Active Nonlinear Tests (ANTs) of Complex Simulation Models*

John H. Miller†

Revised: 5 February, 1996

Keywords: Testing Simulation Models, Nonlinear Sensitivity Analysis, World3 Model, and Genetic Algorithms.

Abstract

Simulation models are becoming increasingly common in the analysis of critical scientific, policy, and management issues. Such models provide a way to analyze complex systems characterized by both large parameter spaces and nonlinear interactions. Unfortunately, these same characteristics make understanding such models using traditional testing techniques extremely difficult. Here we show how a model's structure and robustness can be tested via a simple, automatic, nonlinear search algorithm designed to actively "break" the model's implications. Using the active nonlinear tests (ANTs) developed here, one can easily probe for key weaknesses in a simulation's structure, and thereby begin to improve and refine the model's design. We demonstrate ANTs by testing a well-known model of global dynamics (World3), and show how this technique can be used to uncover small, but powerful, nonlinear effects that may highlight vulnerabilities in the original model.

*This paper has benefited from discussions with Robert Axelrod, Paul Fischbeck, Murray Gell–Mann, Scott Page, Laura Painton, Ken Simons, Sally Sleeper, and other colleagues at both Carnegie Mellon University and the Santa Fe Institute. Research support was generously provided by the National Science Foundation (grant SBR–9411025), Carnegie Mellon University, and the economics program of the Santa Fe Institute.

†Department of Social and Decision Sciences, Carnegie Mellon University, Pittsburgh, PA 15213 (email: miller@santafe.edu, WWW: zia.hss.cmu.edu).

1 Introduction

Complicated, large-scale computational models are becoming increasingly common in the analysis of critical scientific, policy, and management issues. Such models are particularly well suited for the analysis of phenomena that are characterized by high degrees of nonlinearity and enormous search spaces—conditions that confound traditional analytic methods. Unfortunately, the same conditions that make computational techniques so appealing, are also ones that make testing such models inherently difficult. For example, techniques like sensitivity analysis that sequentially perturb individual model parameters, require simple *linear* relationships within the model to be effective. As an alternative, here we suggest that a computational model's structure and robustness can be tested via a simple, automatic, nonlinear search algorithm designed to actively "break" the original model's implications.

The basic idea behind the active nonlinear tests (ANTs) developed here is rather simple: use a nonlinear optimization algorithm to search across a set of reasonable model perturbations with the objective of maximizing the deviation between the original model's prediction and that obtained from the model under the perturbations. Of course, by varying the objective function for this optimization one can attempt to "break" the model in different ways—the actual choice will depend on what aspect of the model one wants to stress, but presumably it will include those implications of the model that are of the greatest importance to the modeler. In a model designed to inform a particular policy or management decision, the goal might be to destroy the predictions of the model that are most in favor of a particular decision. In models designed to create theoretical predictions (see for example, Holland and Miller, 1990), ANTs could be used to find conditions under which the derived theory fails. More generally, ANTs could be used to enhance the exploratory modeling notions put forth by Bankes (1993), and Bankes and Gillogly (1994).

Perhaps the simplest application of this methodology is in testing a model's behavior vis-a-vis its parameters. If the underlying model is highly nonlinear, then information about the impact of altering individual parameters may not be useful in determining the effect of changes in groups of parameters.[1] Because this technique searches across *sets* of parameter values, it is capable of detecting important nonlinear relationships among the parameters—relationships that go unnoticed using standard techniques like sensitivity analysis. While detecting nonlinearities could be done by exhaustively searching over all possible parameter groupings, the implied combinatoric explosion makes this infeasible for even small numbers of parameters. Thus, there is a need for a more directed search mechanism that seeks out *groups* of parameter that affect the model. The use of a nonlinear search algorithm allows such a directed search to occur.

The ANTs developed here automatically probe for weakness in the model's behavior. One view of this process is that it is a way to actively seek worst case scenarios of the model. While such an exercise does not give an estimate of the likelihood of such scenarios (existing

[1] For example, consider the model $y(x_1, x_2) = x_1 x_2$. Then, $y(x_1 + \Delta x_1, x_2) = x_1 x_2 + x_2 \Delta x_1 = y(x_1, x_2) + x_2 \Delta x_1$. Similarly, $y(x_1, x_2 + \Delta x_2) = y(x_1, x_2) + x_1 \Delta x_2$. Yet, $y(x_1 + \Delta x_1, x_2 + \Delta x_2) = y(x_1, x_2) + x_2 \Delta x_1 + x_1 \Delta x_2 + \Delta x_1 \Delta x_2$. Which differs from the combination of the individual perturbations by the last term, $\Delta x_1 \Delta x_2$. Of course, the linear approximation works well if either the perturbations are small or the degree of nonlinearity is low.

techniques like Monte Carlo methods[2] can be used for this task), it does give valuable insight into the maximum error that is possible in the model. More importantly, these searches are a means by which to uncover potential weaknesses in the model's formulation and identify the model's key assumptions. With this information, these assumptions can either be refined or, if they are felt to be sound, additional effort can be focused on better estimating and understanding the behavior of the associated parameters. Note that the inability to "break" a model in this way does not guarantee its quality. For example, models that are completely insensitive to their parameters can obviously not be broken in the above manner—yet, such models are also not likely to be of much value. The tradeoff between the brittleness in a model and its responsiveness to parameters will always need to be carefully considered.

Although the ANTs methodology can be implemented in a variety of ways, here we will use Holland's (1975) genetic algorithm (GA) as the nonlinear optimization procedure. To illustrate the technique, we will test a simulation model of global population and resource dynamics (the World3 model developed by Meadows et al. (1974, 1992)). The choice of this model was simply due to its wide availability and easy accessibility. Underlying the model is a large set of uncertain parameters, for example, current population stocks, technological growth rates, and agricultural productivity. We will use the GA to find new sets of parameters (constrained to lie within a narrow range of the original ones) such that some of the model's main conclusions about the future path of population growth will be most at odds with what the original parameters suggest.

2 An Algorithm for Active Nonlinear Tests

To implement our methodology, we require a nonlinear optimization algorithm that can effectively search over various perturbations in the model's formulation. Although a genetic algorithm is well suited for this task, clearly other nonlinear search algorithms (for example, simulated annealing) could be used. As for the search space, here we will simply restrict the search to a well-defined neighborhood of the original parameters and assume that new parameters must be within a certain percentage of the original ones (for example, all parameters are measured with some error rate). While parameter perturbations are likely to be a common use of ANTs, other more general notions of model perturbations can easily be incorporated into our procedure. For example, the algorithm could be used to search over different formulations of the model's underlying equations, operating assumptions, etc.

Genetic algorithms have proven to be an effective search technique in optimization problems that confront nonlinearities, discontinuities, noise, and enormous search spaces (see, for example, Goldberg, 1989). In a GA, a population of "solutions" is initially created at random. In our case, each solution represents a set of allowable perturbations of the model's original parameters. Each solution is then tested on the problem, and receives a measure of "fitness." Here, fitness is measured by the impact of the perturbations on the model's

[2]For reviews of methods to capture parametric uncertainty see Cox and Baybutt (1981), Iman and Helton (1988), and Morgan and Henrion (1990). Note that most of these methods require a large number of samples to be run on the simulation before they can be applied, and thus the directed nature of ANTs may prove advantageous in those cases where running the underlying simulation model is costly and standard uncertainty methods become infeasible.

behavior given some well-defined objective function (for example, deviation from a specified conclusion). The GA then creates a new population of solutions by both reproducing some of the old solutions based on their performance (with better performers being more likely to be reproduced) and creating some new solutions by applying "genetic operators." GAs typically employ two genetic operators: crossover and mutation. Crossover recombines pieces of existing solutions in a way that preserves those parts of each solution that likely result in good performance. Mutation randomly makes small, unique alterations in a solution and thereby prevents the system from getting prematurely trapped in local optima. Once a new population of solutions has been created, a new "generation" of the algorithm begins and the above testing, reproducing, and modifying stages are iterated. Although the GA requires very little computational power to implement, the above cycle results in a sophisticated sampling scheme on the key patterns underlying effective solutions (Holland, 1975).

To implement the GA for our problem, we need to first define the format of the potential solutions. In the examples below, we simply define strings that contain P elements, where P is the number of parameters in the model we are testing, and each element in the string contains an allowable perturbation of the corresponding parameter. This is done here by letting each element in the string represent a perturbation in the range $\{-10\%, -9\%, \ldots, 0\%, \ldots, 9\%, 10\%\}$ of the original parameter. Thus, in a model of, say, four parameters, the string $\{-5, 0, 0, +8\}$ would test the model in which the first parameter was decreased by 5%, the second and third were not disturbed, and the last was increased by 8%.

Once we have a representation for the solutions we need to define a fitness measure for each solution's performance. For ANTs we need to define an objective function such that its optimization will lead to the destruction of an anticipated model outcome. Thus, the appropriate fitness (objective) function for the optimization will maximize the difference between the expected behavior (null hypothesis) given the original parameters verses the behavior implied by the perturbed parameters, that is, $f(M_n(p), M_n(p + \Delta p)) = |M_n(p) - M_n(p + \Delta p)|$, where $M_n(p)$ is the outcome of the model for hypothesis n under parameters p and Δp is the set of parametric perturbations. The choice of such a function will depend on what conclusions the modeler wishes to infer from the simulation. A variety of modeling aspects can be approached via these functions, including, but certainly not limited to, the stability (or instability) of a predicted pattern, and the occurrence and timing of particular events.

While the objective function defined above will allow the system to actively search for conditions that cause the greatest deviation from what is expected given the original model, it does not impose any parsimony on the number of perturbations. This is problematic since one application of ANTs is for identifying small sets of perturbations in need of further investigation. One way to solve this parsimony problem is to restrict the search space such that only small sets of perturbations are admitted at any given time. An alternative, is to include a penalty term in the objective function that encourages parsimony. Under this approach, the objective function becomes $f(M_n(p), M_n(p+\Delta p), p) = |M_n(p) - M_n(p+\Delta p)| + \alpha\phi(\Delta p)$, where α is a non-positive cost parameter and $\phi(\Delta p)$ is an increasing function of the number (and perhaps size) of perturbations in Δp.

3 An Example

To illustrate ANTs, we apply the above techniques to the World3 model, scenario two, developed by Meadows et al. (1974, 1992). This model was designed to simulate those systems that are of most importance to human sustainability, including population, industrial output, pollution, and agriculture. Hayes (1993) provides a basic review of the model's development and operation. World3 is used because it represents a convenient example of a large-scale simulation model. While many aspects of the model's formulation have been criticized (see, for example, Nordhaus, 1992), the issues of concern here—testing nonlinear simulation models—transcend such controversies. The World3 model has approximately 150 equations that link 272 model variables. Of these 272 variables, 96 of them need to be initialized. Moreover, there are 508 parameters required to specify various function shapes for use in the model.[3]

We test World3 by allowing the GA to choose starting values for the 96 initializing parameters constrained to be within 10% of their original values. We focus on the predicted world population level as the modeling outcome of interest. In Figure 1 (and all subsequent figures) the population path designated as "original" is the one produced by the model with the unperturbed parameters. The model predicts that world population will peak at about 9.4 billion people around the year 2040, and then experience a relatively rapid decline. Our testing of the model will focus on the predicted peak at 2040 (presumably, an event of great policy importance). We will attempt to alter the outcome of the model by attempting to both amplify and destroy the peak.

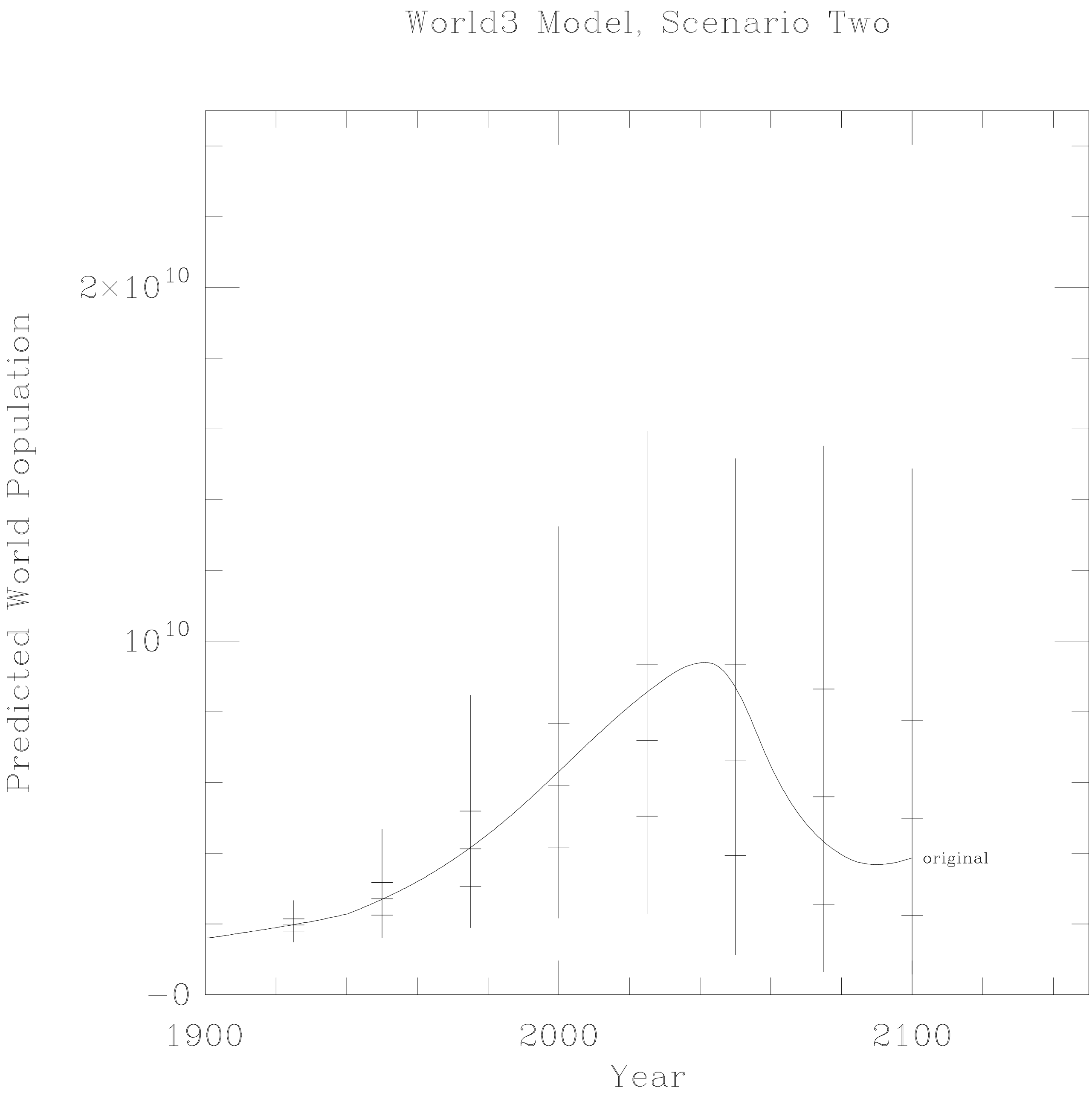
To better understand the behavior of the model under general conditions of uncertainty and to provide some perspective on our results, a Monte Carlo experiment was run for 1000 simulations. During each simulation, we assumed that each parameter was independent and uniformly distributed across the 10% neighborhood defined above.[4] The endpoints of the vertical bars in Figure 1 designate the maximum and minimum population values observed during any of the 1000 simulations at the corresponding time period, with the hash marks on the bars indicating the mean (middle hash) and a one standard deviation interval on either side of the mean of the observed values. The predicted path appears to fall within the one deviation interval, although it appears to be biased slightly upward during the period from 2000 to 2060, and slightly downward thereafter.

The actual GA employed in the procedure worked as follows. During a given run, a population of forty solutions was evolved using a GA. Each solution contained a set of perturbations for the parameters used in the original model. Each perturbation was defined on the range $\{-10\%, -9\%, \ldots, 0\%, \ldots, 9\%, 10\%\}$. After each solution was tried in the simulation, they were reproduced by randomly selecting two solutions (with replacement) and keeping the one with higher fitness. (This is known as tournament selection, and is one of many ways to insure reproduction by performance.) This selection process was repeated

[3]Ken Simons kindly provided the Pascal code for World3, as well as expertise on the model's structure and parameters.

[4]Actually, draws were made on the same grid used for the GA discussed below, and thus the distribution was uniform on this neighborhood in discrete increments of 1%. On average, each parameter set had around 91 perturbed parameters.

Figure 1: Original World3 model and Monte Carlo results.



forty times, resulting in a biased (by better performance) sample of forty solutions drawn from the old population.

The forty new solutions were randomly paired, and with a probability of 0.5 they underwent modification via genetic operators (otherwise, they entered the new population unchanged). If modified, each perturbation was independently and randomly mutated with a probability that implied an expectation of two out of the 96 perturbations being mutated. If a character was mutated, a uniform random draw on the perturbation interval $\{-10\%, -9\%, \ldots, 9\%, 10\%\}$ was made. The mutation rate was annealed over time, decaying in half every fifty generations. (This allows the solutions to converge over time.) Each string that underwent modification was crossed over with its partner by sequentially reading elements down its string and switching to reading elements from the partner string with an independent and identical probability per element that resulted in an average of two reading switches per pair. At the end of the modification process, a generation of the algorithm was concluded and the newly formed solutions were tested in the simulation. This process was repeated for 100 generations. At the end of the 100 generations, the top performing solution in a given population is reported.

The algorithm described above is quite robust to various changes in its underlying parameters and procedures, and a variety of reasonable variants appear to work equally well. Since the algorithm uses stochastic elements, the solution from any given run may vary. Therefore, in the experiments below we report results from two separate runs of the algorithm for each condition.

Along with the unconstrained optimization described in the previous paragraph, a procedure that penalizes excessive perturbations was also implemented. The GA followed the above procedure with two modifications. First, any perturbations that were less then 7% in absolute value were not allowed—that is, each element in the string was restricted to the set $\{-10\%, -9\%, -8\%, 0\%, 8\%, 9\%, 10\%\}$. Note however that the element structure was as defined above (21 possible values), so that the likelihood of a mutation causing a perturbation to go from a 0% value to a nonzero one was 30% (6/20). This modification helps the algorithm by restricting the potential search space to only extreme perturbations. The second modification made was to add a penalty for nonzero perturbations in the objective function. This was done by including a negative cost parameter times the square of the number of nonzero perturbations in the solution. By using the squared number of nonzero terms, the addition of a new perturbation was increasingly penalized.

The first objective function used here was designed to amplify the peak by trying to maximize the predicted population at the end of the simulation (year 2100). In essence, we are seeking a worst case scenario of a maximum population stock at year 2100, assuming only minimal deviations from our expected parameters. The top two lines depicted in Figure 2 are the resulting paths implied by the unconstrained maximization of local parameter perturbation with the objective of maximize the model's population in year 2100. The uppermost path implies an ending population of about 22.3 billion, verses the model's original prediction of 3.9 billion.[5] The other two paths in Figure 2 (both labeled by the parameters

[5]Note that the implied paths of these perturbed runs are very difficult to differentiate from the original run (within around one standard deviation) until around the year 2000, suggesting that even if this scenario was occurring, it would be hard to detect.

that are perturbed) come from runs in which parsimony was encouraged via the penalty function described above. Both of these runs modified only three variables each to achieve the new paths (resulting in ending populations of around 14 billion). In each run, parameters 75 (the fraction of industrial output allocated to agricultural consumption) and 83 (the reproductive lifetime of females) emerged as key perturbations. Note that encouraging parsimony lessens the magnitude of the model's breakage (this will always be the case since the optimization with parsimony is always a constrained version of the one without parsimony), yet still results in a very different outcome from that predicted by the original model.

The solutions found using the constrained procedure provide a simple demonstration of the impact of nonlinear effects. Figure 3 shows the original path along with perturbations in various model parameters (each line is labeled by the parameters involved, and the scale of the ordinate is altered). While each individual perturbation has a noticeable impact, the combination of the two simultaneous perturbations results in a much more extreme effect. Thus, while altering parameter 75 increases ending population by about 2.3 billion and altering parameter 83 increases it by about 1.2 billion, the combination of the two parameters causes a 7.4 billion increase. Similarly, while parameter 63 (the life expectancy normal) alone increases the ending population by only 0.3 billion, when it is combined with parameter 75 the increase is 5.6 billion.[6]

The second objective function we explored was one that attempted to minimize the peak (highest) value observed during any given run of the simulation. Here, the goal is to try and destroy the predicted peak. The two relatively flat lines in the bottom of Figure 4 show the implied paths resulting from two runs of the algorithm with unconstrained perturbations. Obviously, these paths are quite different from the original (however, they achieve this by an early alteration of the path such that the model poorly fits knowledge about current population levels). The other two paths in the diagram are from the runs with the penalty function invoked. In both of these runs, parameters 45 (desired completed family size normal) and 83 appear. The appearance of parameter 83 in both these and the previous maximization runs, indicates that it plays a key role in the model's dynamics concerning population paths.

The above objective functions are only a small subset of those likely to yield interesting insights. For example, another way to destroy the observed peak would be to minimize the predicted population at time 2040 or maximize the observed peak. One could also search for instabilities by trying to maximize the variance of the population path. Another potentially useful objective function would be to maximize the gradient of population change, and thereby discover the likelihood of rapid shifts in population levels.

The above analysis indicates that dramatic changes in the predictions of the World3 model can result from even minor changes in some of its parameters. As discussed earlier, the occurrence of such events does not necessarily imply a faulty model—good models must be responsive to their parameters. Nonetheless, they do indicate the potential for extreme errors, as well as suggest areas of the model that might require further investigation and refinement. Also, note that by implementing a variety of ANTs that use different objective functions, systematic weakness may be revealed (as may be the case with parameter 83 above).

[6] For this demonstration, parameters 63 and 75 were both increased by 10%, and parameter 83 was decreased by 10%.

Figure 2: Original path and new paths generated by maximizing population in year 2100.

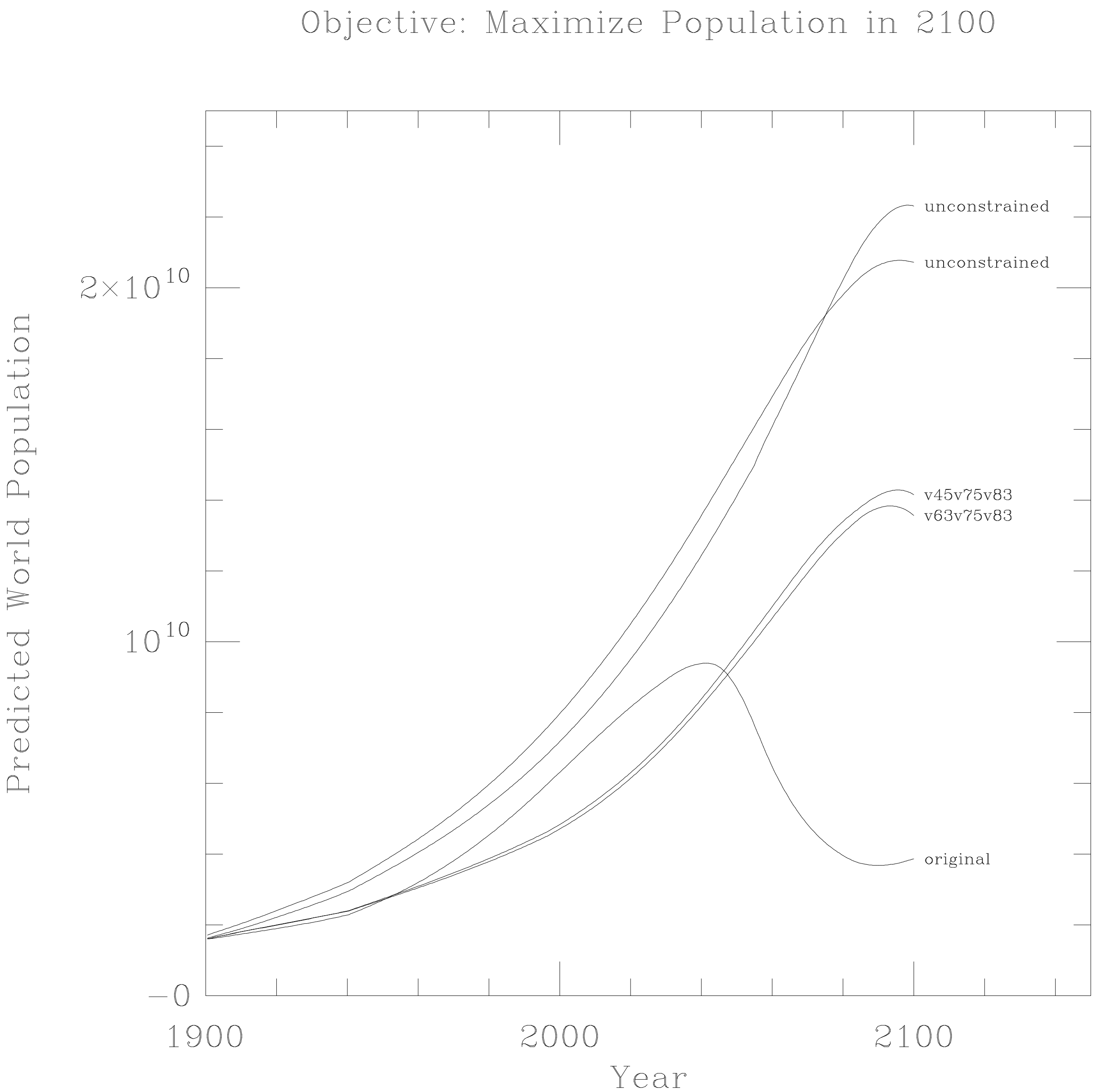


Figure 3: Nonlinear effects of parametric perturbation.

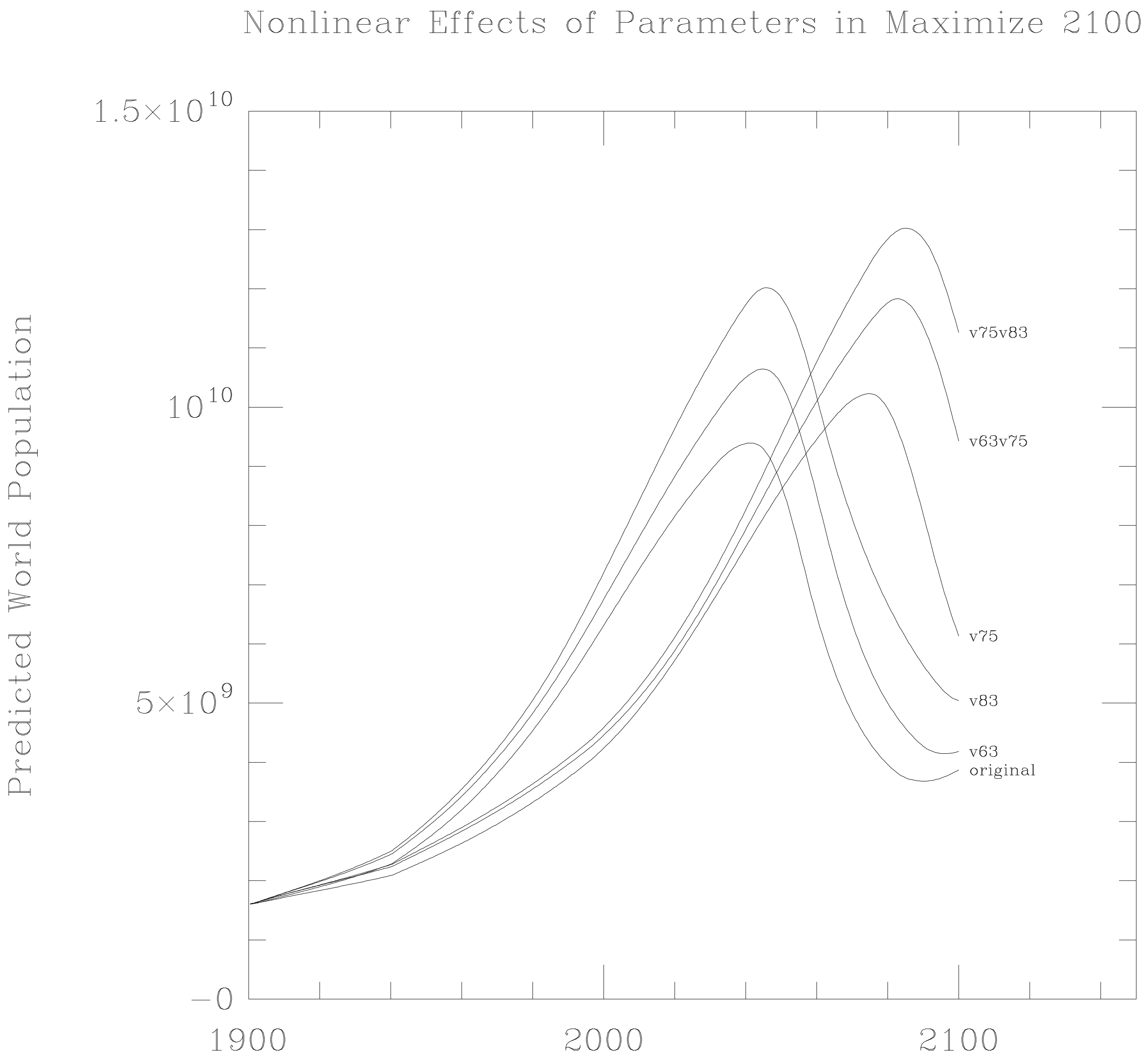
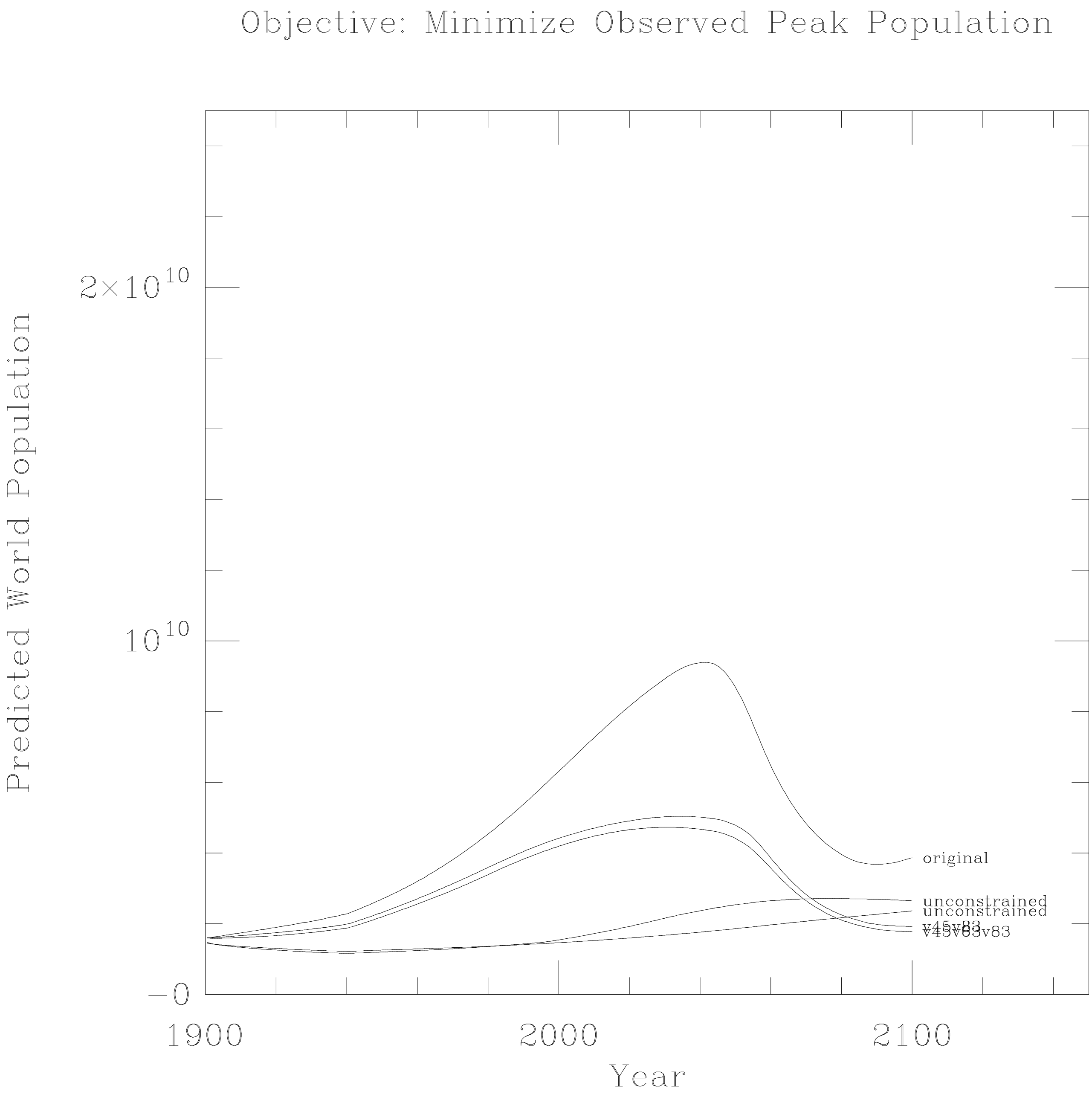


Figure 4: Original model, and paths resulting from minimizing peak population.



4 Conclusions

The growing reliance on large-scale computational models is likely to continue as the cost of computation declines and as researchers and policy makers attempt to confront ever more sophisticated phenomena. The comparative advantage of such models is their ability to embrace systems characterized by large parameter spaces and rampant nonlinearities. Yet, it is these same characteristics that make testing and understanding such models inherently difficult.

ANTs allow a new class of testing to occur on computational models. Given a particular conclusion of the model, an ANT can be easily implemented by defining a corresponding objective function. Once initiated, the ANT automatically probes for weakness in the model's structure. Unlike standard techniques like sensitivity analysis, ANTs are able to find important nonlinear interactions among the model's parameters. Such techniques allow for better models to be developed and refined in this complex environment. ANTs can also be productively employed in those models in which one is confident of the behavior of the underlying model. In these cases, for example, ANTs can be used to discover worst (or best) case scenarios and therein give the user an idea about which parameters should either be altered (if possible) or most closely monitored (for example, one could actively try and break the operation of a simulated aircraft flight system or reactor control system).

Obviously, ANTs alone are not sufficient to guarantee quality modeling. Careful thought and refinement will always be needed on the part of the modeler to insure useful models. Tools like ANTs will certainly help in this process, but cannot substitute for it. The discoveries of ANTs may force changes in the model, closing old, and perhaps opening new, vulnerabilities. The active probing of the model by ANTs provides input from a ready, and tireless, critic. Like ants seeking food at a picnic, a variety of avenues are creatively explored, and it is only with extreme care and foresight that the meal remains untouched.

References

Bankes, Steve. 1993. "Exploratory Modeling for Policy Analysis." *Operations Research* 41: 435–49.

Bankes, Steven and James Gillogly. 1994. *Proceedings of the Third Annual Conference on Evolutionary Programming*, Anthony V. Sebald and Lawrence J. Fogel (eds.). River Edge, New Jersey: World Scientific Publishing Co., pp. 353–60.

Cox, David C. and Paul Baybutt. 1981. "Methods for Uncertainty Analysis: A Comparative Survey." *Risk Analysis* 1:251–8.

Goldberg, David E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Reading, Massachusetts: Addison Wesley.

Hayes, Brian. 1993. "Balanced on a Pencil Point." *American Scientist* 81:510–16.

Holland, John H. 1975. *Adaptation in Natural and Artificial Systems.* Ann Arbor, Michigan: University of Michigan Press.

Holland, John H. and John H. Miller. 1991. "Artificial Adaptive Agents and Economic Theory." *American Economic Review, Papers and Proceedings* 81:365–70.

Iman, Ronald L. and Jon C. Helton. 1988. "An Investigation of Uncertainty and Sensitivity Analysis Techniques for Computer Models." *Risk Analysis* 8:71–90.

Meadows, Dennis L., William W. Behrens III, Donella H. Meadows, Roger F. Naill, Jorgen Randers and Erich K.O. Zahn. 1974. *Dynamics of Growth in a Finite World.* Cambridge: Wright–Allen Press.

Meadows, Donella H., Dennis L. Meadows, and Jorgen Randers. 1992. *Beyond the Limits.* Post Mills, Vermont: Chelsea Green Publishing.

Morgan, M. Granger and Max Henrion. 1990. *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis.* Cambridge: Cambridge University Press.

Nordhaus, William D. 1992. "Lethal Model 2: The Limits to Growth Revisited." Brookings Panel on Economic Activity, draft.