10 Analytica: A Software Tool for Uncertainty Analysis and Model Communication*

> "The real value of computers is communication, not computation."
> Natasha Kalatin

It goes almost without saying that doing quantitative analysis means creating computer models, especially if you want to treat the uncertainty explicitly. It is possible to do back-of-the-envelope calculations for a handful of variables with pencil and paper, using simple error-propagation methods (Section 8.3.5) or probability trees (Section 8.4). Indeed, that is an excellent way to develop your intuitions when you start a project. Any serious uncertainty analysis, however, requires a computer. Fortunately, there are several commercially available software products that make the probabilistic treatment of uncertainty quite straightforward. These tools obviate the need for the analyst to master the intricacies of implementing complicated Monte Carlo codes.

How should we choose and use software for quantitative modeling and risk analysis? At first blush, you might think that quantitative modeling is primarily a matter of number crunching. Certainly, that is a critical part of it; but if the ultimate purpose of the computation is improved insight into complicated situations, and enhanced understanding of the risks and opportunities, then modeling comprises a great deal more. The treatment of uncertainty (the primary focus of this book) is only one of many objectives.

In Chapter 3, we discussed the goals and motivations for risk and policy analysis, culminating in our suggested "ten commandments for good policy analysis" (Section 3.8). Good modeling tools cannot guarantee good analysis, but they can do a great deal to encourage and facilitate an analyst's pursuit of these ideals. Table 10.1 lists a set of design objectives and corresponding features for a modeling tool.

Note that only one of these eight objectives explicitly mentions uncertainty, although adequate treatment of uncertainty requires meeting the other objectives. More generally, a major source of uncertainty is about what the modelers intend to represent, and whether the model reflects their intention. Clarity of communication and representation of the model — at the qualitative,

---

* Chapter 10 of *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis*, M. Granger Morgan and Max Henrion,
Cambridge University Press, New York, 1990, reprinted in 1998.

mathematical, and numerical levels — reduces this uncertainty. As we discuss in the next section, computer models are, too often, an obstacle to rather than a vehicle for clear communication. In the following section, however, we illustrate how software tools can, in fact, facilitate scrutiny of model structure and assumptions, and communication of the key insights.

Table 10.1: Design objectives and corresponding features for a modeling tool for policy and risk analysis.

Design objectives:	**Features:**
Clear communication of the qualitative structure of a model	A visual display and editing of graphs of model structure
Integration of documentation with the model structure	Structured text documentation integrated with mathematical elements
Organization and management of complexity	A hierarchical modular structure, with tools to create, display, and edit the hierarchy
Exploration and understanding of model behavior	Interactive parametric and sensitivity analysis
Comprehensive treatment of uncertainty usable by people with modest training	A suite of methods for expressing, propagating, analyzing, and communicating uncertainty
Review and auditing of mathematical relationships	A simple, nonprocedural language with flexible data abstraction
Progressive refinement and model disaggregation	Hierarchical structure and array abstraction with variable dimensions and granularity
Collaboration among a team of modelers	Modular structure with defined module interfaces and controlled access

While we were writing this book, we were simultaneously designing, using, and evaluating a series of experimental software tools for quantitative policy analysis and the treatment of uncertainty. We designed these tools explicitly in response to the identified objectives. We conducted a series of experiments with these tools to explore, evaluate, and refine our evolving ideas of how software features can support these objectives. These tools included Demos (Henrion & Morgan, 1985; Henrion & Wishbow, 1987); HyperDemos, and Demaps (Wiecha, 1996; Wiecha & Henrion, 1987). This research culminated in the commercial release of the Analytica® software (Lumina Decision Systems, Inc;

1997; 1998), developed by Lumina Decision Systems, Inc., under license from Carnegie Mellon University. Since Analytica reflects much of our approach, we use it in this chapter to illustrate how a software tool can support these objectives. You can download the examples shown here along with an evaluation version of the software from the Web on www.Lumina.com.

10.1. Black Box Models as an Obstacle to Communication

How do we decide whether a policy model is credible? In Section 3.2, we compared policy modeling with the methods of the natural sciences, identifying the essential features of the scientific method for obtaining a deeper understanding of our universe. Scientists have adopted standard practices and institutions to support this process. Among these practices are empirical testing, complete documentation, analysis of errors and uncertainty, peer review, and open debate. Empirical testing is often impractical for policy modeling, because experiments would be unethical or impossible. The other practices are primarily to facilitate communication so that scientists can develop a consensus about what results are credible. These practices are quite applicable to policy analysis, but they are far from universally adopted. All too often, the computer models used for policy analysis are inconsistently documented and include little or no analysis of uncertainty. Thus, models often serve as a major obstacle to, rather than as a vehicle for, the clear communication required for effective peer review and informed debate. The impracticality of empirical testing makes use of the other practices especially critical as checks and balances on the credibility of policy analysis.

The lack of recognized standards for structuring and documenting policy models is part of the problem. Greenberger (1981) complained about *black box models*:

> "The typical policy model is not designed for ease of communication with the decision-maker. Even the model builder may have difficulty comprehending fully the essential working of the model. Ideally, the model should be presented to the policymaker not as a *black box* with assumptions and data feeding into the left and results coming out of the right, but as an *open box* whose inner workings are sufficiently simplified, exposed and elucidated to enable the policymaker to trace the chain of causality from input to output on at least an elementary and fundamental level."

Unfortunately, the situation has not improved as much as we might have hoped since these words were written. In many cases where quantitative modeling would be valuable, it is still avoided precisely because people fear that it will interfere with effective communication. Where quantitative modeling is effective, the analysts must put enormous effort into clearly communicating the model structure and assumptions. The models themselves — represented in spreadsheets or in specialist modeling languages — are rarely suitable as a

medium for communication. Most of the conversation must be at a higher level of abstraction, about the key variables, factors, and relationships, which are hidden among the arcane formulas of the model itself. Effective communication requires multiple levels of abstraction or detail, to suit the concerns, the level of interest, and understanding of each issue.

Typically, the model, documentation, and presentation of key insights each are developed in a separate software application. For example, the model is created in a spreadsheet; the documentation is written in a word processor, and the summary presentation uses a presentation manager. Software suites (e.g., Microsoft Office™ or Lotus Smartsuite™) allow linkages at the level of individual tables, charts, or blocks of text among documents. But, they do not support a high-level visual view that integrates the different representations, and that permits viewing at varying levels of detail and abstraction. Translations among these different representations requires substantial effort. More critically, it introduces many opportunities for translation and synchronization errors among the multiple versions during the iterative refinement process. All too often, the model, documentation, and presentations are prepared by different people at different times, providing ample opportunities for error and confusion.

The quotation at the start of this chapter suggests that, in fact, the primary value of computers is as an instrument for communication. In the rest of this chapter, we illustrate ways in which a modeling tool can satisfy Greenberger's desire for *open box* models – to facilitate, not inhibit communication.

10.2. Visual Display of Model Structure with Influence Diagrams

When people want to communicate with each other about complicated situations, they often draw diagrams with bubbles and arrows. Almost every discipline has formalized this process with specific interpretations of the bubbles and arrows; examples are organization charts, flow charts, PERT charts, semantic networks, entity–relationship diagrams, fault trees, decision trees, causal networks, belief networks, and systems-dynamics graphs. We have found *influence diagrams* an especially intuitive and effective notation for creating, understanding and explaining models for policy and risk analysis.

Influence diagrams were developed by the decision-analysis community as a representation for working with experts, decision makers, and stakeholders to express their knowledge, uncertainties, objectives, and decisions (Howard & Matheson, 1981; Shachter, 1988). Figure 10.1 shows an example influence diagram for a decision model involving the costs and benefits of reducing emissions into the atmosphere of TXC, a potentially toxic chemical.

We expect that reducing emissions of TXC will reduce the atmospheric concentration downwind, and hence reduce exposure, and possible health damage, and hence excess deaths among the population living downwind. The control technology to reduce emissions will have a cost which is a logarithmic

function of the reduction level. The objective is to minimize the expected total cost, defined as the control costs less the benefits. We quantify the benefits as the reduced mortality multiplied by the "value of a life", the monetary investment deemed worthwhile to reduce statistical mortality by one.
It is often convenient initially to position decision variables on the left and objective variables on the right of the screen, and then to fill in the intermediate and chance variables in between. Figure 10.2 shows an early stage during the creation of the diagram shown in Figure 10.1. This initial focus on decisions and objectives helps direct the group to think about other variables that mediate between the decisions and objectives — and to ignore the many variables and issues that, although they may be of interest, are unlikely to be relevant to the current problem. As participants suggest new variables, the analyst asks, "Could this factor change how our decisions influence the objectives?" If the answer is yes, the analyst adds the variable and asks the group to articulate how it might happen. If the answer is no, the factor is omitted. In this way, the focus on relevance to decisions and outcomes helps the group focus on what matters and avoid getting overwhelmed by needless complexity.

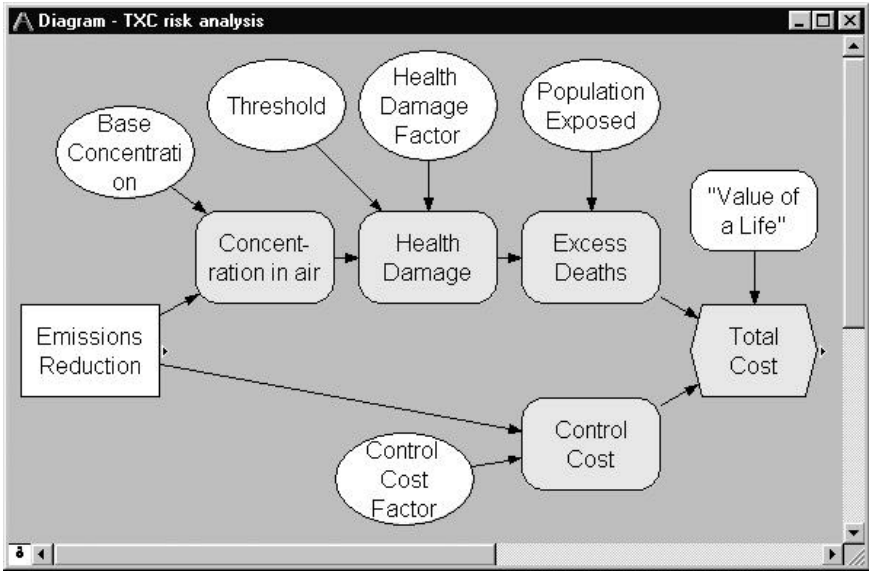


Figure 10.1. Influence diagram for a cost-benefit analysis of reducing emissions of a toxic air pollutant. The rectangle node, Emissions Reduction is a decision. The ovals are uncertain variables, and the hexagon is the objective to minimize, Total Cost.

The rectangular node, Emissions Reduction, depicts the *decision* – a variable that the decision maker (or makers) can affect directly. In this case, the

decision is by how much to reduce the emissions of TXC from the plant, including the option of reducing them by zero.

The oval nodes depict *chance variables* – variables that have an uncertain value that is not under the direct control of the decision maker. We express the value of each chance variable in terms of a probability distribution, fitting it to data or eliciting it from experts, using methods described in Chapter 7.

The rounded rectangle nodes are *deterministic variables* – that is, variables that are defined as deterministic values or as deterministic functions of their input variables. In some notations, deterministic nodes are depicted as ovals that have double outlines.

The hexagonal node on the right labeled Total Cost depicts the *objective* – a quantity to be optimized (either maximized or minimized, depending on how it is expressed). In decision analysis, this node expresses the utility – the objectives and values – of the decision maker. Decision theory prescribes that the decision maker seek the decision that maximizes the expected utility.

10.2.1. Influence Arcs

The arrows represent *influences.* Arrows into a deterministic or objective node indicate that the destination node is a function of the origin nodes. An arrow into a chance node expresses that the probability distribution on the chance node is conditioned on the values of the origin node. An arrow into a decision node is an *information influence*: It indicates that the value of the origin node will be known when the decision is made, and thus may affect that decision.

The absence of a direct arrow between two nodes expresses the belief that the variables are independent of each other, conditional on the values of those variables from which they do have arrows. The absence of an influence arrow, which specifies conditional independence, is actually a stronger statement than is the presence of an arrow, which indicates only the *possibility* of dependence.

An influence arrow does not necessarily represent a causal relationship from one variable to another. Influences express evidential relationships that need not be physical relationships. However, it is often a useful heuristic when eliciting influences, to ask about possible causal relationships among the variables.

10.2.2. Use of Influence Diagrams for Model Creation and Communication

Influence diagrams provide a simple notation for creating and communicating models by clarifying the qualitative issues of what factors need to be included and how they are related. An analyst can use influence diagrams when working alone, with a single decision maker, or with a group of interested people. Influence diagrams are comprehensible even to people not interested in the details of quantitative relationships. Later in the process, influence diagrams provide a simple intuitive representation for communicating the qualitative structure that reflects the underlying quantitative representations. Early

involvement of high-level decision makers in the structuring process, and continuity between the qualitative and quantitative representations, can help to ensure that these people understand and accept the model.

When working with a group, it is helpful to involve the entire group early on in drawing up initial influence diagrams. The group may draw diagrams on a physical whiteboard or may use software, such as Analytica, and project the image of the diagram onto a large screen. The skilled analyst will facilitate the group first in structuring objectives and identifying key decision variables that may affect these objectives. Next, the analyst asks about how the decisions may influence the attainment of the objectives, and elicits intermediate variables that mediate between the decisions and the objectives. The analyst may elicit this information by asking questions such as, "What other factors might affect these variables?"

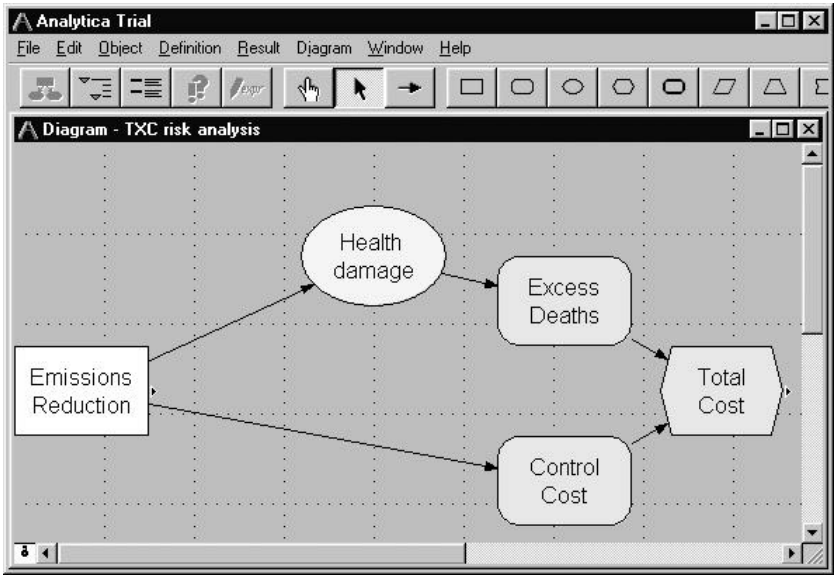


Figure 10.2. An early stage in the creation of the influence diagram, that is shown in final form in Figure 10.1. The analyst, using the Analytica diagram editor, is starting to fill in variables between the decision on the left and the objective on the right.

At first, the analyst may include any variable that somone thinks is relevant. Later, the diagram is pruned. When a creative group works on a complex problem the diagram will often become a tangled web. An advantage of drawing on a computer is that it is easy to clarify the developing diagram, rearranging the variables to reduce intersecting influences and grouping closely related variables. With hierarchical influence diagrams, it is easy to move such

groups into submodules, to simplify the main diagram, as we describe in Section 10.4.

10.3. Integrating Documentation with the Model Structure

An important objective is to integrate the mathematical relationship (variable definitions) and the documentation explaining the meaning of those relationships. With this integration, the model authors can specify and update the computational structure and documentation at the same time, reducing the chance that the two will become inconsistent. Reviewers or auditors can rest assured that the model structure that they see truly reflects the computational structure. Whenever they wish, they can dive down from the visual diagram to inspect the details of mathematical formulas and numeric values.

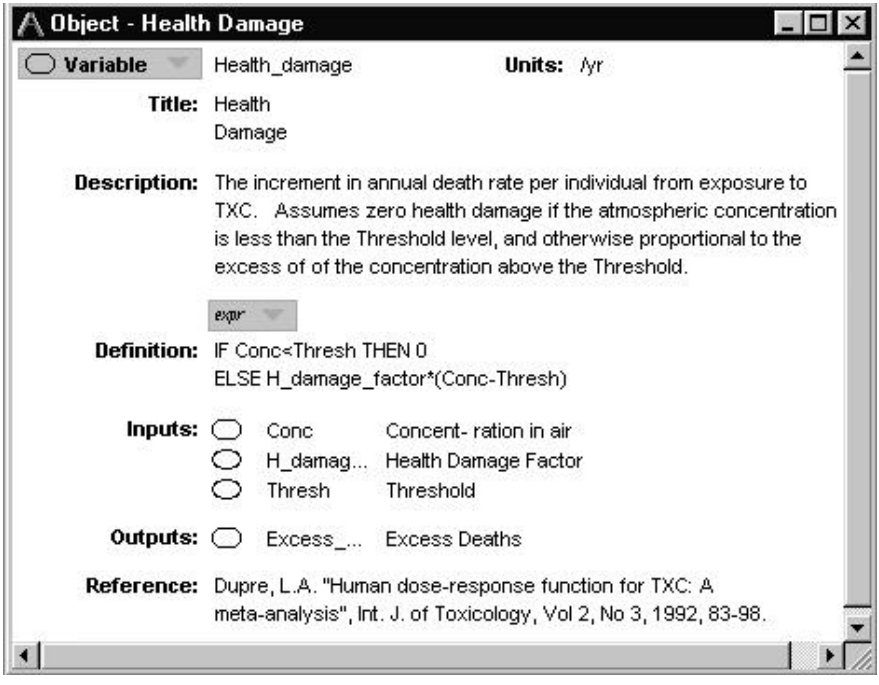


Figure 10.3 An Object card showing detailed information about the variable, Health Damage. The inputs are the variables that appear in the definition above, and also linked in by influence arrows in the influence diagram in Figure 10.1. The reference field may give a bibliographic reference, as in this case, or a name or contact for the source of the relationship, number, or probability distribution.

Figure 10.3 shows an Object card displaying the attributes of a particular variable from the diagram in Figure 10.1. The card shows the key attributes of the variable, including its type, identifier, title, units, description, definition,

inputs, outputs, and reference. The description specifies what the variable represents. The definition is the formula that Analytica uses to compute the value of the variable. It may contain a simple number, a probability distribution, a table, or, as in this case, an expression for computing the value from other variables.

The inputs and outputs lists reflect the dependencies in the definitions, and, also the influence arrows on the diagram (Figure 10.1). Analytica automatically maintains consistency between the diagram and definitions. Drawing an arrow from one variable to another will put the first variable into the inputs list of the second. When you are composing the definition, you can select from the inputs list to enter the variable into the definition. Alternatively, if you type the name of a variable into the definition, Analytica will add a corresponding arrow from that variable to the diagram.

10.4. Organization of a Model as a Hierarchy of Modules

A large model may have hundreds or even thousands of variables, far too many to show on a single diagram. Analytica extends the traditional influence-diagram notation with the addition of a module node. A *module node* opens up to another influence diagram, containing its own variables and modules, as illustrated in Figure 10.4. Using modules, you can organize a model into a hierarchy of modules. When a diagram gets overcrowded, you can simplify it by creating a new module node and dragging a set of interrelated variables into that module node. A module is depicted as rounded rectangle with a thick outline. Analytica displays influence arrows to or from a module node to represent influences to or from variables inside the module.

The module hierarchy in Figure 10.4 and 10.5 are views of the Tracking and Analysis Framework (TAF) model (Henrion, Sonnenblick, & Bloyd, 1997). TAF models the effects of changes in power plants emissions of sulfur oxides, nitrogen oxides, and other air pollutants on human health, acid rain effects on soils, streams, and lakes, and atmospheric visibility, in North America. TAF provides an integrated assessment of actual and proposed reductions in emissions of atmospheric pollutants in North America with a focus on the Clean Air Act Amendments (CAAA) of 1990. It was developed for the National Acid Precipitation Assessment Program. It compares a range of emissions control and regulatory policies against a baseline without CAAA. For each scenario, it projects changes in pollutant emissions over 30 years, for sulfur oxides, nitrogen oxides, and the resulting ambient air quality, and dry and wet depositions. It evaluates each scenario by comparing the cost of reducing emissions against the environmental benefits, including changes in human health effects, acidic deposition on soils, lakes and streams, and visibility. The framework compares costs and benefits, using contingent valuation and other methods to quantify the benefits in dollars terms. The complete TAF model comprises 190 modules, containing a total of over 1500

variables. Each module is based on a detailed analysis of the science – in some cases, comprising a simplified model fitted to results from more complex models, such as for atmospheric transport. The hierarchical structure helps the model's many authors and reviewers to manage and understand the model.

10.4.1. Navigation of a Module Hierarchy

Analytica provides a number of aids to help people navigate large models like TAF. Figure 10.5 shows the module hierarchy for part of the TAF model in the form of an outline window, which works like a standard hierarchical file viewer. You can click on the small triangle next to a module name to list the modules or the variables inside it. A double click on a module or variable in the outline opens its corresponding diagram or object window.

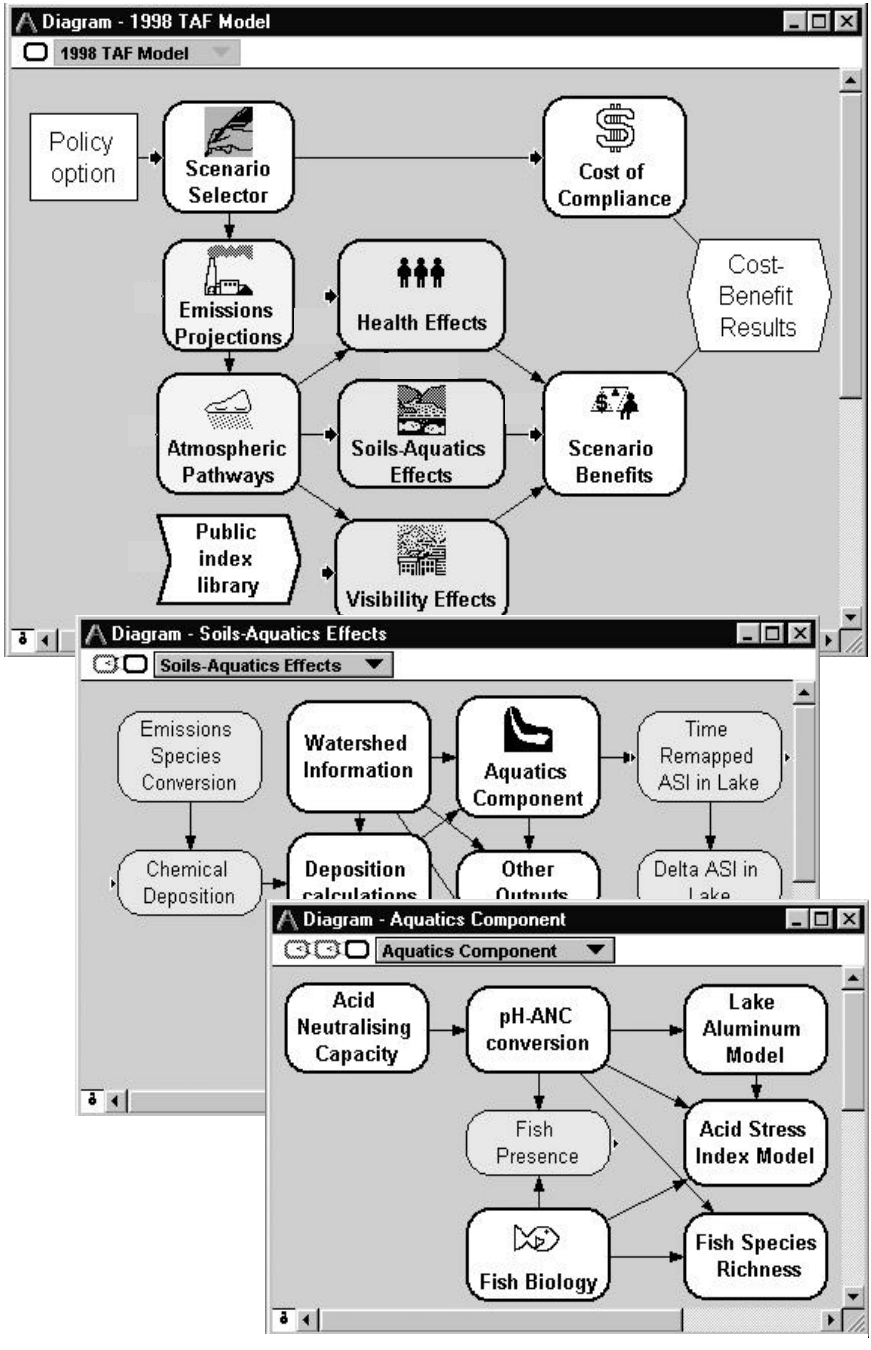


Figure 10.4. Three influence diagrams showing three levels of the module hierarchy for the TAF model. A mouse click in a module opens a diagram to show lower level details.

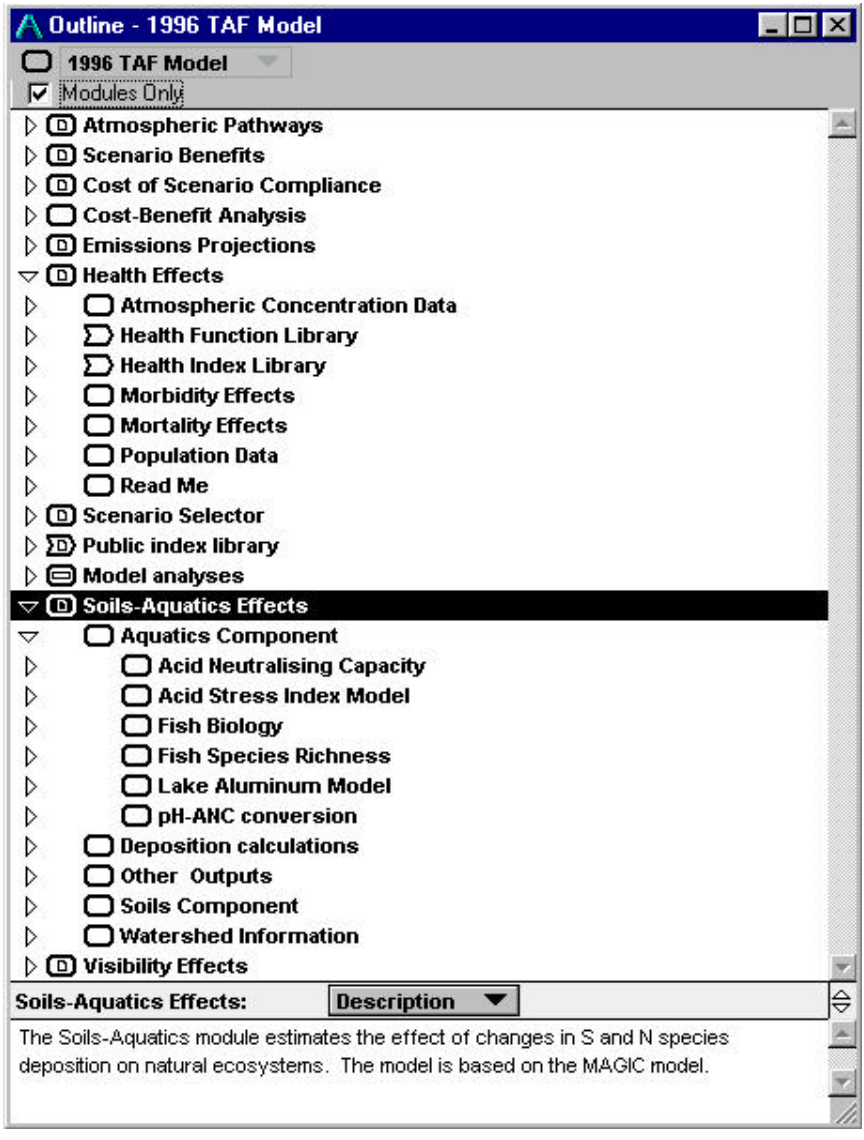


Figure 10.5. An expandable outline view that displays the module hierarchy of the TAF model. A click on a little triangle next to a module name will display the module's contents as an indented list. The TAF model comprises a six-level hierarchy; of which up to three levels are shown here. The panel at the bottom displays the description of the module Soil–Aquatic Effects module that is highlighted above.

There are several methods to trace dependencies across modules. Pressing the mouse button to the left side of a node shows a list of the variables on which

that node depends. Selecting a variable from the list displays its parent diagram, and highlights the variable. Similarly, pressing to the right of a node displays the output variables that depend on that variable. A node whose title in italics is an *alias node:* a variable in another module that is visible and accessible here. Aliases provide a convenient way to show linkages or relationships between different modules.

Another option, visible at the top of each diagram window in Figure 10.4 is the nesting bar, which helps keep users oriented in highly nested modules. The number of module icons along the left of the bar indicates how deep this module is nested in the module hierarchy. The nesting bar also shows a popup menu listing the parent, grandparent, and earlier ancestor modules that enclose this module, giving the user a quick orientation, and allowing her to jump up the hierarchy one or several levels.

10.5. Tools for Modeling and Analysis of Uncertainty

Analytica illustrates a range of methods for expressing uncertainty using probability distributions, for propagating those uncertainties through a model, and for displaying and analyzing uncertain results. Analytica represents each uncertain value internally as a sample of values drawn at random from its underlying probability distribution, using Monte Carlo or Latin hypercube sampling methods. It provides a wide range of methods for displaying the probability distributions, including probability intervals, density, cumulative distributions.

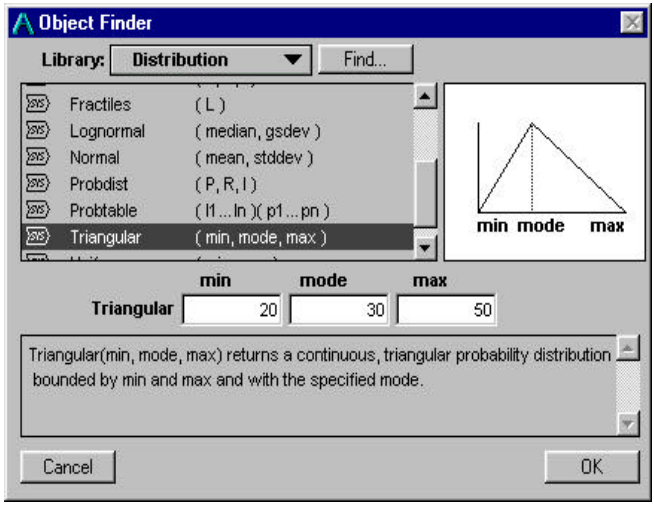


Figure 10.6. A dialog box to select a probability distribution and assess its parameters. The built in distribution library provides a range of standard

continuous and discrete parametric distributions, as well as several ways to express custom distributions.

10.5.1. Selection and Assessment of Probability Distributions

Like most software tools that provide Monte Carlo simulation, Analytica provides a built-in library of probability distributions. It includes standard continuous parametric distributions, such as beta, normal, lognormal, triangular, and uniform, and discrete distributions, such as Bernoulli and binomial. Figure 10.6 shows a dialog box from which the user can select a probability distribution. It also provides a variety of ways for her to specify custom distributions, including a general probability table (discrete distribution, conditioned on other discrete variables), set of fractiles, and sets of points on the cumulative or density functions.

Analytica also contains libraries with additional distributions, and a language in which the user can define further types of distributions and other functions. By writing new functions, users can create libraries to custom tailor the application for particular classes of applications.

10.5.2. Assessment of Dependent Distributions

Standard Monte Carlo plug-in packages for spreadsheet applications, such as Crystal Ball™ (from Decisioneering, Inc) and @Risk™ (from the Palisade Corporation), let the user specify a correlation between two uncertain quantities as a way to express uncertain dependence. Like Crystal Ball and @Risk, Analytica provides functions to define a variables as a probability distribution with a specified rank-correlation with another variable. It also provides library functions to specify an array of uncertain variables with specified serial correlations over time or other dimension. When there are more than two correlated variables, you may express the dependence structure among the variables with a rank-correlation matrix. Analytica's library function implements the method of Iman and Conover (1982b) for generating a vector of variables with specified rank-order correlation.

When the uncertainties are estimated from data, we can often estimate the needed correlations from the same data. When the uncertainties are assessed as expert judgments, the expert may also judge these correlations subjectively. Unfortunately, the experimental literature suggests that people find it hard to express their knowledge reliably and coherently in the form of correlations, as we discussed in Section 6.4.6. When there are more than two dependent variables, assessing the full correlation matrix is particularly difficult. First, the number of correlations increases with square of the number of variables. If the assessor does not judge correlations among all pairs of variables, then we must determine how to fill in omitted correlations. Second, the correlation matrix must be positive definite. For example, correlations between x and y, and between y and z constrain the correlation between x and z. Few people have

intuition sufficiently sharp to assess a set of subjective correlations that will satisfy this constraint constraint.

Instead of using the abstract notion of correlation, it is often easier and more effective to model the underlying reasons for the dependence. For example, suppose that we are interested in modeling the dependence between the Development cost and Time to market for a proposed new product (in Figure 10.7.) Instead of trying to assess a subjective correlation between these two variables, we can extend the model to depict the reason for the dependence. If these variables are dependent because they both depend on how long the product takes to develop, we can include Time to develop in the model. In the diagram on the right of Figure 10.7, we model Development cost as the product of the Development cost per month, and the Time to develop. We model Time to market as the sum of the Time to develop and the Time to productize. Because both Development cost and Time to market depend on Time to develop, the probabilistic simulation will automatically compute them as dependent, with a correlation determined by the relative contributions of uncertainty from their common and independent inputs. The influence diagram clearly depicts the reason for their dependence at a qualitative level.

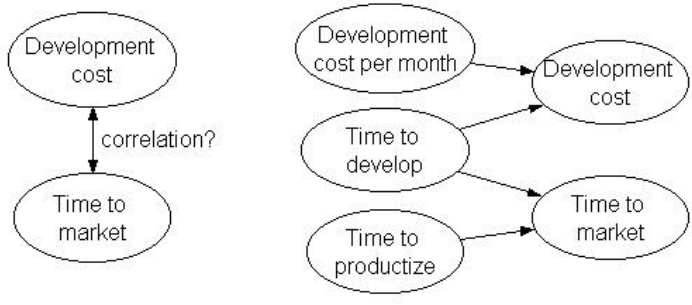


Figure 10.7. Two approaches to expressing dependence. The diagram on the left expresses the, dependence between Development Cost and Time to Market for a new product as a correlation. The diagram on the right expresses their relationship by their common dependence on Time to develop. In the second method, the dependence arises from the model structure and avoiding the need for explicit judgment of a correlation.

10.5.3. Display of Probability Distributions

In Chapter 9, on the graphic communication of uncertainty, we suggested that you suit the method of displaying uncertainty to the kinds of uncertain information that you want to convey and to the interests and skills of audience. Different representations are useful for different purposes and different audiences. Analytica provides six methods to display probabilistic information about uncertain quantities. Each method can be shown as a graph or table of

numbers, by a toggling buttons in the top left of the Result window. Figure 10.8 and 10.9 show the corresponding cumulative probability distribution and probability density function for the Base concentration variable from the TXC model, shown in Figure 10.1.

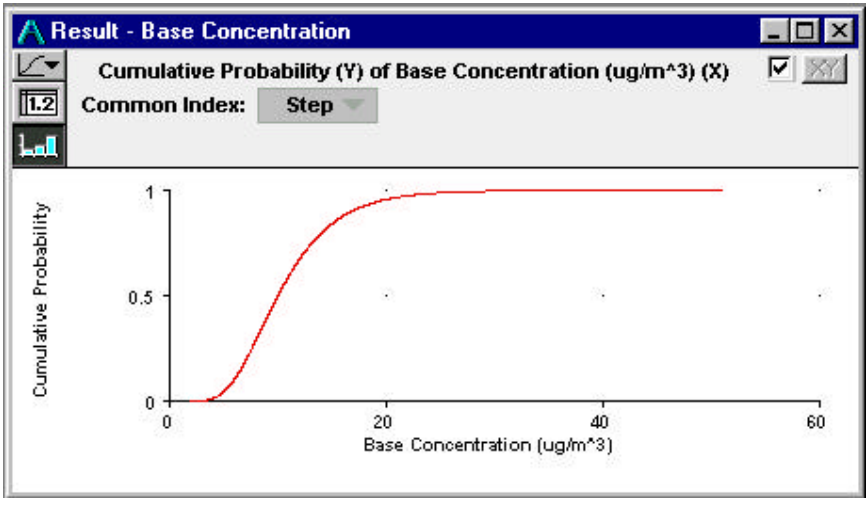


Figure 10.8. The cumulative probability distribution function for the Base concentration from the TXC model, whose influence diagram is depicted in Figure 10.1 This distribution is estimated from a lognormal distribution.

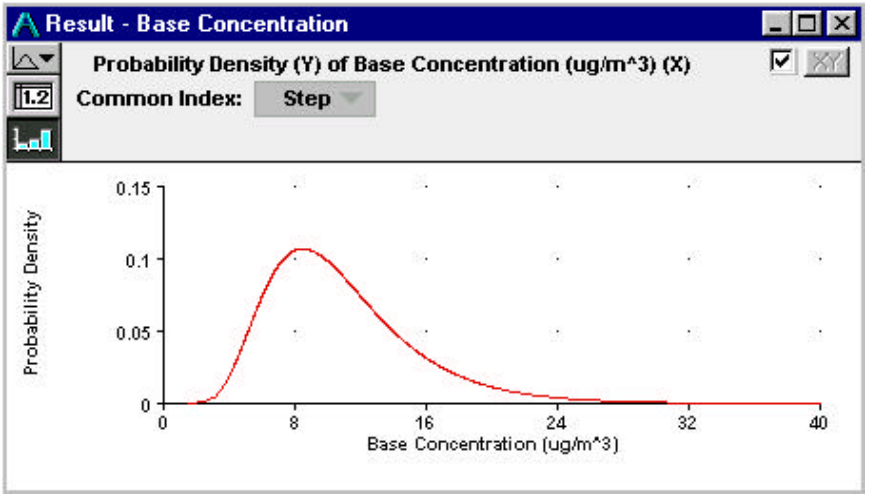
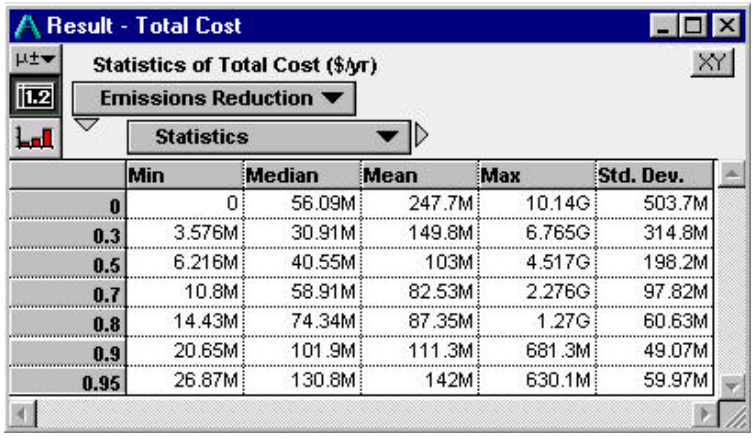


Figure 10.9. The probability density function for the Base concentration from the TXC model corresponding to the distribution in Figure 10.9.

It is often useful to be able to display several distributions on the same graph or table. Figure 10.10 shows a table of statistics for the probability distribution of the Total cost, conditional on each level of Emissions reduction, including the minimum, median, mean, maximum, and standard deviation. Figure 10.11 graphs the mean (or expected) Total cost as a function of Emissions reduction. The Total cost is large for a small Emissions reduction because the many fatalities due to the high emissions. Total cost also increases for higher Emissions reduction, because of the increasing cost of emissions control;. Hence the Emissions reduction resulting in minimum expected Total cost is somewhere in between, at about 0.7.



	Min	Median	Mean	Max	Std. Dev.
0	0	56.09M	247.7M	10.14G	503.7M
0.3	3.576M	30.91M	149.8M	6.765G	314.8M
0.5	6.216M	40.55M	103M	4.517G	198.2M
0.7	10.8M	58.91M	82.53M	2.276G	97.82M
0.8	14.43M	74.34M	87.35M	1.27G	60.63M
0.9	20.65M	101.9M	111.3M	681.3M	49.07M
0.95	26.87M	130.8M	142M	630.1M	59.97M

Figure 10.10. A table of statistics for the probability distributions of Total cost, for each level of Emissions reduction.

Figure 10.12 shows probability bands (fractiles of the distribution) for Total cost, also as a function of Emissions reduction. It shows that the uncertainty about Total cost (e.g. the range between 0.25 and 0.75 fractiles) increases substantially for a low value of Emissions reduction – because of the substantial uncertainty about the Health damage function. Uncertainty for large emissions reduction is much smaller – because there is low mortality and the much lower uncertainty about the control costs.

If you simply want to ignore the uncertainty, Analytica will compute and display a *mid* value for any quantity, computed deterministically from the median of each input distribution. The mid value is useful for initial checking that the model is working correctly, especially in a large model in which the evaluation time for uncertain variables may be appreciable. If you were conducting a deterministic analysis, looking only at the mid value, you would see the single line, close to the median in Figure 10.12. Note that the optimal decision, based only on this deterministic analysis, is an Emissions reduction of about 0.3. This example illustrates how an explicit probabilistic analysis of uncertainty may lead to a very different decision recommendation than a deterministic analysis, in this case, an Emissions reduction of 0.7 instead of 0.3. (See Chapter 12 for an extended examination of this phenomenon.)

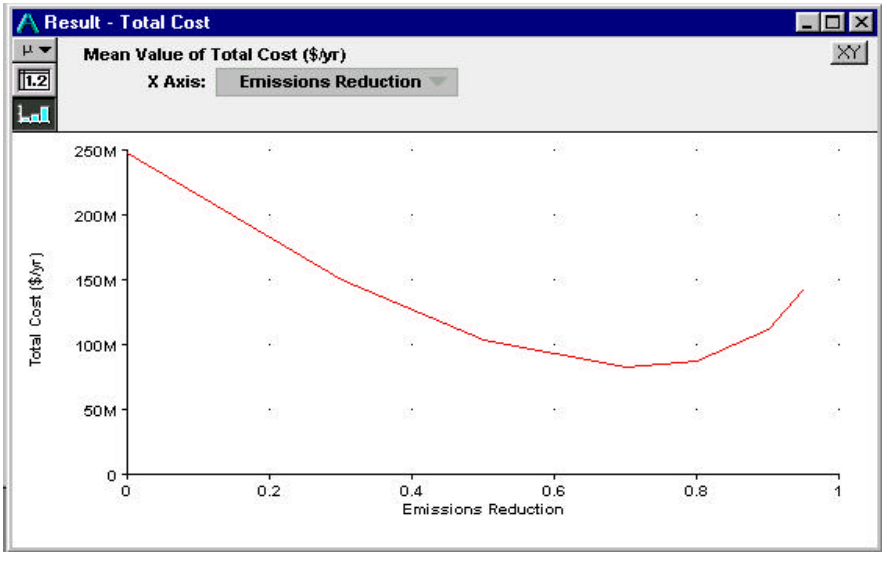


Figure 10.11. The mean Total cost as a function of the Emissions reduction. The mean is estimated from the average of the Monte Carlo sample of the cost for each emissions level. An Emissions Reduction of about 0.7 gives the minimum expected Total Cost.

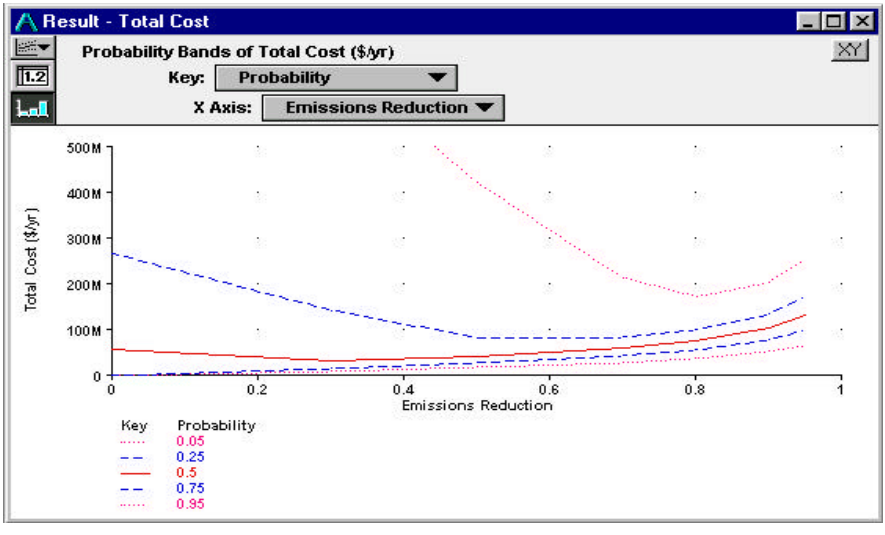
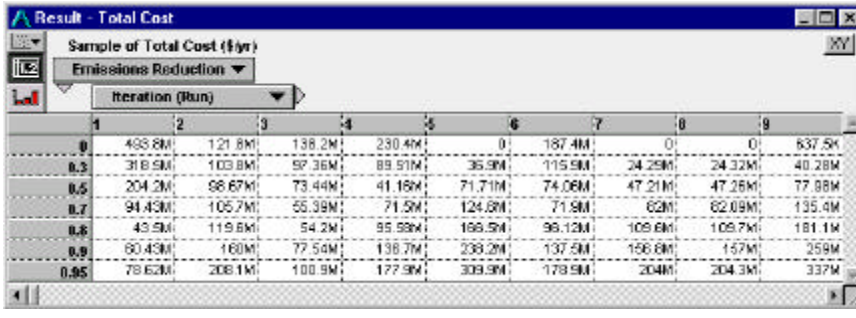


Figure 10.12. Probability bands of the Total Cost as a function of the Emissions Reduction. The inner dashed pairs of lines and outer dotted pair of lines enclose 50% and 90% probability bands, respectively. The uncertainty is much larger at lower emissions reductions (higher emissions) because the cost is dominated by the highly uncertain exposure to and toxicity of TXC.

The final method to display the uncertainty is to shows the underlying sample values generated by the Latin hypercube or Monte Carlo simulation, shown in a table in Figure 10.13. Each column represents a random sample (iteration or run) of the model. This level of detail will be excessive for many users, but it is accessible for scrutiny when required. It is sometimes useful for identifying selected extreme scenarios or sample values.



	1	2	3	4	5	6	7	8	9	
0	493.8M	121.8M	138.2M	230.4M	0	187.4M	0	0	637.5K	
0.3	318.5M	103.8M	97.36M	89.51M	36.9M	115.5M	24.25M	24.32M	40.28M	
0.5	204.2M	98.67M	73.44M	41.16M	71.71M	74.06M	47.21M	47.26M	77.88M	
0.7	94.43M	105.7M	55.39M	71.5M	124.6M	71.9M	62M	62.09M	135.4M	
0.8	43.5M	119.6M	54.2M	95.98M	166.5M	96.12M	109.6M	109.7M	181.1M	
0.9	60.43M	160M	77.54M	138.7M	238.2M	137.5M	156.8M	157M	259M	
0.95	78.62M	208.1M	100.9M	177.9M	309.9M	178.9M	204M	204.3M	337M	

Figure 10.13. A table showing the underlying random sample for the Total Cost for each Emissions reduction from which Analytica estimates the probability distributions. The Iteration (Run) variable identifies each sample from 1 to the number of samples.

10.5.4. Methods of Propagating Uncertainty

A design goal for Analytica was to provide simulation-based methods for propagating and analyzing uncertainty, with minimum need for the user to be concerned with the details of the methods. For many applications, the user can use the default sample size and methods without having to worry about the simulation issues discussed at length in Chapter 8. For cases in which the user wants to change the default values — for example, to expand the sample size to obtain smoother distribution curves — the user can do so from the dialog box shown in Figure 10.14. By clicking on options, the user can change, for example, the sample size, the simulation method, or random-number generator.

Analytica uses *midpoint Latin hypercube* sampling as its default sampling method, since that method usually gives a a faster convergence than simple Monte Carlo or random Latin hypercube sampling, as explained in Section 8.5. The software provides the options of random Latin hypercube sampling for those very rare cases, described in Section 8.5.6, where the midpoint method can be inaccurate, and of simple Monte Carlo sampling for cases where users want convenient statistical properties to estimate sampling errors. It also provides three random sampling methods, explained in the *Analytica User Guide.*

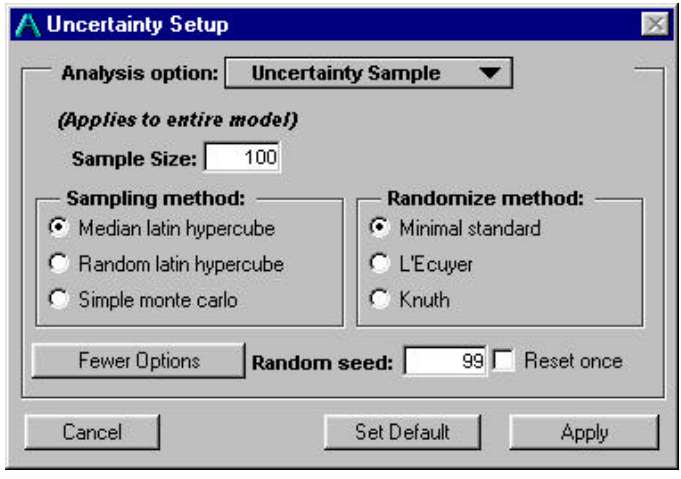


Figure 10.14. Dialog box to set the sample size, simulation method, and random-number generator.

10.5.5. Uncertainty Analysis

Analytica lets users perform *importance analysis*, a convenient method of uncertainty analysis. It computes the rank-order correlation of the selected output with respect to each chance variable in the model. The user selects the output of interest; in the case of the TXC cost-benefit model, she chooses the Total cost variable. Figure 10.15 shows the results of uncertainty analysis for an Emissions reduction of 0.7. Figure 10.16 shows the uncertainty-analysis graph of the importance of each chance variable as a function of emissions-reduction level. This plot indicates that the Control cost factor, which expresses uncertainty about the control costs, contributes the lion's share of the overall uncertainty. If we wanted to refine this analysis, by obtaining more information or elaborating the model, it might therefore make sense to focus on the Control cost factor.

The relative importance of uncertain variables depends on the level of the decision. Figure 10.16 graphs the importance (rank-order correlation) of each uncertain input to the Total cost, as a function of the Emissions reduction, plotted along the X axis. The importance of the Health Damage Factor is largest for low Emissions reduction, and decreases with higher Emissions reduction, because at higher Emissions reduction there is little exposure and so the Health Damage Factor has little effect on human mortality and hence cost. Conversely, the Control Cost Factor increases with Emissions reduction, becoming by far the most important contributor to the uncertainty, as also shown in Figure 10.15 for Emissions reduction of 0.7.

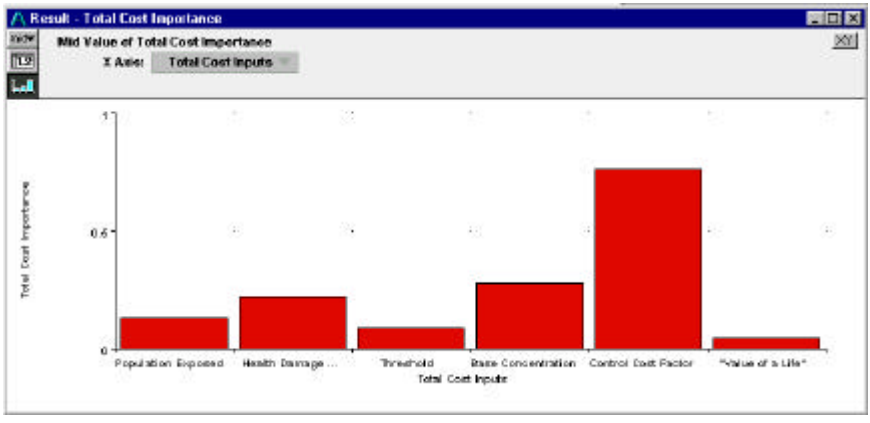


Figure 10.15. Results of an importance analysis showing the rank-order correlation between the Total cost and each chance variable in the TXC model, for an Emission reduction value of 0.7. At this level, the Control Cost Factor is by far the most important in terms of its contribution to the uncertainty in the Total Cost, because few people are likely to be exposed to a toxic level of TXC and the mortality from exposure is therefore low.

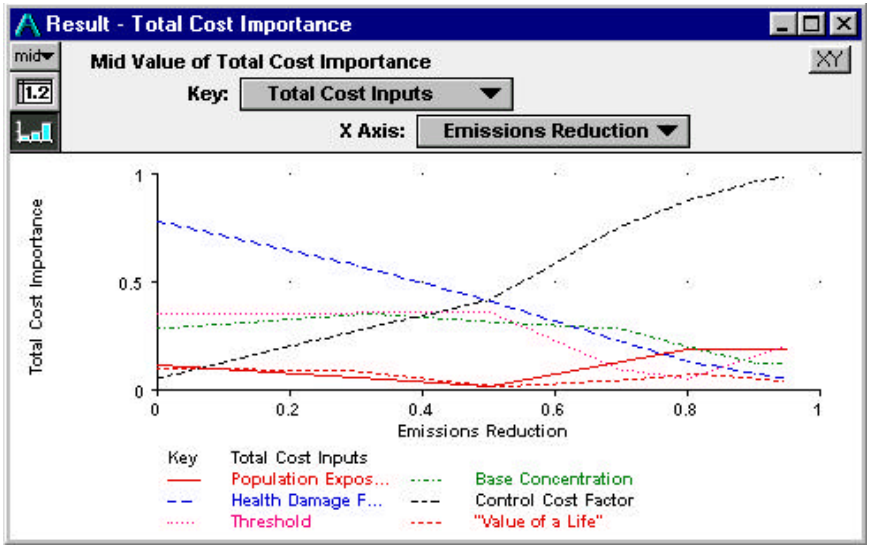


Figure 10.16. A graph of the importance of each uncertain variable versus the level of Emissions reduction. For low Emissions reduction, the Health damage predominates. For high Emissions reduction, the Control Cost factor predominates, as we might expect, since there will be little exposure to the toxic pollutant, TXC.

10.6. Progressive Refinement and Array Abstraction

When building and analyzing a model, ideally you should be thinking at whatever conceptual level is most appropriate for the problem, which is not necessarily the level of spreadsheet cells or program loops. The objects and relationships in the model should reflect directly the entities being modeled. Visual representations like hierarchical influence diagrams offer one way to bring the computer representation closer to the conceptual level. The influence diagram hierarchy offers a natural way to represent a problem at several levels of detail, and is helpful in supporting progressive refinement as an approach to model development. A second set of issues of critical importance in developing models concerns the *dimensions* and *granularity* of the values represented.

Models for substantial applications typically involve vector or array values, indexed by various *dimensions*, such as time periods, spatial locations, products, chemical species, types of people, scenarios, and so on. The *granularity* of each dimension may coarse or fine. Time may be modeled by decade, year, day, or second. Geographic locations may be modeled by nation, state, county, or degree of longitude and latitude. Choosing the dimensions and their granularity is one of the most important tasks for the modeler. If the granularity is too coarse, the model will be too inaccurate to be useful. If it is too fine, the tasks of model building, data collection, or computation will be intractable. It is often hard to know, ahead of time, what level of granularity will be necessary or practical.

Using progressive refinement, you should start out with a simple model using few dimensions and coarse aggregation. Initial sensitivity analysis of the simple model will reveal which variables and dimensions are critical, and guide your choice of directions in which to refine the model. Unfortunately, traditional modeling tools, whether spreadsheets or programming languages, make it quite difficult to change a dimension's granularity or to add a new dimension. Such changes typically require major surgery on the model, making it impractical to perform extensive progressive refinement. Even extending the time horizon by a few years can be painful in a spreadsheet if more than a single table is indexed by time. Adding a dimension to several variables may increase the model size and the entire modeling effort by an order of magnitude.

10.6.1. Array abstraction

The concept of *array abstraction* provides a solution to this problem. The goal of array abstraction is to define an array data type, to which operators and functions can be applied directly, instead of having to apply them to each individual spreadsheet cells or array elements. Ideally, the expressions or formulas involving array variables should not even have to mention the dimensions of those variables, unless the dimension is specifically relevant to

the operation. For example, if variables *A*, *B*, and *C* are defined as arrays, each with the same dimensions, it should be possible to write and evaluate

$C := A + \text{Log}(B)$

and to have the plus operator "+", Log function, and assignment operator ":=" generalize to repeat the operations over all the elements of the arrays. Unlike simple spreadsheet usage, there is no need to repeat the formula for each cell. Unlike common computer languages, there is no need to write loops to iterate over all elements of the arrays. One advantage of array abstraction is that you need to write the formula only once, no matter how many elements or dimensions the variables contain. A second advantage is that you do not need to update the formula if the dimensions are modified.

Some early computer languages, notably APL, introduced a limited form of array abstraction. More recent object-oriented languages provide the means for creating abstract arrays, and libraries that do so. But, most are limited to a fixed number of dimensions, typically one or two. Analytica, drawing on our research in the development of its predecessor, Demos, provides a form of array abstraction termed *Intelligent arrays*™ that considerably extends its power and flexibility for arrays with arbitrary numbers of dimensions.

10.6.2. Index variables

The key to Analytica's Intelligent arrays is the definition of *index variables* — variables that identifies a dimension of a table or array value. Figure 10.17 shows a part of a module from the TAF model (introduced in Section 10.4) which contains several *index variables*, each depicted by a parallelogram node. Example index variables in the TAF model include

- Seasons: Winter, Spring, Summer, and Fall
- Years from 1980 to 2020
- Years by groups of five, for more aggregate analysis
- 60 source regions including the states of the USA, some Canadian provinces, and Northern Mexico, which contain emission sources for pollutants.
- Policy options, including a base and new proposed options for emissions reductions
- Ambient species of pollutant: sulfur dioxide, sulfates, nitrogen dioxide, and nitrates

Each variable in the TAF model may be indexed none, one, or more of these index variables. *Run* is a special index variable that identifies iterations of the Monte Carlo or Latin hypercube simulation from 1 to the number of samples, as illustrated in Figure 10.13. It is treated much like any other index.

To refer to a dimension in an expression, you use the name of that dimension's index. Unlike standard programming languages, you do not need to remember which dimension is the inner, outer, or other dimension. For example,

```
Annual_cost:=Sum(Seasonal_cost, Seasons)
```

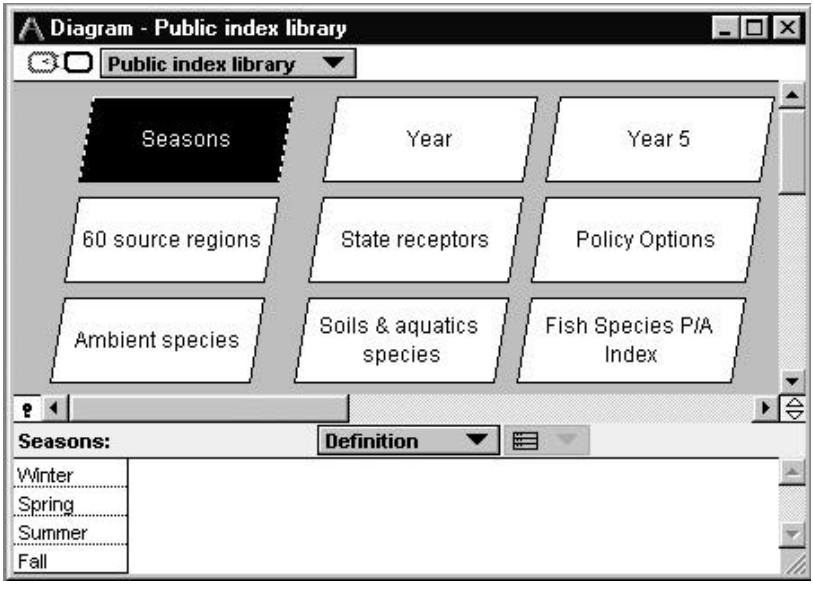


Figure 10.17. A module containing index variables from the TAF model, each shown as a parallelogram node. These indexes are used to identify the dimensions of single and multidimensional variables. The attribute panel below the diagram window displays the definition of the Seasons index as the four seasons. This index is used in emissions and atmospheric transport variables that change by season.

computes the Annual cost by summing the Seasonal costs over the Seasons index. Conversely, if a dimension is irrelevant to the expression, it need not be mentioned in the expression, even if variables in the expression are indexed by that dimension. Therefore, if Seasonal costs is indexed by other dimensions, such as Years, States, and Pollutant species, the relationship does not need to mention them. Those dimensions will pass through the expression, creating an Annual cost indexed by each dimension (except Seasons).

The advantage of this kind of array abstraction is that, if the modeler adds or removes dimensions, the relationship is unaffected and does not need to be updated. If you extend a dimension by adding new elements — for example, extending the horizon year for the Years variable — you need to update only the new cells in those tables that require values specified for each year. Variables that are computed will automatically generalize appropriately. If you remove elements, you do not have to make other changes to compensate. This flexibility makes it much easier to change the dimensionality and granularity of a model and to experiment with alternative schemes, and hence to use progressive refinement to develop your model.

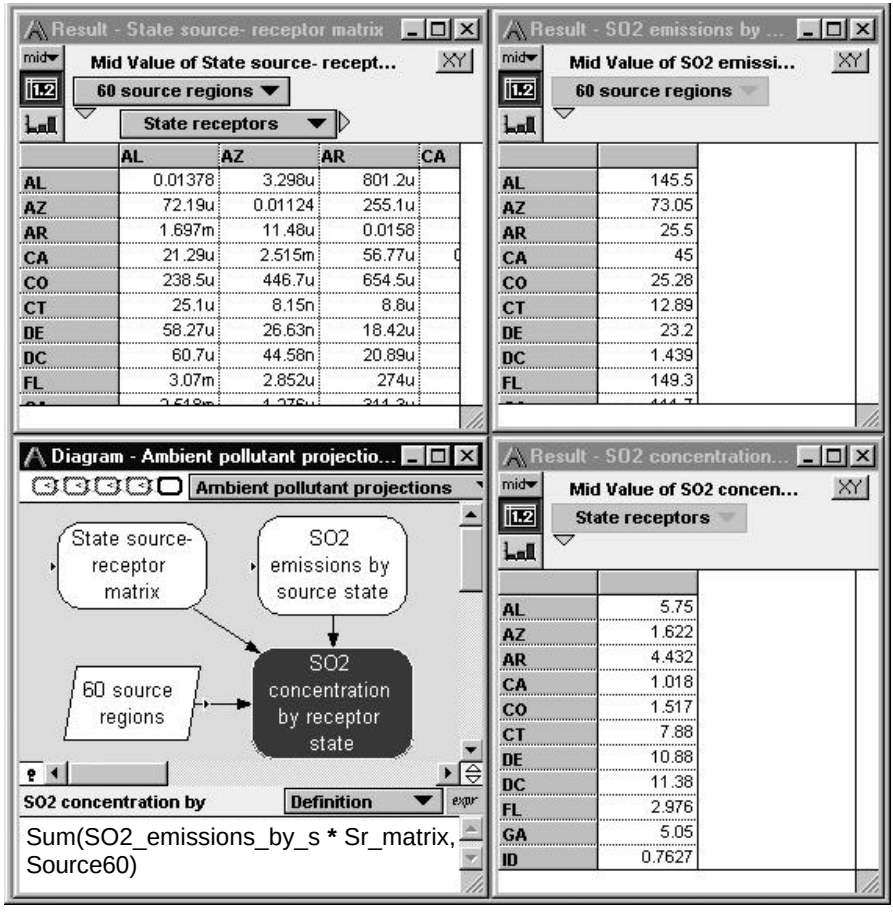


Figure 10.18: Part of the TAF model, showing use of array abstraction. The highlighted node, SO2 concentration by receptor state, shows its definition in the panel under the diagram in the lower left. The formula computes the product of Sr_matrix -- the State source-receptor matrix part of which is shown in the top left – and the SO2_emissions by source state -- part of which vector is shown in the top right -- summed over the Source60, the 60 source regions. The result gives the value of SO2 concentrations by receptor state, shown as the vector in the lower left.

Figure 10.18 illustrates these principles of array abstraction in a calculation of the sulfur dioxide SO2 concentration in each State receptor (in the lower left corner), based on the State source-receptor matrix (in the upper right corner) and the SO2 emissions vector by Source region (in the upper left corner). The diagram in the lower left corner shows the variables, and the index 60 source regions (Source60), and the actual expression used to compute the SO2 concentration below the diagram. The expression mentions Source60,

since it sums the product of the SO2_emissions and Sr_matrix (source-receptor matrix) over the sources to obtain the total SO2 concentration for each State receptor.

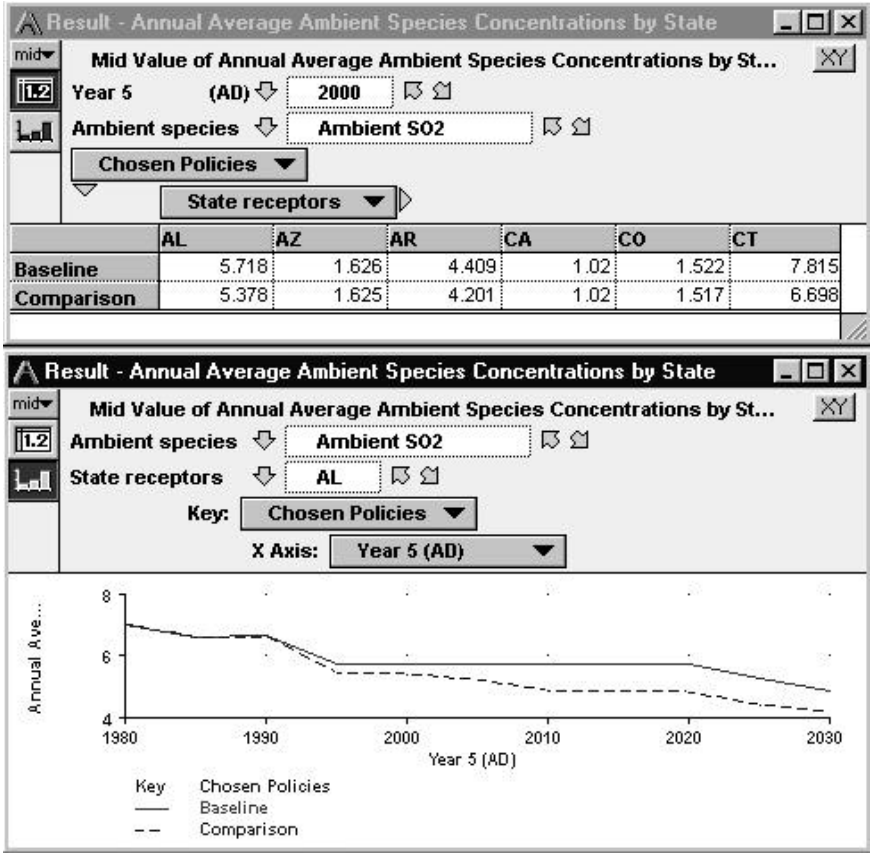


Figure 10.19: Two views of the result for the Annual Average Ambient Species Concentrations by State, showing different two-dimensional slices from a four-dimensional table. The top view shows a table with the projected concentration for Ambient SO2 in the year 2000, for each state receptor for Baseline and Comparison policies. The lower view shows a graph, sliced from the same table, comparing the two policies by Year for the state of Alabama (AL), also for Ambient SO2.

Figure 10.19 shows the result of a similar, but more generalized, calculation as the Annual Average Ambient Species Concentration by State. In this case, the source-receptor matrix was also indexed by Ambient species – that is, it contains a matrix for each of four pollutant species. The Emissions was also indexed by Ambient species, as well as Year 5 and Chosen policies – in other words, it contains emissions of each species over time for two policies

– Baseline and Comparison. Hence, the resulting Ambient Species Concentrations has four dimensions, indexed by Year 5, Ambient Species, Chosen Policies, and State receptors, respectively. The upper part of Figure 10.19 shows a two-dimensional table over Chosen Policies and State Receptors, sliced out of the four-dimensional table. The lower part shows a graph over time (Year 5) for each of the two Chosen Policies. It is a different two-dimensional slice.

The Result windows allow the user to use the index menus to select any two dimensions to display from the full list of dimensions, and to pivot the table or graph, as desired to show the results in the most interesting form. The internal representation identifies no dimension as the inner or outer index, the rows or columns. The choice of rows and columns for the display is up to the user.

10.6.3. Comparison of Array Abstraction in Analytica™ and Spreadsheets

We can compare Analytica's scheme for array abstraction with spreadsheet applications, such as Microsoft Excel™ or Lotus 123™. Copying formulas using relative addresses in a spreadsheet allows you to expand a table from one cell to a row of cells, or from one row to many rows. You can also name a range, that is a reference to a two-dimensional collection of cells. However, a spreadsheet range will not necessarily expand or contract correctly if you change the dimensions. Ultimately, each cell must have its own formula, resulting in massive redundancy in the representation. It requires substantially more effort and allows more scope for error during model authoring, and, equally important, during audit or review. If you want to expand a dimension — such as by extending the time horizon – you must update explicitly every table that is indexed over time. If you miss one table, the model will be corrupted. If you want to add a second dimension, you must clear space for each array.

In an early experiment, we translated a decision-analysis model written in FORTRAN— the ADEPT acid-deposition model (Balsom, Boyd, and North, 1982) — into Demos, the predecessor of Analytica, which used the same method of array abstraction. The FORTRAN model contained 1640 lines, which reduced to 111 lines of Demos code — a reduction factor of about 15. Both these line counts refer to lines of code or mathematical formulas, and exclude comments. This reduction was realized because of both the nonprocedural nature of the language and the array abstraction that eliminated the need for looping and subscripting constructs.

In several cases, modelers have translated spreadsheets from Microsoft Excel into Analytica, demonstrating substantial simplification and reduction in model size, and hence increased ease of comprehension and maintenance. In one case, a business and marketing model examines a proposed global-telecommunications technology. The Excel model consisted of 65 worksheets of 2.3 Megabytes. It was reduced to an Analytica model of 210 Kilobytes. In

creating the Analytica model and comparing its results against the spreadsheet, the analysts identified several significant errors in the spreadsheet that would have made the model's results extremely misleading. The total cost to create the Analytica version, and to identify and remove the defects, was less than the company had budgeted for performing an external audit of the spreadsheet.

A second case was an uncertainty and sensitivity analysis that reconstructed doses of radionuclides to populations living downstream of a nuclear-processing plant. This model examined the organ-specific doses of several radionuclides to populations at different sites over a 45-year period. The original model in Excel contained 39 Workbooks, several with many worksheets, taking 9 Megabytes of storage. In converting it into Analytica, the modeler reduced its size to about 100 kilobytes, a reduction by a factor of about 90. Computation in Excel™, using Crystal Ball™ for the Monte Carlo simulation with 500 samples, took over 5 hours. The same calculation in Analytica, on the same computer, took 1 hour. The modeler, who had no prior experience with Analytica, completed the translation in 2 weeks. He said that the most important advantage of the translation was the simplification of the model structure, which eased review for quality assurance. The main source of simplification was that the array abstraction allowed him to replace the large number of formulas for each combination of multidimensional tables by single formula to express each relationship.

10.7. Dynamic Models: Influence Diagrams and Systems Dynamics

Although Analytica is based on influence diagrams, it also represents dynamic models whose variables change over time and may contain feedback loops. Despite their superficial similarities, there are important differences between influence diagrams and the systems-dynamics notation introduced by Forrester (1969), and used in software such as Stella™ and iThink™.

First, the two notations interpret the nodes and arrows very differently. In systems dynamics, nodes represent stocks, sources, and sinks of conserved quantities, such as materials, water, money, or numbers of humans or other species. The arrows represent flows of these quantities, such as migrations, births, and deaths in populations. Other nodes represent valves, which are controls on flow rates. Influences, on the other hand, do not represent material flows – they represent knowledge and beliefs, about how the value of variables affects the value or probability distributions on other variables, which may reflect knowledge of material flows, or of other evidential relationships. For example, an influence diagram can express the relationship between a disease, its symptoms, and test results. Influence diagrams also identify decisions and objectives, explicitly.

Second, systems-dynamics models are centrally concerned with cyclic relationships that represent positive and negative feedback loops. A conventional influence diagram is necessarily acyclic — there must be no

circuit of directed arrows such that a variable is directly or indirectly influenced by itself. Without this condition, the diagram might represent an incoherent joint probability distribution over the variables. Analytica does allow dynamic cycles with feedback loops, provided that at least one influence in the cycle is time lagged — that is, a variable may depend on its own value at an earlier time step. As shown in Figure 10.20, Analytica depicts a time-lagged dependency as a dashed influence arrow. The time lag ensures that there is no true self-dependency of a single value on itself, and thus allows cycles while it avoids violation of the principles of the influence diagram.

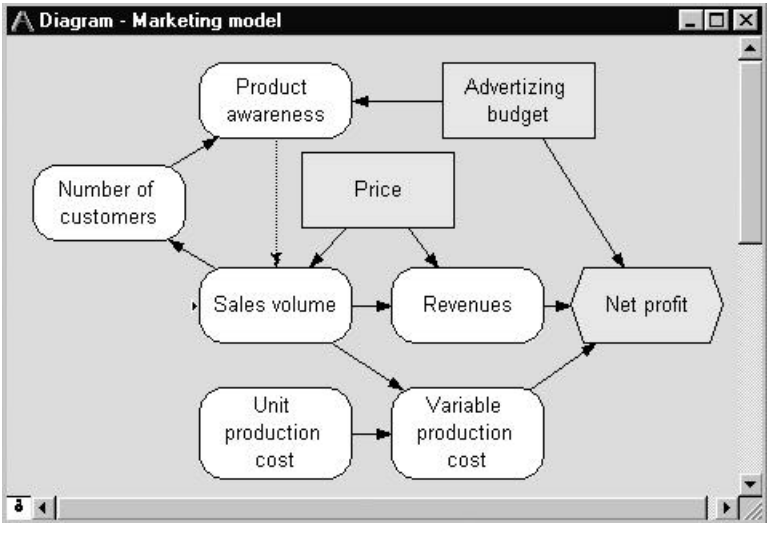


Figure 10.20: An influence diagram in Analytica depicting a dynamic model of marketing strategy. It contains a feedback loop between Product awareness, Sales volume, and Number of customers. The dashed arrow depicts a time-lagged dependency, where Product awareness at time T affects Sales volume at time T+1.

Third, influence diagrams, as a knowledge representation, are centrally concerned with uncertainty. For this reason, we have found them well suited to the creation of models for uncertainty analysis. In principle, it is possible to represent uncertainty in systems-dynamics models, but most existing systems dynamics tools do not support analysis of uncertain trajectories. They generally do provide the means to associate probability distributions with control variables — for example, depicting queues with randomly varying arrival times, or lifetimes. However, these distributions represent variability within a model, rather than uncertainty about the model trajectory. In Analytica, any variable can be uncertain, and be represented by a probability distribution. In a dynamic model in Analytica, this uncertainty gives rise to a distribution of trajectories over time. With such a model, you can explore how the qualitative

and quantitative behavior of the model over time is affected by the uncertainty in parameter values. Systems-dynamics modelers are commonly more concerned with identifying qualitative dynamic behaviors, such as whether the system exhibits oscillations, long-run asymptotes, exponential explosions, or chaotic variation, than with quantitative predictions. Uncertainty analysis of a dynamic model allows examination of whether and how the uncertainty in the parameters affects the qualitative dynamic behavior as well as the quantitative predictions.

Fourth, influence diagrams' depiction of decisions and objectives gives them a pragmatic focus. The goal in creating an influence diagram is to clarify how choice among the decision variables may affect attainment of objectives. Systems-dynamics models can also be used for this purpose, but their primary goal is to clarify complex systems rather than help people to make decisions.

10.8. Collaborative Model Development

Models for significant applications often are developed by a team of topic experts, decision makers, analysts, and reviewers. Participants are often geographically dispersed. Rarely, however, are there more than a few hands-on modelers. Coordinating multiple modelers is usually extraordinarily difficult. With Analytica, however, it is practical for many modelers to create, review, and modify, components of a model simultaneously. The development of TAF involved a collaboration of over 30 people in 10 organizations at 10 different sites around the United States (Henrion, Sonnenblick, and Soo Hoo, 1997). Each team created one or more modules of the complete model. This far-flung collaboration was made practical because the team used a set of methods adapted from software engineering, and the modular facilities in Analytica.

Of crucial importance was the definition of the interface of each module with the rest of the models. Each interface specified the input and output variables of its module. The specification included identification of the index variables that defined the dimensions of each input and output variable. Module authors were free to create and modify any variables in their module, except for those in their agreed interface. Any index used for a variable in an interface was placed in the Public Index Library (see Figure 10.17), whose contents were shared by all teams. No-one was allowed to modify a public index without negotiating clearance from the other teams who used that index. By means of careful definition and separation of public and private variables, module builders were given freedom of action within their module, while continuing compatibility of their module with the rest of the model was guaranteed.

The TAF model was created by progressive refinement in three major phases. Phase 1 started with an initial specification of module interfaces and public indexes. In each phase, the authors of each module renegotiated the interface design, typically adding or changing detail or dimensions. In phase 1, they created a simple schematic version of each module and a general

framework to integrate the modules. In phase 2, they added detail to the modules. In phase 3, they added further detail and a more complete treatment of uncertainty and variability.

Model development continued over 2 years, with face-to-face meetings about three times per year, weekly or biweekly teleconferences, and regular electronic-mail correspondence. New versions of modules and of the integrated model were shared via the Internet. A version of the final model is available at www.lumina.com/taflist.

10.9. Conclusions

There are several commercial software packages that greatly facilitate the expression of uncertainty as probability distributions, and the propagation, analysis, and display of uncertain results. Spreadsheet add-ins, such as Crystal Ball™ and @Risk™, and standalone software, such as Analytica, provide Monte Carlo and related sampling methods. Other decision-analysis products — such as DPL™, Decision maker™, and Precision tree™ — support discrete probability distributions and decision trees. However, support for the treatment of uncertainty is only one criterion for the selection of a modeling tool.

Modeling for policy analysis is a process whose goal is to develop among the experts, analysts, decision makers, analysts, and other stakeholders a shared understanding of the problem, issues, and implications. Effective modeling requires extensive communication among the team, careful auditing, and peer review. Too often, computer models are overly complex, poorly structured, and inadequately documented black boxes, which pose a major obstacle to communication. We have illustrated, however, how a modeling tool can facilitate communication by providing intuitive graphical displays of the model structure, hierarchical modules for managing complexity, integrated documentation, and a nonprocedural language with strong array abstraction.

We believe that wider use of tools that meet the objectives identified here have the potential to improve significantly the practice of quantitative policy analysis — and not just in the treatment of uncertainty. However, even software that meets all the objectives of Table 10.1 is not a panacea. Software is no substitute for knowledge and careful thought. Although software may produce spectacular results in the hands of a skilled analyst, it can also produce spectacular garbage of extraordinary complexity in the hands of an incompetent or careless analyst. The saving grace of such tools is that, if they make it easier to exercise, scrutinize, audit, and critique models, they can facilitate peer review and open debate about policy models, providing a powerful mechanism for improving standards of practice.

References

Amaral, D.A.L. (1983). "Quantifying Uncertainty in the Health Effects of Respirable Sulfates." Ph.D. diss., Department of Engineering and Public Policy, Carnegie Mellon University, Pittsburgh, PA.

Balson, W.E., Boyd, D., and North, D.W. (1982). *Acid Disposition: Decision Framework. Volume 2: User Guide to the Acid Deposition Decision Tree Program*, Research Report No. RP 2156-1, Electric Power Research Institute, Palo Alto, CA.

Dayo, F.B. (1983). "Choice Between Alternative Nuclear Energy Systems: A Decision Analytical Model for Less Developed Countries," Ph.D. diss., Carnegie Mellon University, Pittsburgh, PA.

Ericsson, K.A., and Simon, H.A. (1984) *Protocol Analysis: Verbal Reports as Data*, MIT Press, Cambridge, MA.

Foster, G., and Stefik, M. (1986). "Cognoter, Theory and Practice of a Collaborative Tool," pp. 7-15, in *Conference on Computer-Supported Cooperative Work*, ed. H. Krasner and I. Greif, The Association for Computing Machinery, New York.

Geoffrion, A.M. (1987). "An Introduction to Structured Modeling," *Management Science*, 33:547–588.

Graham, J.G. and Henrion, M. (1984). "The Passive Restraint Question: A Probabilistic Analysis," *Risk Analysis*, 42, no. 2: 25–40.

Greenberger, M. (1981, October). "Humanizing Policy Analysis: Confronting the Paradox in Energy Modeling." *In Validation and Assessment of Energy Models*, Ed. S. Gass, National Bureau of Standards (Special Publication 616).Gaithersburg, MD

Henrion, M.,. Breese, J.S, and Horvitz, E.J. (1991) "Decision Analysis and Expert Systems," *Artificial Intelligence Magazine*", Vol 12, No 4, Winter, pp64-91.

Henrion, M., and Morgan, M.G., (1985). "A Computer Aid for Risk and Other Policy Analysis," *Risk Analysis* 5: 195-208.

Henrion, M., Morgan, M.G., Nair, I., and Wiecha, C. (1986). "Evaluating and Information System for Policy Modeling and Uncertainty Analysis," *Journal of the American Society for Information Science*, 37, no. 5: 319-330.

Henrion, M. & Silva, J. (1994) "Cost savings from Information Technology in U.S. Health Care Reform: Insights from Modeling", , *The J. of Healthcare Information and Management Systems Society*, Vol 8, No 1, 1994, pp23-28.

Henrion, M., Sonnenblick, R.S., and Soo Hoo, K. (1997) "Lessons in the collaborative construction of an environmental integrated assessment model: The Tracking and Analysis Framework." *Proceedings of the Air and Waste Management Conference on Acid Rain and Electric Utilities*, Scottsdale, AZ, Jan 21-23.

Henrion, M., Sonnenblick, R.S., and Bloyd, C. (1997) "Innovations in Integrated Assessment: The Tracking and Analysis Framework (TAF)." *Proceedings of the Air and Waste Management Conference on Acid Rain and Electric Utilities*, Scottsdale, AZ, Jan 21-23.

Henrion, M. and Wishbow, N. (1987). *Demos User's Manual: Version Three*, Department of Engineering and Public Policy, Carnegie Mellon University, Pittsburgh. PA.

Howard, R. A., and Matheson, J. (1981). "Influence Diagrams," reprinted in Howard and Matheson (1984)

Howard, R.A., and Matheson, J. (1984). *The Principles and Applications of Decision Analysis, 2*, Strategic Decisions Group, Menlo Park, Calif., pp. 719-762.

Iman, R.L., and Shortencarier, M.J. (1984, March). *A Fortan-77 Program and User's Guide for the Generation of Latin Hypercube and Random Samples for Use with Computer Models*, SAND83- 2365, Sandia National Laboratory, Albuquerque, N.M.

Keeney, R.L, (1992) *Value-Focused Thinking*, Harvard UP.

Lumina Decision Systems, Inc (1998), *Analytica for Windows 1.2 Users Guide*, , Los Gatos, CA.

Morgan, M.G., Morris, S.C., Henrion, M., Amaral, D.A.L., and Rish, W.B., (1984). "Treating Technical Uncertainty in Policy Analysis: A Sulfur Air Pollution Example, "*Risk Analysis*, 4, no. 3: 201- 216.

Nair, I., Morgan, M.G., and Henrion, M. (1982). "Office Automation: Assessing Energy Implications," *Telecommunications Policy*, 6, no. 3: 207-222.

Prywes, N., Pnueli, A., and Shastry, S. (1979). "Use of a Nonprocedural Specification Language and Associated Program Generator in Software Development," *ACM Transactions on Programming Languages and Systems*, 1, no. 2: 196-217.

Rubin,, E.S, Small, M.J.. Bloyd, C.N and Henrion, M. (1992) "Integrated Assessment of Acid-Deposition Effects on Lake Acidification", *J. Environmental Engineering*, Vol 118, No 1, Jan/Feb, p120-134.

Rubin,, E.S, Small, M.J.. Bloyd, C.N., Marnicio, R. and Henrion, M. (1990) "Atmospheric Deposition Assessment Model: Application to regional aquatic acidification in eastern North America", Chapter 14 in *Impact Models to Assess Regional Acidification*, J. Kamari (ed.), Kluwer Academic Publishers: Dordrecht, The Netherlands,, pp 253-284.

Sonnenblick, R.S., and Henrion, M.. (1997) "Uncertainty in the Tracking and Analysis Framework Integrated Assessment: The Value of Knowing How Little You Know." *Proceedings of the Air and Waste Management Conference on Acid Rain and Electric Utilities*, Scottsdale, AZ, Jan 21-23.

Wiecha, C. (1986). "An Empirical Study of How Visual Programming Aids in Comprehending Quantitative Policy Models," Ph.D. diss., Department of Engineering and Public Policy, Carnegie Mellon University, Pittsburgh, PA.

Wiecha, C., and Henrion, M. (1987a). "Linking Multiple Program Views Using a Visual Cache," *Proceedings of Interact-87*, Stuttgart.

Wiecha, C., and Henrion, M. (1987b). "An Empirical Study of Strategies for Understanding Quantitative Decision Models," *Proceedings of the Eighth International Conference on Information Systems*, Pittsburgh, PA.