Studies in Nonlinear Dynamics and Econometrics

Quarterly Journal
Volume 4, Number 3
The MIT Press

Studies in Nonlinear Dynamics and Econometrics (ISSN 1081-1826) is a quarterly journal published electronically on the Internet by The MIT Press, Cambridge, Massachusetts, 02142-1407. Subscriptions and address changes should be addressed to MIT Press Journals, Five Cambridge Center, Cambridge, MA 02142-1407; (617)253-2889; e-mail: journals-orders@mit.edu. Subscription rates are: Individuals $40.00, Institutions $135.00. Canadians add additional 7% GST. Submit all claims to: journals-claims@mit.edu. Prices subject to change without notice.

An Algorithm for Estimating Multivariate Catastrophe Models: GEMCAT II

Rense Lange
Division of Assessment
Illinois State Board of Education and
Southern Illinois School of Medicine
rlange@smtp.isbe.state.il.us

Terence A. Oliva
Department of Marketing
Temple University
oliva@surfer.sbm.temple.edu

Sean R. McDade
Gallup Organization
mcdade@yahoo.com

Abstract. *Following the framework in Oliva et al. 1987, GEMCAT II implements a flexible method to test catastrophe models containing multivariate (i.e., latent) variables while allowing for a priori variable specifications. The system uses an efficient hybrid minimization algorithm combining the Downhill Simplex and Powell's Conjugate Gradient method. GEMCAT II is compiled in Delphi V3.0 and is sufficiently fast to allow for the use of resampling methods (bootstrap as well as jackknife) to determine the statistical significance of latent variables' indicator weights. In addition, a* Pseudo-R^2 *index of model fit is provided, together with a test of significance, and options are included to facilitate competitive model tests of nested and non-nested catastrophe models as well as linear models. Two simulation studies are reported. Based on 61,250 simulated data sets of varying sizes, the first study addressed the effects of indicator reliability on the quality of the weight estimations, and the second dealt with the problem of false positives in model identification. The results strongly support the viability of the GEMCAT II approach over a wide range of reasonable indicator reliabilities and sample sizes. Moreover, it proved possible to distinguish reliably between cusp catastrophes and linear models based on the* Pseudo-R^2 *values. Finally, GEMCAT II is applied to actual market data in order to demonstrate its use in an economic context. Using 34 quarters of panel data, we examine the fit of a cusp catastrophe model of organizational product adoption as applied to competing software standards in the presence of network externalities. The results are consistent with economic theory and published work on network externalities. This example also illustrates GEMCAT II's bootstrap tests for indicator weights and its options for competitive model testing.*

Keywords. catastrophe theory, competitive model tests, computer simulation, multivariate modeling, parameter estimation, resampling statistics

1 Introduction

The application of nonlinear approaches to model dynamic phenomena has been limited by the availability of supporting statistical methods for developing empirical estimations of such models. In particular, issues surrounding nonlinearities, discontinuities, and multivaluedness have presented difficult methodological problems for developers. This paper reports on techniques for estimating and testing the class of nonlinear dynamic models characterized by René Thom (1975) as catastrophe models. As is shown in Table 1, the present research focuses on four catastrophes with a single behavioral dimension.

One of the first applications of catastrophe theory to economic phenomena was by Zeeman (1974) on the unstable behavior of stock exchanges. Since that time there have been other economic applications of catastrophe theory. Examples include Varian's (1979) work on the business cycle, Ho and Saunders' (1980) model of bank failures, Fischer and Jammernegg's (1986) examination of the Phillips curve, and Oliva's (1994) work on technological choice under changing network externalities. For the most part these works have been theoretical or conceptual in nature. Although such approaches have merits in their own right, theoretical formulations ultimately need to be subjected to empirical verification. Unfortunately, this has proven difficult, as early empirical research in this area has been severely limited by a dearth of econometric techniques to estimate catastrophe models (Oliva et al. 1987).

To redress this, in the early to mid-1980s two different approaches were developed to estimate statistical catastrophe theory models. A first important step forward was Loren Cobb's (1978, 1981) work on statistical distributions for catastrophe models in the biosciences. Cobb developed a technique based on the method-of-moments approach and used maximum-likelihood estimation. Drawing on Cobb's analytical work, Guastello (1982, 1995) developed a statistical specification for the cusp model by starting with the following derivative:

$$dZ = \left(Z^3 - X - YZ\right) dt = 0 \tag{1.1}$$

Letting $dt = 1$ in Equation (1.1) and inserting beta weights yields the following statistical expression that can be fitted to actual data by standard linear regression techniques:

$$\Delta Z = \beta_0 + \beta_1 Z^3 + \beta_2 YZ + \beta_3 X + \varepsilon \tag{1.2}$$

As is the convention throughout, Z represents the dependent variable and ΔZ denotes the change in Z over some fixed time interval. Further, variables X and Y denote the "control factors" (independent variables) that govern the system. The first derivatives of the other catastrophe models considered in this paper are presented in Table 1. Their corresponding regression formulations can be found in Guastello 1995.

A limitation of Guastello's and Cobb's method is that it does not allow researchers to specify models in terms of specific combinations of multiple indicator variables. Rather, if a catastrophe exists, the approach is capable only of identifying individual variables that are associated either with the control factor X or control factor Y in Equation (1.1), but *not* both (Guastello 1992). This limits the usefulness of the approach for

Table 1
Four of Thom's elementary catastrophes

Catastrophe	Control dimensions	Behavior dimensions	Function	First derivative
Fold	1	1	$\frac{1}{2}Z^3 - XZ$	$Z^2 - X$
Cusp	2	1	$\frac{1}{4}Z^4 - XZ - \frac{1}{2}YZ^2$	$Z^3 - X - YZ$
Swallowtail	3	1	$1/5Z^5 - XZ - \frac{1}{2}YZ^2 - \frac{1}{2}VZ^3$	$Z^4 - X - YZ - VZ^2$
Butterfly	4	1	$1/6Z^6 - XZ - \frac{1}{2}YZ^2 - 1/3VZ^3 - \frac{1}{4}UZ^4$	$Z^5 - X - YZ - VZ^2 - UZ^3$

economics and behavioral research, in which realistic catastrophe models often need to rely on multivariate definitions of X, Y, and Z. In addition, the same indicator might contribute to more than one control variable in a catastrophe (e.g., the same variable might be an X as well as a Y indicator). To circumvent these limitations, collections of indicator variables are sometimes averaged or, alternatively, subjected to other grouping techniques like principal-component analysis. Unfortunately, computer simulations indicate that these approaches often result in a loss of valuable information (Oliva et al. 1987).

A solution to this problem was developed by Oliva et al. (1987). Their method, the General Multivariate Methodology for Estimating Catastrophe Models (GEMCAT), uses a scaling approach that allows for a priori variable specifications that allow multivariate constructs in all variables. Hence, GEMCAT is more useful to researchers who need a confirmatory estimation approach for a given model (Alexander et al. 1992). This approach forms the basis of the GEMCAT II software (Lange 1998), which adds a variety of statistical techniques and testing options to the original formulation.

The following section describes the original GEMCAT approach in some detail before focusing on its extensions as incorporated in GEMCAT II. Next, we present the results of simulations that studied the performance of GEMCAT II as a function of measurement errors in the catastrophe variables for various sample sizes. In addition, we study aspects of the often ignored false-positives problem; in particular, the likelihood that linear data will be classified as following a catastrophe model. Finally, we report an illustrative empirical application of the GEMCAT II system involving the organizational adoption of high-technology products when externalities are present. Because the cusp model is the most widely used catastrophe model in actual practice, the presentation focuses on this model. However, the fold catastrophe and linear models are also touched upon in the context of competitive model testing.

2 The GEMCAT Approach

Oliva et al.'s (1987) GEMCAT approach allows all variables in a catastrophe to be latent composites. To accomplish this, the variables X, Y, and Z in the canonical cusp equation (Equation (1.1)) are redefined as

$$X = \sum_{i=1}^{I} \theta_{xi} x_i \tag{2.1}$$

$$Y = \sum_{j=1}^{J} \theta_{yj} y_j \tag{2.2}$$

$$Z = \sum_{k=1}^{K} \theta_{zk} z_k \tag{2.3}$$

where X, Y, and Z are the values of latent variables and θ_{xi}, θ_{yj}, and θ_{zk} are the weights (impact coefficients) of the indicators x_i, y_j, and z_k of the cusp surface (examples are shown in later sections). Substitution of Equations (2.1), (2.2), and (2.3) into Equation (1.1) yields

$$dZ = Z^3 - X - ZY \tag{2.4}$$

The approach used to fit Equation (2.4) is essentially the same as that in other commonly used multivariate techniques. For instance, GEMCAT estimates a cusp model by finding those values for the weights θ_{xi}, θ_{yj}, and θ_{zk} that across all observations $t = 1, \ldots, T$ minimize the objective function Φ relative to Equation (2.4), that is,

$$\min_{\theta_{xi},\theta_{yj},\theta_{zk}} = \Phi = \|e_t^2\| = \sum_{t=1}^{T} \left(Z_t^3 - X_t - Z_t Y_t\right)^2 \tag{2.5}$$

where e_t denotes the error for case t. A major issue in solving Equation (2.5) is to find an optimization method that can deal with the peculiarities of the cusp model while avoiding trivial solutions like

$\theta_{xi} = \theta_{yj} = \theta_{zk} = 0$. Oliva et al. (1987) solved this problem through the use of a modified version of the controlled random search (CRS) proposed by Price (1976). This algorithm does not require the objective function Φ to be differentiable, nor does it require the indicator variables to be continuous. Note that the parameters are not identifiable as a whole, since Φ is unchanged if weights $\theta'_{xi} = -\theta_{xi}$ and $\theta'_{yj} = -\theta_{yj}$ are used. Fortunately, setting $\theta'_{y1} = 1$ helps avoid this and other degeneracies (Oliva et al. 1987).

The actual program used to minimize Φ was implemented by Wayne DeSarbo in the APL computer language, and details of the modified CRS algorithm, including a flowchart, can be found in Oliva et al. 1987. Unfortunately, GEMCAT never achieved widespread acceptance. Being a prototype, it was rather inflexible and user-unfriendly, in large part because it required programming experience in APL. Moreover, because APL is an interpreted language, the program was rather slow. Consequently, although hundreds of copies of the APL listing were requested and sent out to interested parties, only a few researchers actually took the trouble to create the program and use it.

2.1 GEMCAT II

Using the basic multivariate conceptualization of the original GEMCAT, Lange (1998) developed the GEMCAT II system, which added a number of important features to the overall approach. In particular, this system includes a slight generalization of the latent variable definitions and weight constraints, more efficient minimization algorithms, various statistical tests to determine the significance of the indicator weights, options to facilitate competitive model testing, and an R^2-type index of overall model fit. Each of these features is discussed in turn below. A fully operational copy of the latest version of the GEMCAT II software, together with a provisional manual, can be downloaded from the website http://www.sbm.temple.edu/~oliva/cat-theory.htm.

2.2 Latent variables

GEMCAT II provides a slightly more general definition of the latent variables by including offsets θ_{*0} in addition to the multiplicative factors in Equations (2.1), (2.2), and (2.3). Thus, arbitrary linear transformations are allowed. For instance, the latent cusp variables X, Y, and Z can now be defined as:

$$\begin{aligned} X &= \theta_{x0} + \sum_{i=1}^{I} \theta_{xi} x_i \\ Y &= \theta_{y0} + \sum_{j=1}^{J} \theta_{yj} y_j \\ Z &= \theta_{z0} + \sum_{k=1}^{K} \theta_{zk} z_k \end{aligned}$$

For later reference, we define W as the ordered set of all indicator weights (i.e., across catastrophe variables in alphabetical order). Thus, in the above, $W = \{\theta_{x0}, \ldots, \theta_w, \ldots, \theta_{zK}\}$. Each θ_w, including the offsets θ_{*0}, can be estimated by GEMCAT II; alternatively, users can assign a particular fixed value. Hence, W can be partitioned into the set $FREE = \{\theta_1, \ldots, \theta_p, \ldots, \theta_P\}$, containing the weights to be estimated, and the set *FIXED*, containing the remaining weights whose values were preset by the user. Note that the same indicators may contribute to more than one latent variable. Thus, the same indicator variable may have one or more weights in the *FREE* set as well as in the *FIXED* set.

The preceding is illustrated by Dou's (1999; Dou and Ghose 2000) research on on-line store sales in e-commerce. Using a predator-prey analogy, Dou derived a quantitative model that describes the dynamics of the customers attracted by a small website vs. a large website as

$$Z^3 - YZ - X = 0.$$

In this cusp model, the dependent variable Z reflects the unit sales (S) of the small website relative to that of the large website, and X and Y are latent control variables that reflect price (P) and visitation rate (V). The

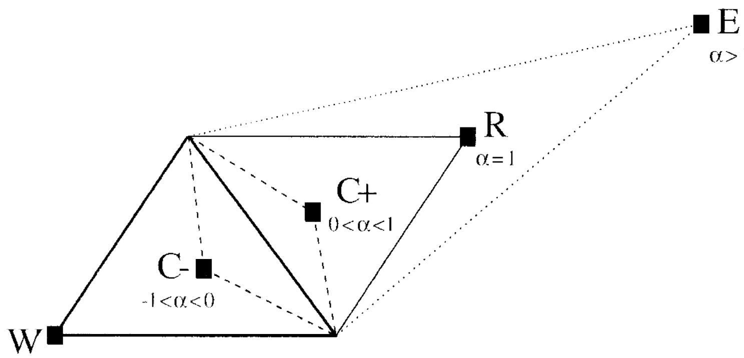

Figure 1
Effects of expansion (E), reflection (R), negative contraction ($C-$), and positive contraction ($C+$) on the worst point (W).

application is interesting because it uses functions of P and V as indicator variables. In particular, the latent GEMCAT variables are

$$\begin{aligned} X &= \theta_{x0} + \theta_{x1} P V^2 + \theta_{x2} V^3 \\ Y &= \theta_{y0} + \theta_{y1} P V + \theta_{y2} V^2 \\ Z &= \theta_{z0} + \theta_{z1} S, \end{aligned}$$

and thus $FREE = \{\theta_{x0}, \theta_{x1}, \theta_{x2}, \theta_{y0}, \theta_{y1}, \theta_{y2}, \theta_{z0}\}$. Further, following the recommendations in Oliva et al. (1987), the Z indicator θ_{z1} is set to unity; hence, $FIXED = \{\theta_{z1}\} = \{1\}$. We note that GEMCAT II currently cannot compute terms like PV^2, and all composite quantities must be precomputed and entered as if they were individual variables.

2.3 Minimizing Φ

The inefficiency of the GEMCAT's CRS algorithm led to a search for alternative approaches. Unfortunately, the inherent discontinuities in the shapes of the various catastrophe surfaces provide a warning sign against the use of Newton-Raphson type approaches, as finding suitable starting values is difficult because of the potentially large numbers of indicator weights. Also, the identification of the appropriate Hessian matrices with function derivatives for each of the catastrophes in the presence of varying numbers of indicator weights is nontrivial. For these reasons, a hybrid approach was implemented that combines two different algorithms: the Downhill Simplex and Powell's Conjugate Gradient method. Like CRS, neither algorithm requires the computation of any derivatives.

2.3.1 The Downhill Simplex algorithm The Simplex algorithm, as originally proposed by Spendley, Hext, and Himsworth (1962) and further generalized by Nelder and Mead (1965), is easily understood by conceptualizing a P-parameter minimization problem ($P > 1$) as being analogous to constructing a simplex that surrounds a solution point in P-space as closely as possible. Obviously, doing this requires $P + 1$ points. Rather amazingly, this can always be done through a series of transformations involving the "worst" point (W) of an existing simplex, provided that repetition is avoided. Specifically, if C denotes the centroid of the points of the simplex with W excluded, then a transformed point P is obtained via

$$P = C + \alpha(C - W). \tag{2.6}$$

As is illustrated in Figure 1, depending on the value of the quantity α, four transformations can be distinguished:

Expansion (E): $\alpha > 1$

Reflection (R): $\alpha = 1$

Positive Contraction ($C+$): $0 < \alpha < 1$

Negative Contraction ($C-$): $-1 < \alpha < 0$

Expansion and reflection are attempted first, as these operations do not decrease the search space. If this does yield an improvement, then contraction around the best value is performed (for additional details, see Appendix A). In the present context, improvement is quantified in terms of the magnitude of Φ in Equation (2.5).

The Downhill Simplex requires that the minimization process be initialized with a nondegenerate starting simplex. Although it is possible to find more efficient methods in particular cases, our experience indicates that the initialization originally proposed by Spendley, Hext, and Himsworth (1962) performs quite well in general. According to this approach, the points of the simplex are represented by the $P + 1$ rows of a matrix **INIT** with P columns, one for each free parameter θ:

$$\mathbf{INIT} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ hi & lo & \cdots & lo \\ \cdots & \cdots & \cdots & \cdots \\ lo & lo & \cdots & hi \end{bmatrix} \tag{2.7}$$

where

$$lo = \frac{\sqrt{P+1} - 1}{P\sqrt{2}}$$

and

$$hi = \frac{(P-1) + \sqrt{P+1}}{P\sqrt{2}}$$

2.3.2 Powell's Conjugate Gradient method Like the Downhill Simplex, Powell's Conjugate Gradient method (cf. Acton 1970; Brent 1973) does not require the use of derivatives. Instead, the algorithm finds a minimum by performing unidimensional "line" minimizations along pairwise perpendicular ("conjugate") directions in the search space. The particular implementation used in GEMCAT II is taken directly from Press et al. 1992, pp. 331–339, which performs the line minimization using either the secant method or the golden ratio search, depending on the local circumstances. Because a full description of this algorithm is rather lengthy, the interested reader is referred to the references listed above.

2.3.3 Combining the two approaches As witnessed by its popularity in engineering applications (see, e.g., Bayne and Rubin 1986), the Downhill Simplex is relatively robust with respect to the presence of noise. For this reason, this procedure is performed first, and Powell's method is invoked next to "polish" the solution. The default procedure is to run the Downhill Simplex until the maximum relative change in the free θ values falls below 10^{-7}, at which point Powell's method is invoked to bring the relative change in the quantity Φ in Equation 2.3 below 10^{-9}. Different termination criteria can be selected by the user.

2.3.4 Implementation GEMCAT II consists of a 32-bit, highly optimized 400KB executable that will run on a standard PC under Windows 95/98/NT. The system is based on more than 4,500 lines of code written in Borland's (Inprise) (1997) Delphi Version 3.0, and it fully exploits the object orientation of this language. In addition, the system makes extensive use of Delphi's built-in point-and-click style interface facilities, thereby providing a flexible and user-friendly graphical interface. Figure 2 summarizes the major software components of the system. It can be seen that all data operations are performed by a specialized object (*TStore*), whereas all catastrophe functions are gathered in a single unit, *CatFuncs*, which is presented in Appendix B. Both

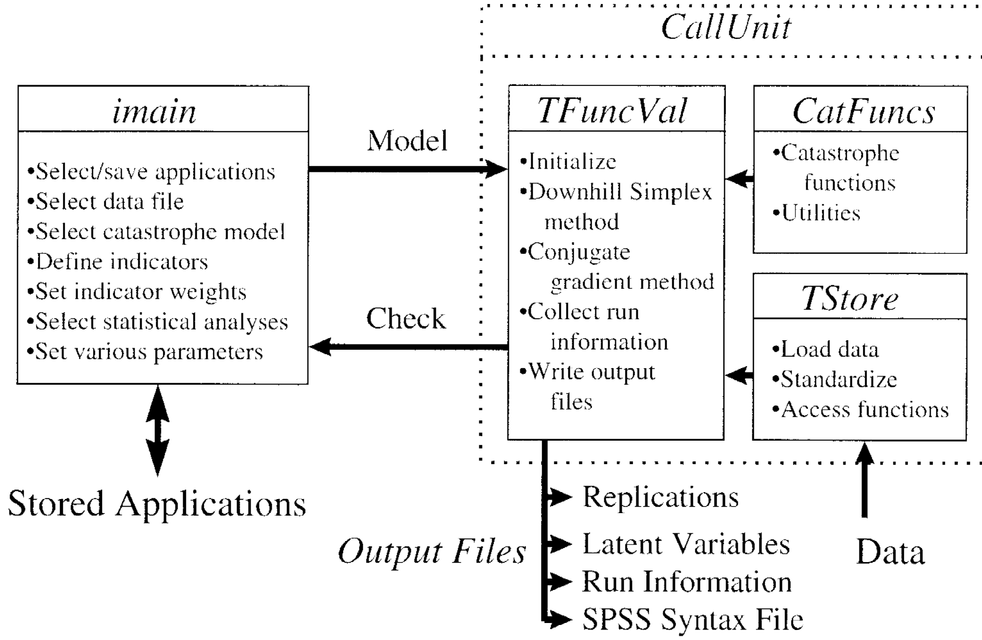

Figure 2
An overview of GEMCAT II's major software components.

types of components are used by *TFuncVal* objects, which perform the actual minimization of Equation (2.5), whereas *CallUnit* performs most of the bookkeeping tasks. All user interaction operations and specification checks are performed by *imain*.

Besides generating a file with the basic results, GEMCAT II produces a file containing the latent variables associated with each case (an example is provided in Section 3.3). Also, the system produces an SPSS syntax file to aid in the processing of the latent variables. Currently, only the cusp, fold, swallowtail, and butterfly catastrophes are implemented, as these occur the most frequently in the literature (see Table 1). Given the system's modular design, however, the remaining three catastrophes as identified by Thom (1975) can be added with relatively little effort, and for implementation details the reader is referred to Appendices A and B.

2.4 Statistical significance of indicator weights

To enhance the utility of the GEMCAT approach it is desirable that specific hypotheses can be tested concerning the free indicator weights θ. Alternatively, users should be able to follow an exploratory approach in which indicators can be selected based on the statistical significance of their weights in one or more latent catastrophe variables. Because there is little reason to assume specific sampling distributions for the θ, GEMCAT II implements hypothesis testing based on resampling techniques (see, e.g., Efron and Tibshirani 1993). In this approach statistical information concerning the indicator weights is obtained by repeatedly recomputing the free θ values based on subsamples of the available sample of (N) observations. Users can select either a jackknife (i.e., "leave one out") approach or a nonparametric bootstrapping approach in which N observations are randomly drawn with replacement from the available sample of observations. The default number of bootstrap replications (B) for the bootstrap is 200. However, any desired number of replications can be specified by the user.

In addition to performance statistics like Φ/N, a *Pseudo-*R^2 value (to be discussed in Section 2.6 below), the number of iterations, and the required CPU time, the system optionally computes the following quantities:

1. The plug-in values $t(\theta_p)$ of each parameter $\theta_p \in FREE$, as determined over the entire sample of available cases.

2. The mean $\theta_p(*)$ of all jackknife or bootstrap values. To avoid dependencies, the θ_p are recomputed from scratch; that is, previously computed values are not used as a starting point.

3. The standard error of the estimates of the $\theta_p(SE_{\theta p})$ as appropriate for the jackknife and bootstrap approach.

4. An estimate (*BIAS*) of the bias of the plug-in values as appropriate for the jackknife and bootstrap methods.

5. An index of the relative bias (%*BIAS*), as

$$\%BIAS = \frac{BIAS}{|\theta_p|} * 100$$

In situations where it is reasonable to assume that the sampling distribution of the indicator weights θ_p is approximately normal, then statistical tests of significance are provided by the quantity z_{θ_p}, where

$$z_{\theta_p} = \frac{t(\theta_p)}{SE_{\theta_p}}$$

which should follow a unit normal distribution under H_0: $\theta_p = 0$.
The following test is specific to the bootstrap: When $1, \ldots, b, \ldots, B$ bootstrap replications are used, an estimate of the Type I error of the hypothesis H_0: $\theta \geq c$ is provided by the achieved significance level (*ASL*):

$$ASL_{boot} = \frac{\#\{\theta_p(b) > c\}}{B}$$

Conversely, the significance of the hypothesis H_0: $\theta \leq c$ is given by

$$ASL'_{boot} = 1 - ASL_{boot}.$$

Note that when the ASL are close to the maximum Type I error level allowed by the user, this approach may require a considerable number of replications to be reliable. By default, the system prints the *ASL* for H_0: $\theta \geq 0$.

2.4.1 The bias-corrected accelerated approach As is shown in Figure 2, the system produces a file ("Replications") with the plug-in values of all θ. Optionally, the jackknife or bootstrap replications can be written to this file as well (and this output can also be used to study the distributions of the $t(\theta_p)$). Thus, although GEMCAT II currently does not implement the "bias-corrected accelerated" (BC_a) approach for percentile estimation, it provides all information needed to compute such statistics (see Efron and Tibshirani 1993, pp. 184–188).

2.5 Competitive model tests

Confirmatory model testing often implicitly entails the hypothesis that a particular favored model should show significantly better performance than a rival or baseline model. In addition, researchers may want to test hypotheses concerning the relative merits of alternative configurations of indicator variables. Competitive tests between two models have to distinguish between nested and non-nested models, depending on whether these models share all free indicator weights.

2.5.1 Non-nested models When two models share all indicator variables (but in a different configuration), they are said to be non-nested. In this case the relative merits of two models can be determined simply by comparing the respective errors for each model as provided by the corresponding e_t^2 terms in Equation (2.5). Each GEMCAT II run causes the signed residual e_t for each case to be saved to a file. Hence, a statistical test for differential fit can be obtained by comparing these residuals after squaring. Given the χ^2-like distribution of the e_t^2, we propose using a nonparametric approach for repeated measures, such as Wilcoxon or Kendall's tests (Marascuilo and McSweeney 1977). Note that bootstrap methods could be used here as well. Given the greater computational effort involved, however, bootstrapping seems worthwhile only when the residuals' estimates are clearly biased.

2.5.2 Comparing nested models When the set of free indicator variables $\mathbf{V}'$ contributing to a partial model M' is a proper subset of the set $\mathbf{V}$ of free indicators of a full model M, then M' is said to be nested within M. Since fitting a catastrophe model relies on minimizing the squared residuals in Equation (2.5), one can use the likelihood ratio approach described by Bates and Watts (1988, pp. 103–104) to test whether inclusion of the set of indicators $\mathbf{V} - \mathbf{V}'$ significantly improves the fit of M'. Assume that the residual sum of squares *SS(RES)*, the degrees of freedom ν, and the number of free indicators P in the respective models are subscripted by M and M'. If N represents the total number of cases, then under the null hypothesis the expression:

$$\textit{Pseudo-F} = \left(\frac{SS(RES)_{M'} - SS(RES)_M}{SS(RES)_M}\right)\left(\frac{N - P_M}{P_M - P_{M'}}\right) \tag{2.8}$$

is approximately distributed as an F distribution with $\nu_1 = P_M - P_{M'}$ and $\nu_2 = N - P_M$.

Equation (2.8) is said to yield a *Pseudo-F* value since the expression will not have an exact F distribution because of the inherent nonlinearity of catastrophe models. According to Bates and Watts (1988), nonlinearity has relatively small effects when M and M' have a similar fit (i.e., *Pseudo-F* ≤ 1). Although the distortion may be stronger when the fit of M and M' differs substantially (i.e., *Pseudo-F* $\gg 1$), it is unlikely that H_0 will be rejected erroneously. We suggest, however, that when the *Pseudo-F* approaches statistical significance, users should inspect the distribution of the residuals as printed for marked deviations of normality and bias. Also, when the estimates of the *SS(RES)* are biased, one might consider using bootstrap estimates thereof.

2.6 A Pseudo-R^2 index of fit

Using the same reasoning as above, the overall fit of a catastrophe M can be expressed by the squared multiple correlation:

$$\textit{Pseudo-}R_V^2 = 1 - \frac{SS_{M(Res)}}{SS_V} \tag{2.9}$$

where $V \in \mathbf{DEP}$ is a dependent variable in this catastrophe (e.g., Z in the case of the cusp), and $SS_{M(Res)}$ and SS_V represent the residual and total sums of squares of variable V, respectively. The statistical significance of *Pseudo-*R_V^2 can be tested by computing the *Pseudo-F* value:

$$\textit{Psuedo-}F_V = \left(\frac{R_V^2}{1 - R_V^2}\right)\left(\frac{N - P - 1}{P}\right) \tag{2.10}$$

where P denotes the number of indicator variables in the model. Under the null hypothesis, this quantity approximately follows an F distribution with $\nu_1 = P$ and $\nu_2 = N - P_M - 1$.

Experience indicates that some important caveats apply. For instance, we have found that the fit of catastrophe models can be so poor that $SS_{M(RES)} > SS_V$ in Equation (2.9), thereby producing a negative R^2 value. Like Cobb (1998), we interpret such a finding to indicate that the catastrophe model involved should be rejected out of hand. Also, analogous to our comments regarding the validity of Equation (2.9), we propose that the distribution of the residuals should be inspected for significant deviations of normality before accepting the values produced by Equation (2.10). Nevertheless, in cases where the model provides at least a moderate fit to the data, the *Pseudo-*R^2 provides a criterion for comparing different models across data sets of varying sizes and with differing indicator variability.

3 Some Examples

The following sections report three studies done with GEMCAT II. Sections 3.1 and 3.2 report the results of two computer simulations. First, to determine the system's behavior in other contexts, a computer simulation study was performed that added one of seven levels of noise to the indicator variables while varying the size of the sample being simulated. A second simulation study addresses the issue of "false positives." In particular, we focus on the likelihood of adopting a cusp catastrophe when the data are actually generated by a completely linear model. Section 3.3 presents an application of catastrophe modeling to the dynamics of organizational product adoption. In addition to the economics contents of this case study, Section 3.3 also provides tutorial information concerning GEMCAT II's model testing facilities.

3.1 A computer simulation of the effect of indicator reliability

To study GEMCAT II's performance in a variety of contexts, we designed a computer simulation study that varied both the size of the sample of cases being analyzed and the reliability of the indicator variables. Because the cusp is probably the most widely used model, we study only this particular catastrophe. Also, catastrophe models are applied increasingly often in areas such as psychology and organizational behavior (for an overview see, e.g., Guastello 1995) that typically use relatively small samples of observations. The experimental error is often considerable in such applications, and the quality of measurement can be poor. We decided therefore to include a wide range of reliability conditions and sample sizes.

3.1.1 Indicator variables of X and Y As in the earlier work by Oliva et al. (1987), the asymmetry (X) and bifurcation variables (Y) consisted of weighted combinations of indicator variables. In particular, X consists of three indicator variables, whereas Y is based on four indicator variables. The $x_I(y_J)$ are presumably indicators of the same latent variable $X(Y)$, and hence we induced moderate linear correlations among these indicators. The following procedure was followed for the x_I:

$$\begin{aligned} x_1 &= N(0, 0.5) \\ x_2 &= N(0, 1) - x_1 \\ x_3 &= N(0, 0.5) - 0.5x_1 \end{aligned}$$

whereas the y_J were created as

$$\begin{aligned} y_1 &= N(0, 1) \\ y_2 &= N(0, 1) - y_1 \\ y_3 &= N(0, 1) - y_1 \\ y_4 &= N(0, 1) - y_1 \end{aligned}$$

In the preceding, $N(\mu, \sigma)$ denotes a random drawing from a normal distribution with mean μ and standard deviation σ as generated by Box, Muller, and Marsaglia's "polar" method (see Knuth 1969, p. 105).

Next, the x_I were given weights of 1, -3, and -2, respectively, to obtain the values of X, whereas the y_J were given weights of 0.5, -1, -1, and 2 in computing Y. In other words,

$$\begin{aligned} X &= x_1 - 3x_2 - 2x_3 \\ Y &= 0.5y_1 - y_2 - y_3 + 2y_4 \end{aligned}$$

3.1.2 Solving for Z Given the values of X and Y, explicit solutions for Z in Equation (2.5) were derived via the Maple symbolic software in Mathcad V5.0 (MathSoft 1994). As is shown below, Z has three roots as represented by the multivalued function $Cusp_Z(X, Y)$:

$$Cusp_Z(X, Y) = \begin{bmatrix} D + \frac{Y}{3D} \\ E + i\sqrt{3F} \\ E - i\sqrt{3F} \end{bmatrix} \tag{3.1}$$

where

$$\begin{aligned} i &= \sqrt{-1} \\ D &= \left[\frac{X}{2} + \frac{\sqrt{3}}{18}\sqrt{-4Y^3 + 27X^2}\right]^{1/3} \\ E &= \frac{-D}{2} - \frac{Y}{6D} \\ F &= \frac{D}{2} - \frac{Y}{6D} \end{aligned}$$

$Cusp_Z(X, Y)$ returns either three or one real (that is, nonimaginary) roots of Z, depending on whether X and Y fall inside or outside the cusp's bifurcation area. When three real roots are obtained, the second largest (i.e., middle) value is *not* a viable solution, and it is therefore ignored. In this case, either the smallest or the largest root was selected with equal probability. All computations involving Equation (3.1) are based on a Delphi implementation of complex arithmetic.

No attempts were made to decompose Z into more elementary indicator variables. In other words, the variable Z has only one indicator variable, and $Z = z_1$. The correlations between the indicator variables resulting from the above procedures are shown in Table 2. It can be seen that the correlations within each of the sets of X and Y indicators tend to be greater than those between the X and Y sets. Also, z_1 is related to the x_I but not to the y_J. Thus, Z depends on X but not on Y.

3.1.3 Independent variables To study the effect of sample size on the estimates of the indicator weights, samples of either 25, 50, 100, 250, or 500 simulated cases were constructed according to the procedures described above. In addition, seven reliability conditions were created by adding normally distributed random noise to the various indicator variables. Specifically, the standard deviation of the noise distributions was set to either 0, 5, 10, 20, 30, 50, or 70% of that of the indicator distribution. To keep the number of experimental conditions manageable, the reliability of the x_I and y_J indicators was kept identical within each of the seven reliability conditions (i.e., an equal percentage of noise was added to each of the X and the Y indicators). In addition, the reliability of z_1 was varied independently at the same seven levels, thereby creating a total of 49 reliability combinations. When crossed with the five sample sizes, a total of 245 "between" experimental conditions result. For reasons explained earlier, the weight of Y indicator variable was set to 1 throughout.

3.1.4 Preliminaries Each condition was replicated 250 times using independently constructed data sets. Thus, the following results are based on 61,250 data sets with well over 11 million simulated individual cases.

Table 2
Pearson correlations between the X, Y, and Z indicator variables

	x_1	x_2	x_3	y_1	y_2	y_3	y_4
x_2	−0.47						
x_3	−0.47	0.20					
y_1	0.00	0.00	0.00				
y_2	0.00	0.00	0.00	−0.70			
y_3	0.00	0.00	0.00	−0.70	0.47		
y_4	0.00	0.00	0.00	0.70	−0.47	−0.47	
z_1	0.38	−0.58	−0.29	0.00	0.00	0.00	0.00

Table 3
Variance explained by ANOVA model and effect sizes of factors that explain at least 5% of the variance

Indicator weight of	Proportion of variance explained by ANOVA model	Reliability of X and Y indicators	Sample size	Reliability of X and Y by sample size interactions
x_1	0.305	0.090	0.143	0.062
x_2	0.207	0.076	0.079	0.052
x_3	0.208	0.063	0.087	0.043
y_1	0.025	0.002	0.007	0.003
y_2	0.027	0.003	0.008	0.006
y_3	0.020	0.003	0.007	0.004
y_4	0.006	0.000	0.000	0.000

GEMCAT II's estimation algorithm was applied to each of the data sets, and the seven X and Y indicator weights, as well as the *Pseudo-*R^2 value, the number of iterations of the minimization algorithm, and the CPU time required to analyze each data set were all saved to a file. Throughout, the Downhill Simplex algorithm was initialized using Equation (2.7), and the default convergence criterion was used (i.e., the Downhill Simplex was run to achieve a relative error below 10^{-7} at which point Powell's method was invoked to bring the relative change in the quantity Φ in Equation (2.5) below 10^{-9}). The simulation program essentially comprises the right side of Figure 2, augmented by data-generating procedures added to the *TStore* component. The simulations required a total of about 11 hours of CPU time on a standard 300 MHz personal computer under Microsoft's Windows 95 operating system.

3.1.5 Results To determine the effects of the experimental manipulations on the indicator weights, each weight was subjected to a 7 (Reliability of the X and Y Indicators) ×7 (Reliability of the Z Indicator) ×5 (Sample Size) analysis of variance (ANOVA). Given the large number of observations it is not surprising that nearly all effects are statistically significant. We decided therefore to report only effects that account for at least 5% of the variance in any of the indicator weights as quantified by their associated η^2 values (i.e., $\eta^2 > .05$). The first column of Table 3 indicates that the ANOVA model explains a significant portion of the variation of the X indicator weights (all $\eta^2 > .20$) but not of the Y weights (all $\eta^2 < .027$). Further, the second, third, and fourth columns show that the X indicator weights, but not the Y weights, are noticeably affected by reliability of the X and Y indicators as well as by the size of the simulated data set. No other ANOVA effects explained more than 5% of the variance for any of the indicator weights, i.e. the reliability of the Z indicator had little or no effect on the weights of the X and Y indicators.

Table 3 further indicates that the x_1 weights are the most affected overall. In particular, a main effect of the reliability of X and Y ($\eta^2 = 0.090$), a main effect of sample size ($\eta^2 = 0.143$), and an interaction involving these two independent variables ($\eta^2 = 0.062$) were obtained. Figure 3 shows that this interaction is due to the large increase in x_1's weights as the X and Y indicators decrease in reliability. Note that whereas this increase is quite dramatic for the smallest sample size ($N = 25$), the effect virtually disappears for moderate sample sizes ($N \geq 100$) and for reasonable reliability levels ($\leq$ 30% error). Under these conditions, GEMCAT II's estimates of all eight indicator weights are quite satisfactory, as Figure 4 reveals that the weights' estimates cluster closely around their theoretical values. (Note: Although the weight of z_1 was fixed at 1.0, it is included for purpose of completeness).

3.1.6 Pseudo-R^2 Analysis of variance of the *Pseudo-*R^2 values revealed powerful effects of sample size ($\eta^2 = .763$) and the reliability of the Z indicator ($\eta^2 = .712$). In addition, a sample size by Z reliability interaction effect was observed ($\eta^2 = .681$). As is shown in Figure 5, this interaction is the result of a rapid decline in R^2 when the indicator error increases for small sample sizes, whereas R^2 is little affected for large

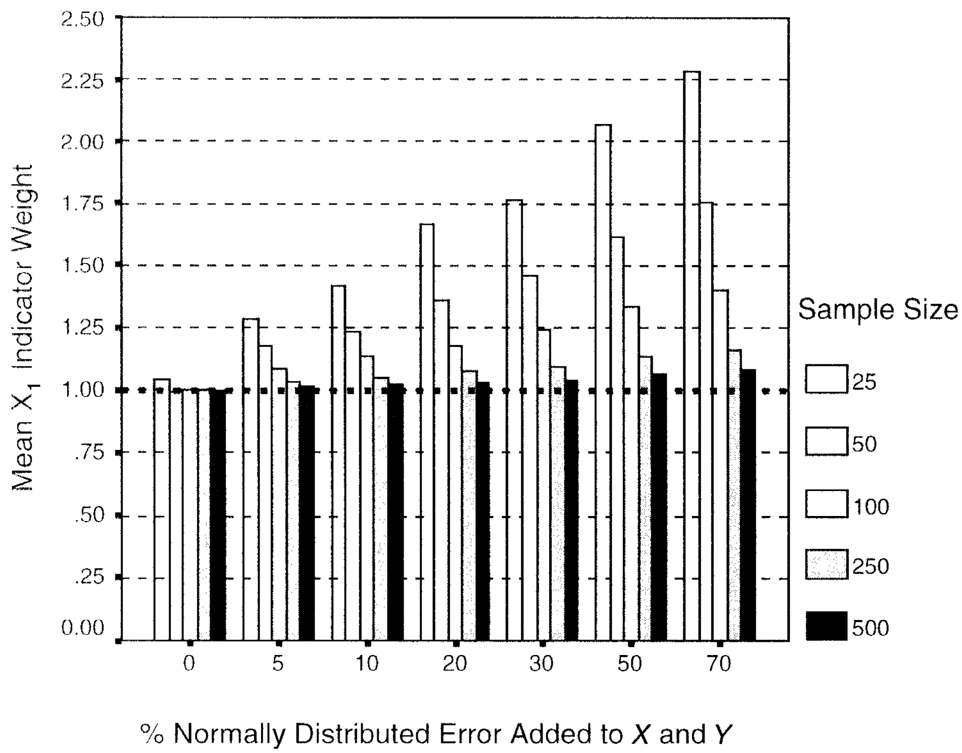

Figure 3
Average weight of x_1 as a function of the error added to X and Y and the sample size (8,750 observations per bar).

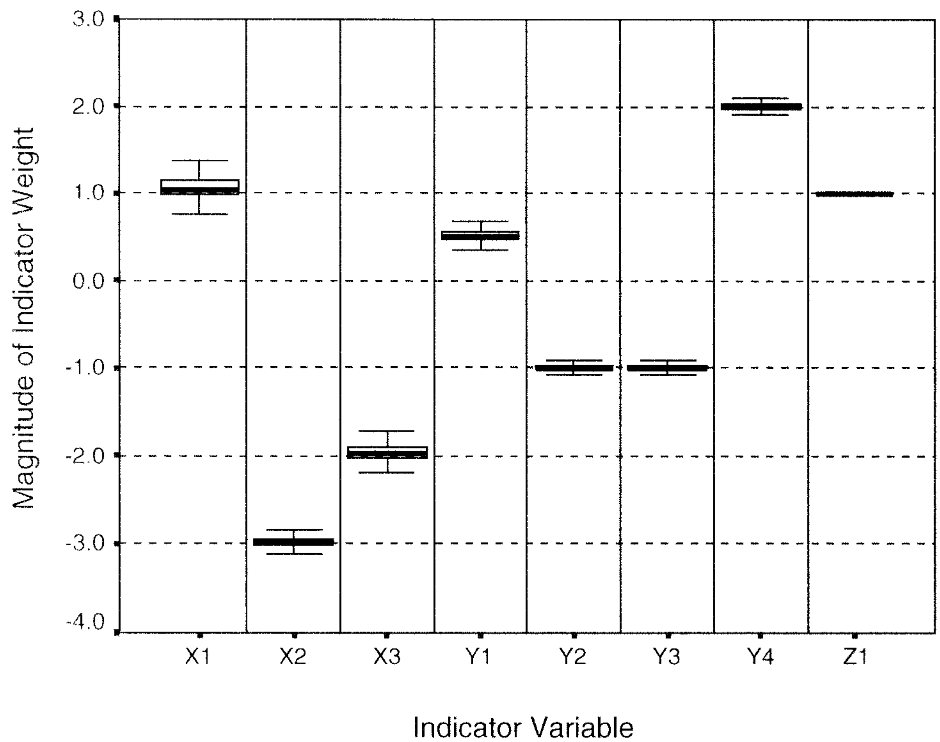

Figure 4
Boxplots of the indicator weights for samples of at least 50, with at most 20% error added to the X, Y, and Z indicators (28,000 observations per box). Boxplots show the median and quartiles; whiskers show the extreme values, except for outliers.

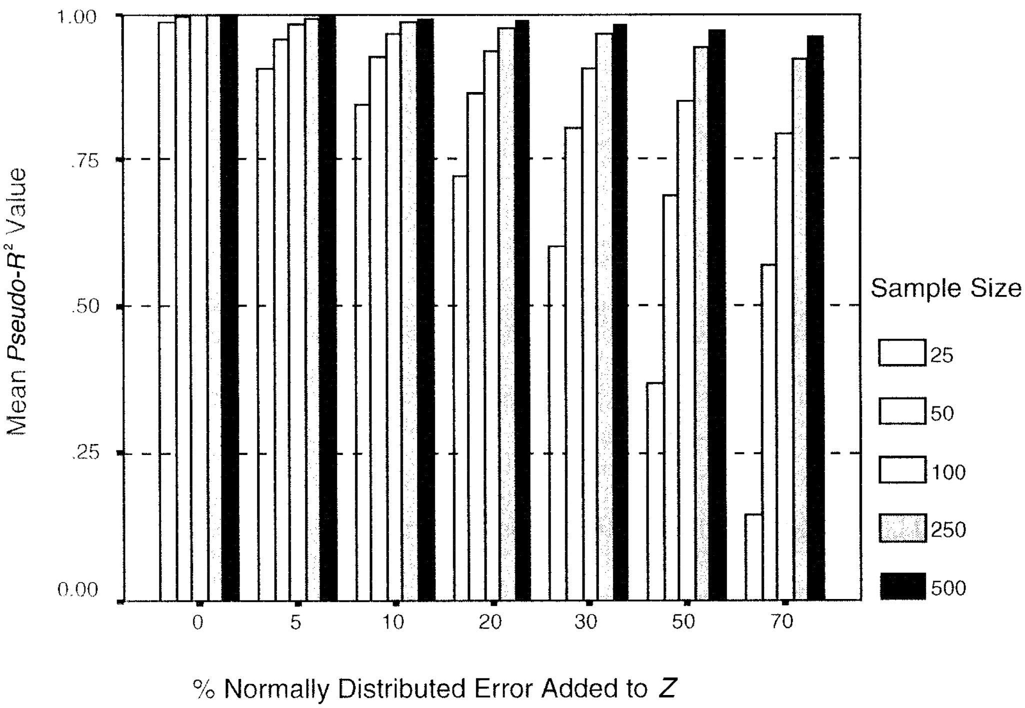

Figure 5
Average *Pseudo-*R^2 as a function of the reliability of the Z indicator and the sample size (8,750 observations per bar).

sample sizes. For instance, for quite reasonable error levels ($\leq$ 30%) and sample sizes ($N \geq 50$) the average R^2 always exceeds 0.75. This pattern of findings implies that large errors in Z, in conjunction with a small sample size, prevent GEMCAT II from recognizing the data as being cusp shaped, sometimes resulting in negative *Pseudo-*R^2 values. This conclusion is supported by the data summarized in Figure 6, which shows boxplots for two sample sizes, $N = 25$ and $N = 250$. It can be seen that negative R^2 values start to appear for $N = 25$ when 30% or more error is added to Z. No negative values are found, however, for the larger sample size of 250.

We further note that the main effect of the X and Y reliability ($\eta^2 = .065$) and the interaction of this factor with sample size ($\eta^2 = .089$) were minor. Thus, given the powerful effects involving Z, the *Pseudo-*R^2 values appear to be affected less by measurement errors in the X and Y variables than by errors in the dependent variable Z. Although this conclusion is clearly limited by the present simulation design, we note that similar results were obtained in Lange (1998), in which a composite Z with two indicator variables was used.

3.1.7 Performance information The CPU time required by GEMCAT II to estimate the indicator weights (i.e., excluding the time needed to generate the data sets) increases nearly linearly with the number of cases ($\eta^2 = 0.234$). On our particular computer system, the average CPU time increased from 0.06, to 0.12, 0.27, 0.75, and 1.50 seconds for 25, 50, 100, 250, and 500 cases, respectively. Interestingly, the number of iterations needed for convergence is affected little by the number of cases ($\eta^2 = 0.001$). However, an interaction involving the indicators' reliabilities was found ($\eta^2 = 0.156$) because the average number of iterations required to achieve convergence was largest when no error was added to X, Y, or Z(M = 7,907, median = 2,977) and smallest in the presence of maximum noise in X, Y, and Z(M = 1,586, median = 1,569).

3.2 A simulation study of false positives

Although the first study established that GEMCAT II is able to recover cusp models under a wide range of reasonable conditions, Alexander et al. (1992) point out that it is equally desirable to avoid false positives.

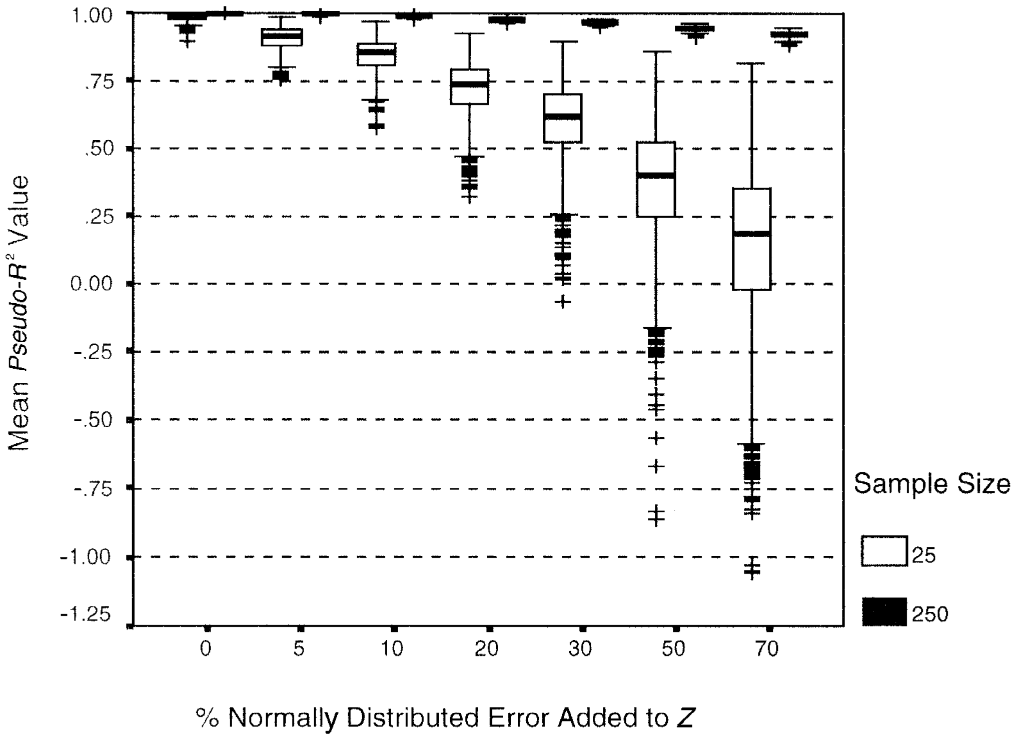

Figure 6
Boxplots of *Pseudo-R*^{2} as a function of the reliability of the Z indicator for sample sizes of 25 and 250 (1,750 observations per boxplot).

That is, noncusp data should not be identified as cusplike. Unfortunately, computer simulations by Alexander et al. showed that regression approaches to catastrophe modeling (Guastello 1982, 1992) may cause some linear models to be identified erroneously as conforming to a cusp catastrophe. For this reason, we decided to approximate the experimental condition that in Alexander et al.'s research produced the most false positives. For greater realism, a small sample size was used ($N = 50$), thereby allowing GEMCAT II to capitalize on accidental nonlinear properties of the data.

3.2.1 Design and hypotheses Following the procedures described on pages 370 and 371 of Alexander et al.'s (1992) article, four normally distributed random variables (v_1, v_2, z_1, and z_2) were generated with $r_{z2,v1} = r_{z2,v2} = 0.40$, $Z = z_2 - z_1$, and all other correlations equal to 0. Since this specifies a completely linear model, there are no compelling reasons to designate either v_1 or v_2 as playing the role of a cusp's Y indicator. Given that the Z^3 term in Equation (2.5) approximates a (curved) line over small ranges of Z, however, we expect that the variables v_1 and v_2 should predict Z^3 in a linear fashion. To test these predictions v_1 and v_2 were defined so as to contribute simultaneously to X and Y. That is:

$$X = \theta_{x1}v_1 + \theta_{x2}v_2,$$
$$Y = \theta_{y1}v_1 + \theta_{y2}v_2.$$

If this hypothesis is correct, v_1 and v_2 should assume the role of X (asymmetry) variables rather than Y (bifurcation) variables (i.e., $\theta_{xi} > 0$ and $\theta_{yj} \approx 0$). In addition, to the extent that the absence of Y-like indicators causes GEMCAT II to reject the data as cusplike, the *Pseudo-R*^{2} values for the linear data should be negative.

3.2.2 Results The design described above was repeated 1,000 times using independently constructed data sets of $N = 50$ simulated cases. Consistent with our hypotheses, Figure 7 shows that the average weights of v_1 and

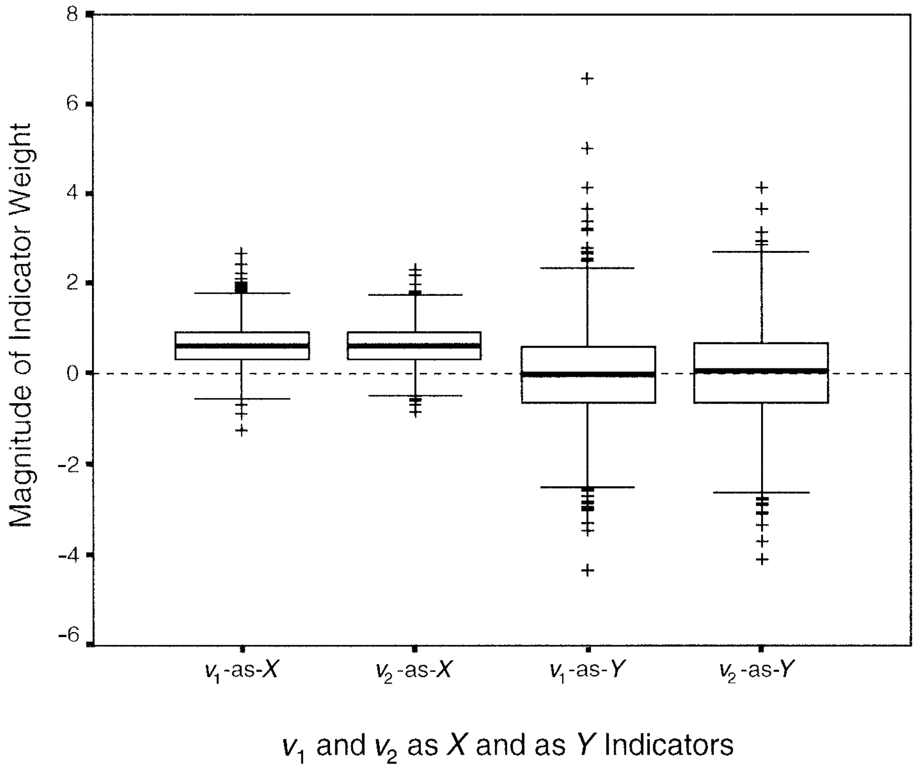

Figure 7
Boxplots of the indicator weights of v_1 and v_2 as indicators of X and as indicators of Y (each boxplot is based on 1,000 samples of 50 simulated cases).

v_2 are positive when used as X variables ($M = .613$ and .594, respectively) but near zero when v_1 and v_2 are used as Y variables ($M = -.013$ and .004, respectively). The relatively large variation in v_1 and v_2's Y weights ($SD = 1.070$ and 1.012, respectively) suggests that the sample size of 50 was indeed sufficiently small to allow the algorithm to exploit accidental Y-like properties of v_1 and v_2. Yet it is possible to reliably spot false positives, since Figure 8 shows that all 1,000 *Pseudo-*R^2 values were negative. Thus, the finding of negative R^2 values can be used as evidence against existence of a cusplike structure, and it appears therefore that GEMCAT II is not susceptible to the shortcomings of the linear regression approach as identified by Alexander et al. (1992).

3.3 A cusp model of organizational product adoption

To make GEMCAT II's features more salient to economists, we present a small application using data on organizational adoptions of competing high-technology product standards when network externalities are present. We caution that the emphasis is on the demonstration rather than the justification of the catastrophe model or of the data presented. We believe, however, that the model is reasonable and that the data are appropriate in the context of demonstrating the program's use. Therefore, only a brief literature overview and statement of the problem is presented, and for a more detailed discussion the interested reader is referred to Oliva 1994.

3.3.1 Background Economists (e.g., Farrell and Saloner 1985, 1986; Katz and Shapiro 1985) have argued that when network externalities are present firms will adopt new innovations, standards, or high-technology products in bandwagons. When a bandwagon occurs, a sudden discontinuous shift takes place in the market. The discontinuity in the diffusion curve is often problematic for most standard modeling approaches but is suitable for nonlinear approaches, particularly when a process-oriented model is used. The literature (Farrell and Saloner 1985) points to three variables as the most relevant: organizational adoptions, firm cost/benefits,

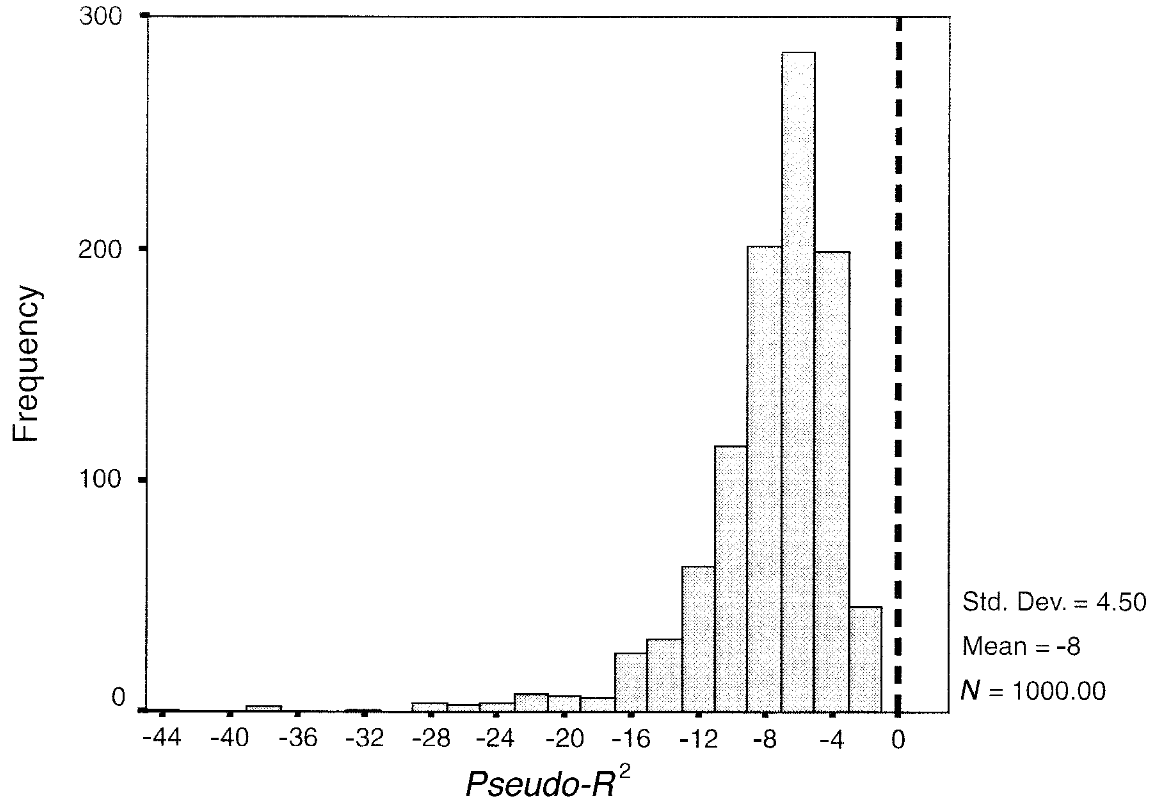

Figure 8
Frequency distribution of the *Pseudo-R^2* values obtained for linear data sets (based on 1,000 samples of 50 simulated cases).

and the level of network externalities desired in the market. Based on these three variables, the market dynamic may be conceptualized in terms of the cusp catastrophe model shown in Figure 9 (Oliva 1994). In this figure, organizational adoption is the dependent variable (Z), whereas cost/firm benefits and network externality levels respectively form the control (X) and the splitting factor (Y), respectively. We presume that such models are relatively self-explanatory to the reader.

3.3.2 Data The data were generously provided by Techtel, Inc., which has a panel of 2,000 firms from 14 industries that provided key information on software and hardware purchases over the previous 15 years. Confidentiality limits exactly how much we can disclose regarding the data, the companies, and the products. The present example is based on a subsample of the data for firms that adopted a spreadsheet program (e.g., Lotus 123, Excel, Quattro Pro) for one of two competing standards (e.g., DOS versus Windows or MAC versus Windows). By holding the application (spreadsheet) constant we help control for preferences in brand and focus on competing operating system standards.

The operationalizations of the dependent and independent variables are summarized in Table 4. We started with the best measures available in the database to generate proxy measures for X, Y, and Z. Ultimately, not all measures for each variable were used in the preferred model. Instead, some experimentation revealed that the data are adequately captured by a cusp model that incorporates just the three starred variables in Table 4, including a constant term for Z. In particular, the normal variable X is defined in terms of “relative opinion” (x_3 with weight θ_{x3}), and the splitting variable Y uses the “installed base” indicator (y_2 with weight θ_{y2}), whereas the dependent variable Z is defined in terms of “relative adoption” (z_2 with weight θ_{z2}) and a constant offset (θ_{Z0}). Throughout the remainder of this section we refer to this cusp as $Cusp_1$. The following section describes the GEMCAT II output associated with this model.

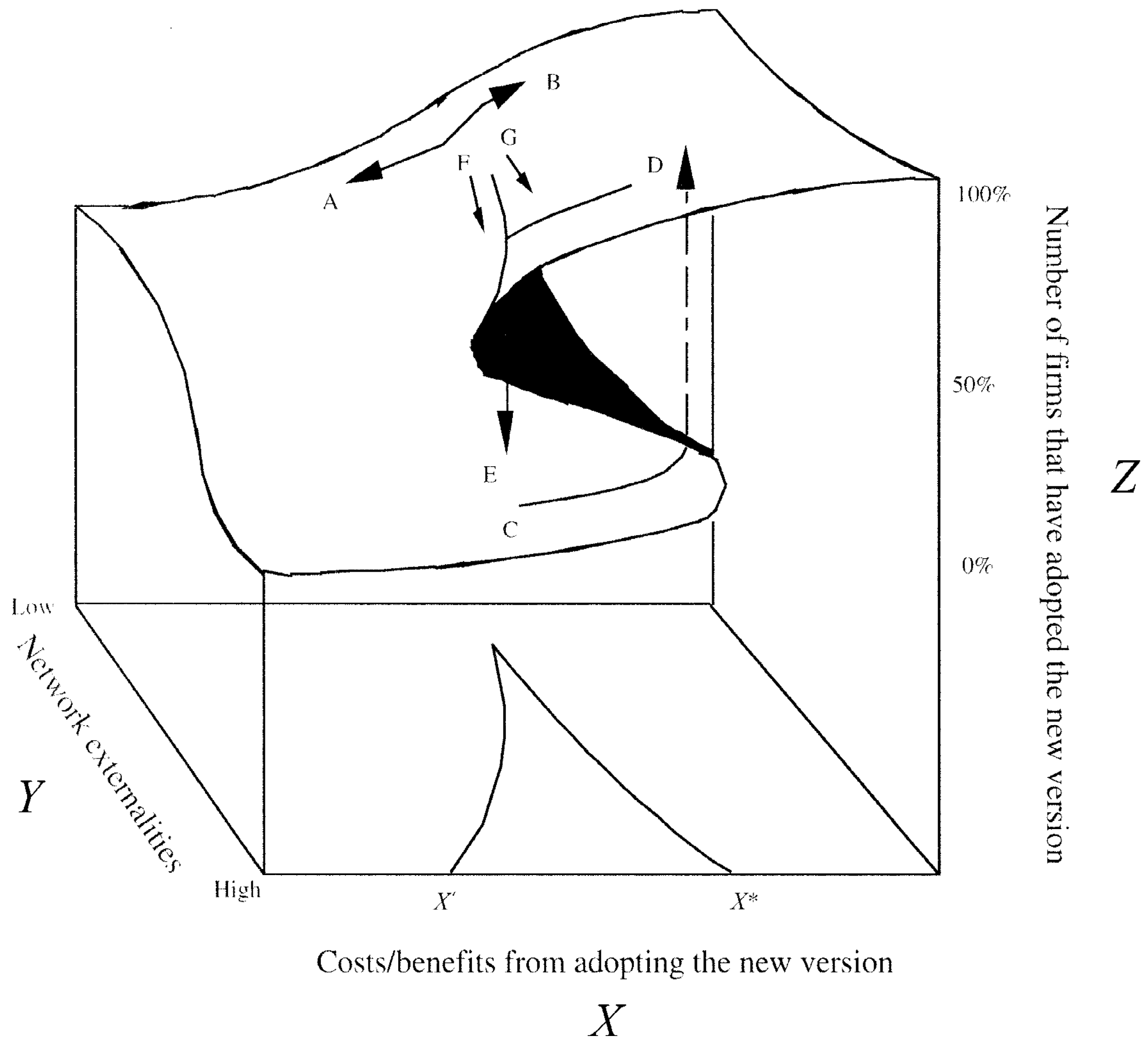

Figure 9
A cusp catastrophe model of product adoption.

Table 4
Indicators used in organizational product adoption example

Dependent variable indicators (Z)	
$*z_0 =$	Z offset
$z_1 =$	Relative adoption (% who bought spreadsheet for OS#2 − % who bought spreadsheet for OS#1)
$*z2 =$	Relative trial (% who tried spreadsheet for OS#2 − % who tried spreadsheet for OS#1).
Cusp asymmetry factor indicators (X)	
$x_1 =$	Relative awareness (% aware of spreadsheet for OS#2 − % aware of spreadsheet for OS#1)
$x_2 =$	Relative consideration (% considering spreadsheet for OS#2 − % considering spreadsheet for OS#1)
$*x3 =$	Relative opinion (% positive opinion of spreadsheet for OS#2 − % positive opinion of spreadsheet for OS#1)
Cusp bifurcation factor indicators (Y)	
$y_1 =$	Complementary goods (average % of firms that adopted complementary goods for either operating system during current quarter)
$*y2 =$	Installed base (average % of firms that adopted either operating system in the previous quarter)

Note: The indicator variables in our preferred cusp model ($Cusp_1$) are based on the marked entries.

3.3.3 GEMCAT II output Table 5 lists a major section of the GEMCAT II output resulting from bootstrapping $Cusp_1$ a total of 500 times. This listing contains most of the statistics that were described earlier in Section 3. That is, Column 1 shows the plug-in values $t(\theta)$ of the free indicator weights as listed under the row heading "Estimated Quantities," and the average bootstrap values $\theta(*)$ are listed in Column 3, whereas bias information is shown in Columns 8 and 9. It can be seen that the weight of the sole X indicator (i.e., x_3 = "relative opinion") as computed over the entire sample is $t(\theta_{x3}) = -0.0231$, whereas its average bootstrap value is $\theta_{x3}(*) = -0.0217$. The difference $\theta_{x3}(*) - t(\theta_{x3}) = 0.0014$ presented in Column 8 reflects a slightly positive bias in the estimate of θ_{x3}. Column 9 shows, however, that this bias is small (6.4%) relative to $\theta(*)$. In other words, the plug-in value of θ_{x3} provides an acceptable estimate of the corresponding population parameter. Column 9 reveals that the relative bias in the estimates of θ_{y2} (−4.8%) and θ_{z0} (−2.2%) is acceptable as well.

Using the $t(\theta)$ or $\theta(*)$ and their estimated standard error, that is, SE_θ in Column 7, yields the z-values in Columns 2 and 4, respectively, thereby providing a test of the null hypothesis H_0: $\theta = 0$ when a normal sampling distribution is assumed. By contrast, a nonparametric test is provided by the *ASL* listed in Column 10. As is often the case, the conclusions to be drawn from Columns 2, 4, and 10 are identical. That is, the two different approaches agree that θ_{x3} and θ_{z0} are nonpositive, whereas θ_{y2} is nonnegative. (*Note*: In case of disagreements, the bootstrap test is to be preferred over the parametric approach.) Additional information is provided by the minimum and maximum bootstrap values shown in Columns 5 and 6, as they reveal that none of the 500 replications θ_{y2} was negative and that none of the θ_{z0} was positive.

Note that the "Performance Statistics" section provides information concerning the plug-in and average values of the "Minimized Residual" Φ/N (i.e., 2.62×10^{-4}), the plug-in and average numbers of iterations needed to achieve convergence, and the plug-in and average time in seconds needed for each replication. (*Note*: The time information may be inaccurate over small intervals.) Finally, note that *Pseudo-*R^2 as defined by Equation (2.9) (0.996), the *Pseudo-F* value as defined by Equation (2.10), and *SS(RES)* used in Equation (2.8) are listed immediately below the main portion of Table 5. The excellent fit of the $Cusp_1$, as reflected by the R^2 and Φ/N values, together with the statistical significance of the various indicator weights, strongly supports the viability of $Cusp_1$ as embodied in the equation

$$(z_2 - 0.489)^3 + 0.023x_3 - 0.195y_2(z_2 - 0.489) \tag{3.2}$$

3.3.4 Linear versus fold versus cusp To determine whether the data are indeed best described by Equation (3.2), a linear model and a fold catastrophe were fitted as well. These last two models use the same indicator variables as $Cusp_1$, including a constant term. The average squared residuals for the linear model, the fold, and $Cusp_1$ were $3.274 * 10^{-3}$, $5.559 * 10^{-3}$, and $2.617 * 10^{-4}$, respectively. The corresponding R^2 and *Pseudo-*R^2 values are 0.954($F = 323.902$, $\nu_1 = 2$, $\nu_2 = 31$, $p < .0001$), 0.924(*Pseudo-F* $= 189.175$, $\nu_1 = 2$, $\nu_2 = 31$, $p < .0001$), and 0.996(*Pseudo-F* $= 4314.109$, $\nu_1 = 2$, $\nu_2 = 31$, $p < .0001$). The results of the nonparametric Kendall's W test indicate that the squared residuals differ significantly ($\chi^2 = 51.353$, $\nu = 2$, $p < 0.0001$, $W = 0.755$) among the three models. In addition, Table 6 shows that each pairwise difference among the three models is statistically significant as well. In particular, the cusp performs better than the linear model, whereas the linear model outperforms the fold catastrophe.

3.3.5 Tests of nested catastrophe models Given the near-perfect fit of $Cusp_1$(*Pseudo-*$R^2 = 0.996$), it seems unlikely that the fit of this model could be improved significantly by including any of the remaining independent variable indicators. For illustrative purposes, however, three additional models were defined. $Cusp_2$ adds the X indicators x_1 and x_2 to $Cusp_1$, $Cusp_3$ adds the Y indicator y_1, and $Cusp_4$ adds all three variables x_1, x_2, and y_1 simultaneously. In addition, a more parsimonious model, $Cusp_5$, was created by deleting the constant θ_{z0} from Z. In other words, $Cusp_1$ is a submodel of $Cusp_2$, $Cusp_3$, and $Cusp_4$, whereas $Cusp_5$ is a submodel of $Cusp_1$. To highlight these differences, the row entries of Table 7 describe the various models in terms of their free indicator weights relative to those included in $Cusp_1$.

Table 5
A portion of the GEMCAT II output for estimating $Cusp_1$ and its indicator weights as based on 500 bootstrap replications

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
Var	Description	Plugin	Z-Plug	Mean	Z-Mean	Min	Max	SE	Bias	RelBias%	ASL
	--- Estimated Quantities ---										
X- 1	X3	-0.0231	-3.8836	-0.0217	-3.6485	-0.0382	+1.01E-003	+5.95E-003	+1.40E-003	6.4432	+4.00E-003
Y- 1	Y2	0.1950	7.2803	0.1861	6.9466	0.0356	0.2423	0.0268	-8.94E-003	-4.8034	1.0000
Z- 1	Constant	-0.4890	-14.6742	-0.5002	-15.0082	-0.7437	-0.4402	0.0333	-0.0111	-2.2255	-0.00E+000
	--- Quantities Fixed by User ---										
Z- 1	Z2	1.0000		1.0000		1.0000	1.0000	0	0	0	0
	--- Performance Statistics ---										
Minimized Residuals		+2.62E-004	3.5002	+2.17E-004	2.8965	+8.50E-006	+4.97E-004	+7.48E-005	-4.51E-005	-20.8445	1.0000
Number of Iterations		495.0000	7.1986	504.1760	7.3321	335.0000	747.0000	68.7631	9.1760	1.8200	1.0000
Execution time in Seconds		0.0410	4.8775	0.0372	4.4302	0.0140	0.0690	+8.41E-003	-3.76E-003	-10.0967	1.0000

SS(RES) = 8.63806784384819E-0003
R2 = 0.9964
DF1 = 3
DF2 = 30
F = 3062.4988

Table 6
Pairwise comparison of individual squared residuals for the linear model, the fold catastrophe, and the cusp catastrophe. Unit normal deviates (z) derived by Wilcoxon's signed ranks test.

Fold	2.573	
Cusp	−4.761	−5.086
	Linear	Fold

Note: A negative value indicates that the row model performs better than the column model, whereas a positive value indicates that the row model performs worse than the column model.

Table 7
Pseudo-R^2 fit of four nested models and *Pseudo-F* test of fit relative to the preferred cusp catastrophe $Cusp_1$

Model	*Pseudo-R*^{2}	Fit relative to $Cusp_1$ *SS(RES)*	v_1	v_2	*Pseudo-F*	p
$Cusp_2 = Cusp_1 + \{\theta_{x_1}, \theta_{x_2}\}$	0.997	0.00679	2	28	0.026	0.974
$Cusp_3 = Cusp_1 + \{\theta_{y_1}\}$	0.997	0.00828	1	29	0.010	0.920
$Cusp_4 = Cusp_1 + \{\theta_{x_1}, \theta_{x_2}, \theta_{y_1}\}$	0.997	0.00661	3	27	0.018	0.997
$Cusp_5 = Cusp_1 - \{\theta_z\}^a$	0.927	0.17708	1	31	5.222	0.029
$Cusp_1$	0.996	0.00864				

[a]Note that $Cusp_5$ is a proper submodel of $Cusp_1$, whereas $Cusp_1$ is nested within $Cusp_2$, $Cusp_3$, and $Cusp_4$.

The fit of $Cusp_2$ through $Cusp_5$ relative to that of $Cusp_1$ was tested using Equation (2.8). The results summarized in Table 7 indicate that the use of additional indicator variables for X or Y as in $Cusp_2$, $Cusp_3$, or $Cusp_4$ does not lead to a statistically significant improvement in fit (all *Pseudo-F* < 1), whereas omitting the constant θ_{z0} as in $Cusp_5$ leads to a significant decrease in fit ($p < .03$). Hence, the inclusion of the offset θ_{z0} is justified, and $Cusp_1$ as summarized by Equation (3.2) is sufficiently complex to capture the most important aspects of our data set. (*Note*: We also attempted adding y_1 as a second dependent variable indicator for Y. Doing so, however, yielded a degenerate solution.)

3.3.6 Discussion The preceding results are consistent with economic theory. For instance, the proxy measures resulting from the GEMCAT II analysis are relative opinion, installed base, and trial. Given what we know about high-technology products, this makes sense. Opinion is a direct comparison of positive preferences for the Windows product over the DOS product. It is reasonable that lower or higher preferences would be a major factor in measuring cost/benefits. Furthermore, it is consistent with the literature on network externalities, which often considers adopter preferences and adoption benefits as synonymous (Katz and Shapiro 1985; Arthur 1989). The inclusion of opinion, rather than awareness and consideration, makes sense from a marketing point of view. For instance, one might argue that in the adoption process, awareness and consideration are likely to occur before an opinion is developed. If so, opinion should subsume the other measures. Further, it is not surprising that installed base was included over complementary goods. Installed base is the more direct manifestation of network externalities. An important result is that the bifurcation variable (installed base) measure had a greater effect (0.1950) than the asymmetry variable (opinion) (0.0231), that is 89% versus 11%, on adoption. The finding that bifurcation plays a larger role than asymmetry strongly supports economic theory, which argues that when network externalities are present, the adoption behavior of firms moves in bandwagons.

The results are made more salient by examining the market trajectories in Figures 10 and 11, which plot the latent variables of $Cusp_1$ as produced by GEMCAT II. These figures tell the same story but from different perspectives (two different planar views of the underlying cusp model). In both figures reference lines help contrast the perspectives relative to the model. Figure 10 shows the trajectory of firm adoptions for 34 quarters (8.5 years). Note that DOS has a small early benefit but externalities are low (location of quarter #1). During this time there were several other operating systems still hanging on (including the early version of Windows).

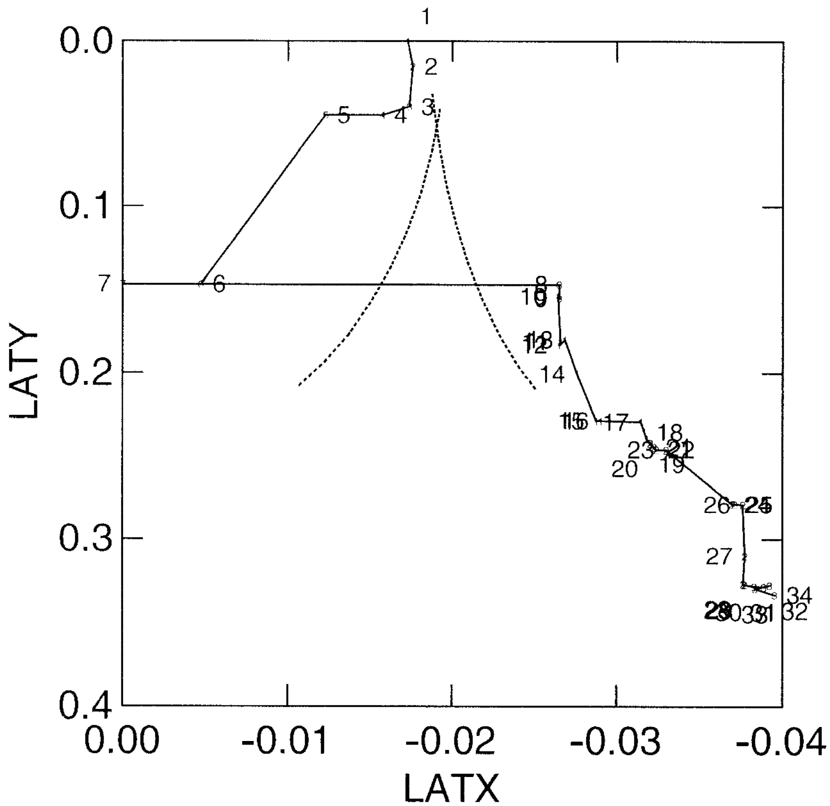

Figure 10
Plot of latent opinion *(LATX)* versus installed base *(LATY)* for 34 quarters. The cusp bifurcation area is shown for reference.

Over time, DOS becomes entrenched and the market moves down and to the left as network externalities become important (periods 1 through 7). Windows, however, keeps making improvements, and a point is reached where the benefits of switching to Windows become overwhelming (period 8). This is noted in the dramatic shift to the right on *LATX*, and in the process crossing the rightmost cusp boundary, signaling a shift should have taken place in the market. Figure 11 tells the same story in terms of adoptions. Here *LATZ* clearly shows the expected shift in the market to the Windows versions of the spreadsheet in period 8. After that time the market continues with Windows, and this operating system dominates from that point on.

Historical back-casting backs up the modeling effort. It was not until Windows 3.1 that the Windows operating system really took off. In fact, there is anecdotal evidence that Microsoft does not get a software product right until the third version is published. Although the main purpose of the present example is to demonstrate GEMCAT II, we suggest that the results are interesting and consistent with economic theory and that they merit further examination. We also note that the approach overcomes some difficulties in other diffusion methods (e.g., Norton and Bass 1987), which require complete replacement of the original by the competitor. This is clearly not the case in software markets, as competing products may coexist here for quite some time.

4 Conclusion

We have demonstrated the efficacy of GEMCAT II for testing catastrophe models in two different contexts. First, the algorithm was subjected to rigorous simulations that evaluate its ability to recover parameters and avoid false positive indications of catastrophes in the face of linear data. Second, the algorithm was used under actual research conditions to demonstrate its ability to function when noisy real-world data are used to

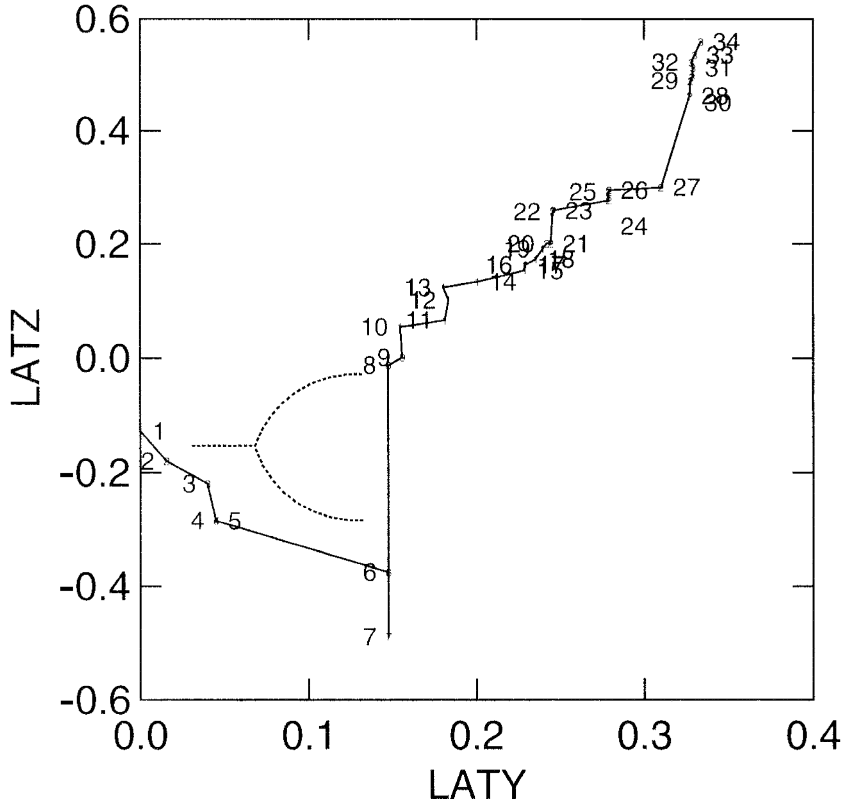

Figure 11
Plot of latent installed base *(LATY)* versus trial *(LATZ)* for 34 quarters. The parabolic split in the cusp surface is shown for reference.

compare competing catastrophe models. We note that this test's primary purpose was illustrative and that a fuller examination of the conceptual model is needed before conclusive statements can be made. Keeping this limitation in mind, however, our analyses clearly show the efficacy of nonlinear methods for modeling dynamic market behavior. We strongly believe therefore that GEMCAT II is a useful tool that can be added to modelers' and researchers' portfolios of nonlinear approaches.

Appendix A: Some Implementation Aspects

As was indicated in Figure 2, the implementation of GEMCAT II revolves around two Delphi class objects, *TStore* and *TFuncVal*, and a set of functions that compute the e_t in Equation (2.5) for the various catastrophes. These functions are contained in the unit *CatFuncs* (see Appendix B for a complete listing). *TModel* records contain all information needed to govern the communication among the various components. Unlike Pascal, Delphi allows the use of procedure and function pointers, and therefore the particular catastrophe being used is represented by the function pointer field *Func* in a *TModel* record. The following describes the heart of the system as embodied in *TStore* and *TFuncVal* objects in a top-down fashion.

TStore Objects. A partial definition of a *TStore* object is shown below, and four of its methods are listed and their functions are commented. Over 20 methods that deal, for instance, with input field selection and variable standardization are omitted.

```
TStore= Class

// Selection style can be bootstrap, jackknife, or input order
```

```
   Constructor Init(FileName : String; S : TSelectStyle; Seed : Integer);
   Destructor Done;

// Reset according to selection style, i.e., bootstrap, jackknife, or input order
   Procedure ResetStore;

// Repeat the sequence created by ResetStore  Procedure RepeatStore;

// Return the data of the next case in R. Return false if none left.
   Function NextCase(var R : TRow) : Boolean;

   ..

   end;
```

TFuncVal Objects. A *TFuncVal* object takes a *TStore* object and a *TModel* record as its arguments and computes the plug-in values of the various free indicator weights during its initialization. *TFuncVal* has over 30 methods, five of which are listed below.

```
TFuncVal = Class
   Constructor Init(Var M : TModel; Var S : TStore);
   Destructor Done;

// Compute the parameter values corresponding to the least
// squares as determined by the Downhill Simplex method followed by an
// application of Powell's method. Return the indicator weights in W.

   Function Minimize(var W : TRow) : RealType;

   Function DoDownHill(var W : Trow) : RealType;
   Function DoPowell(Var W : TRow) : RealType;
   Function SumResid(Var W : TRow) : RealType;
   ..

   end;
```

A run is initiated by assigning the *TModel* fields and creating a *TStore* object. For instance, the following Delphi statements cause GEMCAT II to read a data set from "Infile" and to compute the indicator weights' plug-in values in a cusp catastrophe. Next, *TFuncVal* performs 200 bootstrap replications. Although some simplifications are made for purposes of presentation, the following pseudocode is extremely close to the actual Delphi statements in GEMCAT II.

```
   Var M : TModel;
       S : TStore;
       F : TFuncVal;

   With M do begin
      Func := Address(CuspCat);   // The cusp objective function
```

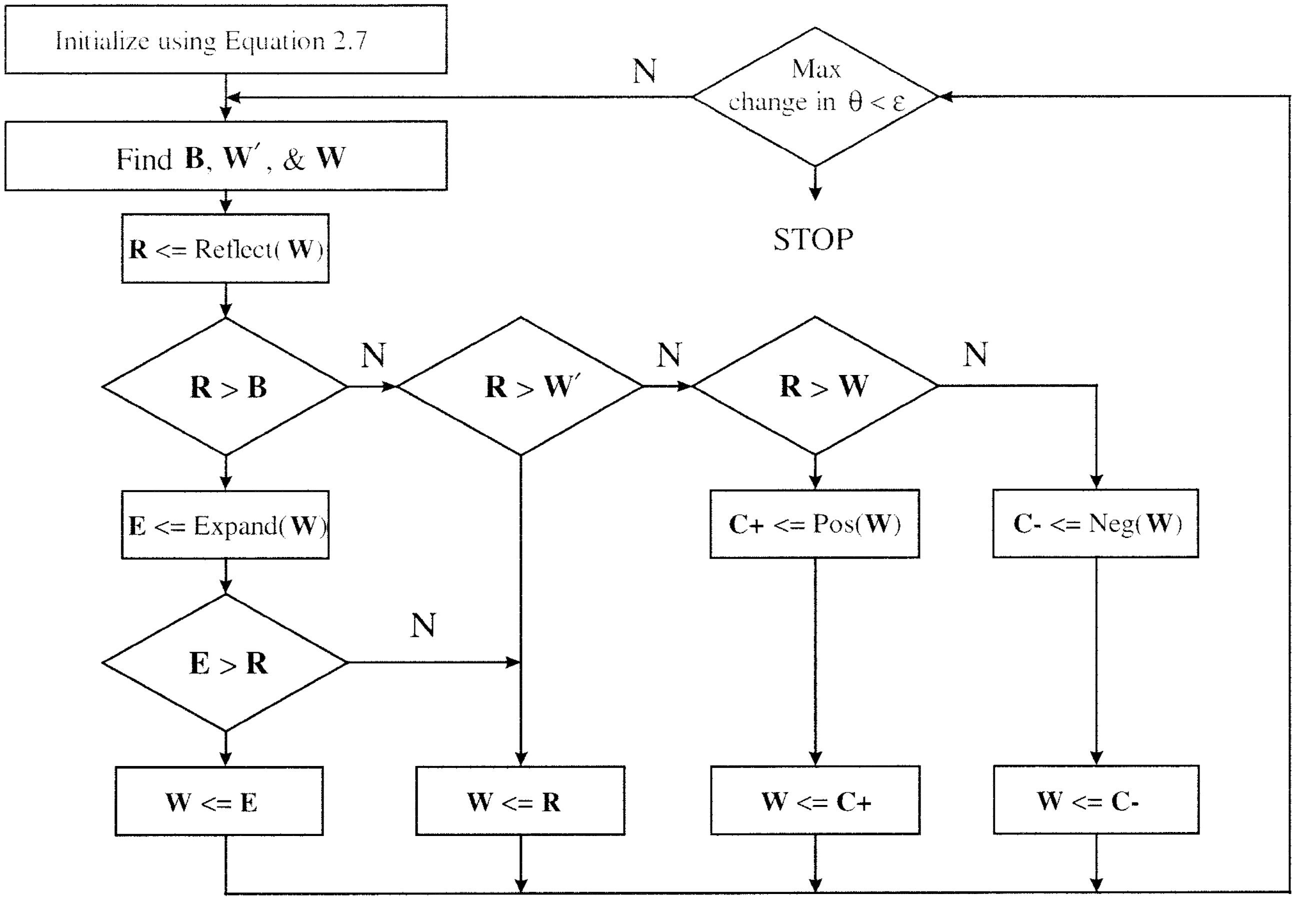

Figure 12
Flowchart of the Downhill Simplex algorithm. **B** = best point, **W** = worst point, and **W**′ = next worst point. The symbol ">" means "is better than," and "<=" represents the assignment operation. The functions "*Reflect*," "*Expand*," "*Pos*," and "*Neg*," respectively, denote the reflection, expansion, positive contraction, and negative contraction operations as shown in Figure 1.

```
    SelectStyle := Tbootstrap;
    NBootSamples:= 200;
    ..
    end;

S := TStore.Init('Infile',M.SelectStyle,-1); // use system default seed

F := TFuncVal.Init(M, S);
..
```

TFuncVal's main function is to encapsulate the *Minimize* method, which calls both the Downhill Simplex method and Powell's Conjugate Gradient algorithm, according to the rules outlined in Section 2.3. The latter algorithm can be found in Press et al. 1992, pp. 331–339, and a flowchart describing GEMCAT II's implementation of the Downhill Simplex is shown in Figure 12. In this figure, the comparisons between the worst point *W* and its reflection, expansion, or contraction are expressed in terms of the sum of residuals, as computed by the function *SumResid*. The following shows how *SumResid* computes such residuals from sets of observations *C* as retrieved from *Store* by *NextCase* and a set of indicator weights *IW* as produced by *DoDownHill* or *DoPowell*.

```
Function TFuncVal.SumResid(Var IW : TRow) : RealType;
   var Store : TStore;
begin

        ...
   with Store do begin

        RepeatStore; // Repeat the Bootstrap or Jackknife sequence
        Result := 0.0;
        while NextCase(C) do Result := Result + Sqr(Func(C,IW,Model));
        ...
        end
   end;
```

Appendix B: Defining Catastrophe Objective Functions and Initialization Procedures

Shown below are the contents of file "*Catfuncs.pas*." When no catastrophe function is provided, *DefaultFunction* is called to return an error message. By default method the Downhill Simplex is initialized via *DefaultInitSimplex*. The fold, cusp, swallowtail, and butterfly catastrophes are represented by the functions *FoldCat*, *CuspCat*, *SwallowTailCat*, and *ButterflyCat*, respectively. By following the conventions outlined in the comments preceding the *FoldCat* function, users can redefine any of these functions to suit their needs or add any other desired function. Naturally, *CatFuncs* must be recompiled in Borland's (Inprise) Delphi V3.0 (or higher) for any changes or additions to take effect. Readers interested in doing so should contact the first author to obtain the required Delphi "**.dcu*" files.

```
unit catfuncs;

interface

Uses Global, Store, Dialogs, SysUtils, Math;

// Do not remove "DefaultInitSimplex" unless you write your own initialization code
   Function DefaultInitSimplex(Var M : TModel; Var S : TStore; Var Initial : TSquare) :
              Boolean;
   Function DefaultFunction(var R, W : TRow; Var M : TModel) : RealType;

// Utilities
   Procedure FuncTerms(Var R : TRow);
   Procedure InitPosNegVal(N : Integer);
   Function GetPosInitVal : RealType;
   Function GetNegInitVal : RealType;
   Procedure SetInitWeights(W : TRow);
   Function GetInitWeigths : TRow;

// Catastrophe functions
   Function FoldCat(var R, W : TRow; Var M : TModel) : RealType;
```

```
   Function CuspCat(var R, W : TRow; Var M : TModel) : RealType;
   Function SwallowTailCat(var R, W : TRow; Var M : TModel) : RealType;
   Function ButterflyCat(var R, W : TRow; Var M : TModel) : RealType;


implementation

var
   T1,T2,T3,T4,T5,T6 : RealType;
   PosVal, NegVal : RealType;
   InitWeights : TRow;


// Will be called if no other initialization procedure is specified by user

   Function DefaultInitSimplex(Var M : TModel; Var S : TStore; Var Initial : TSquare) :
      Boolean;
      var i,j,NW : Integer;
   begin
      NW := M.NWeights;
      Result := NW > 0;
      if not Result then exit;

   // Make vectors defining the simplex completely orthogonal
      For j := 1 to NW do Initial[1,j] := 0.0;
      For i := 2 to Succ(NW) do
         for j := 1 to NW do
            if j = Pred(i) then Initial[i,j] := InitWeights[j] + PosVal
            else Initial[i,j] := InitWeights[j] + NegVal
      end;


   Function DefaultFunction(var R, W : TRow; Var M : TModel) : RealType;
   begin
      ShowMessage('No function was specified');
      Result := -99
      end;


// -------------------------------- Utilities ----------------------------------

// Return the term variables as used in the catastrophe definitions. Use them
// as temporary variables if you need the intermediate results.

   Procedure FuncTerms(Var R : TRow);
   begin
      R[1] := T1;
```

```
      R[2] := T2;
      R[3] := T3;
      R[4] := T4;
      R[5] := T5;
      R[6] := T6
      end;

   procedure InitPosNegVal(N : Integer);
   begin
      PosVal := (1.0 / (N*sqrt(2))) * (Pred(N) + Sqrt(Succ(N)));
      NegVal := -(1.0 / (N*sqrt(2))) * (Sqrt(Succ(N)) - 1.0)
      end;


   Function GetPosInitVal : RealType; begin Result := PosVal end;

   Function GetNegInitVal : RealType; begin Result := NegVal end;

   Procedure SetInitWeights(W : TRow); begin InitWeights := W end;

   Function GetInitWeigths : TRow; begin Result := InitWeights end;




// ---------------------------------- Catastrophes ----------------------------
// The variable names X, Y, Z, U, V, and W, are taken from Oliva et.al. (p. 123)
//
// FOLD        Zˆ2 - X
// CUSP        Zˆ3 - X - YZ
// SWALLOWTAIL Zˆ4 - X - YZ - VZˆ2
// BUTTERFLY   Zˆ5 - X - YZ - VZˆ2 - UZˆ3
//
// For "historical" reasons, the following convention developed:
//
// variables suffixed by 1x = X     (e.g., N11, N12, ...)
// variables suffixed by 2x = Y
// variables suffixed by 3x = Z
// variables suffixed by 4x = U
// variables suffixed by 5x = V
// variables suffixed by 6x = W
// NOTE : This does NOT apply to terms Tx to maintain contiguity
//
// The constants set by the user are denoted by Cval[*], and there are Cx1 to Cx2
   of these.

// FOLD Zˆ2 - X
```

```
   Function FoldCat(var R, W : TRow; Var M : TModel) : RealType;
      var i : Integer;
   begin
      with M do begin
         T1 := 0.0;   // X
         For i := N11 to N12 do T1 := T1 + R[i]*W[i];
         For i := C11 to C12 do T1 := T1 + R[i]*Cval[i];

         T2 := 0.0;   // Z
         For i := N31 to N32 do T2 := T2 + R[i]*W[i];
         For i := C31 to C32 do T2 := T2 + R[i]*Cval[i]
         end;

         Result := sqr(T2) - T1
      end;


// CUSP Z^3 - X - YZ

   Function CuspCat(var R, W : TRow; Var M : TModel) : RealType;
      var i : Integer;
   begin
      with M do begin
         T1 := 0.0;          // X
         for i := N11 to N12 do T1 := T1 + R[i]*W[i];
         For i := C11 to C12 do T1 := T1 + R[i]*CVal[i];

         T2 := 0.0;          // Y
         for i := N21 to N22 do T2 := T2 + R[i]*W[i];
         For i := C21 to C22 do T2 := T2 + R[i]*CVal[i];

         T3 := 0.0;          // Z
         for i := N31 to N32 do T3 := T3 + R[i]*W[i];
         For i := C31 to C32 do T3 := T3 + R[i]*CVal[i];

         Result := T3 * (Sqr(T3) - T2) - T1
         end
      end;


// SWALLOWTAIL Z^4 - X - YZ - VZ^2

   Function SwallowTailCat(var R, W : TRow; Var M : TModel) : RealType;
         var i : Integer;
   begin
```

```
      with M do begin
         T1 := 0.0;          // X
         for i := N11 to N12 do T1 := T1 + R[i]*W[i];
         For i := C11 to C12 do T1 := T1 + R[i]*CVal[i];

         T2 := 0.0;          // Y
         for i := N21 to N22 do T2 := T2 + R[i]*W[i];
         For i := C21 to C22 do T2 := T2 + R[i]*CVal[i];

         T3 := 0.0;          // Z
         for i := N31 to N32 do T3 := T3 + R[i]*W[i];
         For i := C31 to C32 do T3 := T3 + R[i]*CVal[i];

         T4 := 0.0;          // V
         for i := N51 to N52 do T4 := T4 + R[i]*W[i];
         For i := C51 to C52 do T4 := T4 + R[i]*CVal[i];

         Result := sqr(T3);
         Result := Result * (Result - T4) - T1 - T2*T3
         end
      end;


// BUTTERFLY Z^5 - X - YZ - VZ^2 - UZ^3

   Function ButterflyCat(var R, W : TRow; Var M : TModel) : RealType;
         var i : Integer;
   begin
      with M do begin
         T1 := 0.0;          // X
         for i := N11 to N12 do T1 := T1 + R[i]*W[i];
         For i := C11 to C12 do T1 := T1 + R[i]*CVal[i];

         T2 := 0.0;          // Y
         for i := N21 to N22 do T2 := T2 + R[i]*W[i];
         For i := C21 to C22 do T2 := T2 + R[i]*CVal[i];

         T3 := 0.0;          // Z
         for i := N31 to N32 do T3 := T3 + R[i]*W[i];
         For i := C31 to C32 do T3 := T3 + R[i]*CVal[i];

         T4 := 0.0;          // U
         for i := N41 to N42 do T4 := T4 + R[i]*W[i];
         For i := C41 to C42 do T4 := T4 + R[i]*CVal[i];

         T5 := 0.0;          // V
         for i := N51 to N52 do T5 := T5 + R[i]*W[i];
```

```
        For i := C51 to C52 do T5 := T5 + R[i]*CVal[i];

        Result := sqr(T3);
        Result := Result * (T3*(-T4 + Result)-T5) - T1 - T2*T3
        end
    end;
end.
```

References

Acton, F. S. (1970). *Numerical Methods.* New York: Harper and Row.

Alexander, R., G. Herbert, R. DeShon, and P. Hanges. (1992). An examination of least-squares regression modeling of catastrophe theory. *Psychological Bulletin*, 111: 366–374.

Arthur, B. W. (1989). "Competing technologies, increasing returns, and lock-in by historical events." *Economic Journal*, 99: 116–131.

Bates, D. M., and D. G. Watts. (1988). *Nonlinear Regression Analysis and Its Applications.* New York: Wiley & Sons.

Bayne, C. K., and I. B. Rubin. (1986). *Practical Experimental Designs and Optimization Methods for Chemists.* Deerfield Beach, FL: VHC.

Borland International Inc. (1997). *Delphi Version 3.0.* Scotts Valley, CA: Author.

Brent, R. P. (1973). *Algorithms for Minimization without Derivatives.* Englewood Cliffs, NJ: Prentice-Hall.

Cobb, L. (1978). "Stochastic catastrophe models and multimodal distributions." *Behavioral Science*, 23: 360–416.

Cobb, L. (1981). "Parameter estimation of the cusp catastrophe model." *Behavioral Science*, 26: 75–78.

Cobb, L. (1998). "An introduction to cusp surface analysis." Corrales Software Development, Carbondale, CO. Available on-line at http://www.rof.net/yp/corrales/models/cusp/Intro.html

Dou, W. (1999). "Data-driven marketing strategy design and catastrophe modeling-based implementation on the internet." Unpublished Ph.D. diss., School of Business Administration, University of Wisconsin–Milwaukee.

Dou, W., and S. Ghose. (2000). "A catastrophe model of online store sales." Working paper, St. Cloud University, St. Cloud, MN.

Efron, B., and R. J. Tibshirani. (1993). *An Introduction to the Bootstrap.* New York: Chapman & Hall.

Farrell, J., and G. Saloner. (1985). "Standardization, compatibility, and innovation." *Rand Journal of Economics*, 16: 70–83.

Farrell, J., and G. Saloner. (1986). "Installed base and compatibility: Innovation, product preannouncements, and predation." *American Economic Review*, 75: 940–955.

Fischer, E., and W. Jammernegg. (1986). "Empirical investigation of a catastrophe theory extension of the Phillips curve." *Review of Economics and Statistics*, 68: 9–17.

Guastello, S. J. (1982). "Moderator regression and the cusp catastrophe: Application of two-stage personnel selection, training, therapy, and policy evaluation." *Behavioral Science*, 27: 259–272.

Guastello, S. J. (1992). "Clash of the paradigm: A critique of an examination of the least squares regression technique for evaluating catastrophe theory hypotheses." *Psychological Bulletin*, 111: 375–379.

Guastello, S. J. (1995). *Chaos, Catastrophe, and Human Affairs: Applications of Nonlinear Dynamics to Work, Organizations, and Social Evolution.* Mawah, NJ: Lawrence Erlbaum Associates.

Ho, T., and A. Saunders. (1980). "A catastrophe model of bank failure." *Journal of Finance*, 35(5): 1189–1207.

Katz, M., and C. Shapiro. (1985). "Network externalities, competition, and compatibility." *American Economic Review*, 75(June): 425–440.

Knuth, D. E. (1969). *The Art of Computer Programming. Vol. 2: Seminumerical Algorithms.* Reading, MA: Addison-Wesley.

Lange, R. (1998). "GEMCAT II : A distribution-free method for testing the statistical properties of Thom's catastrophe equations as defined by sets of arbitrary latent variables." Paper presented at the Eighth Annual Conference of the Society for Chaos Theory in Psychology and the Life Sciences, Boston, MA., July 28–August 1.

Marascuilo, L. A., and M. McSweeney. (1977). *Nonparametric and Distribution-Free Methods for the Social Sciences*. Monterey, CA: Brooks/Cole.

MathSoft Inc. (1994). *Mathcad V5.0 User's Guide*. Cambridge, MA: Author.

Nelder, J. A., and R. Mead. (1965). "A simplex method for function minimization." *Computer Journal*, 7: 308–313.

Norton, J., and F. Bass. (1987). "A diffusion theory of adoption and substitution for successive generations of high-technology products." *Management Science*, 33(9): 1069–1086.

Oliva, T. (1994). "Technological choice under conditions of changing network externalities." *Journal of High Technology Management Research*, 5(2): 279–298.

Oliva, T., W. DeSarbo, D. Day, and K. Jedidi. (1987). "GEMCAT: A general multivariate methodology for estimating catastrophe models." *Behavioral Science*, 32: 121–137.

Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. (1992). *Numerical Recipes in Pascal*. Cambridge: Cambridge University Press.

Price, W. (1976). "A controlled random search procedure of global optimization." *Computer Journal*, 20: 368–370.

Spendley, W., G. R. Hext, and F. R. Himsworth. (1962). "Sequential application of simplex designs in optimisation and evolutionary operation." *Technometrics*, 4: 441–461.

Thom, R. (1975). *Structural Stability and Morphogenesis*. New York: Benjamin-Addison-Wesley.

Varian, H. (1979). "Catastrophe theory and the business cycle." *Economic Inquiry*, 17: 14–28.

Zeeman, E. (1974). "On the unstable behavior of stock exchanges." *Journal of Mathematical Economics*, 1: 39–49.

Advisory Panel

Editorial Board

Editorial Policy

The *SNDE* is formed in recognition that advances in statistics and dynamical systems theory may increase our understanding of economic and financial markets. The journal will seek both theoretical and applied papers that characterize and motivate nonlinear phenomena. Researchers will be encouraged to assist replication of empirical results by providing copies of data and programs online. Algorithms and rapid communications will also be published.

ISSN 1081-1826