To test if the residuals of a system of equations are distributed multivariate normal, Spanos (1986) suggests the following multivariate skewness and kurtosis tests:

$$\frac{T}{6}\hat{a}^{2}_{3,m} \overset{H_0}{\underset{\propto}{\sim}} c^{2}(l),\ l=\frac{1}{6}m(m+1)(m+2) \text{ and} \tag{C1}$$

$$\frac{T}{8m(m+2)}\left(\hat{a}_{4,m}-m(m+2)\right)^{2} \overset{H_0}{\underset{\propto}{\sim}} c^{2}(1), \tag{C2}$$

where $\hat{a}_{3,m}=\left[1/T^{2}\right]\sum_{t=1}^{T}\sum_{s=1}^{T}g_{ts}^{3}$, $\hat{a}_{4,m}=\frac{1}{T}\sum_{t=1}^{T}g_{tt}^{2}$, $g_{ts}=\hat{u}_t'\hat{\Omega}^{-1}\hat{u}_s$, *T* is the total number of observations, *m* is the number of equations estimated in the system, $\hat{u}_t$ is the *t*th row of the $(T\times m)$ matrix of residuals from the estimated system of equations, and Ω is the variance/covariance matrix of the residuals of the estimated system of equations.

C.2 Functional Form Tests

A RESET type functional form test (see Ramsey, 1969; Spanos, 1986) was used to test each individual equation, and was based on the significance of g (the null hypothesis being $H_0 : g = 0$) in the following auxiliary regression:

$$\hat{u}_{it,k} = d_0 + d_1 \hat{w}_{it,k} + g\hat{w}_{it,k}^2 + v_{it,k},\ i \in M,\ t \in T,\ k = 1,...,m, \tag{C3}$$

where $\hat{u}_{it,k}$ are the fitted residuals of the k^{th} equation of the system of equations being examined and $\hat{w}_{it,k}$ is the estimated share for the processed food category associated with that k^{th} equation. The set *M* represents the set of markets examined and the set *T* represents the set of time periods examined in the panel data set.

The RESET type functional form test for each system of equations is based on the significance of Γ (the null hypothesis being $H_0 : \Gamma = 0$) in the following auxiliary regression:

$$\hat{\mathbf{u}}_{it} = \Delta_0 + \Delta_1' \hat{\mathbf{w}}_{it} + \Gamma \boldsymbol{\Psi}_{it} + \mathbf{v}_{it},\ i \in M,\ t \in T, \tag{C4}$$

where Δ_0 is a $(m \times 1)$ vector of parameters, Δ_1 is a $(m \times m)$ matrix of parameters , Γ is a $((m(m+1)/2) \times m)$ matrix of parameters, $\mathbf{v}_{it}$ is a $(m \times 1)$ vector of residuals from the auxiliary regression, $\hat{\mathbf{u}}_{it} = (\hat{u}_{it,1},...,\hat{u}_{it,m})'$ is a $(m \times 1)$ vector of fitted residuals from the estimated system of equations being examined, $\hat{\mathbf{w}}_{it} = (\hat{w}_{it,1},...,\hat{w}_{it,m})'$ is a vector of estimated shares for the m processed food products represented in the system of equations, and

$$\boldsymbol{\Psi}_{it} = \left(\hat{w}_{it,1}^2, \hat{w}_{it,1}\hat{w}_{it,2},..., \hat{w}_{it,1}\hat{w}_{it,m}, \hat{w}_{it,2}^2, \hat{w}_{it,2}\hat{w}_{it,3},..., w_{it,m}^2\right)'.$$

C.3 Heterogeneity Tests

C.3.1 Market Heterogeneity

The test for market heterogeneity for each individual equation is based on the significance of Γ (the null hypothesis being $H_0 : \Gamma = 0$) in the following auxiliary regression:

$$\hat{u}_{it,k} = d_0 + d_1 \hat{w}_{it,k} + \Gamma' D_{it} + v_{it,k},\ i \in M,\ t \in T,\ k = 1,...,m, \tag{C5}$$

where Γ is a $(z \times 1)$ vector of parameters and D_{it} is a $(z \times 1)$ vector of *z* market dummies. In this paper, *z* represents the cardinality of the set *M* (i.e. $z = |M|$) , the number of markets represented in the panel data set used in the paper, 39.

The test for market heterogeneity for each system of equations is based on the significance of Γ (the null hypothesis being $H_0 : \Gamma = 0$) in the following auxiliary regression:

$$\hat{\mathbf{u}}_{it} = \Delta_0 + \Delta_1' \hat{\mathbf{w}}_{it} + \Gamma' D_{it} + \mathbf{v}_{it},\ i \in M,\ t \in T, \tag{C6}$$

where Γ is a $(z \times m)$ matrix of parameters.

C.3.2 Trend Heterogeneity

The test for trend heterogeneity for each individual equation is based on the significance of g (the null hypothesis being $H_0 : g = 0$) in the following auxiliary regression:

$$\hat{u}_{it,k} = d_0 + d_1 \hat{w}_{it,k} + g t_{it} + v_{it,k},\ i \in M,\ t \in T,\ k = 1,...,m, \tag{C7}$$

where $t = (t_1, t_2, ..., t_z)'$, t_i is a separate time trend for market $i \in M$ and $z = |M|$.

The test for trend heterogeneity for each system of equations is based on the significance of Γ (the null hypothesis being $H_0 : \Gamma = 0$) in the following auxiliary regression:

$$\hat{\mathbf{u}}_{it} = \Delta_0 + \Delta_1' \hat{\mathbf{w}}_{it} + \Gamma' t_{it} + \mathbf{v}_{it},\ i \in M,\ t \in T, \tag{C8}$$

where Γ is a $(1 \times m)$ vector of parameters.

C.3.3 Variance Heterogeneity

The test for variance heterogeneity for each individual equation is based on the significance of Γ (the null hypothesis being $H_0 : \Gamma = 0$) in the following auxiliary regression:

$$\hat{u}_{it,k}^2 = d_0 + \Gamma' D_{it,k} + v_{it,k},\ i \in M,\ t \in T,\ k = 1,...,m, \tag{C9}$$

where D_{it} is a $(z+4\times 1)$ vector of market and quarterly dummy variables and Γ is $(z+4\times 1)$ vector of parameters.

The test for variance heterogeneity for each system of equations is based on the significance of Γ (the null hypothesis being $H_0:\Gamma=0$) in the following auxiliary regression:

$$\hat{\mathbf{u}}_{it}^2=\Delta_0+\Gamma' D_{it}+\mathbf{v}_{it},\, i\in M,\, t\in T, \qquad \text{(C10)}$$

where $\mathbf{u}_{it}^2$ are the squared elements of $\mathbf{u}_{it}$, D_{it} is defined in equation (C7), and Γ is a $(z+4\times m)$ matrix of parameters.

C.4 Temporal Dependence Tests

The test for temporal dependence (or autocorrelation of order *p*) for each individual equation is based on the significance of Γ (the null hypothesis being $H_0:\Gamma=0$) in the following auxiliary regression:

$$\hat{u}_{it,k}=d_0+d_1\hat{w}_{it,k}+\Gamma' U_{it,p,k}+v_{it,k},\, i\in M,\, t\in T,\, k=1,...,m, \qquad \text{(C11)}$$

where Γ is a $(p\times 1)$ vector of parameters and $U_{it,p,k}=(\hat{u}_{i,t-1,k},...,\hat{u}_{i,t-p,k})'$ is a $(p\times 1)$ vector of lagged fitted residuals (Spanos, 1986). In this paper, tests were conducted for $p=1$ and $p=2$.

The test for temporal dependence for each system of equations is based on the significance of Γ (the null hypothesis being $H_0:\Gamma=0$) in the following auxiliary regression:

$$\hat{\mathbf{u}}_{it}=\Delta_0+\Delta_1'\hat{\mathbf{w}}_{it}+\Gamma'\mathbf{U}_{it,p}+\mathbf{v}_{it},\, i\in M,\, t\in T, \qquad \text{(C12)}$$

where Γ is a $(pm\times m)$ matrix of parameters and $\mathbf{U}_{it,p}=(U_{it,p,1},...,U_{it,p,m})$ (Spanos, 1986).

C.5 Heteroskedasticity Tests

The test for heteroskedasticity for each individual equation is based on the significance of g (the null hypothesis being $H_0:g=0$) in the following auxiliary regression (Spanos, 1986):

$$\hat{u}^2_{it,k} = d_0 + g\hat{w}^2_{it,k} + v_{it,k} \,.\, i \in M,\, t \in T,\, k = 1,...,m\,, \tag{C13}$$

The test for heteroskedasticity for each system of equations is based on the significance of Γ (the null hypothesis being $H_0 : \Gamma = 0$) in the following auxiliary regression:

$$\Pi_{it} = \Delta_0 + \Gamma'\Psi_{it} + \mathbf{v}_{it},\, i \in M,\, t \in T\,, \tag{C14}$$

where $\Pi_{it} = \left(\hat{u}^2_{it,1}, \hat{u}_{it,1}\hat{u}_{it,2},..., \hat{u}_{it,1}\hat{u}_{it,m}, \hat{u}_{it,2}, \hat{u}_{it,2}\hat{u}_{it,3},..., \hat{u}^2_{it,m}\right)$, Δ_0 is a $\left(m(m+1)/2 \times 1\right)$ vector of parameters, Γ is a $\left((m(m+1)/2) \times m\right)$ matrix of parameters, Ψ_{it} is defined in equation (C4), and $\mathbf{v}_{it}$ is $\left(m(m+1)/2 \times 1\right)$ vector of residuals for the auxiliary regression (Spanos, 1986).

C.6 Joint Conditional Mean Tests

The joint conditional mean tests conducted in this study are based upon the joint tests proposed by McGuirk, Driscoll and Alwang (1993), McGuirk *et al*. (1995), and Spanos (1999).

The joint conditional mean test for each individual equation is based on the significance of Γ (the null hypothesis being $H_0 : \Gamma = 0$) in the following auxiliary regression:

$$\hat{u}_{it,k} = d_0 + d_1\hat{w}_{it,k} + \Gamma'\Lambda_{it,k} + v_{it,k},\, i \in M,\, t \in T,\, k = 1,...,m\,, \tag{C15}$$

where Γ is a $\left(p + z + 2 \times 1\right)$ vector of parameters, $\Lambda_{it,k} = \left(U_{it,p,k}, D_{it}, t_{it}, \hat{w}^2_{it,k}\right)$, $U_{it,p,k}$ is defined in equation (C11) and D_{it} is defined in equation (C5). In addition, tests examining the significance of the parameters of Γ corresponding to the different elements of Λ using equation (C15) as the unrestricted model were performed.

The joint conditional mean test for each system of equations is based on the significance of Γ (the null hypothesis being $H_0 : \Gamma = 0$) in the following auxiliary regression:

$$\hat{\mathbf{u}}_{it} = \Delta_0 + \Delta_1'\hat{\mathbf{w}}_{it} + \Gamma'\Lambda_{it} + \mathbf{v}_{it},\, i \in M,\, t \in T\,, \tag{C16}$$

where Γ is a $\left(pm+z+m+1\times m\right)$ matrix of parameters, $\Lambda_{it}=\left(\mathbf{U}_{it,p},D_{it},t_{it},\hat{\mathbf{w}}_{it}^{2}\right)$, $\mathbf{U}_{it,p}$ is defined in equation (C12), and D_{it} is defined in equation (C5). In addition, tests examining the significance of the parameters of Γ corresponding to the different elements of Λ using equation (C16) as the unrestricted model were performed.

C.7 Spatial Dependence Tests

The tests for spatial dependence for each individual equation are based on the significance of Γ (the null hypothesis being $H_0:\Gamma=0$) in the following auxiliary regression:

$$\hat{u}_{it,k}=d_0+d_1\hat{w}_{it,k}+\Gamma'\Phi_{it,k}+v_{it,k},\ i\in M,\ t\in T,\ k=1,...,m \qquad \text{(C17)}$$

where Γ is a $\left(h\times1\right)$ vector of parameters and $\Phi_{it,k}=\left(\hat{u}_{pt,k},\hat{u}_{qt,k},\hat{u}_{rt,k},...\right)$ (with *h* elements) where the (market) indices *p*,*q*,*r*,… are chosen appropriately from the set *M*. In this paper, these indices were chosen to be other markets in geographical proximity to the market being examined. Spatial dependence for eight sets of markets were examined. The sets are:

Set 1 – Cincinnati/Dayton, Cleveland, Detroit, Grand Rapids, Pittsburgh;

Set 2 – Chicago, Hartford/Springfield, Indianapolis, Milwaukee, Minneapolis/St. Paul;

Set 3 – Albany, Baltimore/Washington, Boston, Buffalo/Rochester, New York, Philadelphia;

Set 4 – Louisville, Memphis, Nashville, St. Louis;

Set 5 – Dallas/Ft. Worth, Houston, Oklahoma City, San Antonio;

Set 6 – Denver, Omaha, Phoenix/Tucson, Salt Lake City, Wichita;

Set 7 – Los Angeles, Sacramento, San Diego, San Francisco/Oakland, Seattle/Tacoma; and

Set 8 – Atlanta, Birmingham, Miami/Ft. Lauderdale, Raleigh/Greensboro, Tampa/St. Pete.

The tests for spatial dependence for each system of equations are based on the significance of Γ (the null hypothesis being $H_0:\Gamma=0$) in the following auxiliary regression:

$$\hat{\mathbf{u}}_{it} = \Delta_0 + \Delta'\hat{\mathbf{w}}_{it} + \Gamma'\Phi_{it} + \mathbf{v}_{it},\ i \in M,\ t \in T\ , \tag{C18}$$

where Γ is a $(hm \times m)$ matrix of parameters and $\Phi_{it} = (\hat{\mathbf{u}}_{pt}, \hat{\mathbf{u}}_{qt}, \hat{\mathbf{u}}_{rt}, ...)$ (with $h \cdot m$ elements) where the (market) indices *p*,*q*,*r*,… are chosen appropriately from the set *M*. The indices were chosen using the same method described above.

Table C1: P-Values for Individual Misspecification Tests

Equation[1]	Normality Tests		Functional Form Test	Heterogeneity Tests			Temporal Dependence Tests		Heteroskedasticity Test
	Skewness	**Kurtosis**		**Market**	**Time Trend**	**Variance**	***p* = 1**	***p* = 2**	
				Conditional Share Equations					
Beverages[2]	0.000	0.000	0.000	0.000	0.446	0.000	0.000	0.000	0.000
Bottled Water	0.000	0.000	0.326	0.000	0.995	0.000	0.000	0.000	0.047
Low-Calorie Soft Drinks	0.000	0.000	0.314	0.000	0.919	0.000	0.000	0.000	0.227
Regular Soft Drinks	0.719	0.000	0.762	0.000	0.920	0.000	0.000	0.000	0.336
Coffee	0.000	0.011	0.006	0.000	0.932	0.000	0.000	0.000	0.005
Coffee Creamer and Flavorings	0.067	0.442	0.597	0.000	0.983	0.000	0.000	0.000	0.001
Tea	0.077	0.001	0.119	0.000	0.954	0.000	0.000	0.000	0.792
Refrigerated Juices	0.000	0.022	0.298	0.000	0.921	0.000	0.000	0.000	0.936
Shelf-Stable Juices	0.007	0.000	0.862	0.000	0.868	0.000	0.000	0.000	0.332
Diary Products[2]	0.000	0.000	0.000	0.000	0.999	0.000	0.000	0.000	0.000
Cheese (not shredded)	0.137	0.000	0.001	0.000	0.728	0.000	0.000	0.000	0.419
Shredded Cheese	0.931	0.000	0.005	0.000	0.934	0.000	0.000	0.000	0.000
Imitation Cheese	0.000	0.028	0.223	0.000	0.921	0.000	0.000	0.000	0.001
Whole Milk	0.376	0.014	0.225	0.000	0.437	0.000	0.000	0.000	0.013
Skim/Low-Fat Milk	0.000	0.054	0.011	0.000	0.467	0.000	0.000	0.000	0.658
Powdered/Condensed Milk	0.000	0.000	0.294	0.000	0.952	0.000	0.000	0.000	0.247
Cocoa Mix/Flavored Milks	0.000	0.000	0.324	0.000	0.969	0.000	0.000	0.000	0.000
Ice Cream and Yogurt	0.003	0.034	0.550	0.000	0.947	0.000	0.000	0.000	0.073
Milled Grain and Pasta[2]	0.000	0.000	0.000	0.000	0.840	0.000	0.000	0.000	0.000
Peanut Butter	0.698	0.022	0.856	0.000	0.860	0.000	0.000	0.000	0.010
Jams and Jellies	0.001	0.751	0.668	0.000	0.704	0.000	0.000	0.000	0.720
Bread	0.927	0.000	0.891	0.000	0.991	0.000	0.000	0.000	0.001
Muffins and Rolls	0.839	0.029	0.667	0.000	0.998	0.000	0.000	0.000	0.001
Rice	0.000	0.000	0.077	0.000	0.998	0.000	0.000	0.000	0.695
Pasta	0.000	0.003	0.425	0.000	0.999	0.000	0.000	0.000	0.514

Table C1 cont’d.

	Normality Tests		**Functional Form Test**	**Heterogeneity Tests**			**Temporal Dependence Tests**		**Heteroskedasticity Test**
Equation[1]	**Skewness**	**Kurtosis**		**Market**	**Time Trend**	**Variance**	***p* = 1**	***p* = 2**	
Spaghetti Sauce	0.000	0.654	0.079	0.000	0.990	0.000	0.000	0.000	0.003
Canned Soup	0.632	0.004	0.195	0.000	0.984	0.000	0.000	0.000	0.000
Fruits and Vegetables[2]	0.000	0.010	0.000	0.000	0.836	0.000	0.000	0.000	0.000
Dried Fruits	0.000	0.010	0.037	0.000	0.941	0.000	0.000	0.000	0.080
Shelf-Stable Fruits	0.000	0.839	0.918	0.000	0.990	0.000	0.000	0.000	0.047
Baked Beans	0.007	0.008	0.705	0.000	0.999	0.000	0.000	0.000	0.000
Shelf-Stable Vegetables	0.314	0.000	0.001	0.000	0.969	0.000	0.000	0.000	0.002
Frozen Vegetables	0.000	0.003	0.283	0.000	0.885	0.000	0.000	0.000	0.041
Baking, Condiments and Deserts[2]	0.000	0.031	0.000	0.000	0.888	0.000	0.000	0.000	0.000
Sour Cream	0.000	0.000	0.014	0.000	0.428	0.000	0.000	0.000	0.863
Refrigerated Pickles/Relish	0.000	0.062	0.118	0.000	0.408	0.000	0.000	0.000	0.159
Shelf-Stable Pickles/Relish	0.002	0.119	0.132	0.000	0.970	0.000	0.000	0.000	0.013
Pourable Salad Dressings	0.014	0.585	0.093	0.000	0.973	0.000	0.000	0.000	0.009
Dry Dressings/Toppings	0.038	0.039	0.134	0.000	0.975	0.000	0.000	0.000	0.860
Mayonnaise	0.700	0.677	0.785	0.000	0.941	0.000	0.000	0.000	0.000
Ketchup	0.591	0.000	0.047	0.000	0.833	0.000	0.000	0.000	0.000
Sauces and Marinades	0.000	0.000	0.128	0.000	0.962	0.000	0.000	0.000	0.000
Gelatin/Pudding Mix	0.000	0.202	0.147	0.000	0.402	0.000	0.000	0.000	0.000
Popcorn	0.005	0.080	0.568	0.000	0.786	0.000	0.000	0.000	0.000
Snack Nuts	0.000	0.102	0.375	0.000	0.785	0.000	0.000	0.000	0.001
Candy and Mints	0.000	0.000	0.968	0.000	0.905	0.000	0.000	0.000	0.101
Mixes	0.361	0.087	0.008	0.000	0.998	0.000	0.000	0.000	0.432
Seasonings/Preservatives	0.000	0.381	0.773	0.000	0.879	0.000	0.000	0.000	0.479
Syrups	0.848	0.011	0.719	0.000	0.985	0.000	0.000	0.000	0.011
Group Share Equations									
System[2]	0.000	0.000	0.000	0.000	0.105	0.000	0.000	0.000	0.000

Table C1 cont'd.

	Normality Tests		**Functional Form Test**	**Heterogeneity Tests**			**Temporal Dependence Tests**		**Heteroskedasticity Test**
Equation[1]	**Skewness**	**Kurtosis**		**Market**	**Time Trend**	**Variance**	***p* = 1**	***p* = 2**	
Beverages	0.000	0.000	0.841	0.000	0.885	0.000	0.000	0.000	0.000
Dairy Products	0.000	0.000	0.947	0.000	0.731	0.000	0.000	0.000	0.002
Milled Grain and Pasta	0.000	0.000	0.816	0.000	0.826	0.000	0.000	0.000	0.000
Fruits and Vegetables	0.000	0.298	0.230	0.000	0.829	0.000	0.000	0.000	0.000
Baking, Condiments and Deserts	0.000	0.000	0.048	0.000	0.770	0.000	0.000	0.000	0.000

[1] In order to estimate the five systems of conditional share equations in the model, the conditional share equations for frozen juices, cheese spreads, dry soup, frozen fries and onion rings, and flour are dropped during estimation. The group share equation for the "all other goods" composite good is dropped in order to estimate the system of group share equations.

[2] The results for these categories are the p-values for the system wide misspecification tests for the systems of conditional and group share equations.

Table C2: P-Values for Joint Conditional Mean Misspecification Test and Spatial Dependence Tests

	Joint Conditional Mean Test					Spatial Dependence							
Equation[1]	Overall	1 Lag	Market Dummies	Time Trend	RESET (2)	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7	Set 8
Conditional Share Equations													
Beverages[2]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Bottled Water	0.000	0.000	0.000	0.230	0.018	0.000	0.000	0.000	0.000	0.001	0.000	0.000	0.000
Low-Calorie Soft Drinks	0.000	0.000	0.000	0.573	0.054	0.000	0.000	0.000	0.959	0.000	0.000	0.000	0.000
Regular Soft Drinks	0.000	0.000	0.000	0.006	0.127	0.000	0.000	0.000	0.774	0.000	0.000	0.019	0.000
Coffee	0.000	0.000	0.000	0.000	0.002	0.000	0.317	0.000	0.000	0.115	0.000	0.004	0.001
Coffee Creamer and Flavorings	0.000	0.000	0.000	0.221	0.054	0.000	0.000	0.000	0.000	0.000	0.000	0.011	0.000
Tea	0.000	0.000	0.000	0.013	0.745	0.000	0.000	0.000	0.000	0.000	0.000	0.011	0.000
Refrigerated Juices	0.000	0.000	0.000	0.000	0.156	0.000	0.000	0.000	0.000	0.000	0.397	0.000	0.000
Shelf-Stable Juices	0.000	0.000	0.000	0.000	0.116	0.044	0.000	0.000	0.000	0.007	0.000	0.000	0.000
Diary Products[2]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Cheese (not shredded)	0.000	0.000	0.000	0.000	0.319	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.025
Shredded Cheese	0.000	0.000	0.000	0.000	0.014	0.000	0.038	0.000	0.000	0.000	0.000	0.000	0.004
Imitation Cheese	0.000	0.000	0.013	0.054	0.000	0.000	0.000	0.000	0.000	0.397	0.000	0.000	0.053
Whole Milk	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.000	0.001	0.000	0.000	0.000	0.010
Skim/Low-Fat Milk	0.000	0.000	0.000	0.808	0.958	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001
Powdered/Condensed Milk	0.000	0.006	0.000	0.000	0.018	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Cocoa Mix/Flavored Milks	0.000	0.959	0.000	0.000	0.000	0.000	0.044	0.001	0.000	0.090	0.000	0.035	0.000
Ice Cream and Yogurt	0.000	0.000	0.000	0.000	0.096	0.043	0.000	0.000	0.021	0.000	0.000	0.000	0.442
Milled Grain and Pasta[2]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Peanut Butter	0.000	0.000	0.000	0.001	0.844	0.000	0.000	0.004	0.000	0.000	0.000	0.000	0.000
Jams and Jellies	0.000	0.000	0.000	0.235	0.365	0.020	0.000	0.057	0.000	0.144	0.000	0.000	0.000
Bread	0.000	0.000	0.000	0.252	0.858	0.000	0.000	0.000	0.000	0.035	0.000	0.000	0.000
Muffins and Rolls	0.000	0.000	0.000	0.035	0.812	0.000	0.000	0.013	0.000	0.000	0.000	0.000	0.000
Rice	0.000	0.000	0.000	0.026	0.000	0.076	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Pasta	0.000	0.000	0.000	0.967	0.017	0.000	0.000	0.000	0.000	0.000	0.000	0.998	0.000
Spaghetti Sauce	0.000	0.000	0.000	0.963	0.149	0.003	0.000	0.449	0.002	0.005	0.000	0.000	0.000

Table C2 cont'd.

	Joint Conditional Mean Test					Spatial Dependence							
Equation[1]	Overall	1 Lag	Market Dummies	Time Trend	RESET (2)	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7	Set 8
Canned Soup	0.000	0.024	0.000	0.684	0.001	0.000	0.000	0.000	0.009	0.000	0.015	0.000	0.000
Fruits and Vegetables[2]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Dried Fruits	0.000	0.000	0.000	0.000	0.000	0.005	0.004	0.000	0.000	0.153	0.001	0.000	0.000
Shelf-Stable Fruits	0.000	0.000	0.000	0.000	0.773	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.004
Baked Beans	0.000	0.000	0.000	0.000	0.479	0.000	0.000	0.004	0.313	0.000	0.000	0.000	0.000
Shelf-Stable Vegetables	0.000	0.000	0.000	0.000	0.004	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Frozen Vegetables	0.000	0.000	0.000	0.000	0.002	0.031	0.000	0.030	0.000	0.001	0.000	0.044	0.000
Baking, Condiments and Deserts[2]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Sour Cream	0.000	0.000	0.000	0.175	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Refrigerated Pickles/Relish	0.000	0.000	0.000	0.841	0.013	0.000	0.000	0.102	0.000	0.000	0.000	0.000	0.000
Shelf-Stable Pickles/Relish	0.000	0.378	0.000	0.031	0.002	0.001	0.000	0.000	0.141	0.252	0.000	0.000	0.000
Pourable Salad Dressings	0.000	0.004	0.000	0.400	0.761	0.000	0.090	0.000	0.000	0.000	0.001	0.227	0.903
Dry Dressings/Toppings	0.000	0.000	0.000	0.756	0.000	0.001	0.000	0.000	0.000	0.000	0.003	0.005	0.000
Mayonnaise	0.000	0.000	0.000	0.000	0.572	0.000	0.058	0.000	0.000	0.000	0.000	0.607	0.207
Ketchup	0.000	0.000	0.000	0.001	0.915	0.000	0.000	0.000	0.000	0.006	0.000	0.036	0.000
Sauces and Marinades	0.000	0.003	0.000	0.295	0.006	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Gelatin/Pudding Mix	0.000	0.000	0.000	0.051	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.005	0.013
Popcorn	0.000	0.000	0.000	0.001	0.001	0.001	0.000	0.000	0.528	0.396	0.000	0.000	0.000
Snack Nuts	0.000	0.000	0.000	0.016	0.000	0.001	0.000	0.000	0.000	0.077	0.000	0.002	0.000
Candy and Mints	0.000	0.147	0.000	0.000	0.900	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000
Mixes	0.000	0.000	0.000	0.100	0.440	0.234	0.000	0.000	0.006	0.000	0.000	0.038	0.000
Seasonings/Preservatives	0.000	0.000	0.000	0.089	0.598	0.000	0.000	0.000	0.000	0.000	0.002	0.000	0.000
Syrups	0.000	0.000	0.000	0.872	0.867	0.000	0.000	0.000	0.000	0.977	0.035	0.012	0.000
Group Share Equations													
System[2]	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Beverages	0.000	0.000	0.001	0.380	0.008	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.193
Dairy Products	0.000	0.000	0.000	0.100	0.000	0.000	0.000	0.000	0.009	0.000	0.000	0.048	0.000

Table C2 cont'd.

	Joint Conditional Mean Test					**Spatial Dependence**							
Equation[1]	**Overall**	**1 Lag**	**Market Dummies**	**Time Trend**	**RESET (2)**	**Set 1**	**Set 2**	**Set 3**	**Set 4**	**Set 5**	**Set 6**	**Set 7**	**Set 8**
Milled Grain and Pasta	0.000	0.000	0.000	0.059	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.007
Fruits and Vegetables	0.000	0.000	0.000	0.881	0.004	0.000	0.000	0.000	0.000	0.000	0.000	0.155	0.007
Baking, Condiments and Deserts	0.000	0.000	0.000	0.481	0.162	0.000	0.000	0.000	0.000	0.000	0.000	0.547	0.000

[1] In order to estimate the five systems of conditional share equations in the model, the conditional share equations for frozen juices, cheese spreads, dry soup, frozen fries and onion rings, and flour are dropped during estimation. The group share equation for the "all other goods" composite good is dropped in order to estimate the system of group share equations.

[2] The results for these categories are the p-values for the system wide misspecification tests for the systems of conditional and group share equations.