analytica®

Beyond   the Spreadsheet

Tutorial

Release 4.2
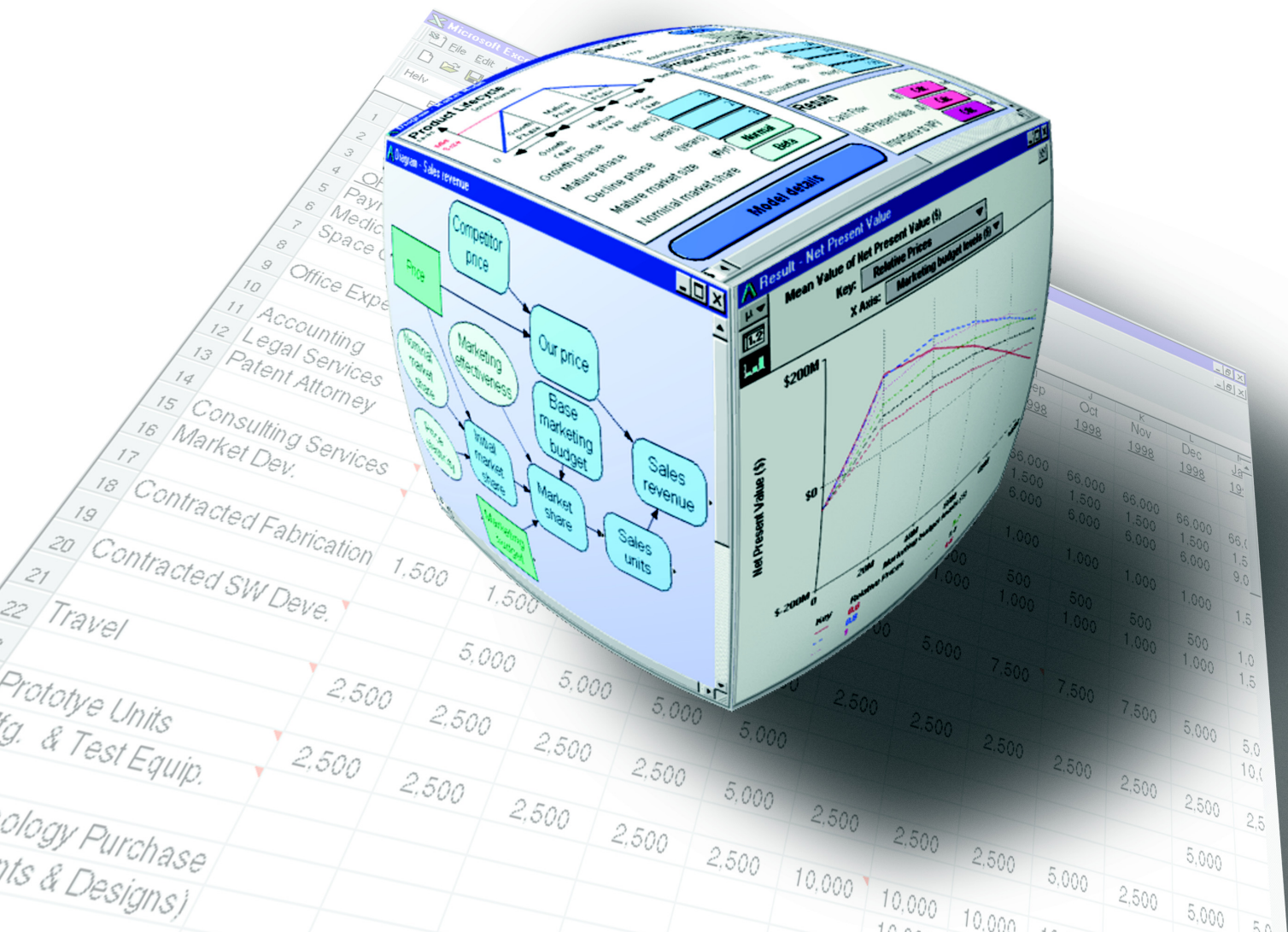
Lumina®
DECISION SYSTEMS

Copyright notice

Credits

The Analytica Tutorial was written by Brian Arnold and Lynda Korsan with Max Henrion and Randa Mulford (Expert Support, Inc.). New releases have been edited by Lonnie Chrisman, Richard Morgan, Paul Sanford and Fred Brunton.

Contents

Contents

Contents

Contents

Introduction

About Analytica

This section introduces Analytica and its uses, explains what is included in this tutorial, and tells you how to use this manual.
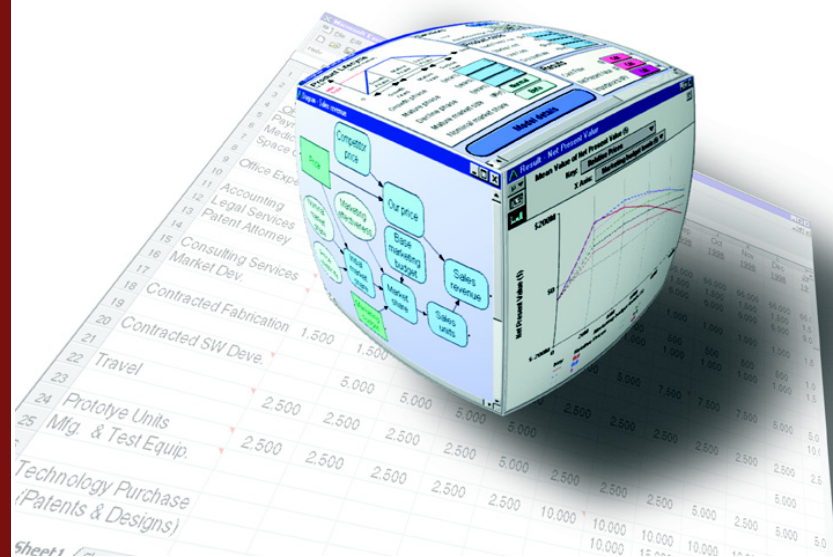
Welcome to Analytica

You are about to discover a powerful tool for real-world modeling and analysis. Analytica embodies the idea of using a white board for problem solving. Using a visual, point-and-click approach, you draw nodes and arrows to depict the relationships between model components. This approach allows you to describe the essential *qualitative* nature of the problem without getting lost in the details. As the model develops and your understanding of the problem becomes clear, you can define the exact *quantitative* details of the model.

A key feature of Analytica is its ability to create hierarchies of models. By grouping related components of a problem into separate submodels, you can impose a multi-level organization to your model. This helps you to manage complex relationships and allows other users to more easily grasp important concepts.
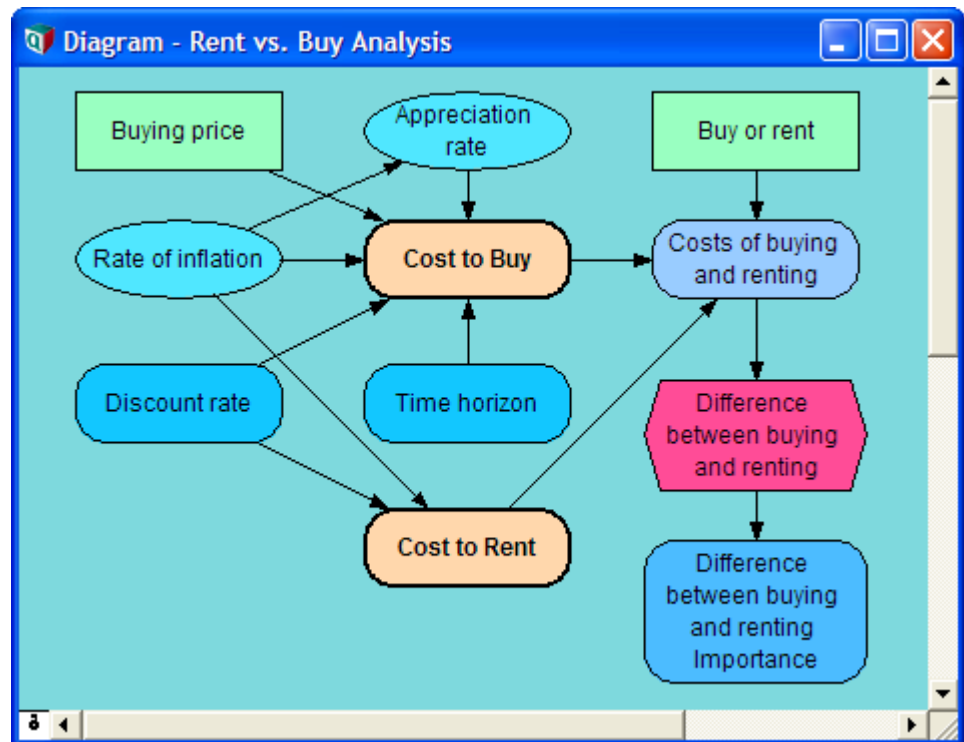
Another key feature of Analytica is the use of **Intelligent Arrays**™. These enable you to add or remove dimensions such as time periods, geographic regions, alternative decisions, etc., with minimal changes to the model structure. Unlike spreadsheets, which require you to repeat formulas with each new dimension, Analytica separates the dimensions from the relationships so that models remain simple. As the dimensions change, Analytica automatically updates, reports, and graphs the results.

Each node, or object, in an Analytica model has a window that displays the node's inputs and outputs, and allows you to enter definitions, descriptions, units of measure, and other documentary information. This self-documenting capability, combined with hierarchical models and Intelligent Arrays, makes it easier to understand and communicate how models work.

Analytica features fully integrated risk and sensitivity analysis for analyzing models with uncertain inputs; powerful facilities for time-dependent, dynamic simulations; powerful graphing capabilities; and over 200 financial, statistical, and scientific functions for calculating just about any type of mathematical expression.

Who can use Analytica

Analytica is for the modeler and problem solver — from the financial analyst modeling business opportunities to the engineer designing new products to the scientist investigating the behavior of physical phenomena.

It is particularly suited to users in the fields where you have to reason with uncertainties, or arrays of data, or both, i.e., management consulting, health and environmental sciences, aerospace, oil and gas, construction, manufacturing, financial services, and investing.

Tutorial overview

This tutorial is a hands-on introduction to using Analytica. Step-by-step instructions show you how to explore and analyze an existing Analytica model and how to create a new Analytica model. Because later tutorial sections build on the material in earlier chapters, you should work through the chapters in order.

We recommend that everyone new to Analytica complete Chapters 1 through 5, which takes two to three hours. If you want to work more quickly, skip the text and only follow the instructions in the boxed steps. Then, if you are unsure about any terms or concepts, look them up in the Glossary or review the text. And before you start your own modeling, you should review Chapter 8, which describes the sample models included with Analytica. This way, you can benefit from examples similar to what you might be modeling.

This tutorial is designed to introduce you to some of Analytica's basic features. When you are familiar with the basics, refer to the *Analytica User Guide* for more detailed information on Analytica's features.

- **Chapter 1: Using the Rent vs. Buy Model**

    This chapter shows how to open and run an Analytica model. Using a simple interface to an example model that analyzes the total costs of buying or renting a house, you will calculate results and change input values to see the effects on the results. You will display uncertain results in a variety of ways.

- **Chapter 2: Exploring the Rent vs. Buy Model**

    This chapter shows you how to browse a model's structure and assumptions by examining its influence diagrams, variables, and definitions.

- **Chapter 3: Analyzing the Rent vs. Buy Analysis Model**

    This chapter shows you how to perform importance analysis and sensitivity analysis to see which uncertain variables most heavily influence the outcome.

- **Chapter 4: Creating Models**

    This chapter shows you how to create a new Analytica model. In the process of building a model that analyzes the costs of owning and operating an automobile, you will create variables, define relationships between variables, add documentary text, and compute results. In addition, you will create modules and add dependencies between modules.

- **Chapter 5: Working with Arrays (Tables)**

    This chapter shows you how to add *index variables* and *edit tables* (these will be defined later) to a model, and demonstrates how tables work in Analytica, including an introduction to table functions.

- **Chapter 6: Creating the Party Problem Model**

    This chapter walks you through a familiar problem: where to have your next party. This model introduces probability tables and conditional deterministic tables. You should complete this chapter if your models will use discrete or conditional uncertainties.

- **Chapter 7: Creating the Foxes and Hares Model**

    In this chapter you create a dynamic model of population sizes that depend on each other and that change with time. You should complete this chapter if your models will use dynamic simulation or variables that change over time.

- **Chapter 8: On Your Own**

    This chapter briefly describes all the example models provided with Analytica. You should

investigate these as you begin to build your own models.

Installing Analytica

Before you start this tutorial, follow these steps to install the Analytica application and associated model files on your computer.

Installing from the Web

1. Go to the Lumina web site: www.lumina.com.
2. Click the **Download Software and Manuals** link.
3. Read the instructions on the web page that comes up.
    Click the **Setupana.exe** link next to the **Analytica 4.2 for Windows** heading.
4. A file download dialog box appears. Click **Run** to download the installation program and start running it to install Analytica. Or you can click **Save** to download the installation program to run later.
5. If you clicked **Run**, the Windows Installer should automatically start up and begin installing Analytica.

The setup program requires some responses from you. For example, you are asked to verify the directory name in which Analytica will be installed. Most users can accept the defaults provided by the setup program. The default installation location for Analytica is `C:\Program Files\Lumina\Analytica 4.2`.

Installing from a CD-ROM

1. Start Windows.
2. Insert the Analytica CD in your computer's CD-ROM drive.
3. The Windows Installer should automatically start up and begin installing Analytica.

If the AutoRun function does not work, follow these alternate steps:

4. Click the **Start** button on the Windows taskbar.
5. Select **Run** from the popup menu.
6. In the **Run** dialog box, specify the program `SETUP.EXE` on your CD-ROM drive (usually either the D: or E: drive).
7. Click **OK**.

The setup program requires some responses from you. For example, you are asked to verify the directory name in which Analytica will be installed. Most users can accept the defaults provided by the setup program. The default installation location for Analytica is `C:\Program Files\Lumina\Analytica 4.2`.

Conventions used in this tutorial

The conventions used in this tutorial are as follows:

* Boxed, numbered instructions along the left side of the page give you the steps to take.

   Explanatory text often follows a step, accompanied by pictures of the Analytica screens similar to the ones you see on your computer.

For example:

1. This is an instruction.

2. In a step, ***this is the text you will enter.***

   This is explanatory text.
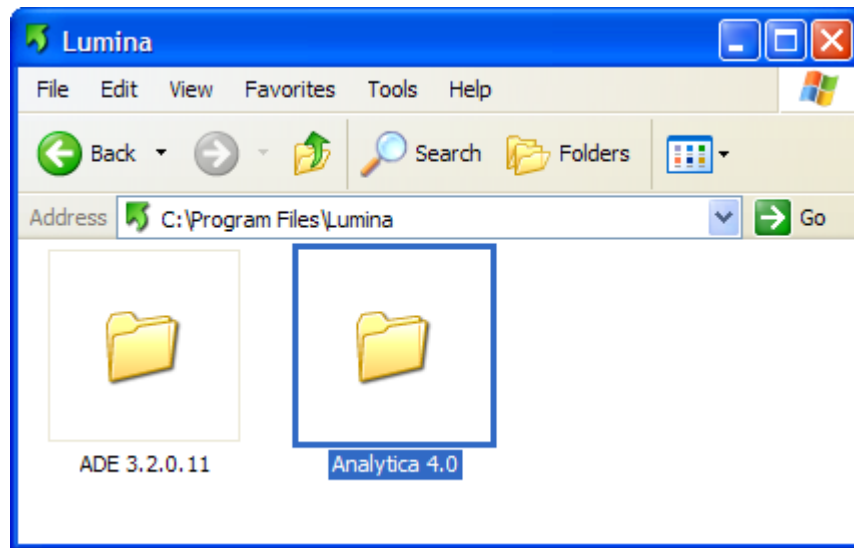
- Variable and model titles are displayed in *italic type*.
- The following keys on the keyboard are shown in *italic type*: *Enter*, *Return*, *Tab*, *Delete*, *Shift*, *Alt*, *F1*.
- Special Analytica terms are displayed in ***bold italic type***; they are defined when they are first introduced.

For your reference, a glossary at the end of the tutorial defines the terms used in this tutorial.

Tips alert you to useful or important information. They look like this:

Tip These alert you to useful or important information.

Assumed background

This tutorial assumes that you already have the basic skills needed to run Windows programs, including the following:

Term	Meaning
click	Press and release the mouse button one time.
double-click	Quickly press and release the mouse button two times.
drag	Press and hold down the mouse button while moving the cursor to a new location on the screen, then release the mouse button.
press	Press and hold down the mouse button.
select	Click an interface object, such as a menu command or a cell in a table; selected objects usually appear highlighted.

You also need to know how to use pull-down and popup menus, scroll bars, and windows.

If you are not familiar with these basic operations, look at the reference material that came with your computer.

This tutorial also assumes that you have basic skills of financial or quantitative modeling — for example, from previously using a spreadsheet program.

It assumes that you are acquainted with elementary statistics and are comfortable with the concepts of mean, median, and standard deviation. It also assumes that you have some understand-

ing of probability distributions, such as the normal and uniform, and are familiar with the concepts of probability density function and cumulative distribution function. These terms are reviewed briefly in the Glossary at the end of the tutorial.

Chapter 1

Using the Rent vs. Buy Model

This chapter shows you how to:

- Open an existing model
- Calculate results
- Change input values to calculate different results

Chapter **1** *Using the Rent vs. Buy Model*

Opening the Rent vs. Buy model
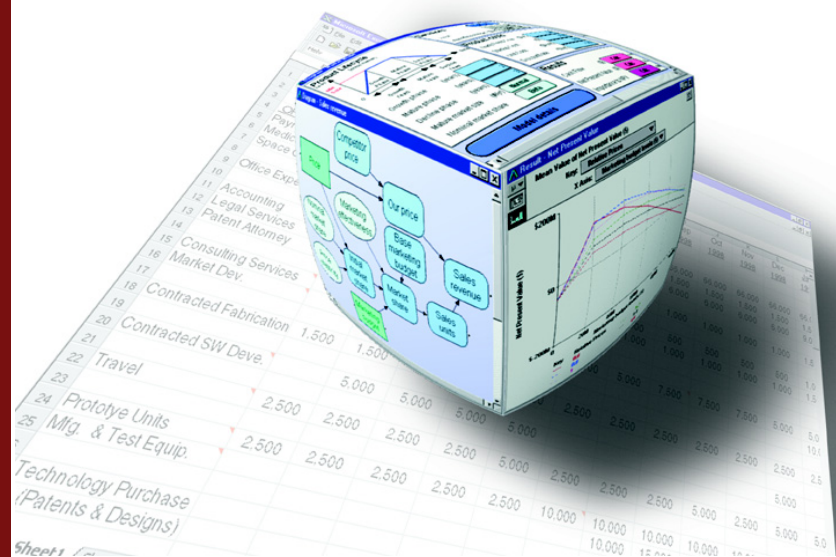
In this chapter, you use the *Rent vs. Buy* model, an Analytica model that compares the cost of renting a house to the cost of buying one. After working through the chapter, you will know how to open an existing model, use it to calculate results, and change input values to calculate different results.

Opening the Rent vs. Buy model

To begin, follow these steps.

1. Click the **Start** button on the Windows taskbar.
2. Click **All Programs** in the **Start** menu.
3. Click **Analytica** in the **Programs** menu.
4. Click **Tutorial Models** in the **Analytica** menu.
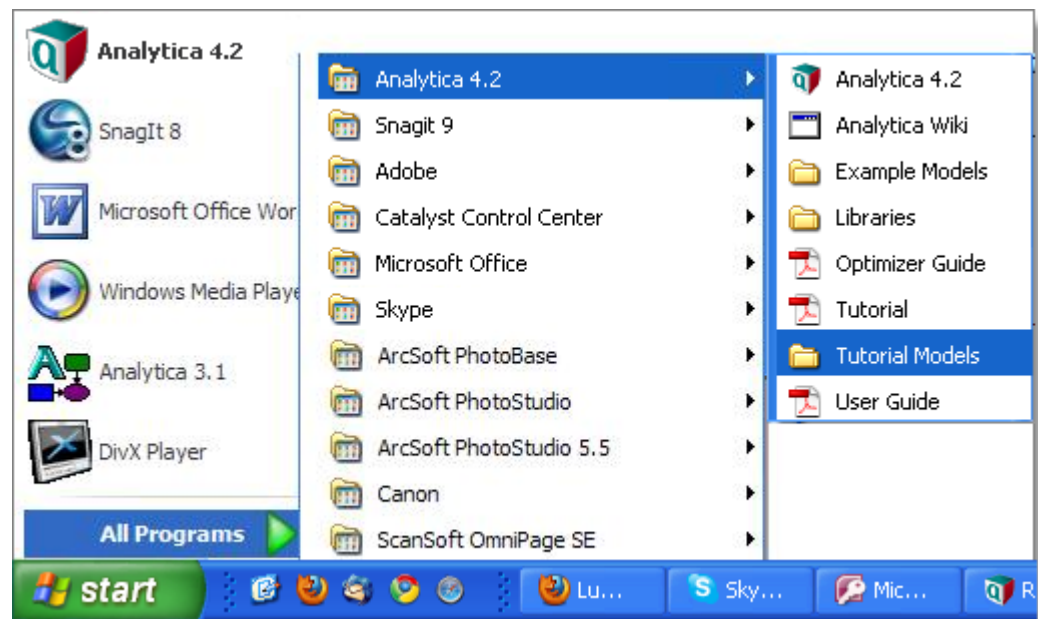
5. Open the *Rent vs. Buy* model.

1. Double-click the icon for the *Rent vs. Buy* model to start Analytica.
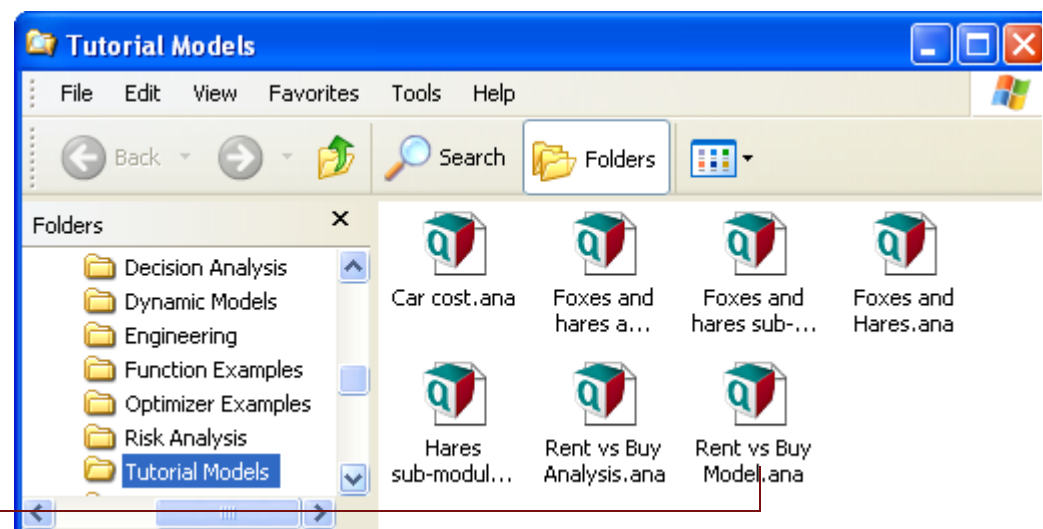
Analytica reads in the *Rent vs. Buy* model.

Chapter 1 *Using the Rent vs. Buy Model*

Becoming familiar with the Diagram window

Becoming familiar with the Diagram window

When you open a model, Analytica first displays a top-level **Diagram** window. The *Rent vs. Buy* model diagram shows several input variables that affect the trade-offs between renting and buying, **Normal** buttons, a **Calc** button, and a node labeled *Model*.
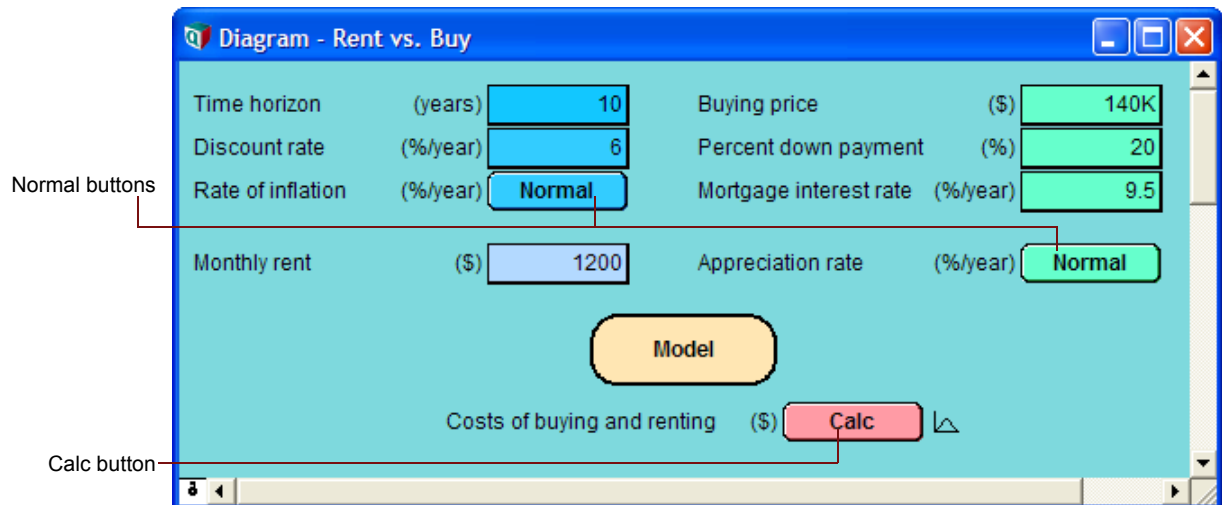
This top-level diagram is an end-user interface to the model itself, which is contained in the *Model* node. In this chapter, you use only the interface in this top level diagram; in the following chapters you will explore the model in more depth.

Across the top of the screen is a horizontal palette of buttons. This is called the ***tools palette***.

When you first open the *Rent vs. Buy* model, the ***browse tool*** is highlighted on the palette. With the browse tool selected, the cursor looks like a hand 👆 when it is over the diagram. The browse tool allows you to calculate the model, change input values, and examine — but not change — the structure of the model. In this chapter, you only use the browse tool.

Accessing Help Resources

At any time, you can press the *F1* key on the keyboard or use the **Help** pull-down menu to access Analytica's help resources. These include User Guide and Tutorial documents as well as Analytica's online Wiki pages.

Computing output values

In the *Rent vs. Buy* model, the output value of interest is at the bottom, *Costs of buying and renting*.

1. Click the **Calc** button to compute the comparison of the cost of buying to the cost of renting.

The output value displays in a **Result** window. This **Result** window shows a graph of two ***probability density*** curves, one for buying and one for renting. In a probability density graph, the units of the vertical scale are chosen so that the total area under each curve is 1 (100%). 25µ corresponds to 25 x 10^{-6} or 0.000025.

Tip Numerical suffixes like µ and K are used extensively throughout Analytica. A quick reference for these suffixes is given on the back page of this tutorial.

Since the graph is of probability densities, both buying and renting have probabilistic, or uncertain, inputs. The probability density graph for Rent appears to be a bell-shaped curve (***normal*** dis-

tribution). The graph for Buy appears "noisy," with two peaks, indicating it is a more complicated function of uncertain inputs.

This probability density graph appears jagged because Analytica is using a sample size of 100. A larger sample size would produce a smoother curve, but the curve would take longer to compute. You can also make this graph appear smoother by increasing the number of samples per plot point. See "Uncertainty View Options" in the *Analytica User Guide* for more information.

The graphs show that the cost of renting, given the model's inputs, are between about $105,000 and $155,000 (the negative numbers mean cost — cash flowing out), while the cost of buying is between $125,000 and a gain of $100,000.
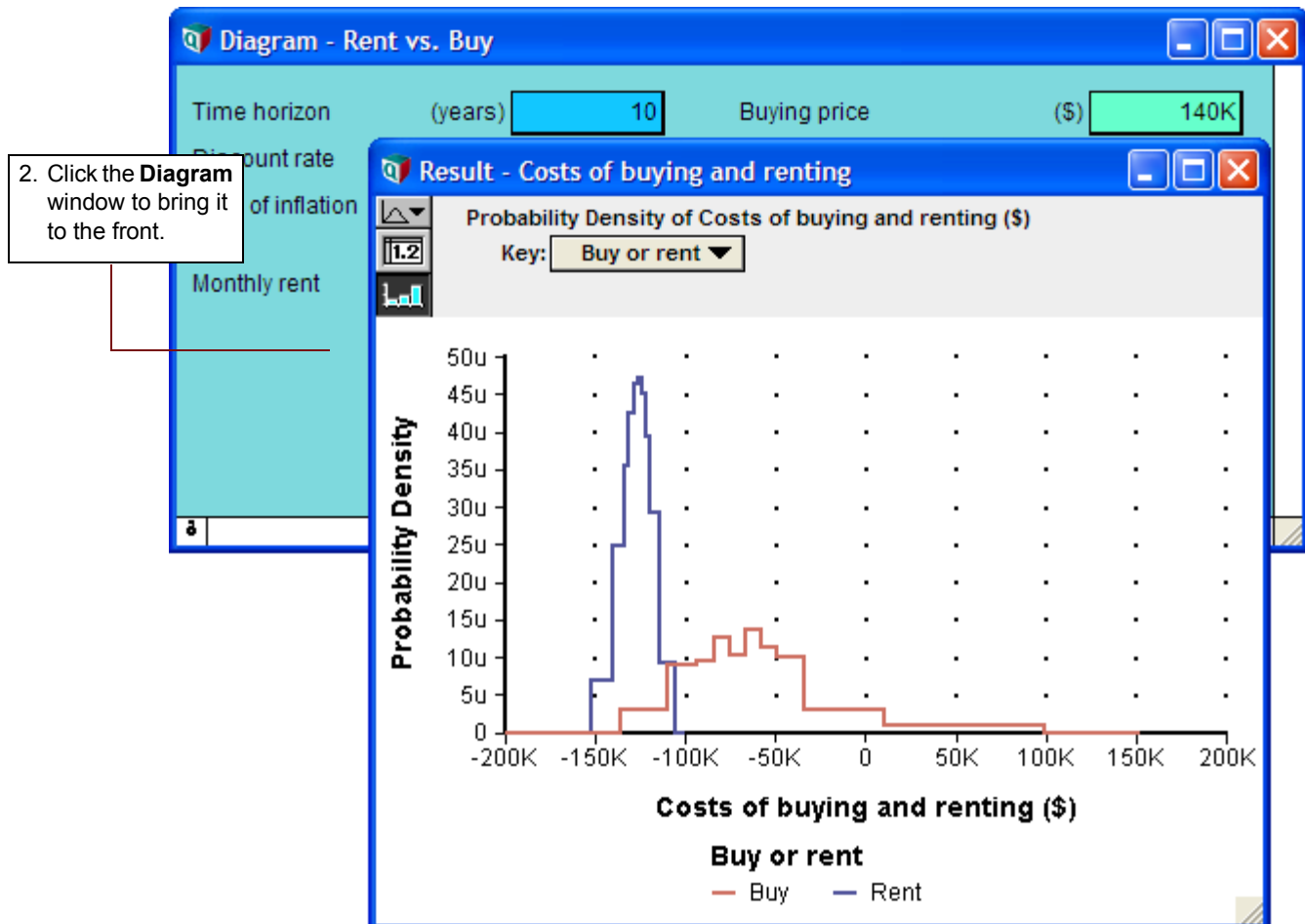
Note: *Your results can vary slightly, since the model is generating random inputs based on a normal distribution for the uncertainty of the rate of inflation and for the appreciation rate.*

Click the model **Diagram** window to bring it to the front. Notice that the button next to *Costs of buying and renting* has changed to **Result**. The **Result** button indicates that the value has been computed; clicking the **Result** button re-displays the computed values.

The **Calc** button has changed to **Result**

Changing input values and recomputing

Now you will change some input values to the model and recompute the rent vs. buy comparison. You will change the values of *Time horizon*, *Monthly rent*, and *Buying price*.

1. Click the box next to *Time horizon*. Change the value to **7** and press *Alt+Enter*.

The main *Enter* key and the numeric keypad *Enter* key are not interchangeable. They have different functions in Analytica. *Alt+Enter* is equivalent to the numeric keypad *Enter* key.

As soon as you change an input, the **Result** button changes to a **Calc** button, indicating that *Costs of buying and renting* needs to be recomputed.

2. Click the box next to *Monthly rent*. Change the value to *1400* and press *Alt+Enter*.

3. Click the box next to *Buying price*. Change the value to *180K* (or 180000) and press *Alt+Enter*.

Now you are ready to recompute to see the new results.

4. Click the Calc button to compute the comparison of the cost of buying to the cost of renting.

The graphs show that the cost of renting, given these changed inputs, is between $90,000 and $120,000, while the cost of buying is between $135,000 and a gain of $70,000.

5. Click the Diagram window to bring it to the front.

Examining and changing uncertain input

When an input is defined as a probability distribution, a button with the name of the distribution appears next to the input's name. Clicking this button opens the **Object Finder** window, in which you can see details and change the distribution's parameters or type of distribution.

Rate of inflation's button says **Normal**, indicating that it is defined as a normal distribution.

1. Click the **Normal** button next to *Rate of inflation*.
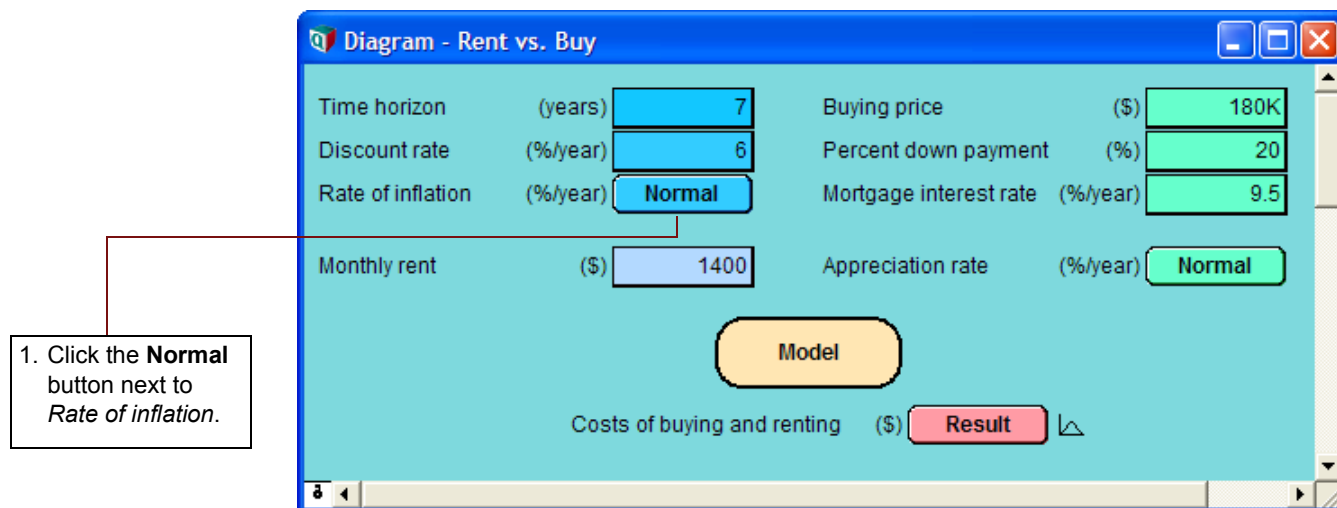
The **Object Finder** window appears. It shows that *Rate of inflation* is defined as a normal distribution with a **mean** of 3.5 and a **standard deviation** of 1.3.

You will now modify the probability distribution that defines *Rate of inflation*. Rather than using the normal distribution, you will use the uniform distribution, and assume that inflation has an equal probability of being anywhere between 3% and 4% per year.

2. Scroll down the list of distributions and select **Uniform**.

Scroll bar
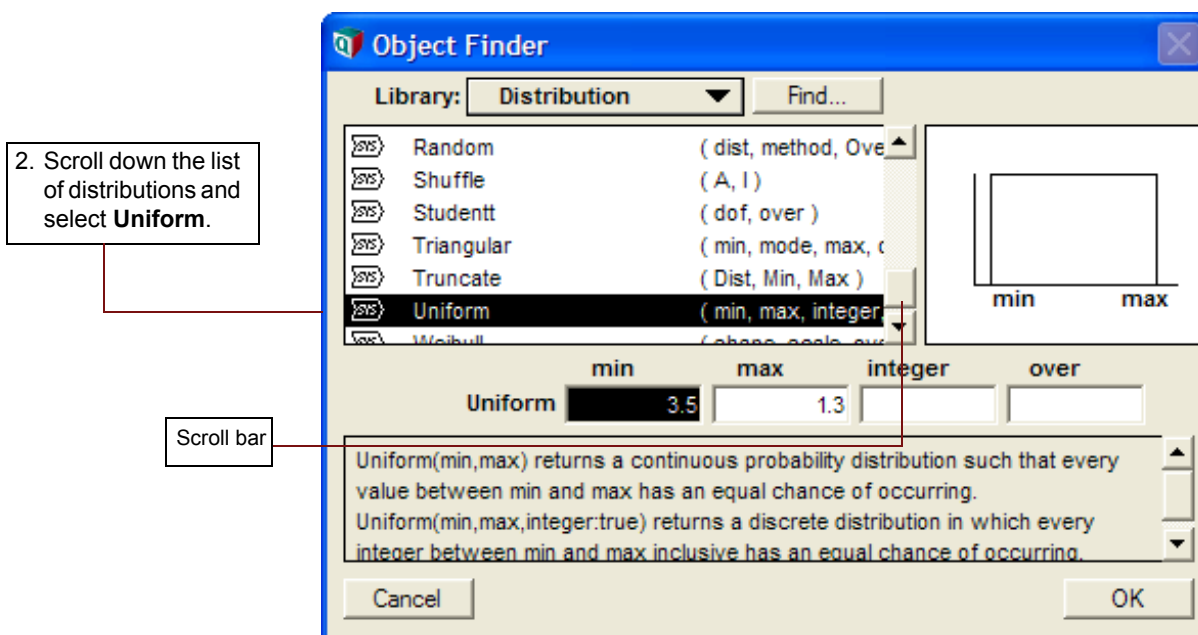
3. Change the minimum to **3** and the maximum to **4**.

4. Click **OK** to accept the change.

Object Finder

Library:	Distribution ▼	Find...

[SYS]	Random	(dist, method, Ove
[SYS]	Shuffle	(A, I)
[SYS]	Studentt	(dof, over)
[SYS]	Triangular	(min, mode, max, (
[SYS]	Truncate	(Dist, Min, Max)
[SYS]	Uniform	(min, max, integer
[SYS]	Weibull	(shape, scale, ove

	min	max	integer	over
Uniform	3	4		

Uniform(min,max) returns a continuous probability distribution such that every value between min and max has an equal chance of occurring.
Uniform(min,max,integer:true) returns a discrete distribution in which every integer between min and max inclusive has an equal chance of occurring.

Cancel   OK

5. Click the **Calc** button to compute the new comparison of the cost of buying to the cost of renting.

Diagram - Rent vs. Buy

Time horizon	(years)	7	Buying price	($)	180K
Discount rate	(%/year)	6	Percent down payment	(%)	20
Rate of inflation	(%/year)	Uniform	Mortgage interest rate	(%/year)	9.5
Monthly rent	($)	1400	Appreciation rate	(%/year)	Normal

Model

Costs of buying and renting   ($)   Calc
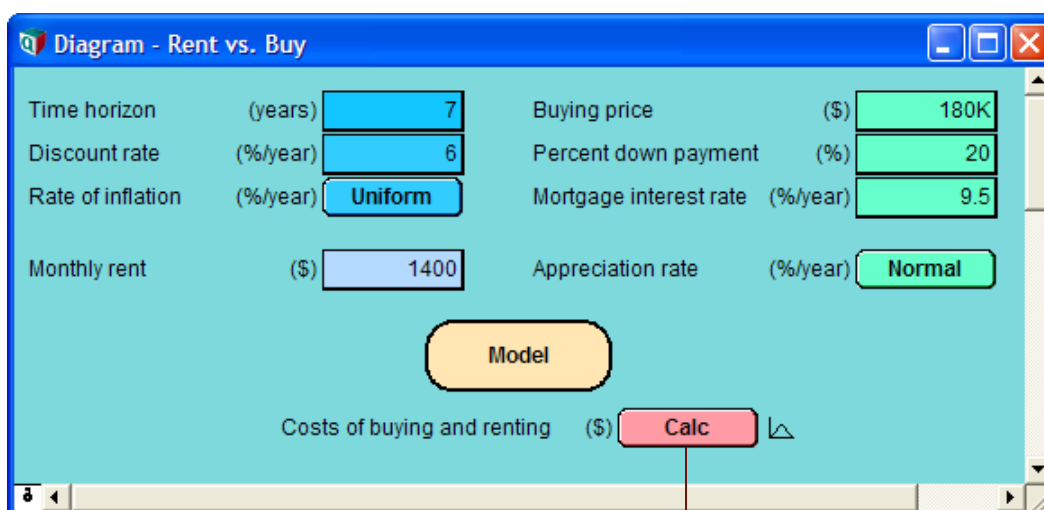
The graphs show that the uncertainty in the cost of renting has narrowed to between about $123,000 and $130,000, while the uncertainty in the cost of buying has flattened to between about $135,000 and a gain of $37,000.

Displaying alternative uncertain views

Analytica offers a variety of views to display uncertain values, including selected statistics, ***probability bands***, the ***probability density*** function, the ***cumulative probability*** distribution function, measures of central tendency, and the table of random numbers from which the uncertain distribution is estimated.

You will now examine several of these views.

In the upper-left corner of the **Result** window is the Uncertainty View popup menu.

Uncertainty View
popup menu
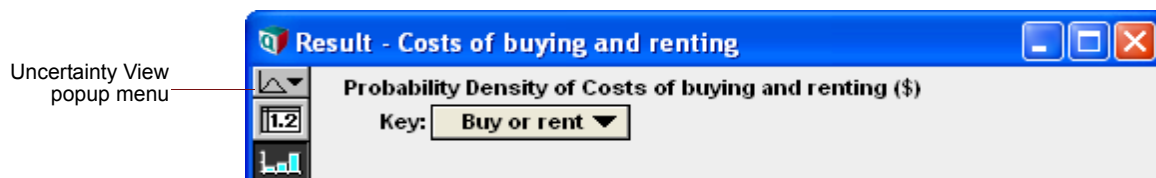
The miniature probability distribution [△▾] indicates that **Probability Density** is selected.

1. Press on the Uncertainty View popup menu and select **Cumulative Probability** [△▾].

mid	Mid Value
μ	Mean Value
μ±	Statistics
▦	Probability Bands
✓ △	Probability Density
⟋	Cumulative Probability
▨	Sample
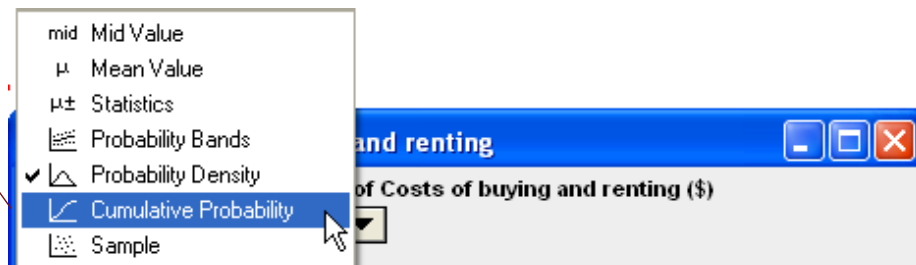
The **Result** window now shows two cumulative probability curves. Along the vertical axis, these curves give the probability that each cost is less than a given value along the horizontal axis.

Result - Costs of buying and renting

Cumulative Probability of Costs of buying and renting ($)

Key: **Buy or rent** ▾

Costs of buying and renting ($)

Buy or rent

— Buy   — Rent
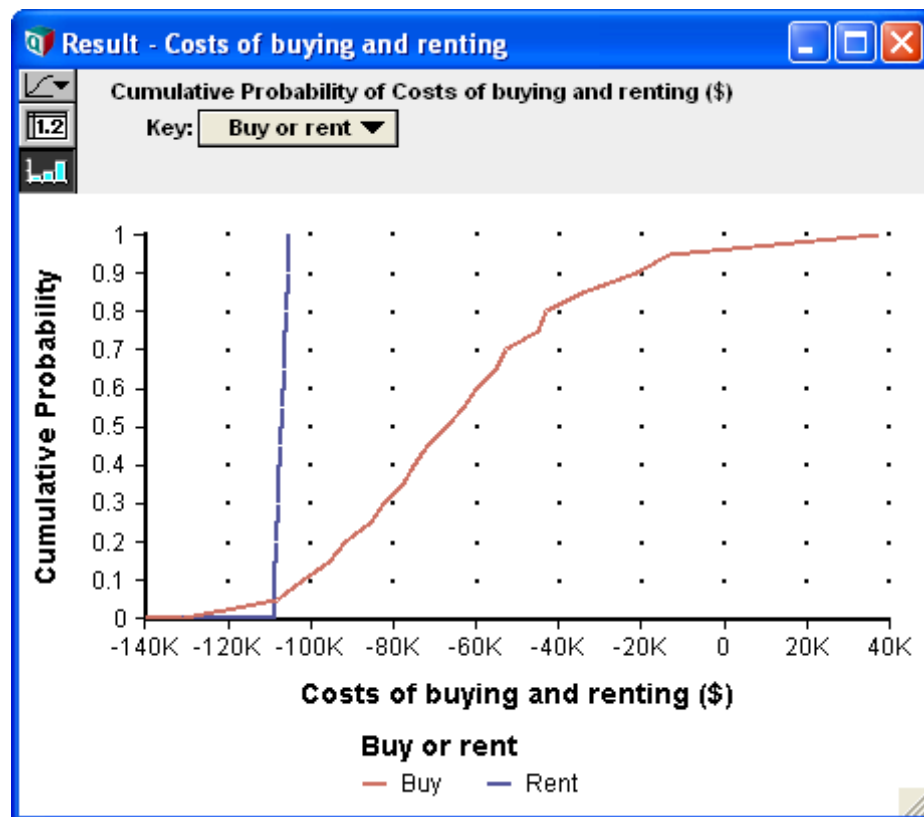
There appears to be about a 50% probability that the cost to buy is below $70,000, while the cost to rent has a 50% probability of being below about $130,000.

Sometimes you might want to see an uncertain value expressed as a single number — a measure of central tendency. Analytica computes the **mid value** (sometimes called the **deterministic value**) by fixing all input probability distributions at their **median** (50% probability) values. The mid value is the only uncertainty view available for nonprobabilistic results.

2. Select **Mid Value** [mid▾] from the Uncertainty View popup menu.

mid	Mid Value
μ	Mean Value
μ±	Statistics
	Probability Bands
	Probability Density
✓	Cumulative Probability
	Sample
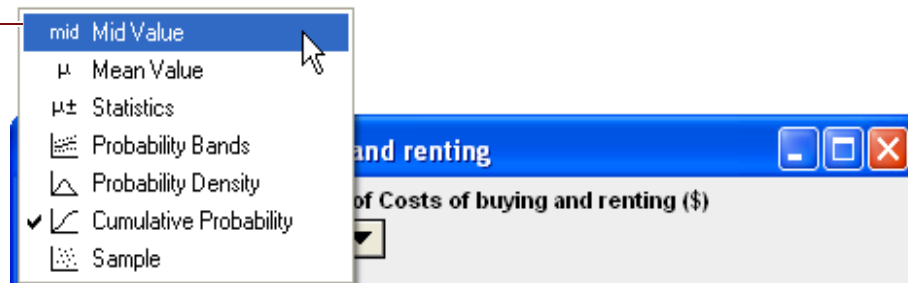
The **Result** window now displays bar graphs for the two mid values.

Under the Uncertainty View popup menu are two buttons, [1.2] and [ᴸᴬᴸ]. The [ᴸᴬᴸ] is highlighted, indicating that the **Result** window is displaying a graph view. The **Result** window can also display numeric values in a spreadsheet-like table view.

3. Click the **table** view button [1.2] to select the table view.

Result - Costs of buying and renting

Mid Value of Costs of buying and renting ($)

Buy or rent ▾ ☐ Totals

Buy	-67.2K
Rent	-107K
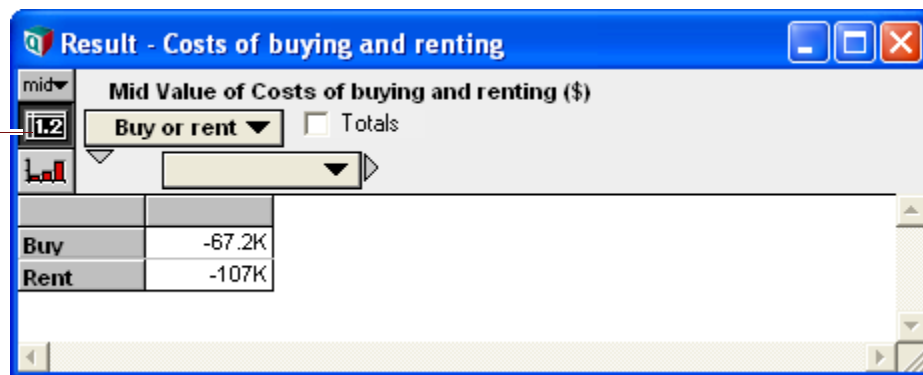
Analytica also provides the *mean* (or average) value.

4. Select **Mean Value** [μ ▾] from the Uncertainty View popup menu.

Result - Costs of buying and renting

✓ mid	Mid Value
μ	Mean Value
μ±	Statistics
	Probability Bands
	Probability Density
	Cumulative Probability
	Sample

Result - Costs of buying and renting

Mean Value of Costs of buying and renting ($)

Buy or rent ▾ ☐ Totals

Buy	-64.75K
Rent	-107K
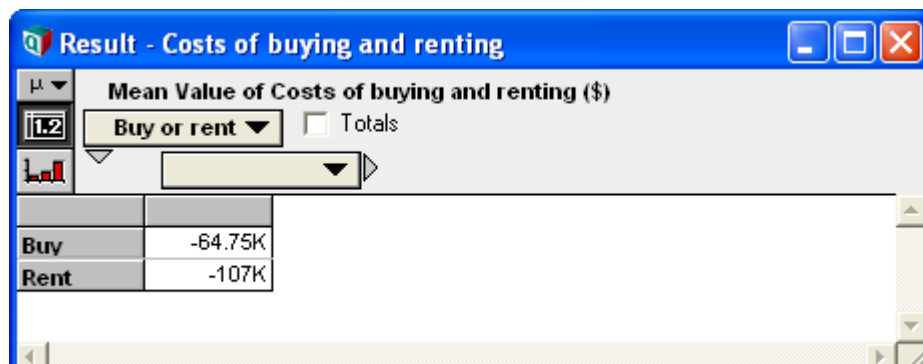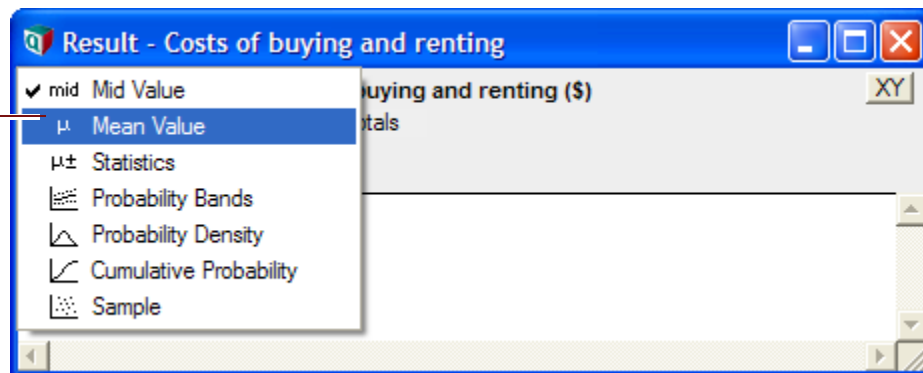
You can also view a set of statistics, including both the median and mean, the ranges (minimum and maximum), and the standard deviation.

5. Select **Statistics**
$\mu\pm$ from the
Uncertainty View
popup menu.

mid	Mid Value
✓ μ	Mean Value
μ±	Statistics
	Probability Bands
	Probability Density
	Cumulative Probability
	Sample
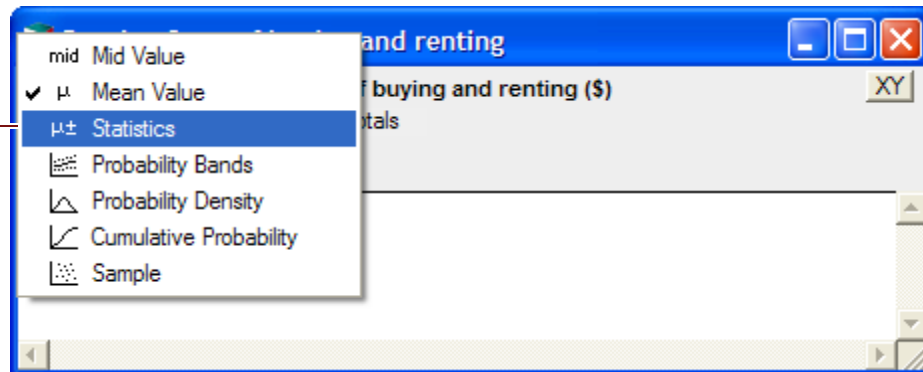
The **Result** window now displays the minimum, median[1], mean, maximum, and standard deviation for *Costs of buying and renting*.

Result - Costs of buying and renting

Statistics of Costs of buying and renting ($)

Statistics ▼   ☐ Totals

Buy or rent ▼ ▷ ☐ Totals

	Buy	Rent
Min	-126.4K	-109K
Median	-67.49K	-107K
Mean	-64.75K	-107K
Max	28.22K	-105K
Std. Dev.	30.02K	1171

The statistics might not be exact, because they are estimated from a sample of values from the distribution.

---

1. Note that the median value is slightly different from the mid value. The mid value is composed of non-probabilistic results generated by using the mean value for each input. The median value is calculated using probabilistic inputs and taking the median of the resulting distribution.

Finally, you see the sample values.

6. Select **Sample** from the Uncertainty View popup menu.

7. Click this pull-down menu to select "Buy or rent" for the columns.

8. Use the scroll bar to examine more sample values in the table.

Result - Costs of buying and renting

Sample of Costs of buying and renting ($)

Iteration (Run) ▼ ☐ Totals

Buy or rent ▼ ▷ ☐ Totals

	Buy	Rent
1	-95.31K	-107.6K
2	-17.64K	-108.8K
3	-94.02K	-106.3K
4	-32.29K	-108.1K
5	-126.4K	-106.6K
6	-43.28K	-108.8K
7	-30.69K	-105.8K
8	-36.65K	-106.3K
9	-75.48K	-109K
10	-65.84K	-107.3K
11	-60.36K	-106.9K

The table above lists the 100 sample values that Analytica randomly generated from the probability distribution to estimate the statistics.

A sample size of 100 is adequate for most applications; however, if you need more precise estimates, you can increase the sample size. See "Uncertainty Setup dialog box" in Chapter 13 of the *Analytica User Guide*.

9. Click the **Result** window's close button to return to the **Diagram** window.

Result - Costs of buying and renting

Sample of Costs of buying and renting ($)

Iteration (Run) ▼ ☐ Totals

Buy or rent ▼ ▷ ☐ Totals

Using the Rent vs. Buy model: summary

You have now used the *Rent vs. Buy* model to calculate the results of a model, change input values and probability distributions, and display the uncertain results in a variety of ways. These are the basic techniques for using any quantitative model.

After you create your own models, you might want to give them a top-level input and output diagram like the one used in this chapter. For information about customizing a model for end users, see the *Analytica User Guide*, Chapter 9.

In the next chapter, you will navigate the details of the *Rent vs. Buy* model, exploring its structure and contents.

Saving your model

If you want to save changes to your model, you can do so at this point. (For instructions on quitting without saving, see the next section.)

Chapter **1** *Using the Rent vs. Buy Model*

Saving your model

File	Edit	Object	Definition	Result	Diagram

New Model	Ctrl+N
Open Model...	Ctrl+O
Add Module...	Ctrl+L
Add Library...	
Close	Ctrl+W
Close Model	
Save	**Ctrl+S**
Save As...	
Save A Copy In...	
Import...	
Export...	
Print Setup...	
Print Preview	
Print...	Ctrl+P
Print Report...	
Rent vs Buy Model.ana	
Mo1.ANA	
Rent vs Buy Model2.ana	
Rent vs Buy Model.ana Copy.ANA	
My Expenses at work.ANA	
EW 5723[1].ANA	
Exit	Ctrl+Q

1. Select **Save** from the **File** menu.

   You can also type the keyboard shortcut, *Control+S*.
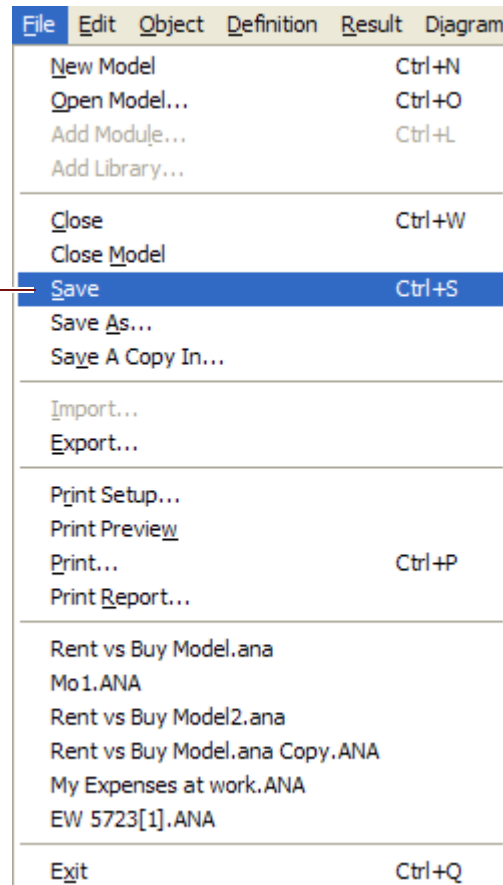
If you wish to save your model as a different file, so that you do not change the original model, select **Save As** from the **File** menu.

Quitting Analytica

When you have finished using a model, you might want to quit Analytica.

1. Select **Exit** from the **File** menu.

File	Edit	Object	Definition	Result	Diagram

New Model	Ctrl+N
Open Model...	Ctrl+O
Add Module...	Ctrl+L
Add Library...	
Close	Ctrl+W
Close Model	
Save	Ctrl+S
Save As...	
Save A Copy In...	
Import...	
Export...	
Print Setup...	
Print Preview	
Print...	Ctrl+P
Print Report...	
Rent vs Buy Model.ana	
Mo1.ANA	
Rent vs Buy Model2.ana	
Rent vs Buy Model.ana Copy.ANA	
My Expenses at work.ANA	
EW 5723[1].ANA	
Exit	Ctrl+Q

Chapter 2

Exploring the Rent vs. Buy Model

This chapter shows you how to explore a model by examining its:

- Influence diagrams
- Variables
- Attributes
- Definitions
- Results
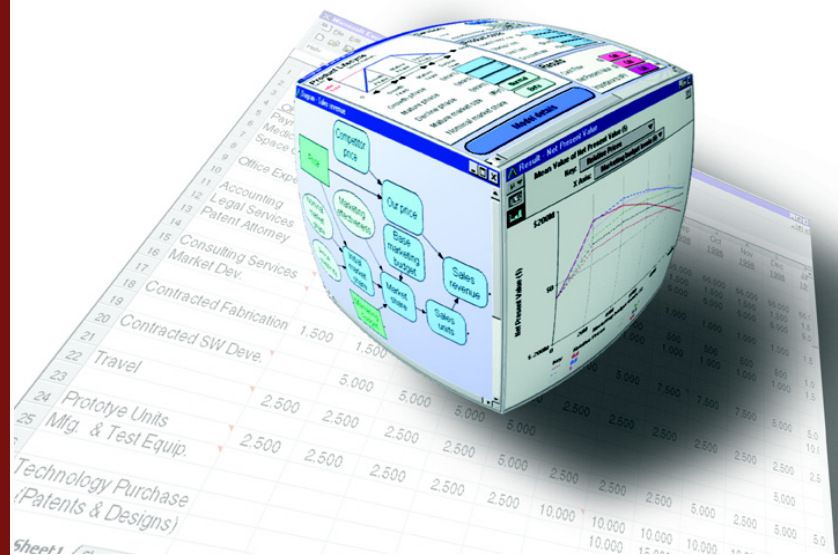
This chapter assumes you have started Analytica and have opened the *Rent vs. Buy* model. If this is not the case, see "Opening the Rent vs. Buy model" on page 8. If you are using the model as modified from Chapter 1, change the value of *Time horizon* back to **10**, the value of *Monthly rent* back to **1200**, and the value of *Buying price* back to **140K**. Also change the *Rate of inflation* back to a normal distribution with a mean of **3.5** and a standard deviation of **1.3**.

In this chapter, you will examine the structure and contents of the *Rent vs. Buy* model.

The *Rent vs. Buy* model uses financial flow conventions: funds flowing in (received) have positive values; funds flowing out (expended) have negative values.

Recognizing influence diagrams

In this chapter, you will delve into some of the details of the *Rent vs. Buy* model. You will not use the top diagram that you used in Chapter 1.
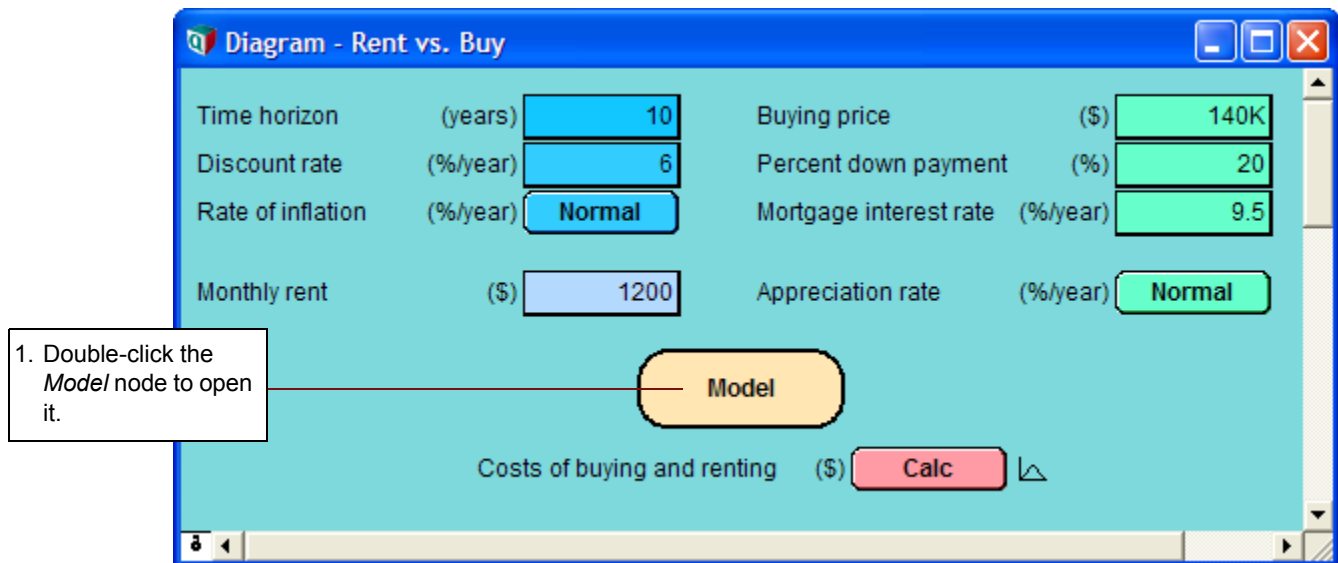
The details of an Analytica model display in an ***influence diagram*** window. An influence diagram (shown on the next page) is a graphical representation of a model, showing how different variables in the model interact with each other. A typical influence diagram consists of a number of ***nodes*** connected by ***arrows***.

Tools palette ——

Nodes ——

Influence
diagram
window ——
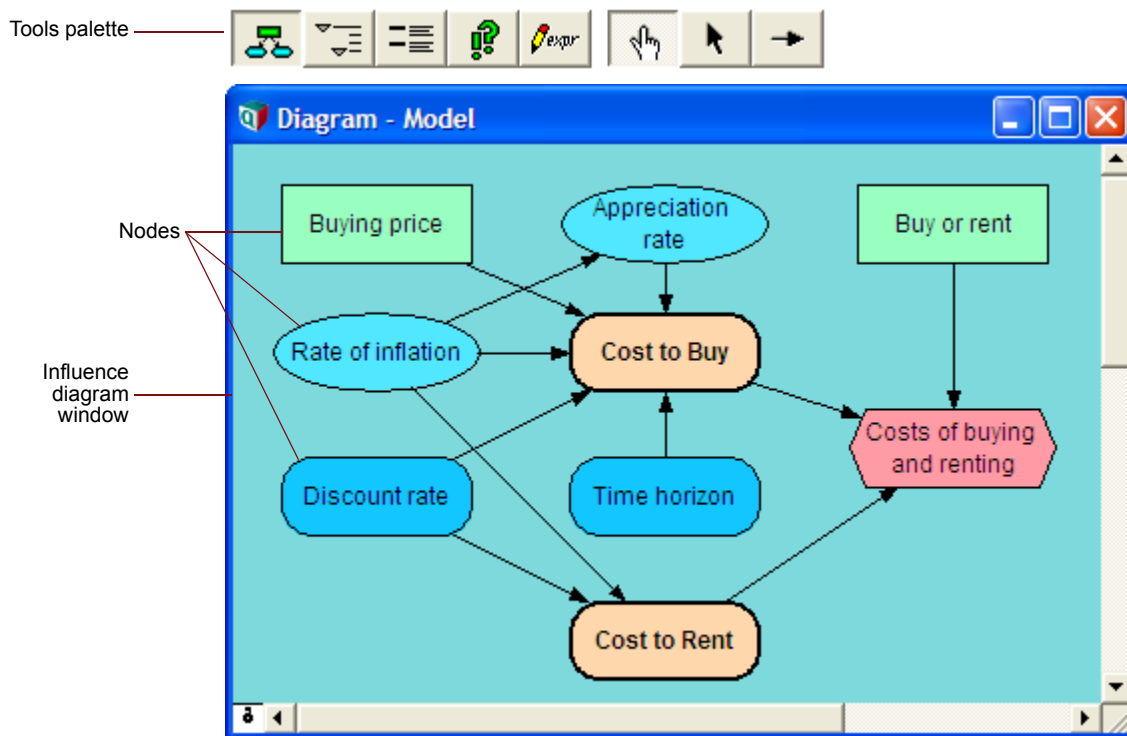
Nodes represent variables and appear as boxes, ovals, hexagons, and other shapes. Different node shapes represent different types of variables. Analytica uses the term ***variable*** broadly to include anything that has a value or can be evaluated. Note that many of the variables have the same names as the inputs and output at the top diagram that you used in Chapter 1. The top diagram provides an easy way to see and change these nodes' values.

Arrows connecting different variables indicate a relation between the variables. The arrow connecting *Rate of inflation* to *Appreciation rate* indicates that the value of the *Appreciation rate* variable depends on the value of the *Rate of inflation* variable. In the *Rent vs. Buy* model influence diagram, *Cost to Buy* depends on the *Buying price*, *Rate of inflation*, *Appreciation rate*, *Discount rate*, and *Time horizon* variables.

The following figure illustrates different types of nodes.

A *general variable* is represented by a rounded rectangle. It can represent any type of variable and is useful when you don't know what the type is. Typically, a general variable is used to represent a deterministic quantity or functional relationship.

A *decision variable* is represented by a rectangular node. A decision variable is directly under the control of the decision maker.

A *chance variable* is represented by an oval node. A chance variable cannot be controlled directly by the decision maker. It has an uncertain value represented by a probability distribution.
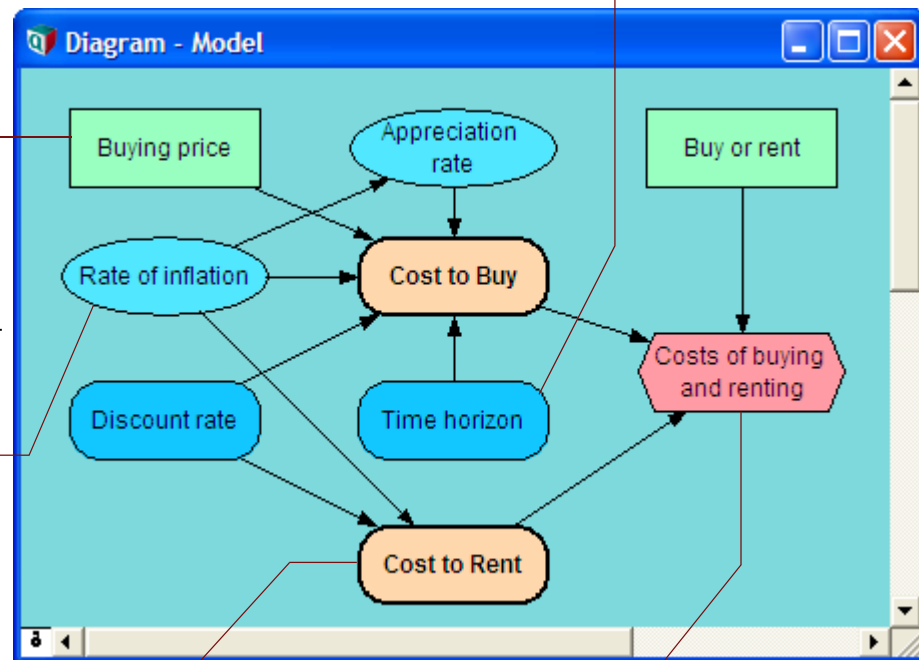
A *module* is represented by a thick-lined rounded rectangle. A module contains its own influence diagram, allowing nesting of multiple modules within a model.

An *objective variable* is represented by a hexagon. This variable is the model's "goal" and evaluates the overall value or desirability of possible outcomes. In this model, the goal is to evaluate the cost difference between renting and buying. A decision model usually contains a single objective variable.

Opening Object windows

Every object in Analytica has an associated ***Object window*** containing detailed information about it. You can display the **Object** window of any variable by double-clicking its node in the influence diagram.

1. Double-click the *Buying price* node to open the *Buying price* **Object** window.
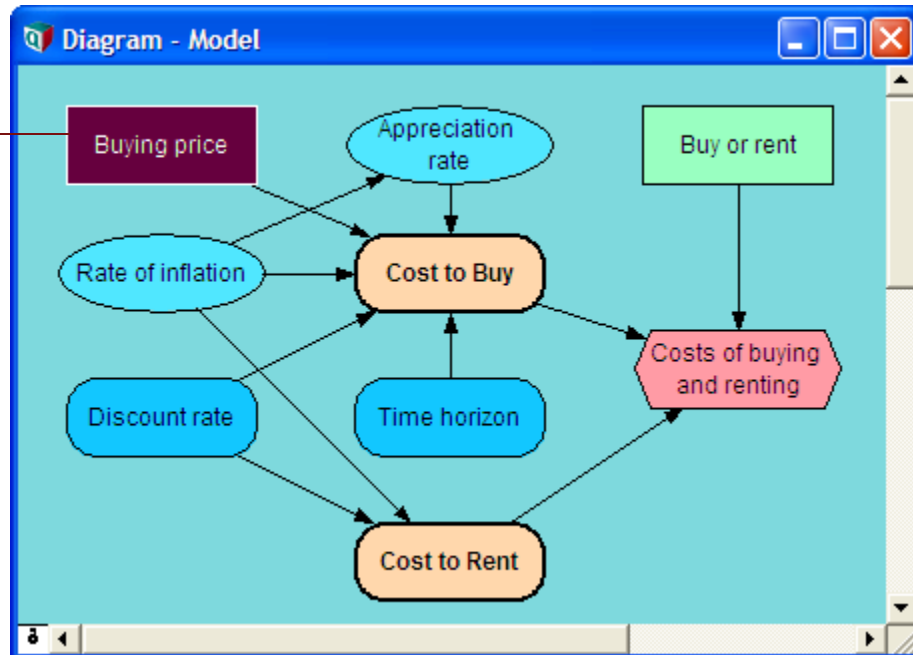
Information about a variable is provided in a list of attributes. Attributes include the variable's class (for example, decision, chance, or constant), identifier, units, title, description, definition, inputs, and outputs. See the illustration on the next page.

This variable's *identifier* is *Price*. The identifier is used to refer to this variable in definitions of other variables. It can contain up to 20 characters and cannot contain spaces.

The *Units* attribute indicates the units of measurement for the variable. The *Buying Price* variable is measured in dollars.

This variable's *class* is decision variable.

The *Title* describes briefly what the variable represents. The title is also shown in the node on the influence diagram.

The *Description* provides more complete documentation (unlimited length) about this variable.

Object - Buying price

| Decision ▼ | Price | Units: $ |

Title: Buying price

Description: Buying price of house.

expr ▼

Definition: 140K

Outputs:
- Downpaymt — Down payment
- Insurance — Insurance
- Maintenance — Maintenance
- Mortgage — Mortgage loan amount
- Moving_costs — Moving costs
- Property_tax — Property taxes
- Sell_price — Selling price

The *Definition* specifies the variable value, or how to compute the value, sometimes using other variables as inputs. The definition might also be a probability distribution or any other mathematical expression.

Outputs are other variables that depend on this variable. For each output, its identifier and title are shown.

Tip   You can enter numbers with a suffix abbreviation, so *Buying Price* can be defined as either 140K or 140000. A quick reference for these suffixes is given on the back page of this tutorial.

Moving between Object windows

You have opened the **Object** window of a variable (*Buying price*) by double-clicking its node in the influence diagram.
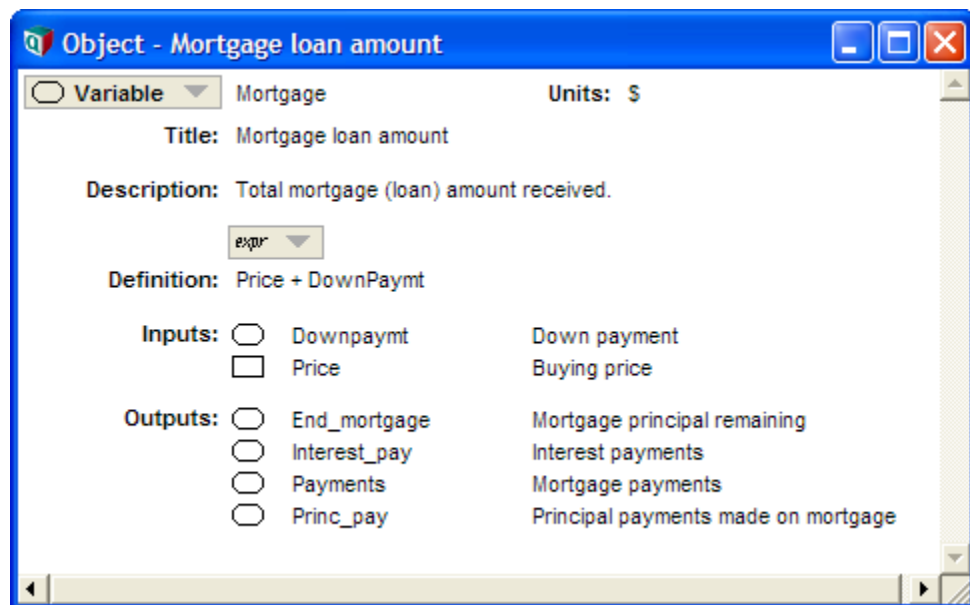
The **Object** window contains a list of the variable's *inputs* and *outputs*, if there are any.

You can open the **Object** window for any input or output variable by double-clicking the one you wish to view.

1. Double-click the output variable titled *Mortgage loan amount*.

Analytica switches to the **Object** window for *Mortgage loan amount*.

Note in the figure above that the ***Title*** of *Mortgage loan amount* is different from the variable's ***identifier***, *Mortgage*. The title is what the model user normally sees; the identifier is used as a mathematical symbol in the definitions of other variables that depend on this variable.

The ***definition*** of the *Mortgage loan amount* is an ***expression***, the sum of *Buying price* and *Down payment* (which is a negative amount). The definition refers to these variables by their identifiers.

Inputs lists the identifiers and titles of the variables in the definition. *Buying price*, the variable you just examined, is one of the inputs. The other input of *Mortgage loan amount* is *Down payment*.

2. Double-click the *Down payment* input.

Object - Mortgage loan amount

Variable ▾   Mortgage   Units: $

Title: Mortgage loan amount

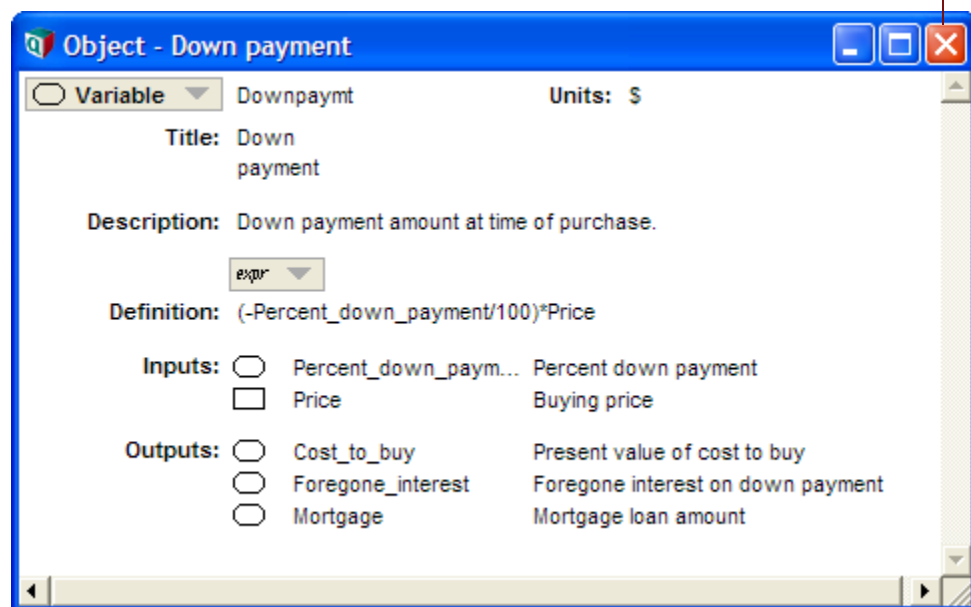Description: Total mortgage (loan) amount received.

expr ▾

Definition: Price + DownPaymt

Inputs: ◯ Downpaymt   Down payment
          ☐ Price   Buying price

Outputs: ◯ End_mortgage   Mortgage principal remaining
          ◯ Interest_pay   Interest payments
          ◯ Payments   Mortgage payments
          ◯ Princ_pay   Principal payments made on mortgage

The **Object** window now displays the attributes of *Down payment.*

3. Click the close button ⊗ to close the window and return to the diagram.

Object - Down payment

Variable ▾   Downpaymt   Units: $

Title: Down payment

Description: Down payment amount at time of purchase.

expr ▾

Definition: (-Percent_down_payment/100)*Price

Inputs: ◯ Percent_down_paym... Percent down payment
          ☐ Price   Buying price

Outputs: ◯ Cost_to_buy   Present value of cost to buy
          ◯ Foregone_interest   Foregone interest on down payment
   ◯ Mortgage   Mortgage loan amount
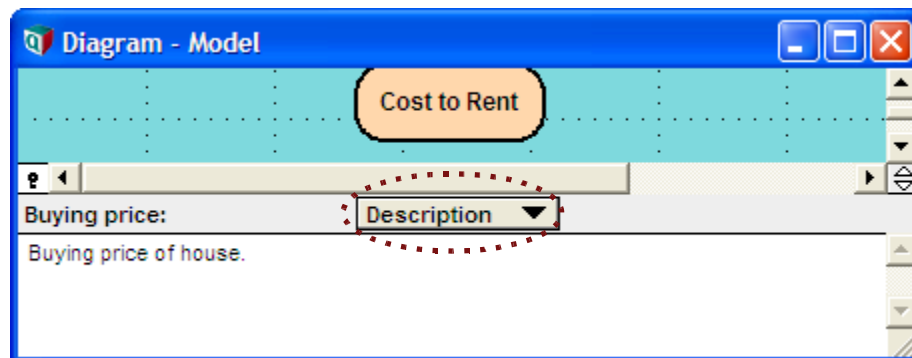
Using the Attribute panel

As an alternative to viewing a variable's attributes in a separate window, you can inspect them in the **Attribute panel**, which is an auxiliary window pane that you can open below the influence diagram.

The **Attribute** panel allows you to rapidly examine one attribute at a time of any variable in the model. You select the variable you wish to view and select the attribute to examine from a popup menu.

The variable *Buying price* should be highlighted with a title in white, indicating that it is selected; if it is not, select it by clicking it once.

1. Click the key icon ♂ to open the **Attribute** panel.
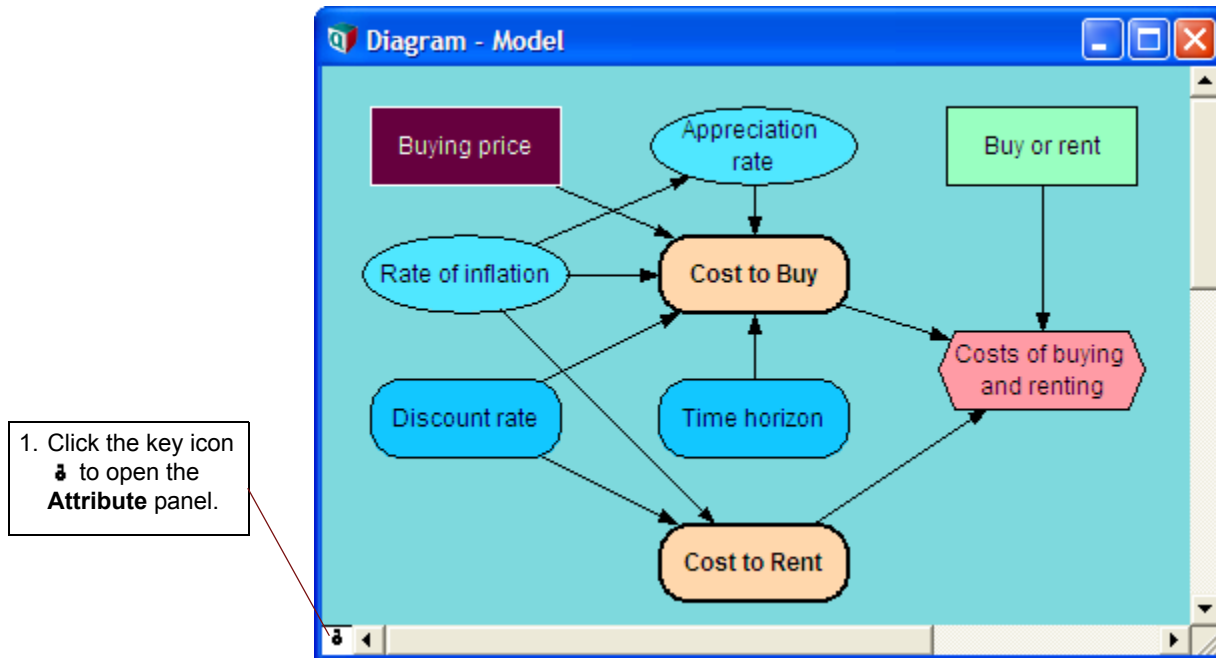
By default, Analytica displays the ***description*** of the selected node (e.g., *Buying price*) in the **Attribute** panel.
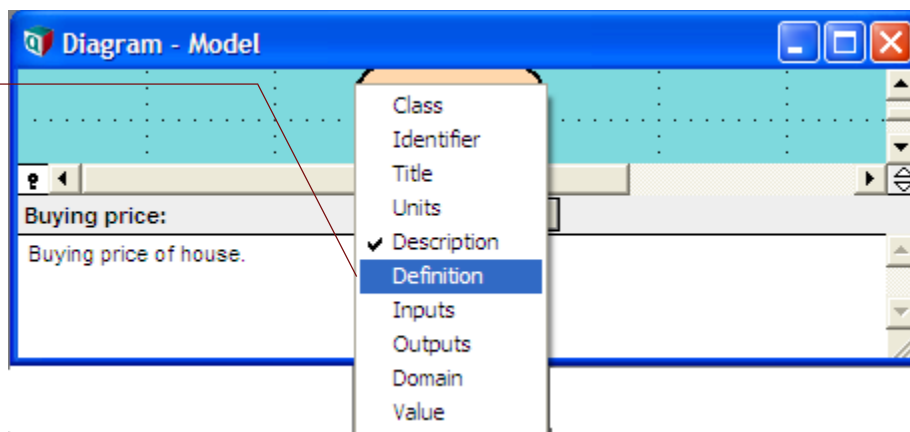

Inspecting definitions in the Attribute panel

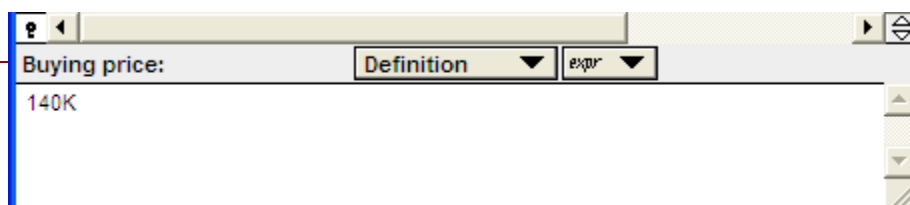The **Attribute** panel allows you to inspect any attribute of a variable.

In this section, you will see the definition of two variables that you viewed in the top-level diagram in Chapter 1.

1. Click **Description** to view the Attribute popup menu.
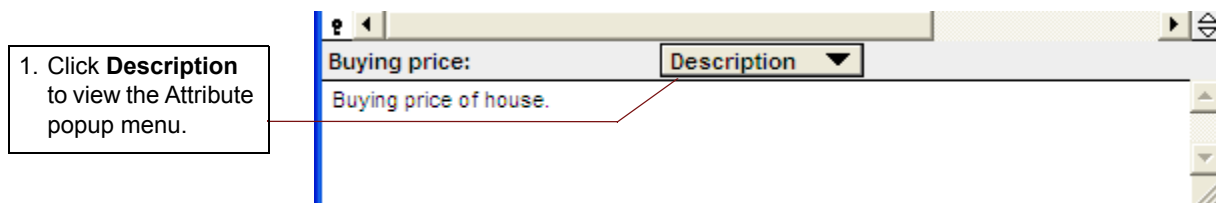
2. Select **Definition** as the new current attribute to display.

When a variable is defined as an ***uncertainty distribution***, a button appears in the **Definition** field.

3. The definition of *Buying price* is displayed. It is a single number: 140K (140,000).

4. Select the *Rate of inflation* node.

   The definition in this example is a normal distribution; hence, the button is labeled **Normal**.
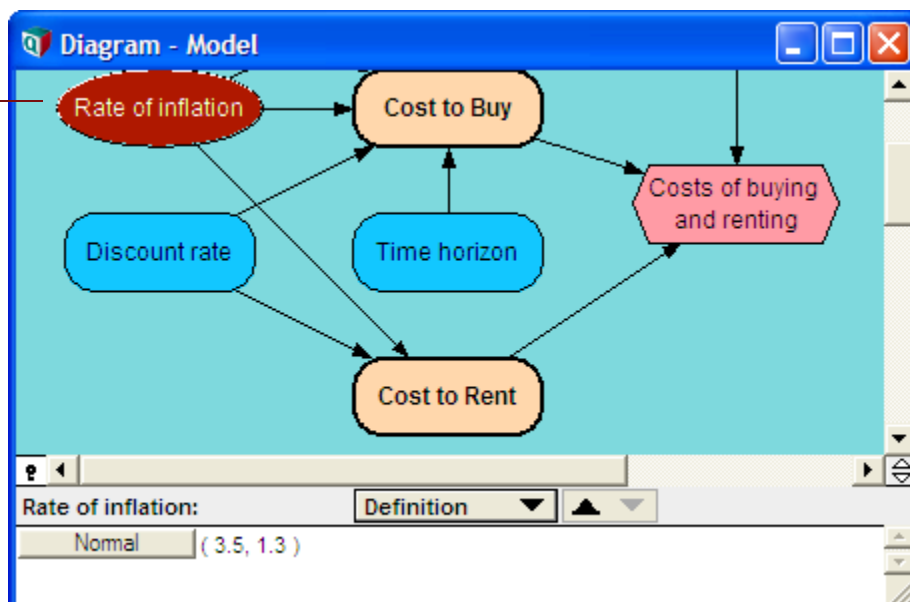
In Chapter 1 you saw that *Rate of inflation* is defined as a normal distribution with a mean of 3.5 and a standard deviation of 1.3. These are the input parameters to the normal distribution, and they display to the right of the **Normal** button.

Opening modules

Analytica models generally contain ***modules***. Each module contains the details of a part of the model, also represented as an influence diagram. In the *Rent vs. Buy* model, *Cost to Buy* and *Cost to Rent* are both modules.

Modules can also contain other modules. In this manner, a large model with hundreds of variables can be organized into a hierarchy of modules, each small enough to be easily understood.

1. Double-click the *Cost to Buy* node to open the module.
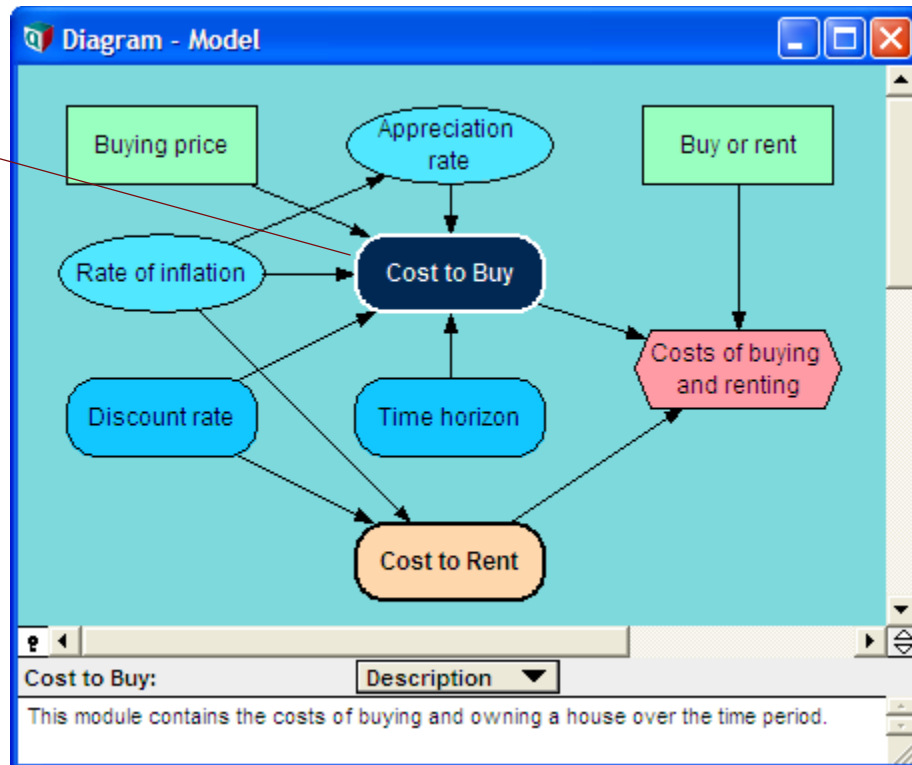
Analytica displays the influence diagram of the *Cost to Buy* module. This module contains three additional modules: *Out-of-pocket costs to own*, *Future sales proceeds*, and *Opportunity cost*.

Output arrowhead
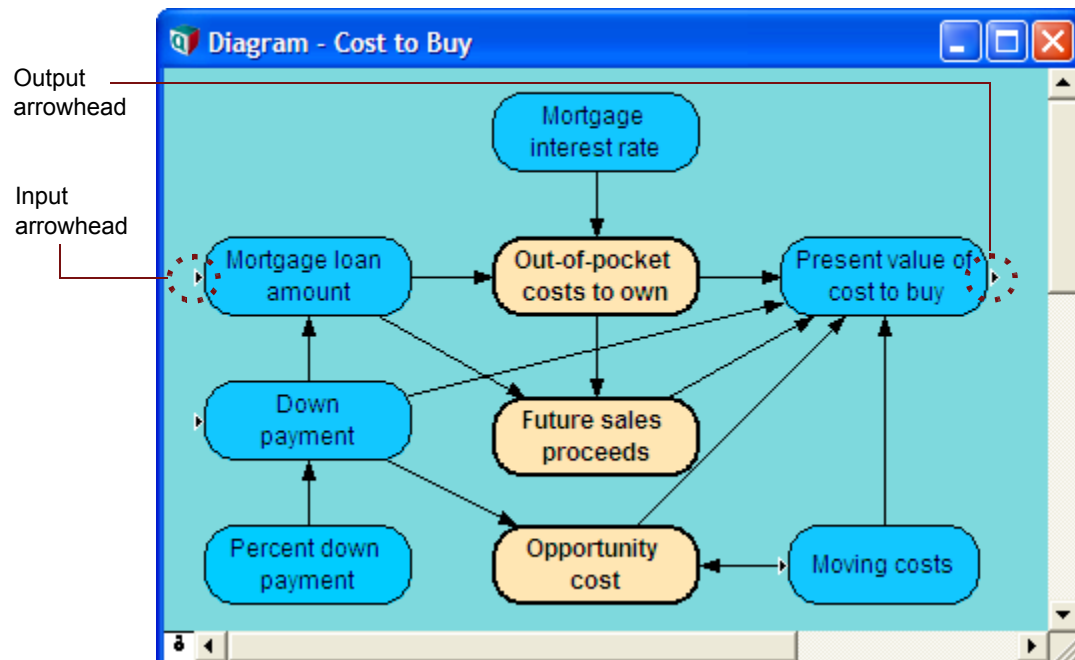
Input arrowhead

The ***input arrowhead*** (without a trailing line) shows that the node to the right of the arrow has one or more inputs from outside this module.
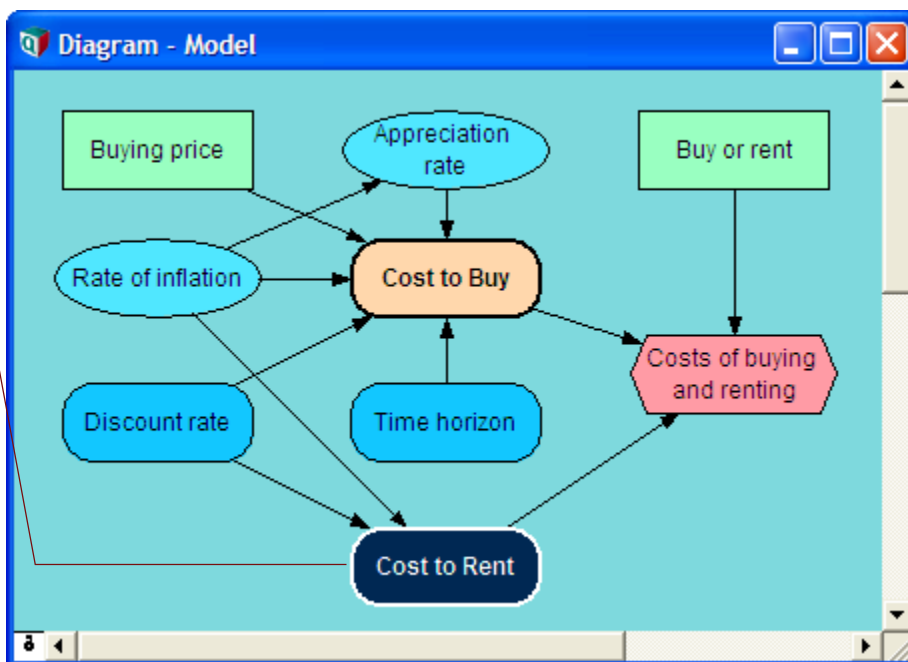
The *output arrowhead* shows that the node to the left of the arrow has one or more outputs outside this module.

2. Click the **Diagram** button to return to the parent diagram, *Model*.

3. Double-click the *Cost to Rent* node to open the module.

   The *Cost to Rent* diagram opens (see the figure below).

Analytica limits the number of open windows at each level of the model hierarchy to minimize clutter on your screen. See "Managing Windows" in Chapter 19 of the *Analytica User Guide* for information on how to open more than one module **Diagram** window at a time.

4. Click the **Diagram** button to return to the parent diagram, *Model*.
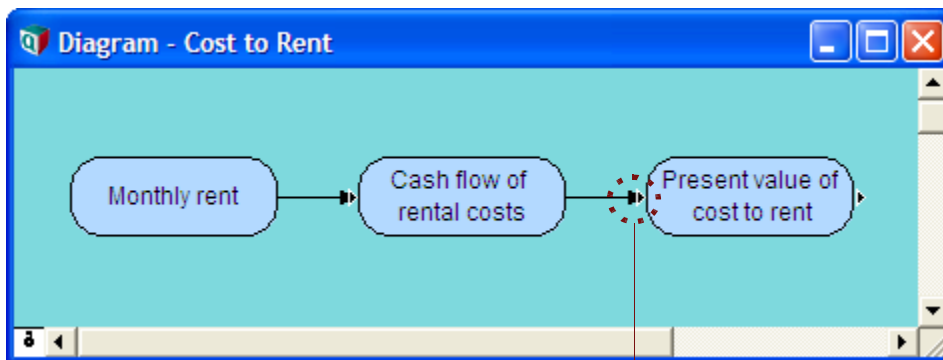
Combined arrowhead

The combined arrowhead, shown above, indicates that the node has one or more inputs from outside this module, plus the input variable in this module.

You can also navigate the model by tracking a variable's inputs or outputs.

5. Select the *Buying price* node.

6. Select **Outputs** from the Attribute popup menu to view a list of variables that depend on the *Buying price* variable.

7. Double-click *Insurance*.
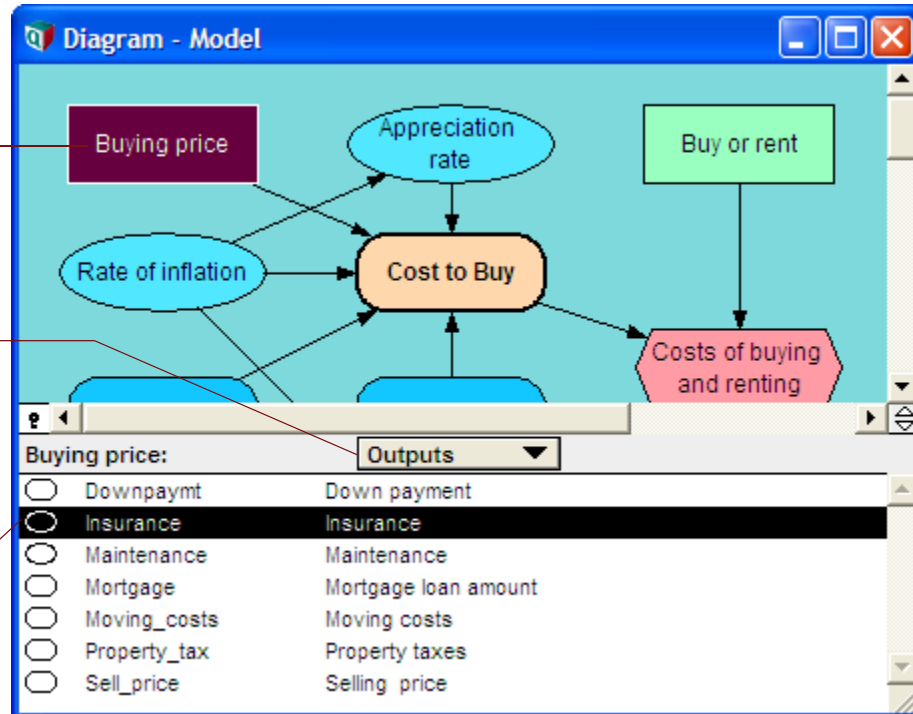
The *Out-of-pocket costs to own* module diagram is brought to the front, with the *Insurance* node selected.

8. Click the **Diagram** button to view the *Cost to buy* diagram.
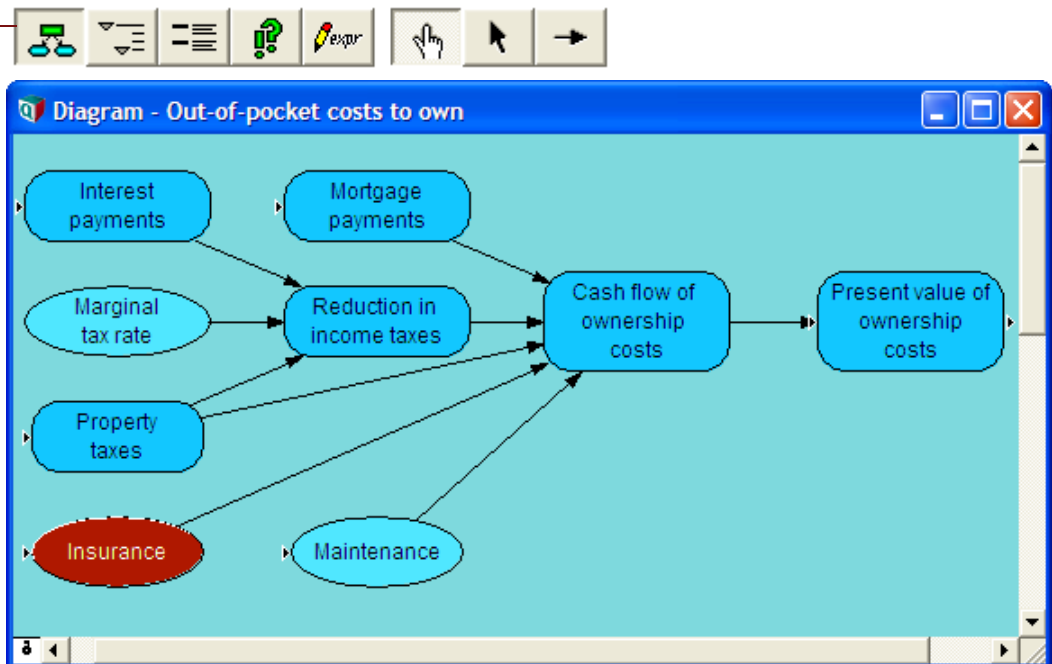
Inspecting values in the Attribute panel

The **Attribute** panel allows you to view certain attributes, such as a variable's *value*, that are not (initially) displayed in an **Object** window.

Chapter **2** *Exploring the Rent vs. Buy Model*

Inspecting values in the Attribute panel

1. Select the *Present value of cost to buy* node.

2. Click the key icon 🔑 to open the **Attribute** panel.

3. Press the Attribute popup menu and select **Value**.

The deterministic (or mid) value of *Present value of cost to buy* displays, in this case, -67.2K.

Diagram - Cost to Buy

Mortgage interest rate

Mortgage loan amount

Out-of-pocket costs to own

Present value of cost to buy

Down payment

Future sales proceeds

Percent down payment

Opportunity

Moving costs

Class
Identifier
Title
Units
✓ Description
Definition
Inputs
Outputs
Domain
Value
Probvalue

Present value of cost to

Total present value of the cash fl[o]... including the down payment, annual out-of-pocket costs, proce... [a]nd forgone interest on the down payment (opportunity cost).

Present value of cost to   Value

-67.2K

If *Value* was not previously computed, Analytica computes the variable's value *deterministically*, assuming that all of the input probability distributions are fixed at their median values. *Mid value* is an abbreviation for this deterministically computed value.

You can use the **Attribute** panel in this manner to examine the mid value of any variable in the model.

It is faster to compute a mid (deterministic) value than an uncertain (probabilistic) value, so it is useful for conducting initial checks of a model before performing any uncertainty analysis.

Displaying results

When you are viewing a model's influence diagram, you can evaluate any variable and display its value in a **Result** window.
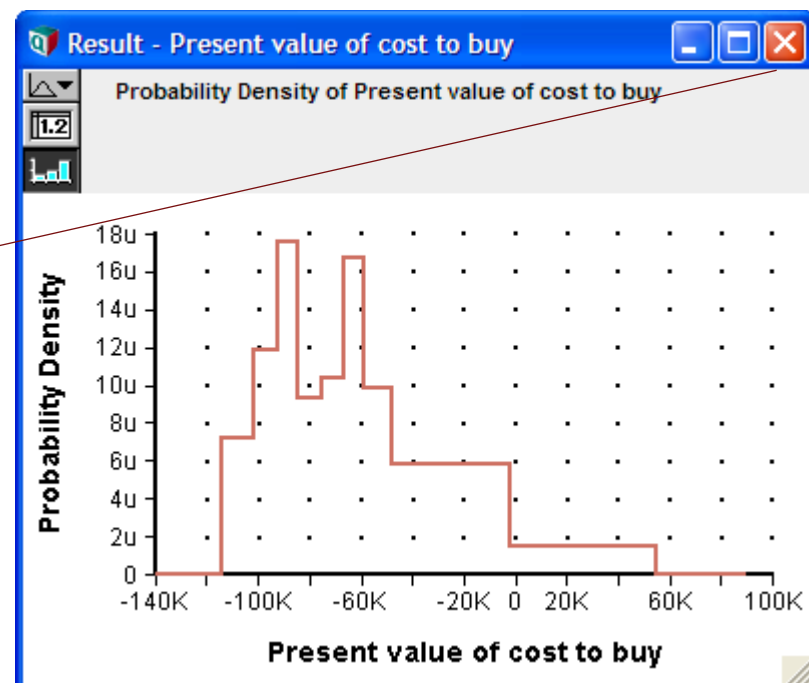
1. With *Present value of cost to buy* still selected, click the **Result** button to evaluate it.
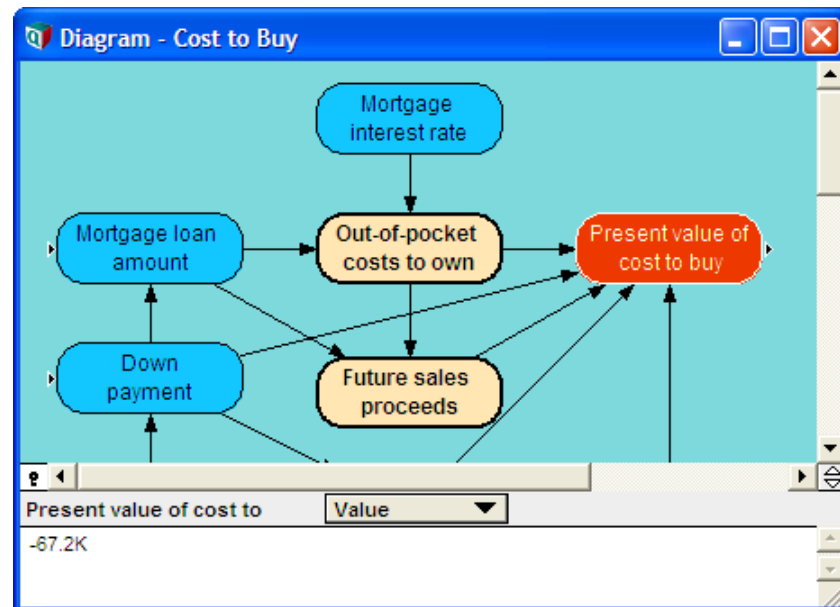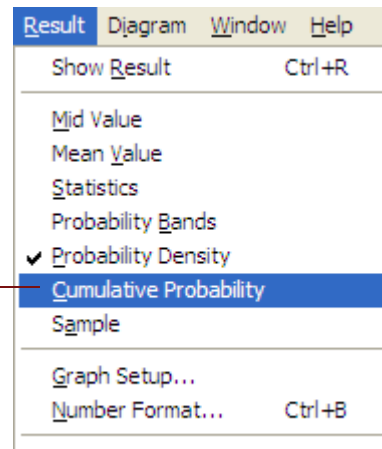
A **Result** window displays the probability density function graph for this variable. Analytica displays the uncertainty view that was most recently selected from the Uncertainty View popup menu, or that was saved with the model.

2. Click the close button to close the **Result** window.

As an alternative to clicking the **Result** button [icon] and then selecting an uncertainty view, you can use the **Result** menu to evaluate a variable and select the uncertainty view of the result.

3. With *Present value of cost to buy* still selected, select the **Result** menu. The check mark next to **Probability Density** indicates that the **Probability Density** was last displayed. Select **Cumulative Probability**.

Result	Diagram	Window	Help
Show Result			Ctrl+R

Mid Value
Mean Value
Statistics
Probability Bands
✔ Probability Density
Cumulative Probability
Sample

Graph Setup...
Number Format...   Ctrl+B

The **Result** window appears displaying the variable's cumulative probability distribution.

4. Click the **Diagram** button [icon] to display the *Cost to buy* diagram.

Result - Present value of cost to buy

Cumulative Probability of Present value of cost to buy
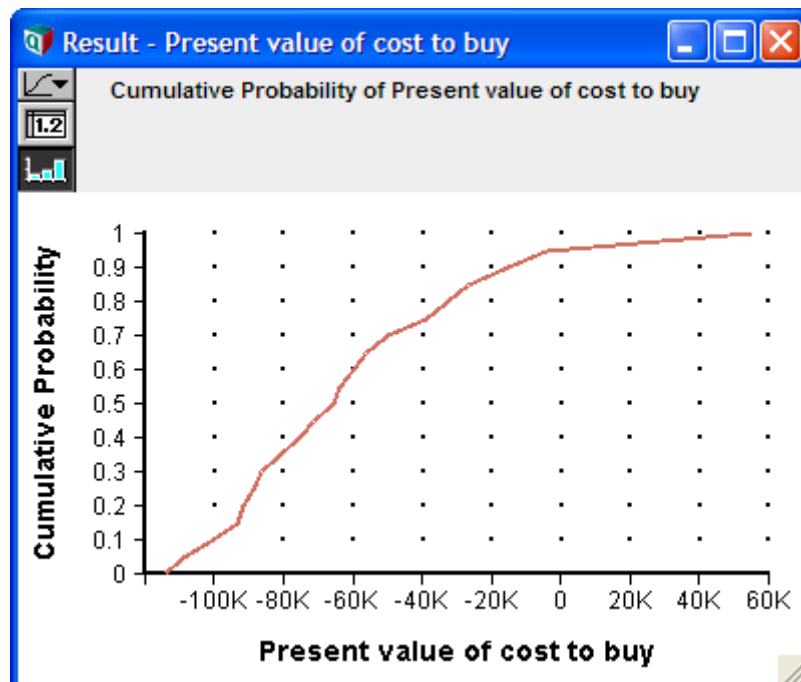
5. Select *Moving costs* and then click the **Result** button [icon] to evaluate.
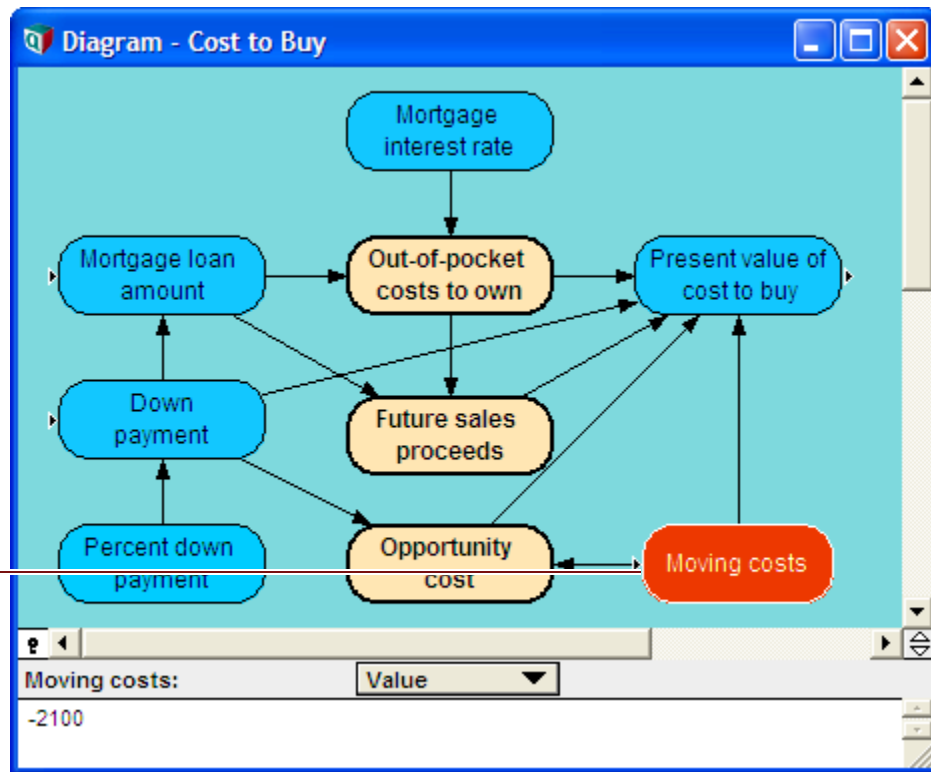
A single mid value appears in table view.

Uncertainty View Popup Menu

The mid value in table view is the only result view available for a nonprobabilistic variable with a single value.

6. Select **Probability Density** from the Uncertainty View popup menu.
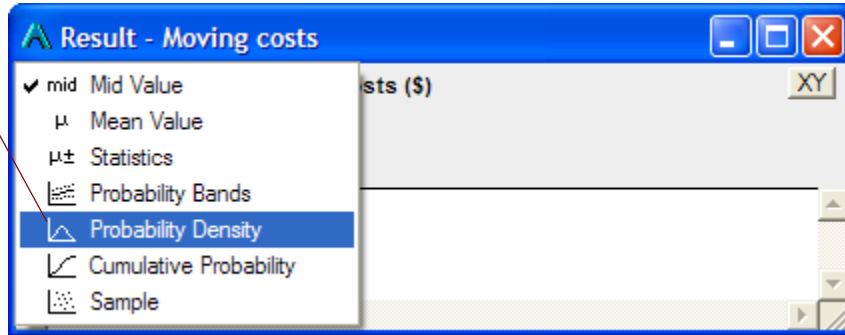
Analytica tells you that this is a nonprobabilistic value.

7. Click the **OK** button.
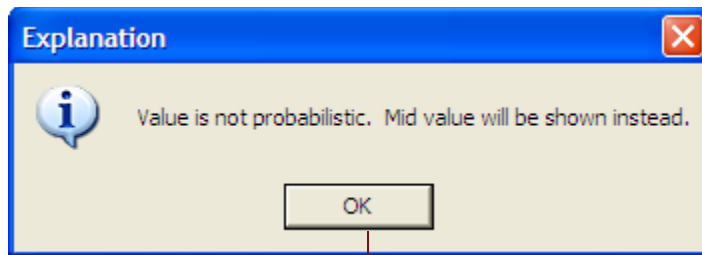
Exploring the Rent vs. Buy model: summary

You now have browsed the *Rent vs. Buy* model by examining its influence diagrams, variables, attributes, definitions, and results. These are the basic techniques for exploring any Analytica model.

The next chapter shows you how to analyze the *Rent vs. Buy* model.

You can quit Analytica at this point. See "Quitting Analytica" on page 23.

Chapter 3

Analyzing the Rent vs. Buy Analysis Model

This chapter shows you how to:

- Perform importance analysis
- Perform parametric analysis
- Set up and compare alternative decisions

In this chapter you will analyze the *Rent vs. Buy Analysis* model, a modified version of the model that you used in Chapter 1, "Using the Rent vs. Buy Model" and Chapter 2, "Exploring the Rent vs. Buy Model." You will identify its key sources of uncertainty through **importance analysis, perform parametric analysis**, and **compare alternative** decisions.

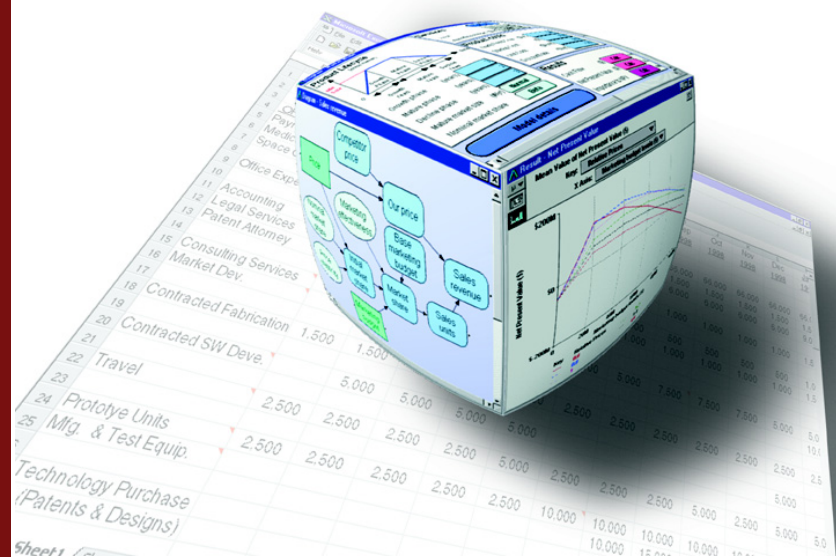For instructions on how to open a model, see "Opening the Rent vs. Buy model" on page 8. In this case, however, open the *Rent vs. Buy Analysis* model by double-clicking the file labeled `Rent vs. Buy Analysis.ana`.

Examining the difference between renting and buying

The *Rent vs. Buy Analysis* model is the module called *Model* that you explored in Chapter 2, "Exploring the Rent vs. Buy Model," with the addition of nodes to help you understand the importance of the uncertain inputs to the uncertainty in the output.

In Chapter 1, "Using the Rent vs. Buy Model," you saw that evaluating *Costs of buying and renting* produces a graph of two uncertain values. To understand whether it would be financially advantageous to rent or buy, the *Rent vs. Buy Analysis* model includes the objective node, *Difference between buying and renting*.

1. Click the *Difference between buying and renting* node to select it.

2. Click the **Result** button to evaluate it.
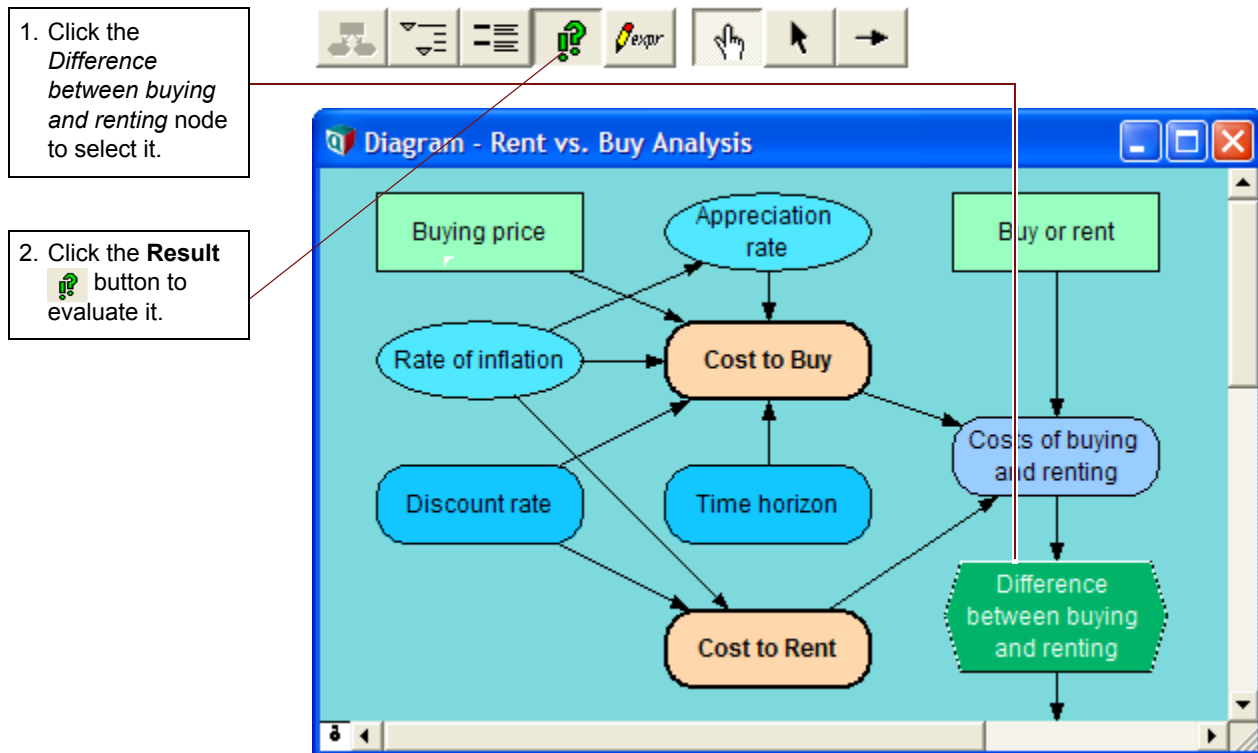
The difference between the two uncertain values is also uncertain. The difference is positive if buying costs less over the time period, and negative if renting costs less over the time period.

3. Click the **Diagram** button 🔲 to return to the *Rent vs. Buy Analysis Diagram* window.
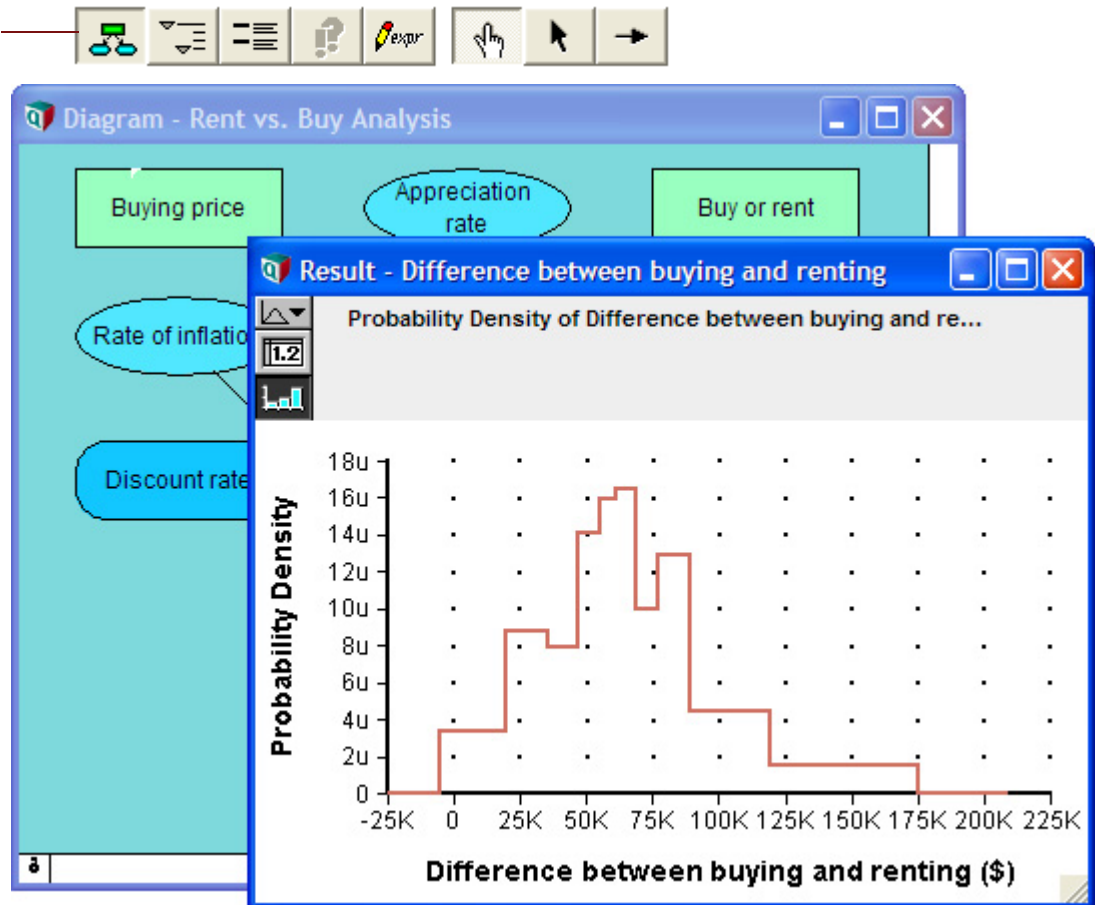
Importance analysis

In the *Rent vs. Buy Analysis* model, as in most complex models, several of the input variables are uncertain.

It is often useful to understand how much each uncertain input contributes to the uncertainty in the output. Typically, a few key uncertain inputs are responsible for the lion's share of the uncertainty in the output, while the rest of the inputs have little impact.

Analytica's ***importance analysis*** features can help you understand which uncertain inputs contribute most to the uncertainty in the output. You can then concentrate on getting more precise estimates or building a more detailed model for the one or two most "important" inputs.

1. Select the *Difference between buying and renting* node.

2. Click the **Result** button to display the importance values.
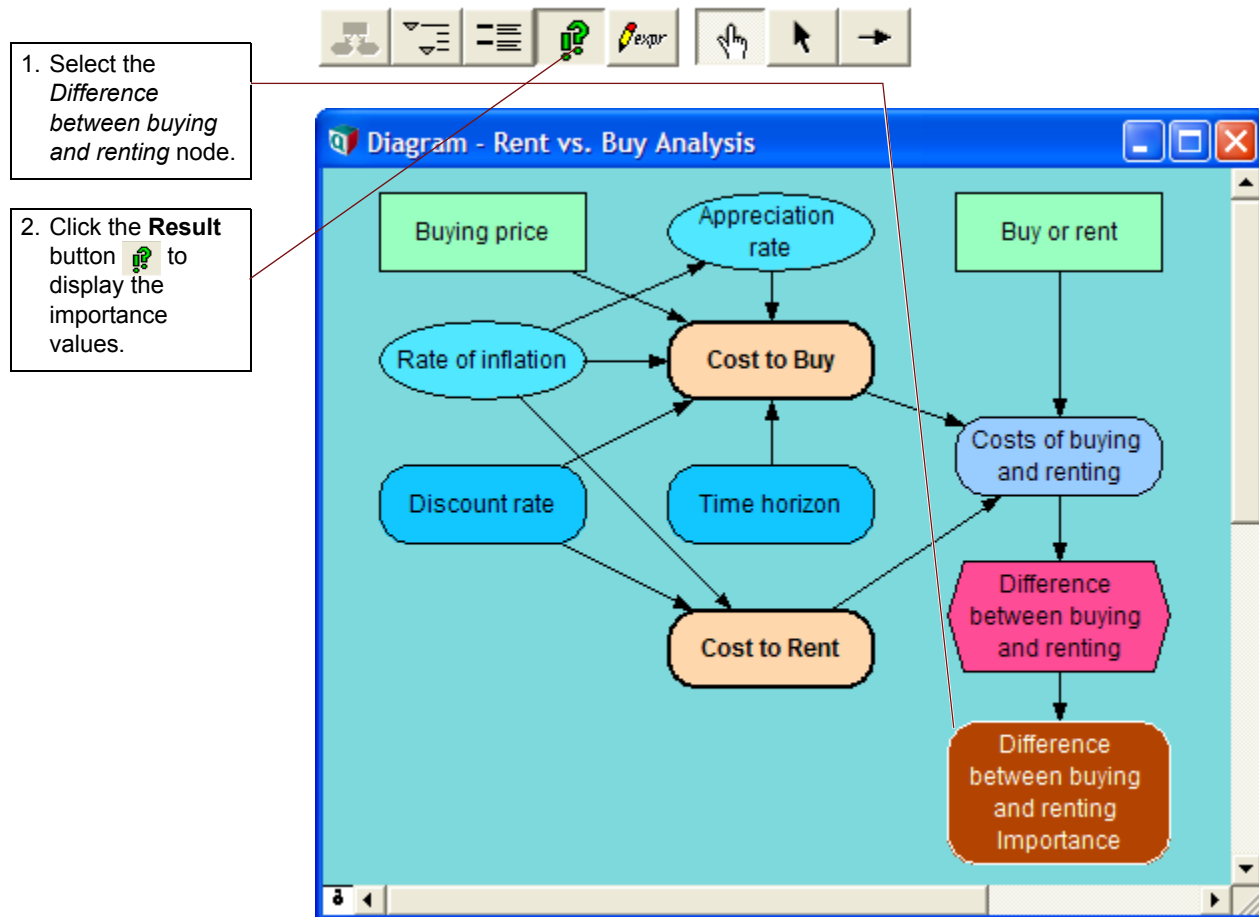
Analytica defines **importance** as the rank order correlation between the output value and each uncertain input. Each variable's importance is calculated on a relative scale from 0 to 1. An importance value of 0 indicates that the uncertain input variable has no effect on the uncertainty in the output. A value of 1 implies total correlation, where all of the uncertainty in the output is due to the uncertainty of a single input.
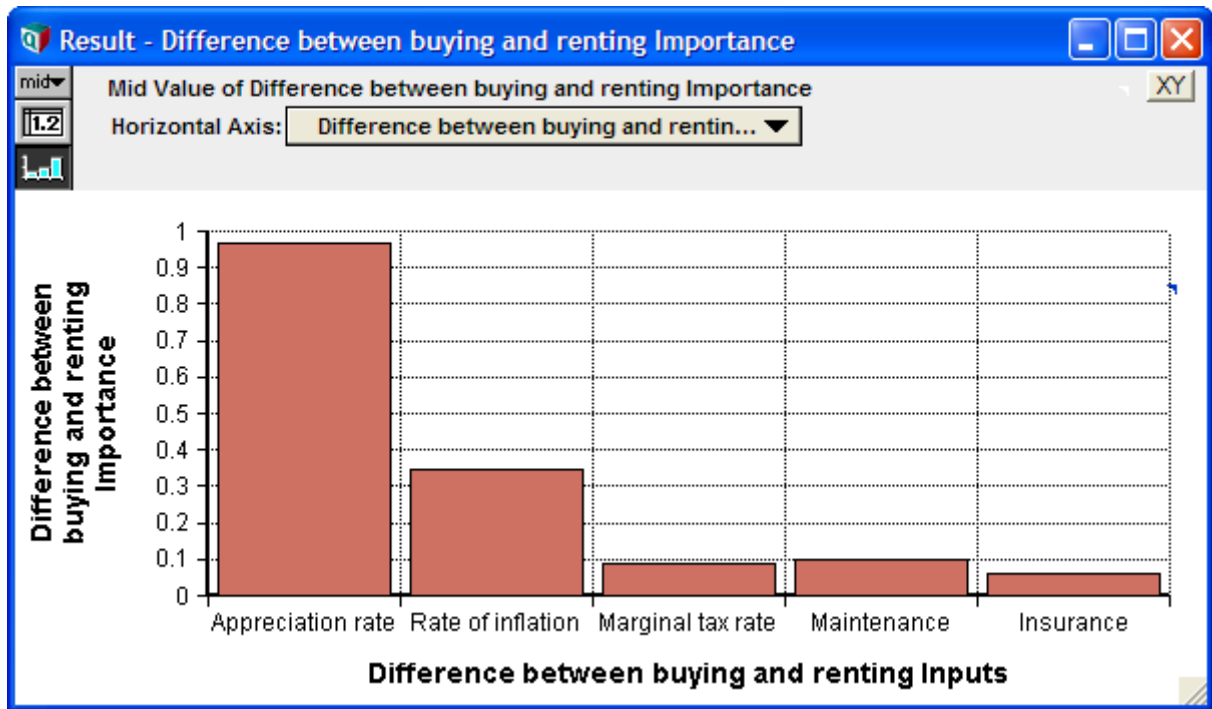
It is clear in the figure above that the input *Appreciation Rate* is contributing most of the uncertainty in the *Difference between buying and renting*.

3. Click the **Diagram** button to return to the *Rent vs. Buy Analysis* **Diagram** window.
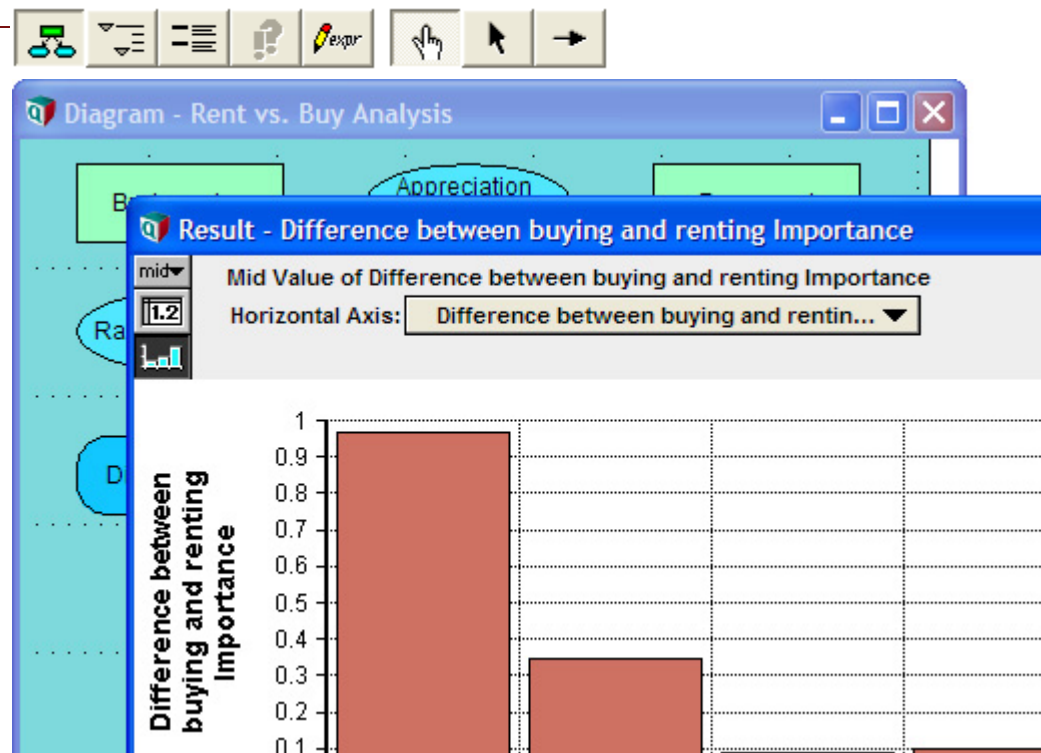
For more information about importance analysis and the steps to create an importance variable in your own model, see "Scatter plots" in the "Sensitivity and Uncertainty Analysis" chapter of the *Analytica User Guide*.

Performing parametric (sensitivity) analysis

Parametric analysis (also called *sensitivity analysis*) involves varying the value of an input variable to examine its effect on a selected output. Performing sensitivity analysis often provides useful insights into how small changes in input variable values affect the desired outcome.

Because the importance analysis in the section "Importance analysis" revealed that *Appreciation rate* caused most of the uncertainty in *Difference between buying and renting,* you will start the parametric analysis with that input variable. You will change *Appreciation rate*'s definition from a probability distribution to a list of alternative values, and analyze the effect on the *Difference between buying and renting* output.
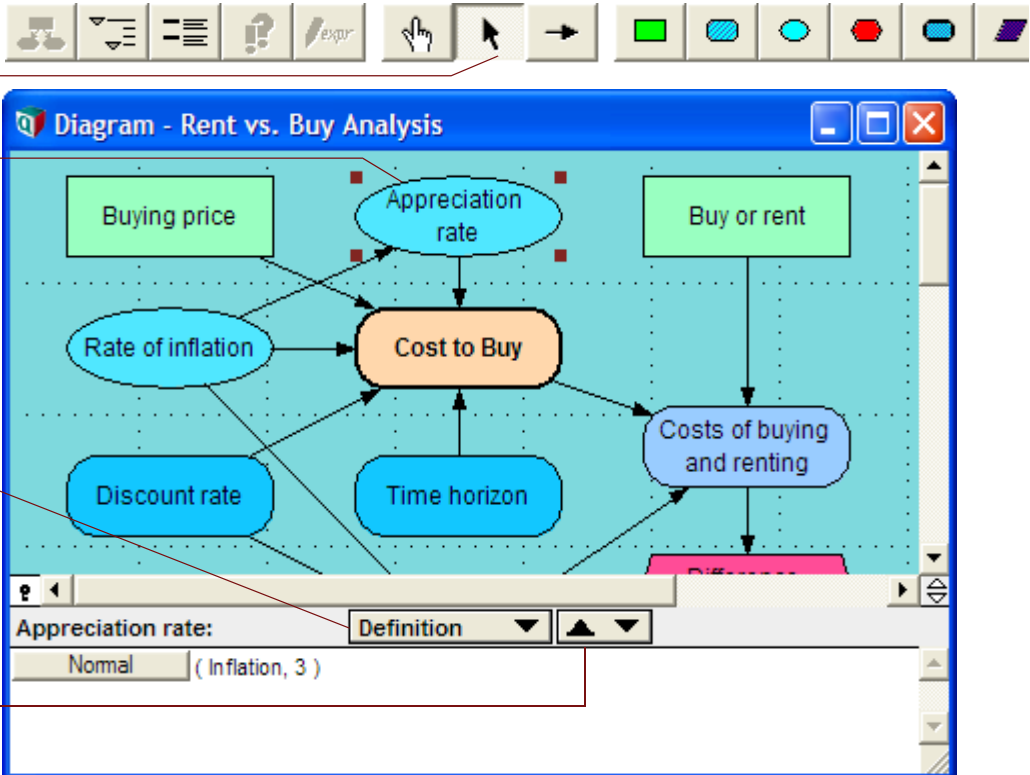
Before proceeding, click the edit button ⬆ in the toolbar to switch into edit mode. In edit mode you can modify the model: adding and removing nodes, and modifying existing nodes. Then click the key icon 🔑 to open the **Attribute** panel, then select the *Appreciation rate* node, and then select **Definition** from the Attribute popup menu to view its definition.

1. Click the **edit** button to enter edit mode.

2. Select the *Appreciation rate* node.

3. Select **Definition** from the Attribute popup menu to view its definition.

Expression popup menu

When the **Definition** attribute is displayed, the Expression popup menu ▲▽ appears. The distribution icon ▲ in the menu indicates that the definition is a probability distribution.

Before proceeding, click the **edit** tool ⬆ to switch to edit mode.

The Expression popup menu allows you to change the definition of a variable to one of several different types of expressions.

Expression types include:

- Expression, or mathematical formula ⊏expr▼⊐
- List ⊏▤▼⊐
- List of Labels ⊏▦▼⊐
- Table ⊏▥▼⊐
- Probability table ⊏▥▼⊐
- Distribution ⊏▲▽⊐

- Choice

You will now use the Expression popup menu to change the definition of *Appreciation rate* from a probability distribution to a list. You will redefine *Appreciation rate* as a list of alternative values from -10% to 10%.

<table>
<tr>
<td>

4. While pressing on the Expression popup menu, drag the mouse to **List**, and release the mouse button to select **List**.

</td>
<td>

expr Expression

☰ List

🔢 Table

🔢 Probability Table

✓ ▲ Distribution

▱ Other...

Appreciation rate: Definition ▼

Normal (Inflation, 3)

</td>
</tr>
<tr>
<td>

5. Click the **OK** button or press the *Enter* key to confirm that you want to change the definition from a distribution to a list.

</td>
<td>

Question

❓ Replace current definition with a List?

OK   Cancel

</td>
</tr>
</table>
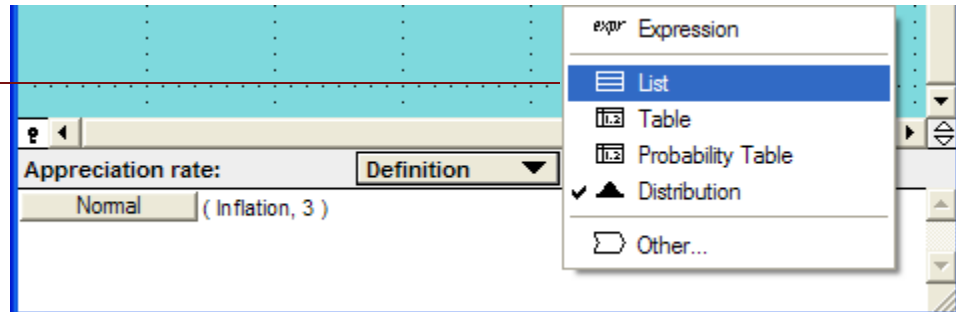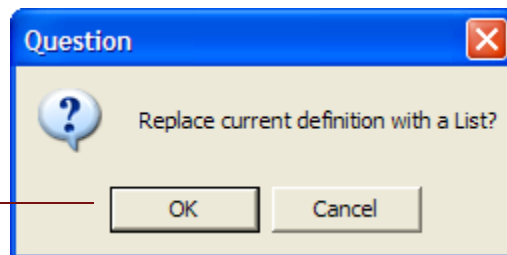
Note that the icon on the Expression popup menu changes to indicate that **List** ☰ ▼ is selected.

When a definition is first changed to a list, a cell (indicated by a box around it) appears in the definition. The first cell in the list initially contains the expression that was previously in the definition. In this case, you see the expression for a normal distribution (`Normal(Inflation,3)`).

You will replace the entry with a number and add cells to perform parametric analysis.

<table>
<tr>
<td>

6. Select the cell by clicking in it. Type the value **-10** and press the *Enter* key.

</td>
<td>

Appreciation rate: Definition ▼ ☰ ▼

-10

</td>
</tr>
</table>
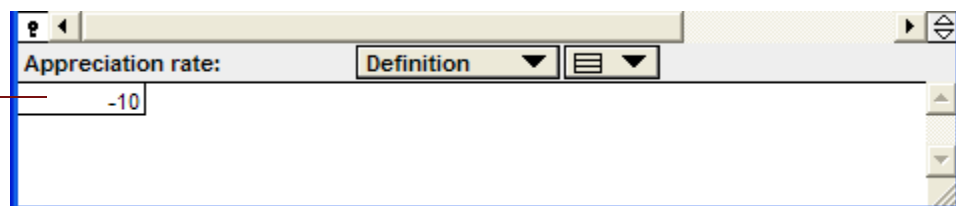
Tip   In Analytica, you add cells to a list by pressing the main *Enter* key, not the numeric keypad *Enter* key.

A new cell appears with the value -9. Change its value to -5. After you have entered two values, as you press *Enter* to add a new cell, Analytica automatically fills in the new cell with a value based on the difference between the last two values. You can override the automatic value by typing the desired value.

7. Type *-5*, followed by the *Enter* key. *0* automatically appears. Press the *Enter* key two more times; automatically, *5* and *10* appear.

Appreciation rate:	Definition ▼	▼
-10		
-5		
0		
5		
10		

8. Select the *Costs of buying and renting* node.

Diagram - Rent vs. Buy Analysis

- Buying price
- Appreciation rate
- Buy or rent
- Rate of inflation
- Cost to Buy
- Costs of buying and renting
- Discount rate
- Time horizon
- Cost to Rent
- Difference between buying and renting
- Difference between buying and renting Importance

9. Select **Mid Value** from the **Result** menu.

File   Edit   Object   Definition   Result   Diagram   Window   Help

Show Result   Ctrl+R

Mid Value

Mean Value
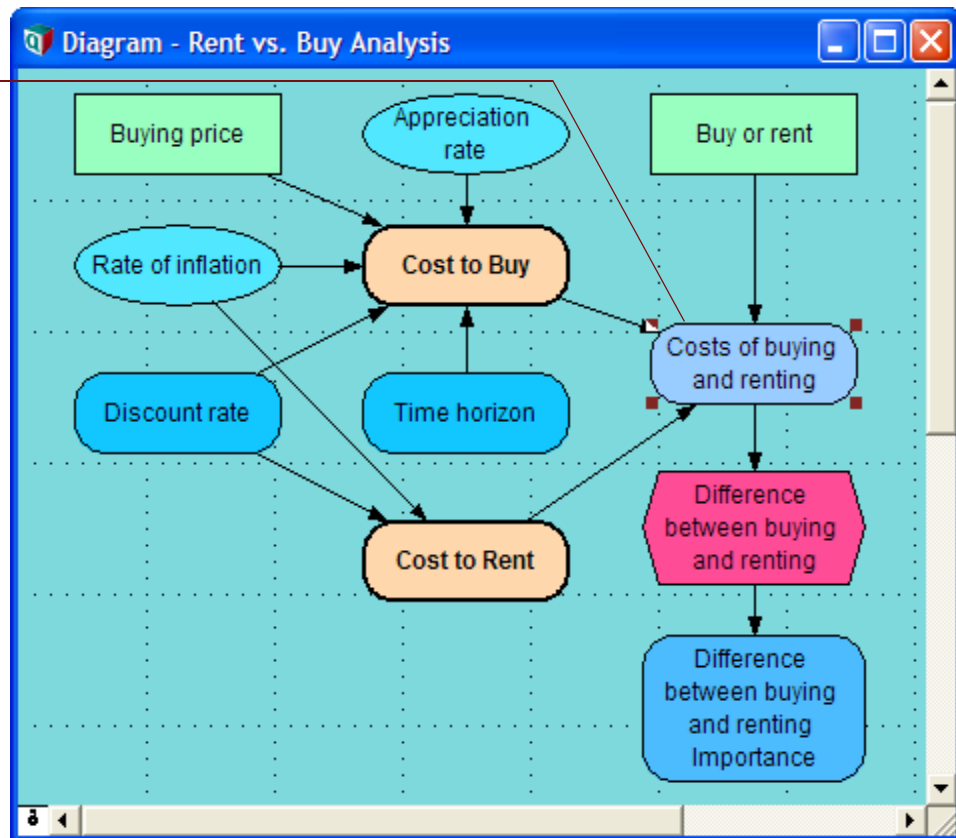
The resulting graph shows the mid value of buying and renting as a function of *Appreciation rate*, which varies from -10% to 10%, as you just entered.

Appreciation rate is informally called an **index** because it characterizes a dimension of another variable's value, in this case, *Costs of buying and renting*.

The graph shows that at an *Appreciation rate* of about -5% per year, renting and buying costs the same. If it is less than -5%, it would be better to rent; if it is greater than -5%, it would be better to buy.

10.Click the **Table** button **1.2** to view the result as a table.
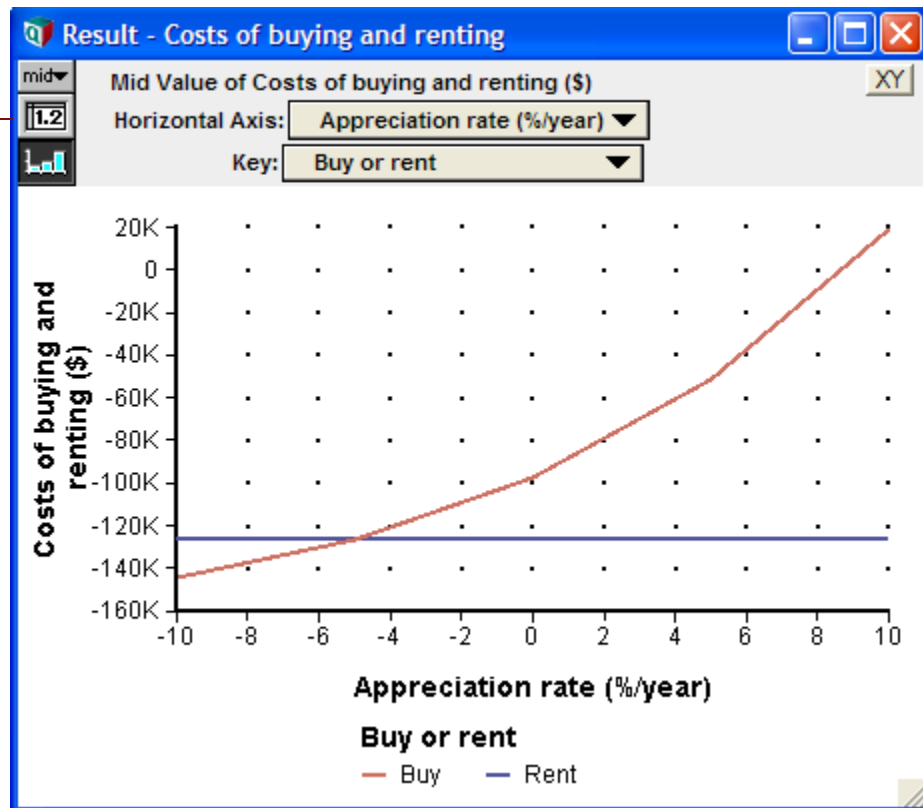
The table shows the values computed for each parameterized value of *Appreciation rate*.

11.Click the **Diagram** button to return to the *Rent vs. Buy Analysis Diagram* window.

Evaluating alternative decisions

Analytica allows you to perform sensitivity analysis on several variables simultaneously.

In this section, you will change *Buying price* to compare results based on alternative decisions. In doing so, you will perform parametric analysis on both *Buying price* and *Appreciation rate* at the same time.

Chapter **3** *Analyzing the Rent vs. Buy Analysis Model*

Evaluating alternative decisions

1. Select the *Buying price* node.

2. Select **List** from the Expression popup menu.

   Analytica asks you to confirm that you want to make this selection.
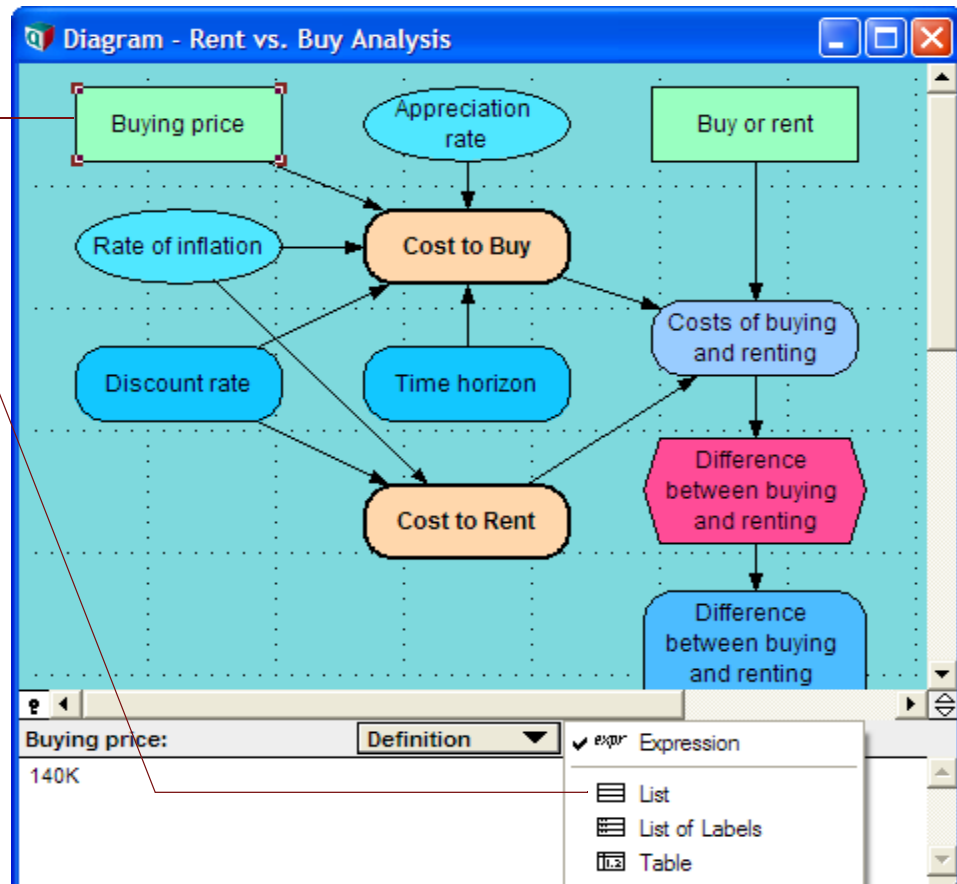
3. Click the **OK** button to proceed.
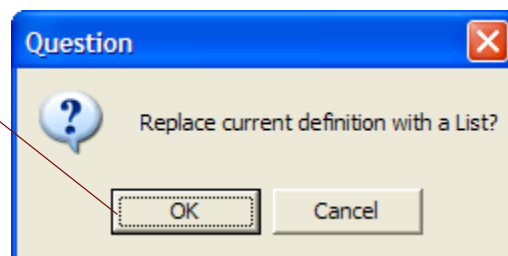
The first cell in this list contains the expression for the previous definition, **140K**. You will change this value, and add additional cells, as you did in Step 6 on page 49 and Step 7 on page 50.

4. Click in the cell to select it. Type *120K* and press the *Enter* key.

5. Type *140K* and press the *Enter* key.

   160K is automatically entered in the next cell.
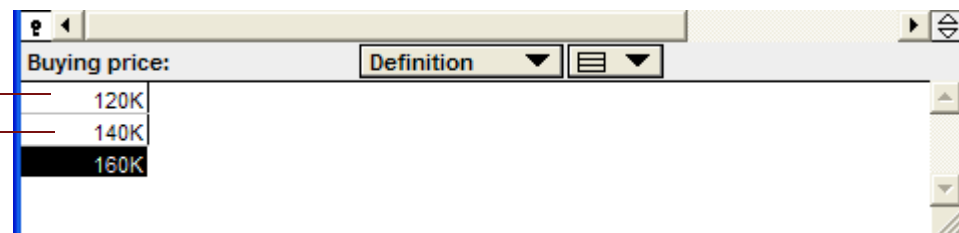
6. Select the *Difference between buying and renting* node.
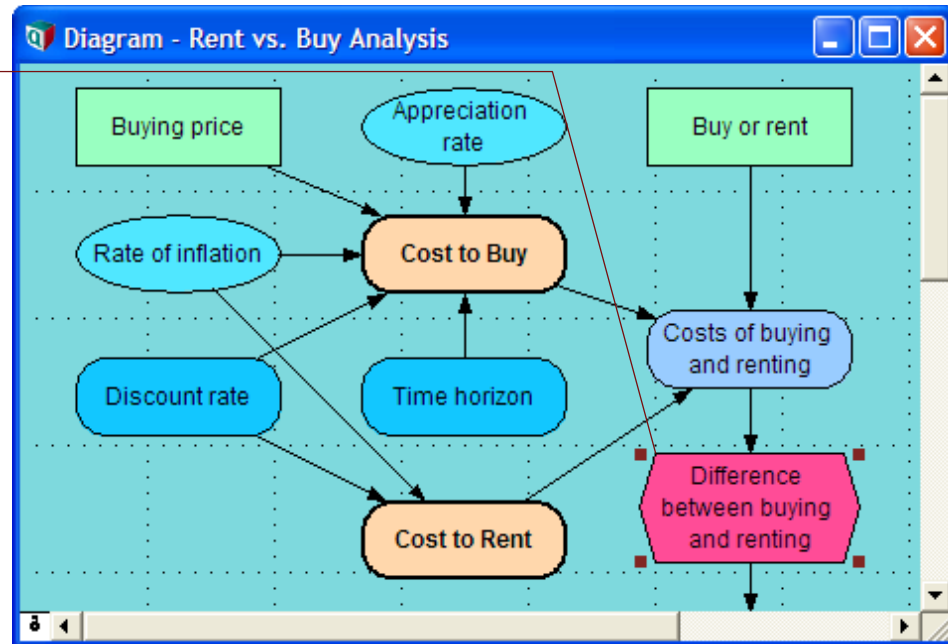
7. Select **Mid Value** from the **Result** menu to recompute and display its value.
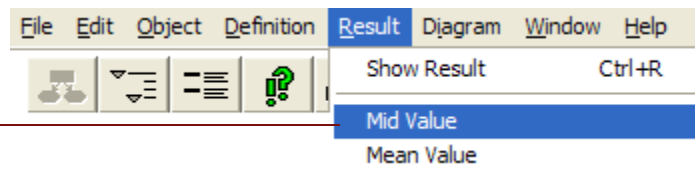
The **Result** window appears displaying the variable's mid value. The *Difference between buying and renting* variable is three curves, one for each *Buying price*. Below the graph is a key to identify each curve.

When you examine the mid value results, you can see that only a $160K home, coupled with an appreciation rate of -2%/year or less, or a $140K home, coupled with an appreciation rate of -6%/year or less, results in renting being cheaper than buying. So, what is the best buy, a 120K home or a 160K home? That depends on what you anticipate the appreciation rate will be. For appreciation rates less than 9% per year, the less expensive home is the better investment. For higher appreciation rates above 9%, the more expensive home provides a larger return.

8. Click the **Diagram** button 🔲 to return to the *Rent vs. Buy Analysis Diagram* window.
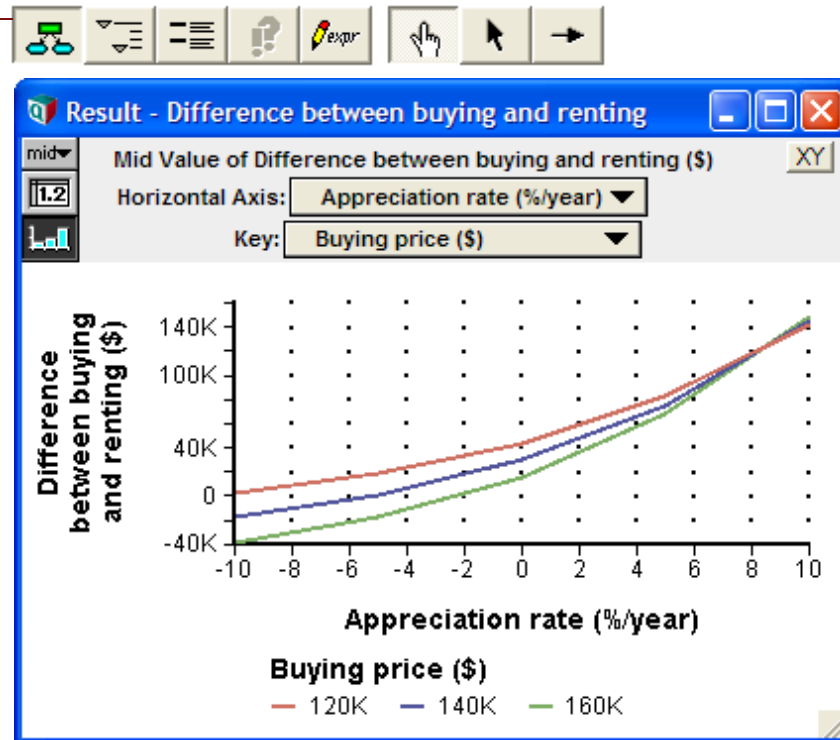
Remember that the cost of renting has been held constant. To further investigate the effect of this, you will examine the *Costs of renting and buying* node.

9. Select the *Costs of buying and renting* node.

Chapter **3** *Analyzing the Rent vs. Buy Analysis Model*

Evaluating alternative decisions
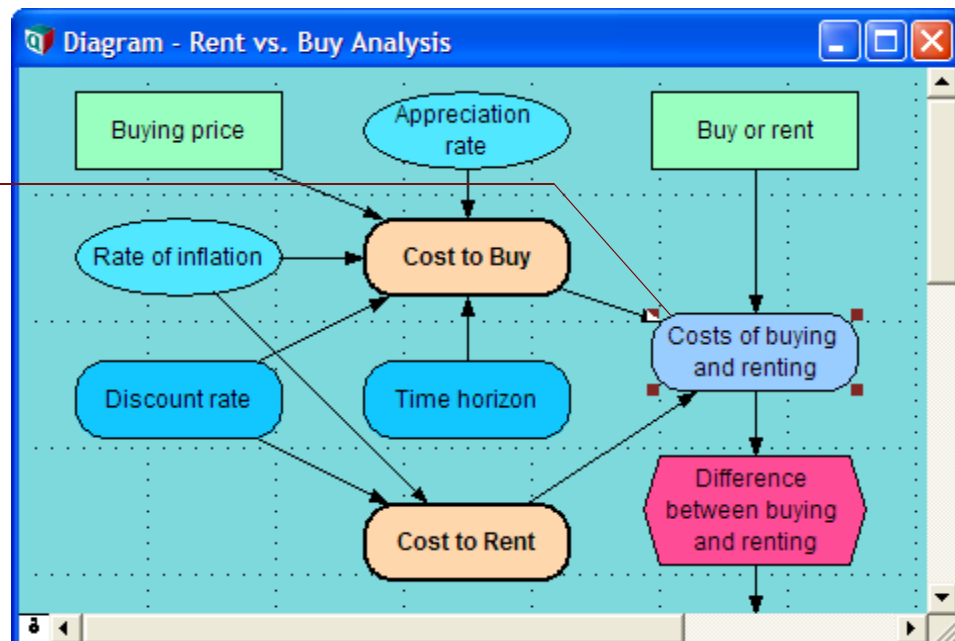
10. Select **Mid Value** from the **Result** menu to recompute and display its value.

Navigating arrows
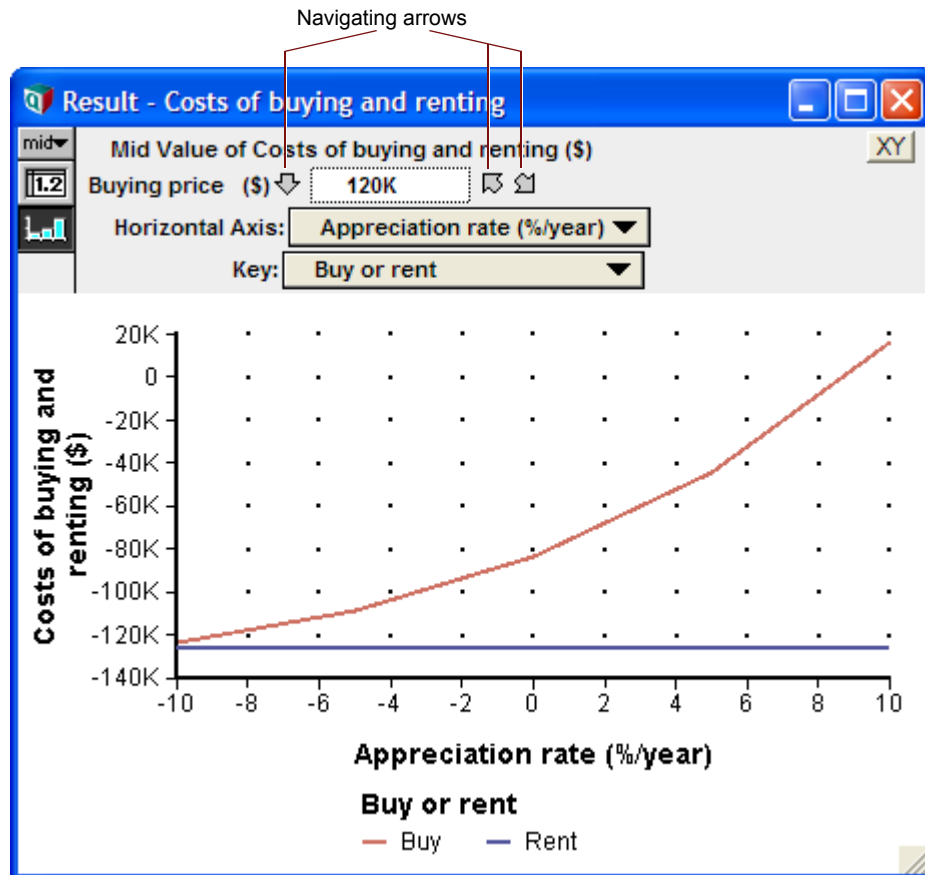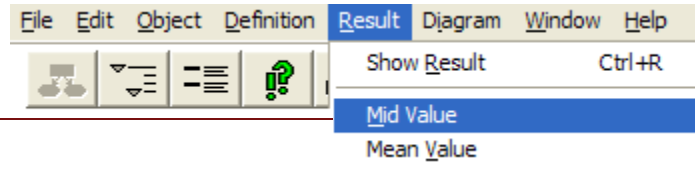
The result has three dimensions, *Buying price*, *Buy or rent*, and *Appreciation rate*, shown in the figure above.

Because only two dimensions can be shown in the graph, Analytica chooses one value of the third dimension to display, in this case, *Buying price* equals **$120K**.

Use the navigating arrows to display different values of the *Buying price* index.

11. Click the down arrow to display a scrolling list of alternative values for the variable.
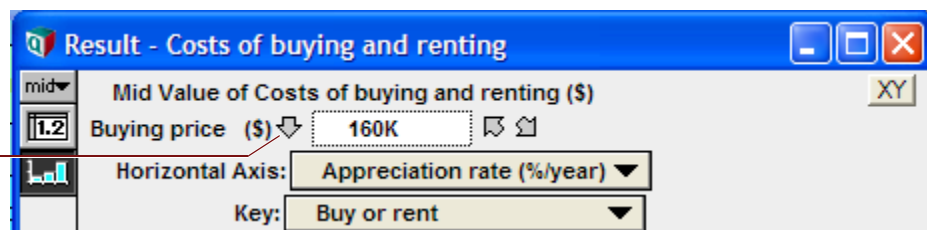
12. Click **160K** to select it.
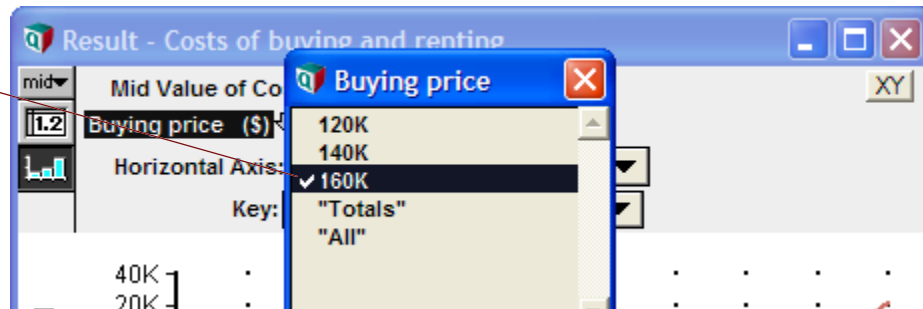
The graph changes to show the mid value of *Costs of buying and renting* given that the *Buying price* equals **$160K**.

13. Click the **Table** button **1.2** to see the table view.
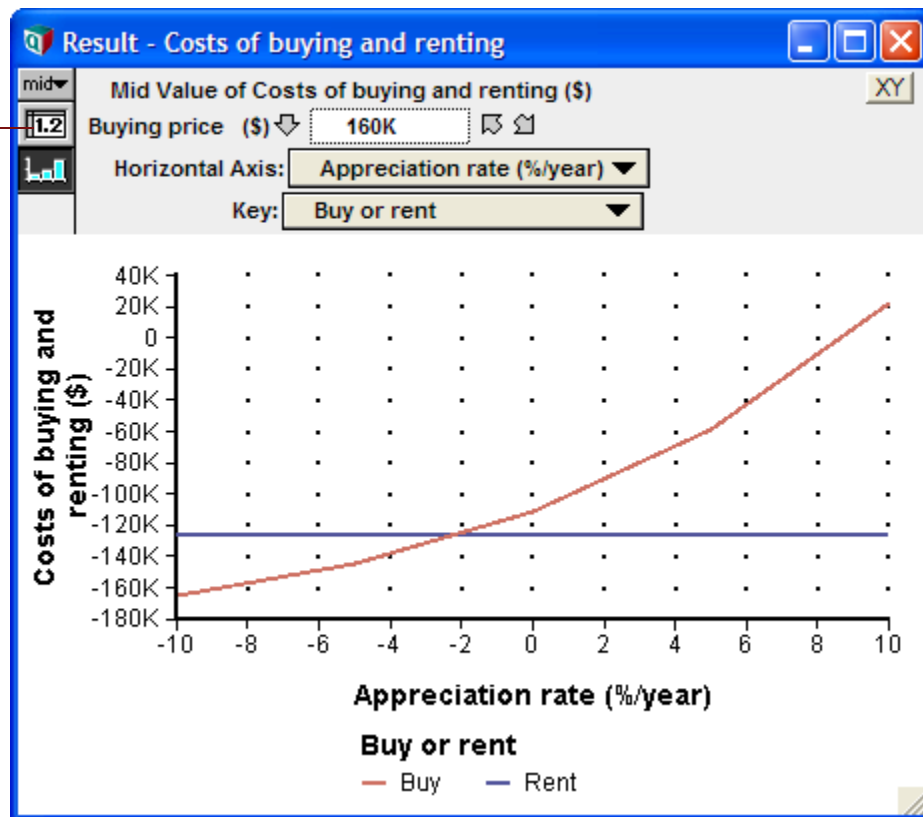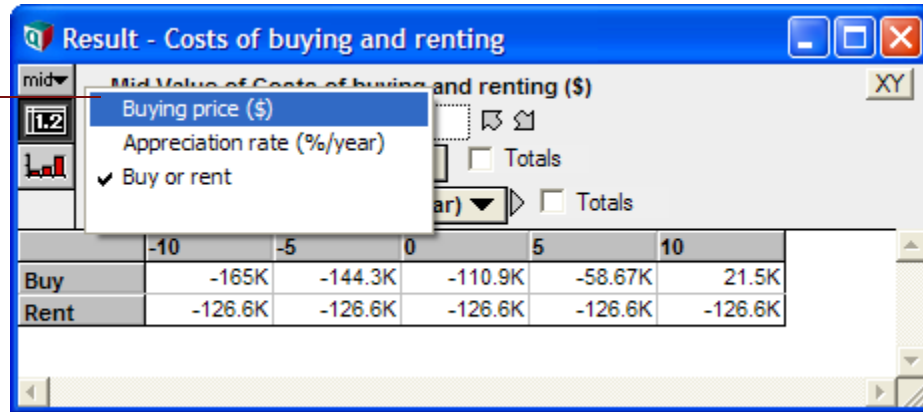
Row index popup menu ─

Column index popup menu ─

	-10	-5	0	5	10
Buy	-165K	-144.3K	-110.9K	-58.67K	21.5K
Rent	-126.6K	-126.6K	-126.6K	-126.6K	-126.6K
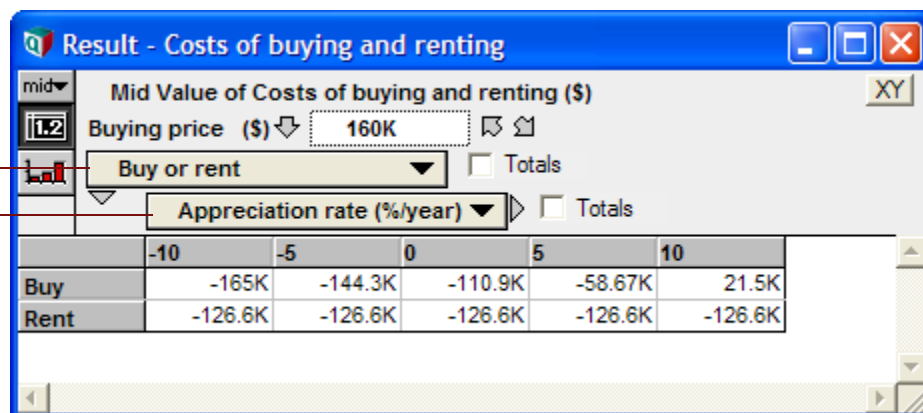
14. Select *Buying Price ($)* from the Row index popup menu.

    Buy or Rent becomes the third dimension with one value (Buy) displayed.

This table shows the mid value cost of buying for the parameterized values of *Buying Price* and *Appreciation Rate*.

15. Click the navigating arrow to show the corresponding table for *Rent*.

This table shows that *Cost to Rent* does not vary with *Buying Price* or *Appreciation rate*.

Analyzing the Rent vs. Buy Analysis model: summary

In this chapter, you have:

- Performed importance analysis.
- Performed parametric analysis.
- Set up and compared alternative decisions.

The next chapter introduces you to creating a new Analytica model.

You can quit Analytica at this point. See "Quitting Analytica" on page 23.

Chapter 4 | *Creating Models*

This chapter shows you how to:

- Create a new model
- Create new variable nodes and enter attributes
- Move and delete variable nodes
- Draw influence arrows to define dependencies between variables
- Define variables as explicit values, functions and lists
- Invoke a built-in function to define a variable
- Do a simple Parametric Analysis
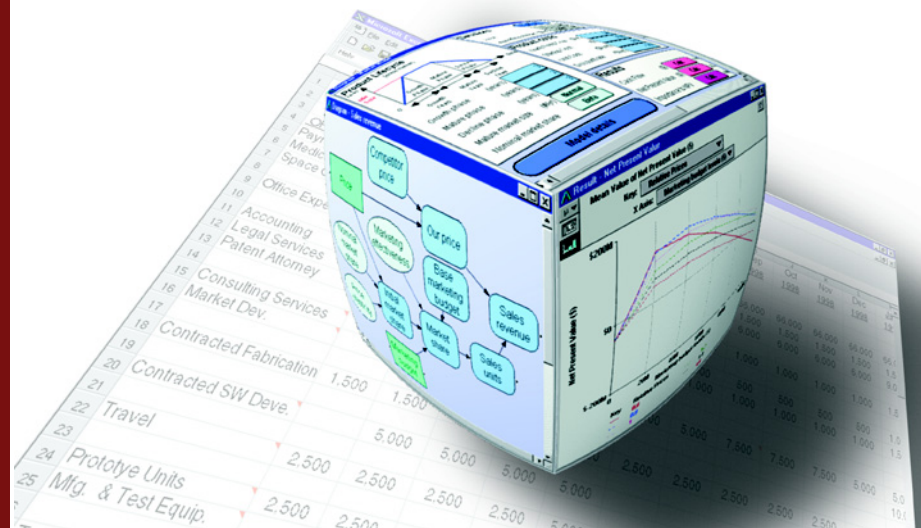
This chapter shows you how to create a new Analytica model.

In the process of building a model to analyze the costs of owning and operating an automobile, you will create variables, define dependencies, add documentary text, and compute results.

If Analytica is not open, start it by double-clicking its icon. If Analytica is already open with an active model go to the **file** menu, select **close model**, then select **new model**.

Creating a new model

When you start up a new Analytica session it shows an Object window for an untitled model. Behind the Object window you will see an empty Diagram window.

First you should enter a Title for your model:

1.   Click in the Title attribute field (to the right of **Title**). Type in *Car cost model* and press Alt-enter. In the space just above the Title you entered, you will see that the Identifier field automatically changes to *Car_cost_model*.

2.   Press Tab to go to the Description attribute and type in:
      A demonstration model to show how to calculate the cost of running a car

Tip   If this is not a first-time entry for Title information, a dialog box will appear to confirm an automatic change to the Identifier.

The Identifier field is automatically filled in to match the first 20 characters of the Title. You can also change this manually if desired.

1 Tab to, or click in, the Title field (to the right of the label **Title**) and type *Car cost model*.

2. *Tab* to the Description field (to the right of the label **Description**) to enter description information.

Object - Untitled

Model ▼ Car_cost_model

Title: Car cost model

Description: A demonstration model to show how to calculate the cost of running a car

Author(s): Renee

Created: Fri, Mar 16, 2007 10:47 AM

Last Saved:

File info: (not saved yet)

Author and creation date are filled in automatically. You can also change this manually if desired.

Analytica automatically updates the *Last Saved* and *File info* information

Tip   You can either press the *Tab* key or use the mouse to move between fields.

3. Add your name as the author if your computer does not automatically register your name.

4. Click the **Diagram** button 🔲 to display the model's **Diagram** window.
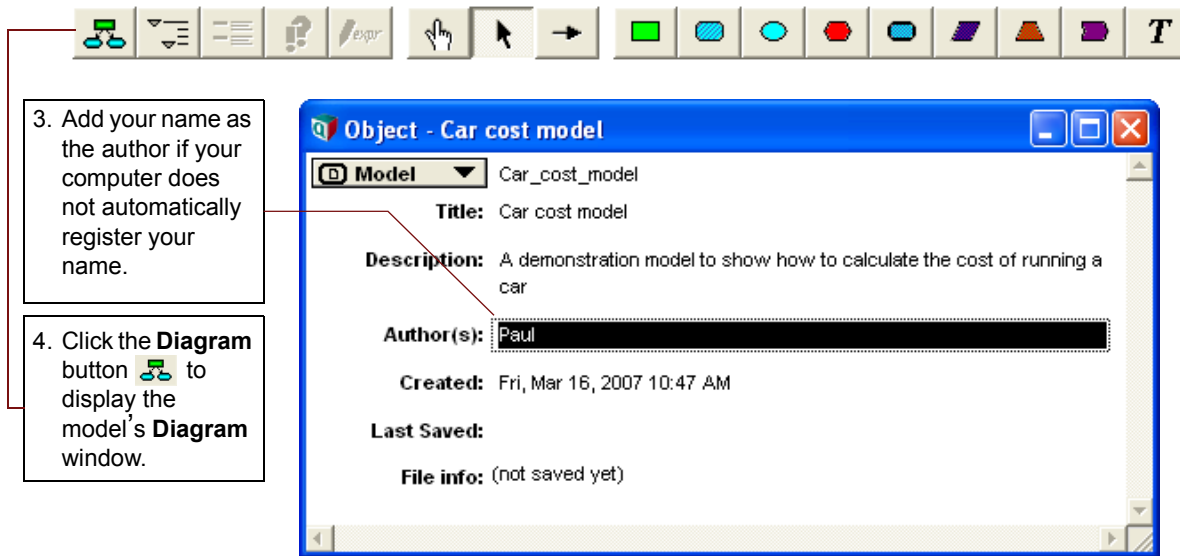
Editing a diagram

In most of the three previous chapters you were in the browse mode, with the ***browse tool*** 🖑 highlighted in the tool palette. In browse mode you can view an existing model without changing its structure. When you create a new model the ***edit tool*** ▶ is selected by default. You use the edit tool to create or change a model.

Be sure to note which tool is selected throughout the remainder of this tutorial.
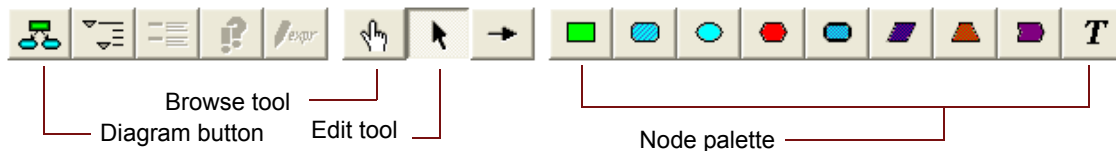
When the edit tool is selected, a menu of icons is displayed in the ***node palette***. These icons represent the different node types and allow you to add nodes to the diagram.

Creating variable nodes

In this chapter you will create variables in the *Car Cost* model.

Each variable is represented in the influence diagram as a shape that depicts the Class of Variable. These shapes are generally referred to as ***nodes***. Select the node shape based on what you know about the variable. If you are not sure what kind of variable to use, choose a General variable (🔲).

See "Recognizing influence diagrams" on page 26 for a description of the node shapes.

The first variable you will create is for the cost of fuel.

1. Choose the edit tool

2. Drag the Variable Node icon to a position in the influence diagram.

   A new Variable Node appears in the diagram at the location where you release the mouse button.
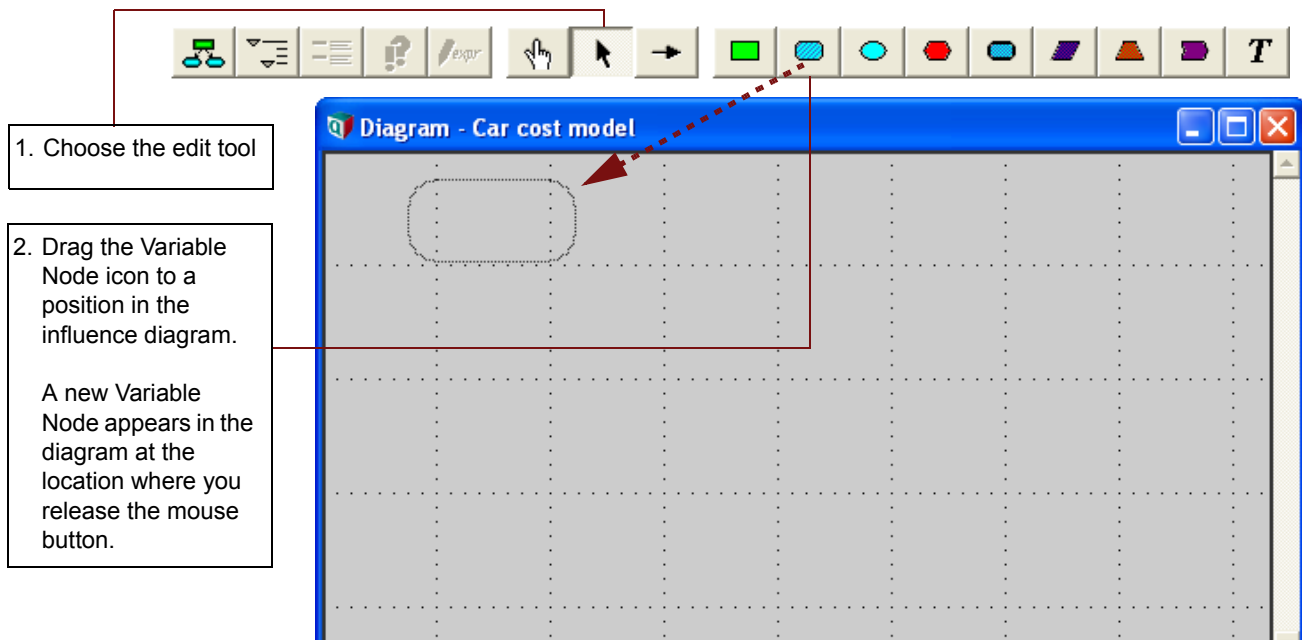
Tip As you build a model, you should select descriptive titles for your variables. Descriptive variable titles remind you of the model's logic and help to inform others about how the model operates.

3. Type **Fuel cost** for the variable title.

   Press *Alt+Enter* or click any other location in the diagram to accept the title.
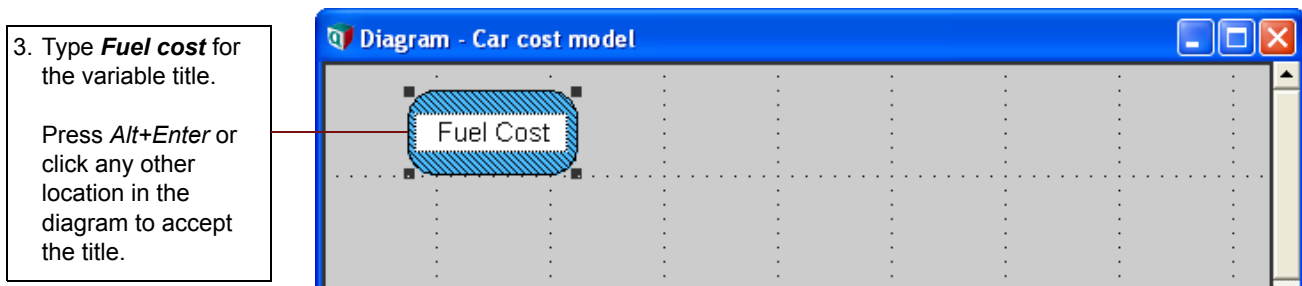
Tip *Fuel cost* is filled with a diagonal line pattern around its text, indicating that it does not yet have a valid definition.

Repeat Steps 1 and 2 four times to create four more chance variables that affect fuel cost. Title these variables as follows:

- *Fuel price* (price per gallon of gasoline)
- *Annual miles* (number of miles driven each year)
- *Mpg* (miles per gallon of gasoline)
- *Age* (driver's age)

4. Drag the Variable Node icon to a position on the diagram
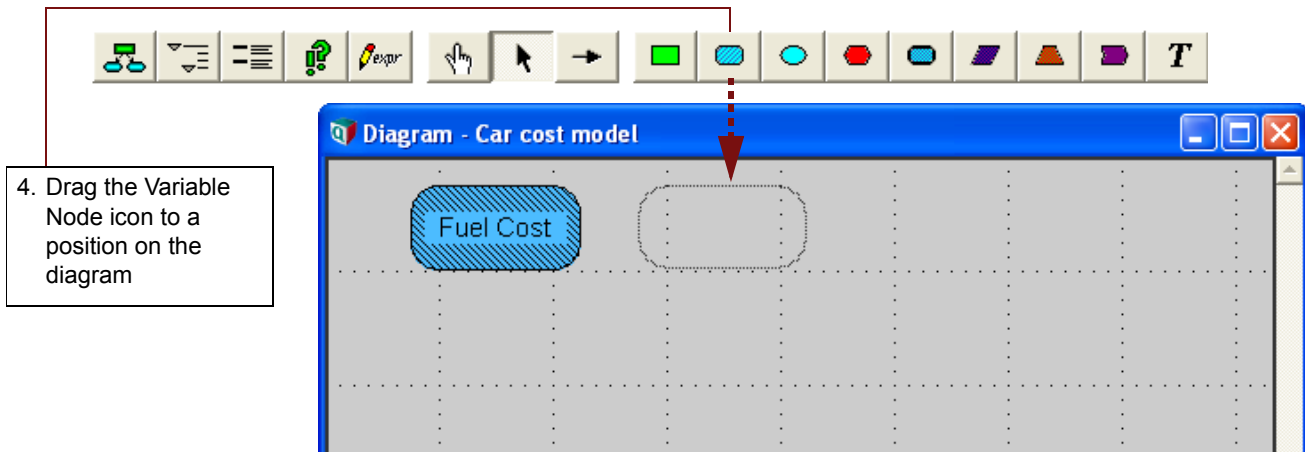
Title text wraps to fit within the node as you type. You can manually create a new line in a title by pressing the *Enter* key at the desired break point(s). You can also expand the size of the node by dragging one of the corner handles.

5. Type **Fuel** and press the *Enter* key to create a second line.

Type **price** and press *Alt+Enter* to indicate that you are finished.

6. Repeat Steps 3 and 4 to create three more variables, as shown.

Title the variables **Annual Miles**, **Mpg**, and **Age**.
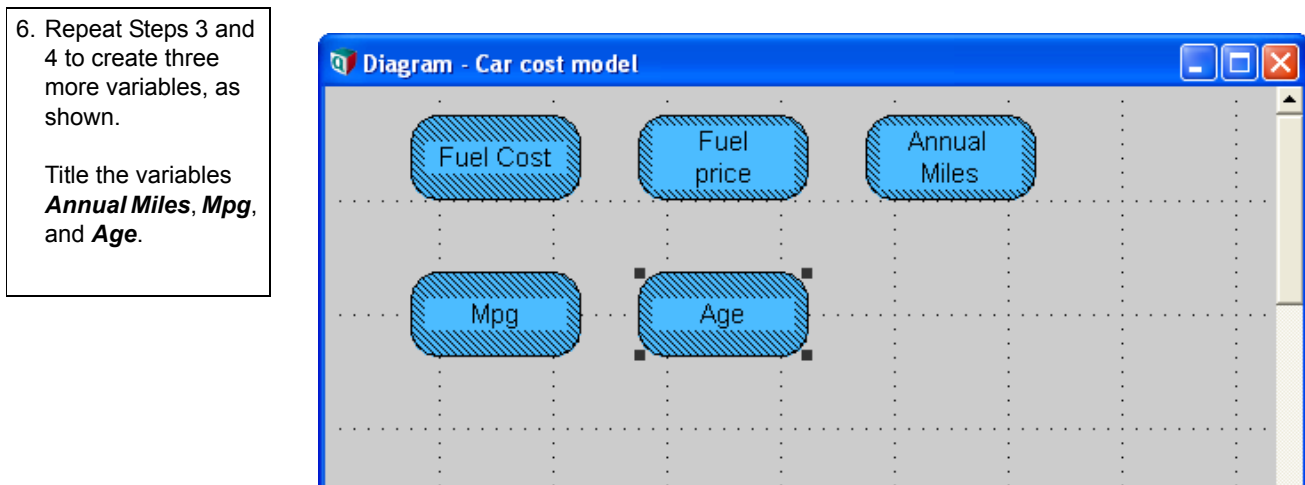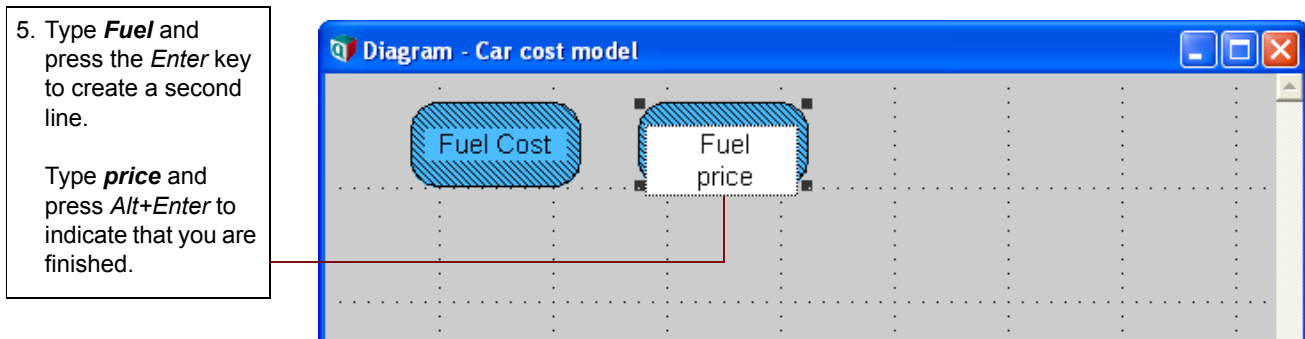
Saving your model

Analytica automatically saves each change you make to a backup file. If your computer (or Analytica) should terminate unexpectedly, it will offer to recover your changes the next time you start Analytica. Even so, it's a good idea to save your changes periodically.

1. Select **Save** from the **File** menu (or press *control-s*).
   The first time you try to save a model, it shows a **Save As** dialog, offering to save the model in the **My Documents** folder using the model Title as the default file name. You can select another folder and modify the file name

2. Click **Save** or press **Enter** to save the model in the selected folder.

1. Select **Save As** from the **File** menu.
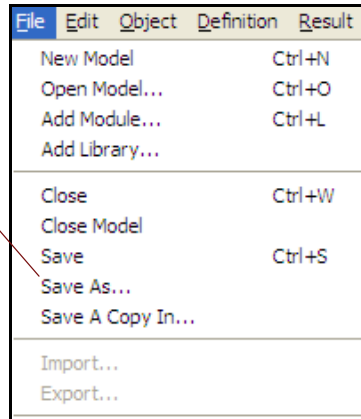
2. Type the new file name here.
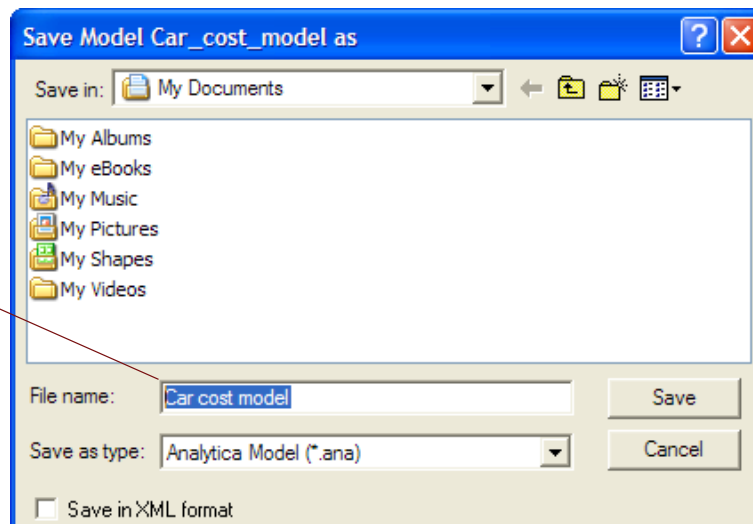
Deleting a variable

Sometimes you might want to delete a variable that you previously created.

In this example you realize that the driver's age is not relevant to your understanding of the *Fuel Cost* variable. Therefore, you will delete the *Age* variable.

1. Select the Age node if it is not already selected.

   Handles surround the node to indicate that it is selected

2. Select **Clear** from the **Edit** menu, or press the **Delete** key
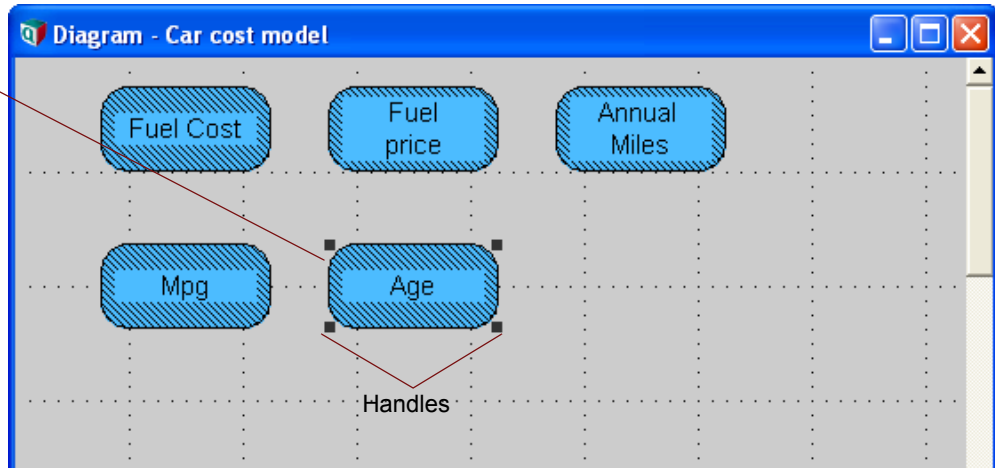
The **Delete** command cannot be undone; Analytica asks you to confirm that you want to delete.

3. Click **OK** to confirm that you want to delete the selected object.
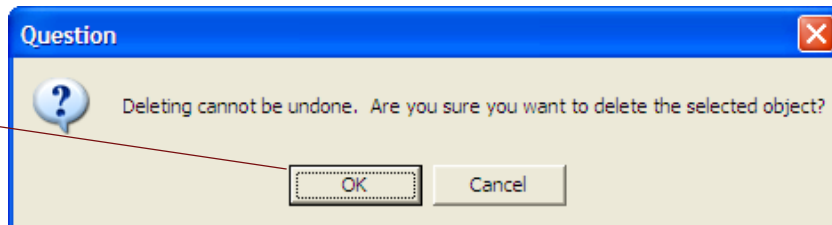
Moving nodes

When you create a model you should try to structure the layout to make the influence diagram easy to understand. As you refine your model, however, you will undoubtedly want to group nodes in different ways to achieve this goal.

In this case we anticipate that *Fuel cost* will be derived from the other three variables. Therefore we want to place it apart.

Moving nodes is a simple matter of dragging them to the desired position. Try moving the nodes into the new configuration below.

Tip You can select multiple nodes by holding down the *Control* or *Shift* keys, or by placing the cursor in a blank area and dragging diagonally to draw a rectangular frame across the nodes you want to select.

Drag the nodes to a new configuration.
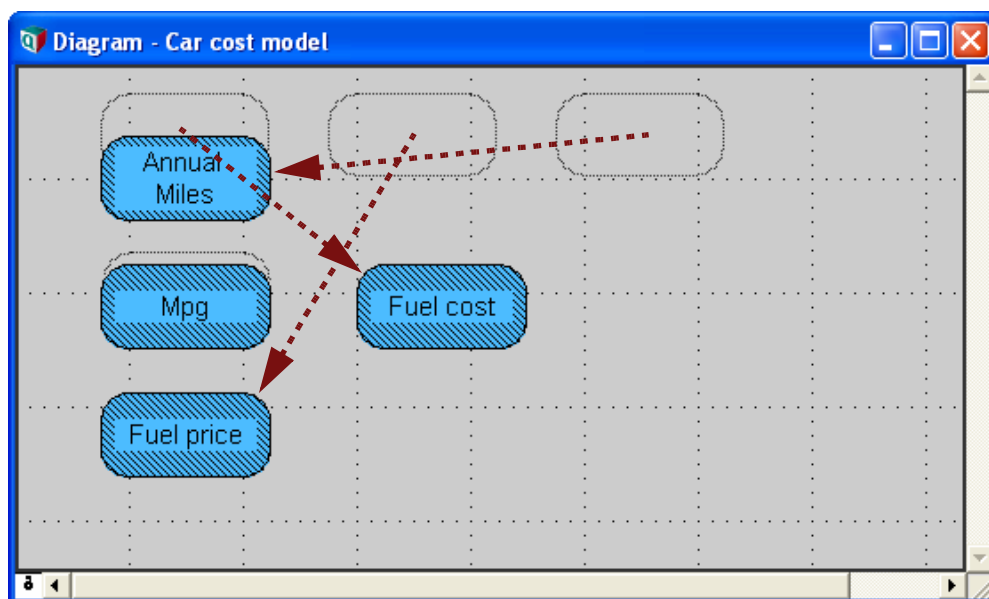
Tip You can undo or redo a drag operation by selecting **Undo/Redo** from the **Edit** menu, or by typing the keyboard shortcut, *Control+Z*.

Editing variable titles

The name displayed inside each variable node is referred to as a *Title*. In this example you will expand the title *Mpg* to *Miles per gallon*:

1. Select the *Mpg* node.
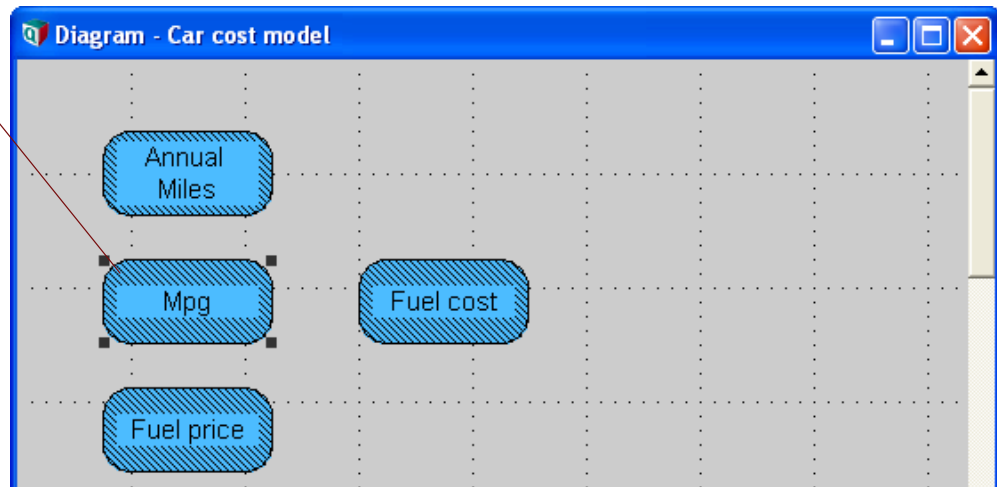
2. Click again inside the node's title to select its text for editing.
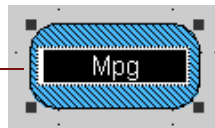
Tip Pause briefly between the click to select the node and the click to select the text within it. If you complete two single-clicks too quickly, Analytica interprets your actions as a double-click and opens an **Object** window.

If you accidentally open the **Object** window, return to the **Diagram** window by clicking the **Diagram** button.

3. Type ***Miles per gallon*** and press *Alt+Enter*.

   The new title is displayed.
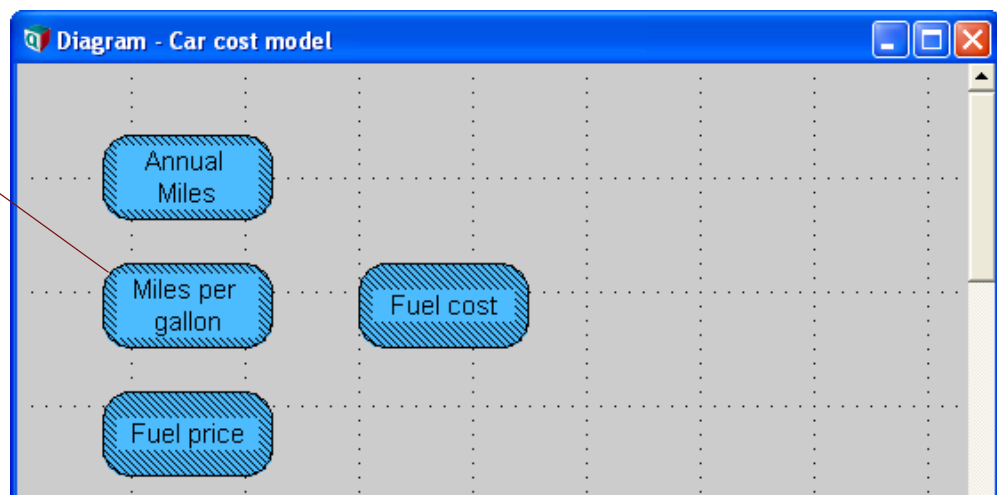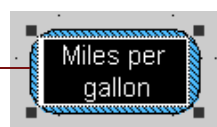
When you change the title of a node, Analytica asks you if you want the Identifier to be automatically changed as well.

4. Click **No** to close this dialog box and keep the identifier as *Mpg*.

Question

?   Change Identifier from "Mpg" to "Miles_per_gallon" to match changed Title "Miles per gallon"?

Yes   No
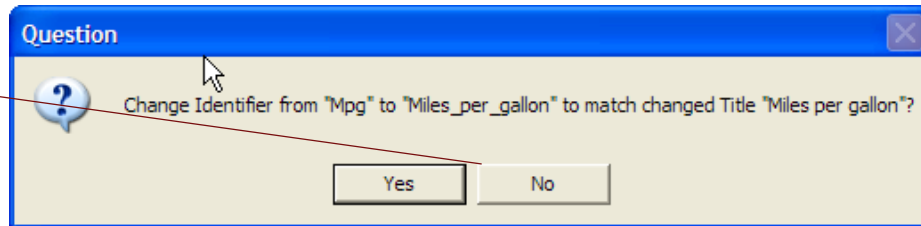
Tip   You can change this behavior, to either turn off automatic updating of the identifier or to make it fully automatic, so that you are not asked. See the "Preferences dialog" section in Chapter 4 of the *Analytica User Guide* for details.

Drawing arrows between nodes

One of Analytica's most powerful features is its ability to show relationships between variables in the influence diagram. **Influence arrows** are used to specify the dependencies between variables.

Because the *Miles per gallon* variable influences the *Fuel cost* variable, you will draw an arrow connecting the two nodes.

1. Select the **arrow** tool ➔ to begin drawing arrows.

Notice that the cursor turns into an arrow.

Diagram - Car cost model

Annual Miles

Miles per gallon

Fuel cost

Fuel price
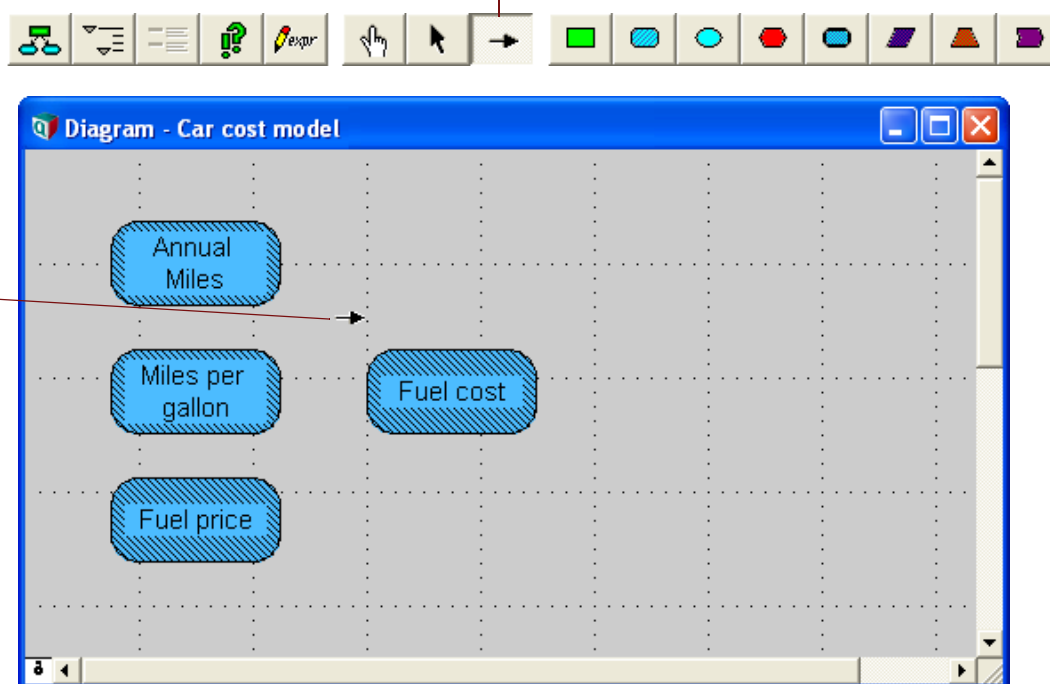
2. Drag from the *Miles per gallon* node to the *Fuel cost* node.

Both nodes will become highlighted
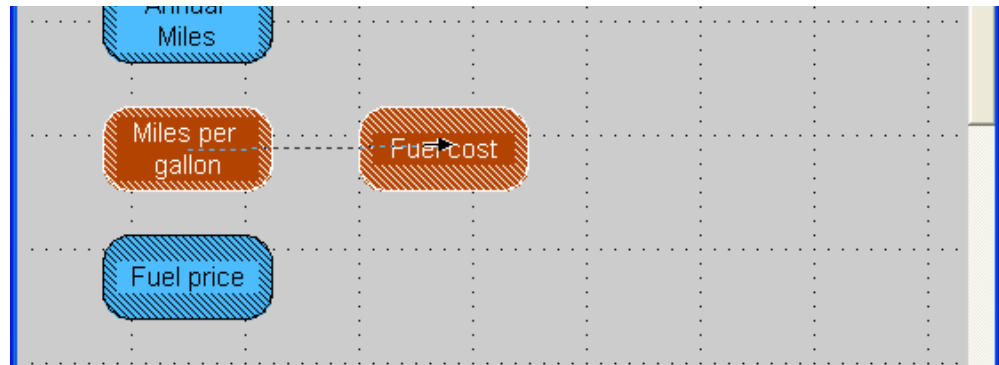
3. Release the mouse button when *Fuel cost* is highlighted.

The two nodes are now connected by an arrow, indicating that *Miles per gallon* affects *Fuel cost*.
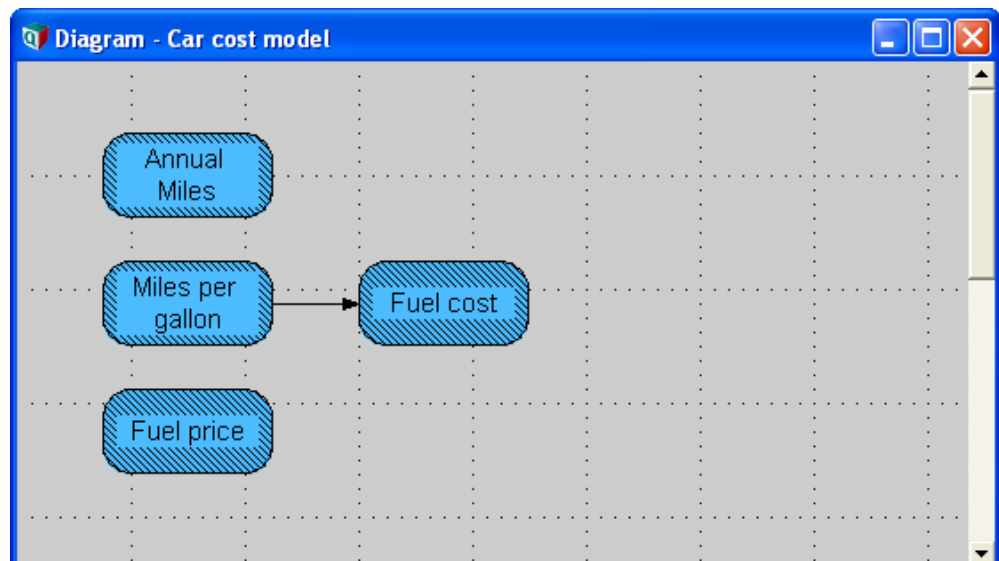
If the nodes are not connected by an arrow, repeat Steps #1 through #3.

Deleting an arrow

Occasionally, you might need to delete an arrow because of an earlier mistake or a change in your understanding of the model. This section shows you how to delete the arrow that connects *Miles per gallon* to *Fuel cost*.

You can delete an arrow using either the edit tool or the arrow tool.

First, make sure you have either the edit tool or arrow tool selected.

1. Make sure either the arrow tool or the edit tool is selected.

2. Select the arrow.

   It is easiest to select the arrow by clicking the arrow head. Handles appear when the arrow is selected.

3. Press the *Delete* key to delete the arrow.

   The arrow disappears.
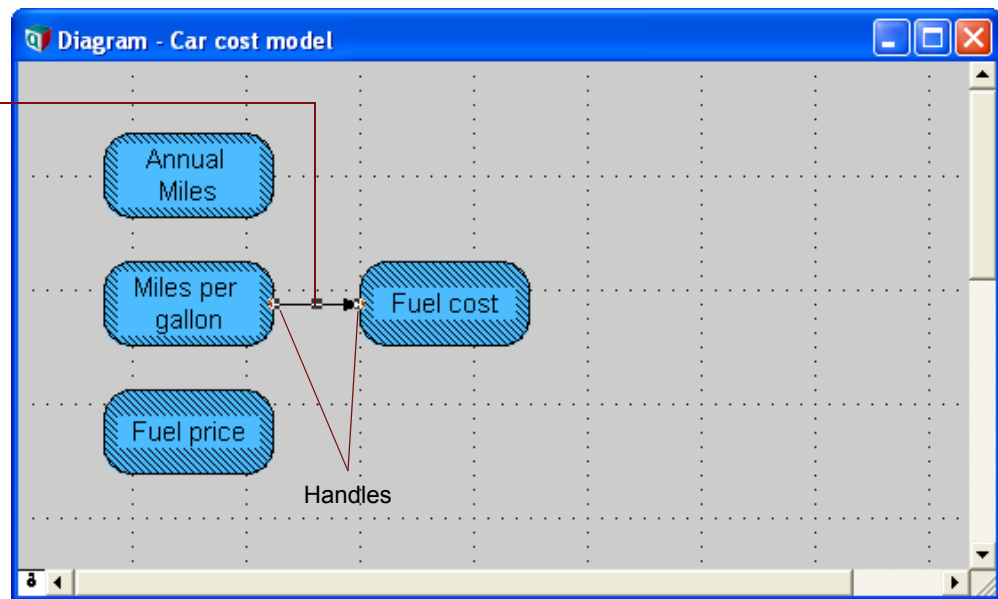
Connecting multiple arrows

When one variable is influenced by several other variables, you can draw multiple arrows at once. This example shows you how to connect the three variables contributing to the *Fuel cost* variable.

First make sure the arrow tool is selected.
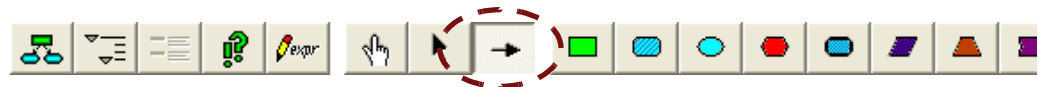
1. Select the *Annual Miles*, *Miles per gallon,* and *Fuel price* nodes simultaneously.

   (Hold down *Control* or *Shift* while selecting or drag diagonally across the group.)
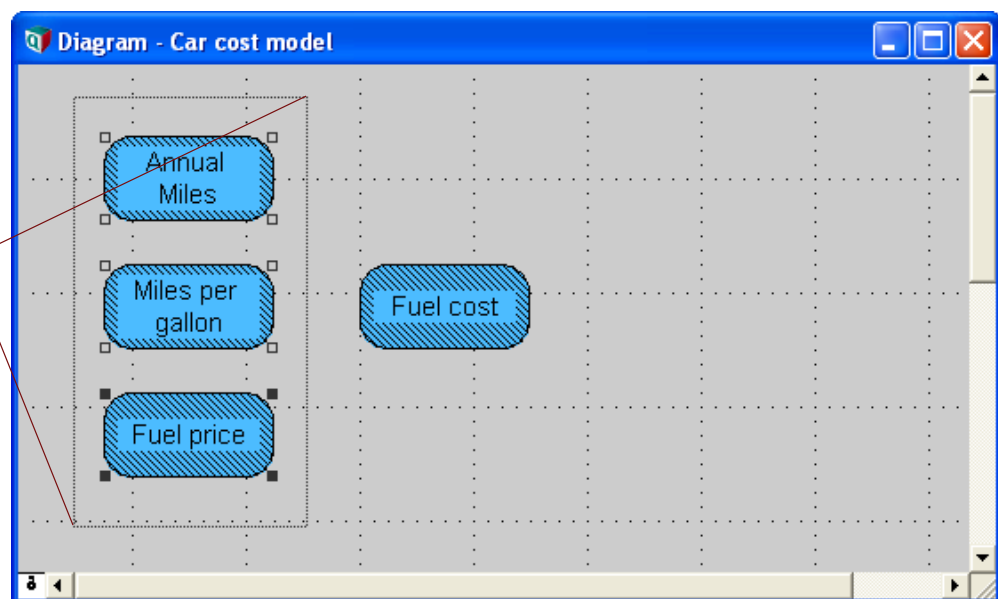
2. Drag from any one of the selected nodes to the *Fuel cost* node.

3. Release the mouse button when the *Fuel cost* node is highlighted.

   Three arrows should now point to the *Fuel cost* node (as shown in the following diagram).
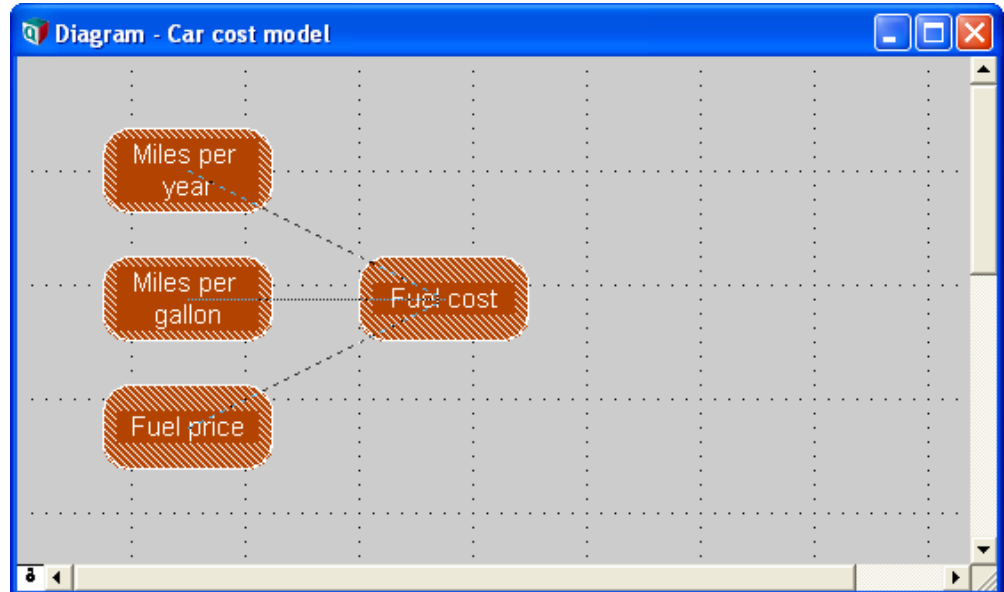
4. Deselect all of the nodes by clicking in any location in the diagram that is not on a node.
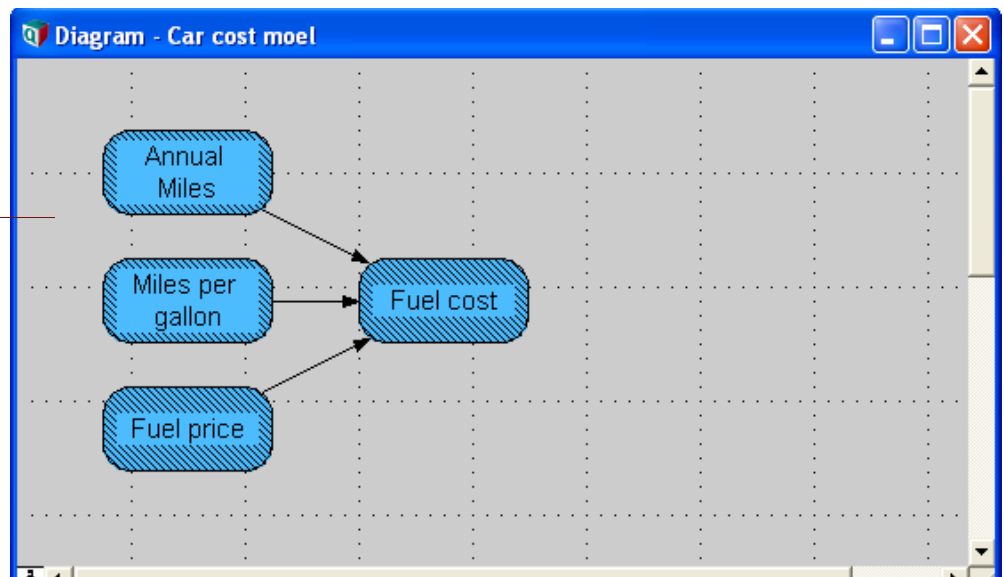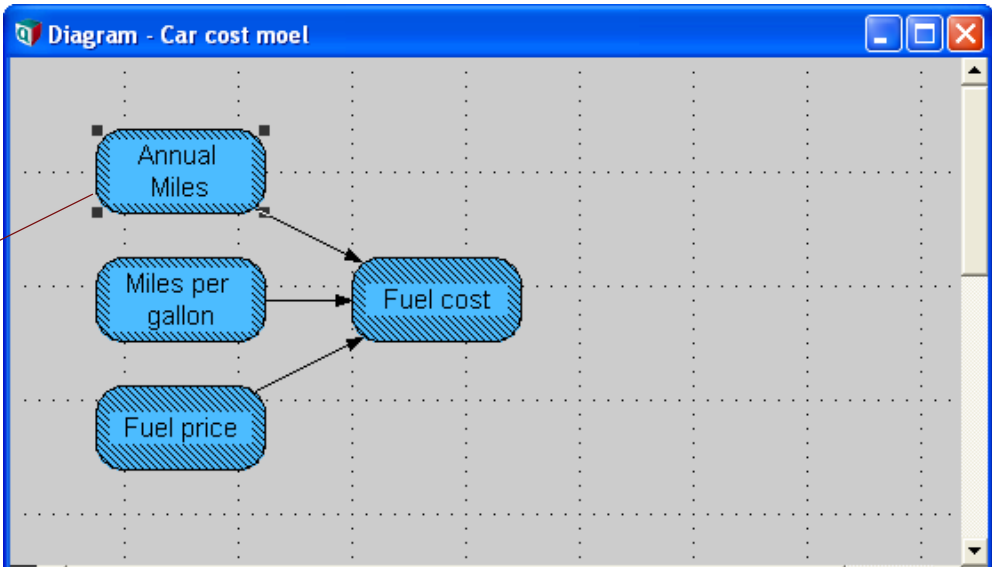
Entering attributes into the Object window

Each variable (or other Object) has an ***Object window*** that lets you see and edit its Attributes - including its Identifier, Title, Units, Description and Definition.

In this section we will use the Object window to enter attributes for the *Annual Miles* variable.
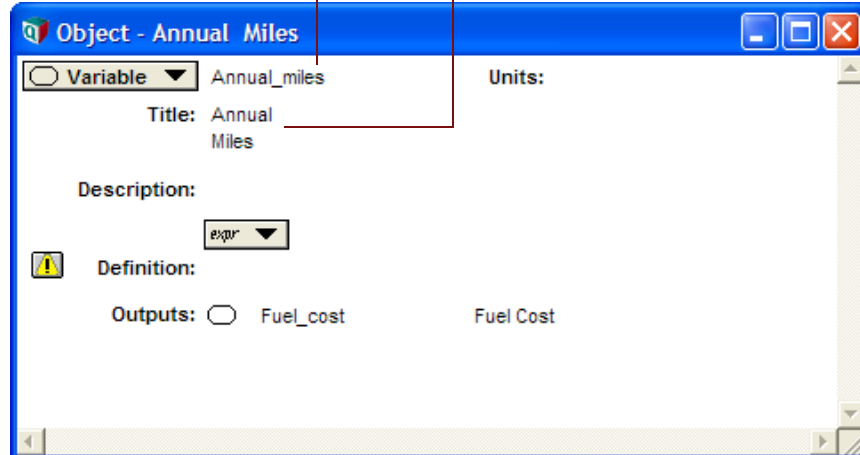
Make sure to select the **edit** button ( ) before continuing.

1. Select the **edit** button to turn off arrow drawing.

2. Click the *Object window button* ( ) to open its Object window.

Tip   You can also open a variable's Object window by double-clicking the node using the browse or arrow tools. If you are using the browse tool, you will not be able to enter or change documentation.

Identifier   Title

The identifier in the Object window shown above is ***Annual_miles***. Analytica assigns the identifier when the title is created. It uses the first 20 characters of the title except for spaces or punctuation, which are replaced by underscores (_). Analytica does not differentiate between uppercase and lowercase letters in identifiers.

You can directly edit both the identifier and the title.

First, you will change the variable's identifier to a short abbreviation so that it can easily be used later in the definitions of other variables. You will then document the variable more fully.

<table>
<tr>
<td>2. Select the contents of the Identifier field.</td>
<td>

Object - Annual Miles
Chance ▾ | Annual_miles | Units:
Title: Annual Miles
Description:
</td>
</tr>
<tr>
<td>3. Type ***Mpy*** and press *Alt+Enter*.</td>
<td>

Object - Annual Miles
Chance ▾ | Mpy | Units:
Title: Annual Miles
Description:
</td>
</tr>
<tr>
<td>4. Click the Units field. Type ***miles/year*** and press *Alt+Enter*.</td>
<td>

Object - Annual miles
Chance ▾ | Mpy | Units: miles/year
Title: Annual miles
Description:
</td>
</tr>
</table>

Tip Analytica uses the information from the Units field to label graphs or tables that use the *Miles per year* variable; Analytica does not use it in any mathematical computations.

<table>
<tr>
<td>5. Select the contents of the Title field and press the *Delete* key to erase **Annual miles**.</td>
<td>

Object - Annual miles
Chance ▾ | Mpy | Units: miles/year
Title: Annual miles
Description:
</td>
</tr>
<tr>
<td>6. Type ***Miles per year*** and press *Alt+Enter*.</td>
<td>

Object - Annual miles
Chance ▾ | Mpy | Units: miles/year
Title: Miles per year
Description:
</td>
</tr>
</table>
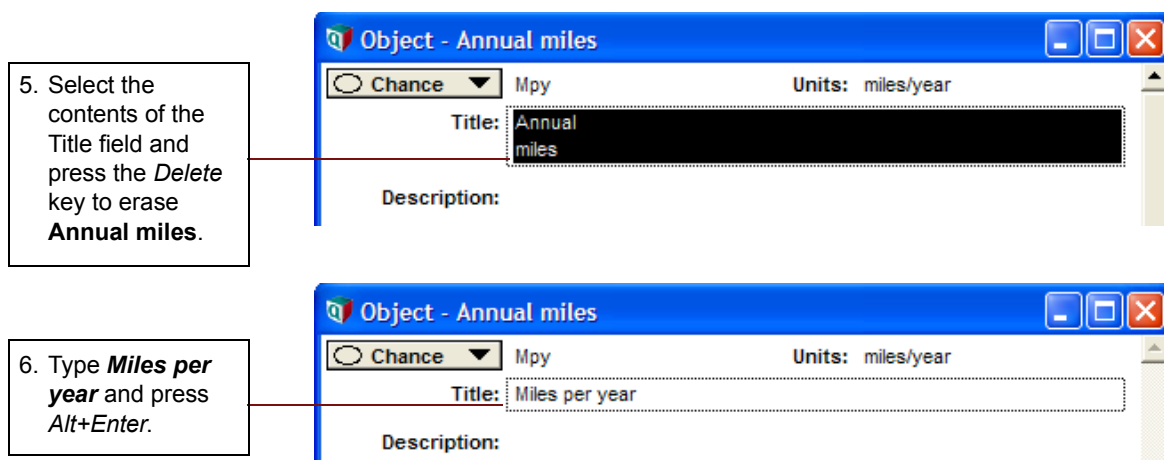
Tip When you change the title of a node, Analytica asks you if you want it to automatically change the identifier to match the new title.

7. Click **No** to keep the identifier as **Mpy**.
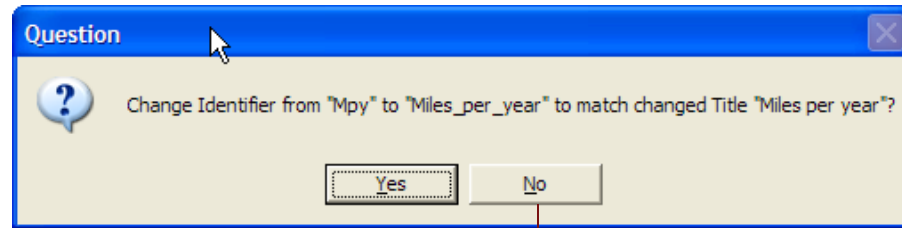
8. Click in the Description field, type *Average miles driven per year*, and press *Alt+Enter*.
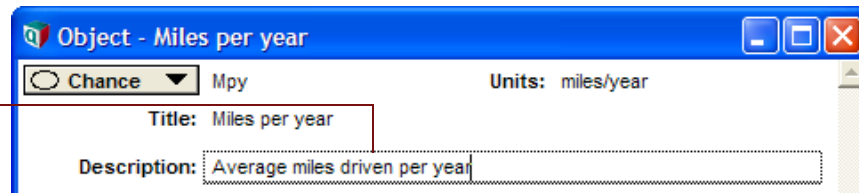
Defining a variable as an explicit value

Analytica uses a wide range of variable types. In this section we simply enter an explicit value for the variable. Functional expressions and lists are described later in this chapter. Other variable types are addressed in later chapters.

For variables as functional expressions see "Defining a variable as a function of other variables" on page 76.

For variables as lists see "Defining a variable as a list" on page 81.

For variable as tables see "Working with Arrays (Tables)" on page 89.

For a demonstration of built-in functions see "Defining a variable using a built-in function" on page 84.

First let's define *Miles per year* as 12,000

1. Click in the Definition field to enter a mathematical expression for the variable; type *12K*.

A warning icon indicates that this variable's definition has not yet been accepted.
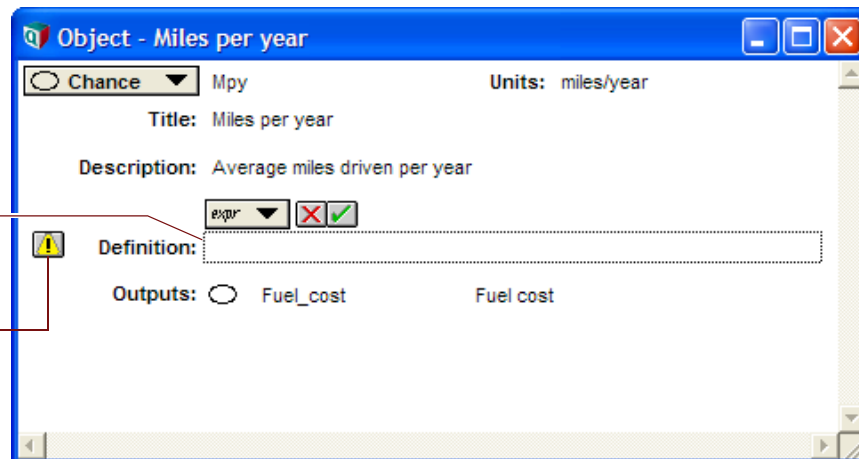
Tip Numerical suffixes like μ and K are used extensively throughout Analytica. A quick reference for these suffixes is given on the back page of this tutorial.

2. Click the check button ☑ or press *Alt+Enter* to accept the new definition.

The warning icon disappears because the variable now has a valid definition.

If you want to cancel what you entered, click the cancel button ☒.
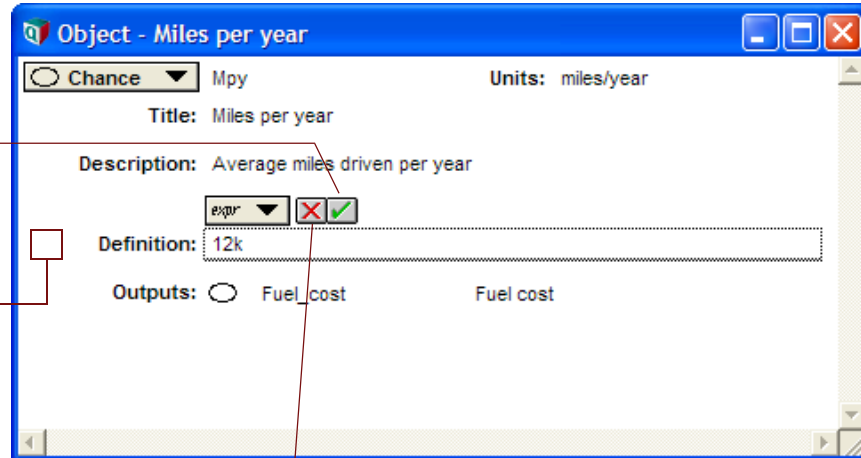
Whenever another variable's definition includes the *identifier* (*Mpy*) of *Miles per year*, this defined value, **12K**, is used as its value.

3. Click the **Diagram** button to return to the influence diagram.

Miles per year is no longer filled with a diagonal line pattern around its title, as shown in the figure below. The clear node indicates that *Miles per year* now has a valid definition.
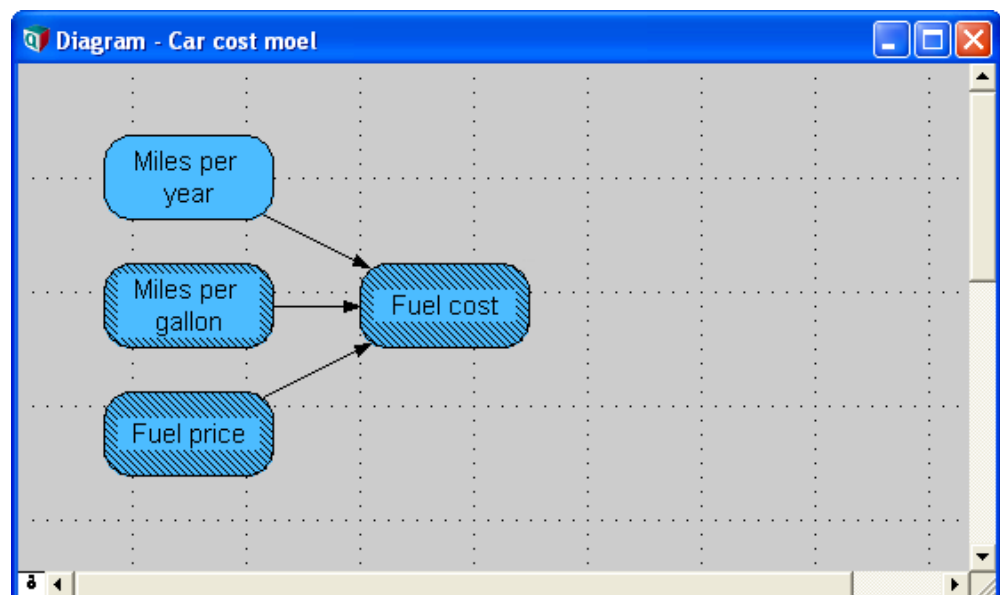
Defining a variable as a function of other variables

When one variable is dependent upon another variable, you must provide an expression that describes the relationship between the variables. Influence arrows connecting other nodes to *Fuel cost* show the direction of this dependency.

In this section, you will enter a definition for *Fuel cost* in terms of the values of *Miles per year*, *Miles per gallon*, and *Fuel price*.

1. Double-click the *Fuel cost* node to open its Object window.
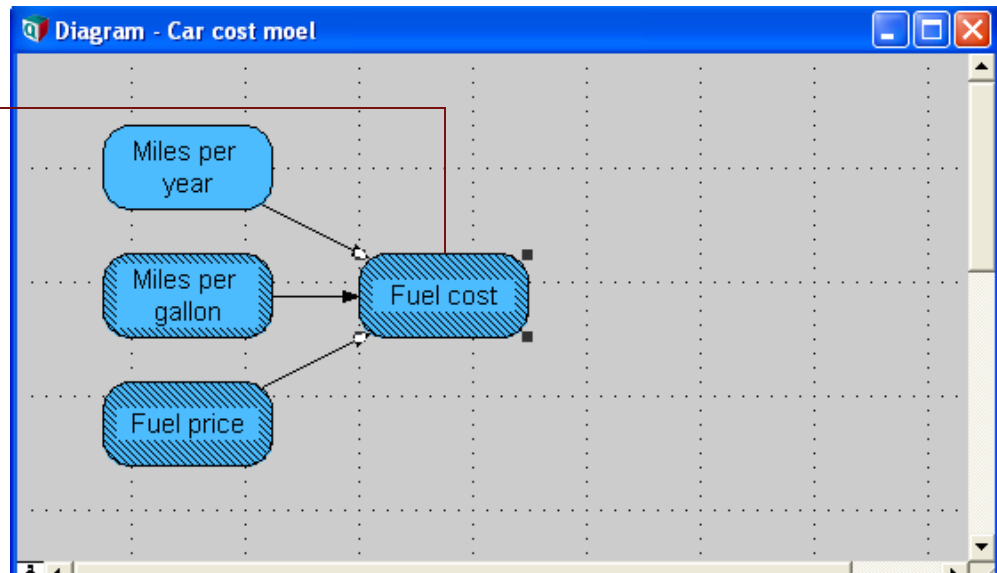
2. The identifiers and titles of the three input variables appear in the Inputs field for the *Fuel cost* variable.

Because the form of the dependence has not been specified, the Definition field is blank.

3. Enter the variable's units as **$/year**, and description as **Annual cost of fuel**.

4. Click in the Definition field to enter a mathematical expression.

Because Fuel cost is equal to fuel price times miles driven, divided by miles per gallon, you will enter the following expression into the Definition field:

```
Fuel_price * Mpy/Mpg
```

An asterisk (*****) represents multiplication; a slash (**/**) represents division.

5. From the Inputs popup menu select the name of the variable that you want to add, in this case, *Fuel price*. *Fuel_price* appears in the Definition field.
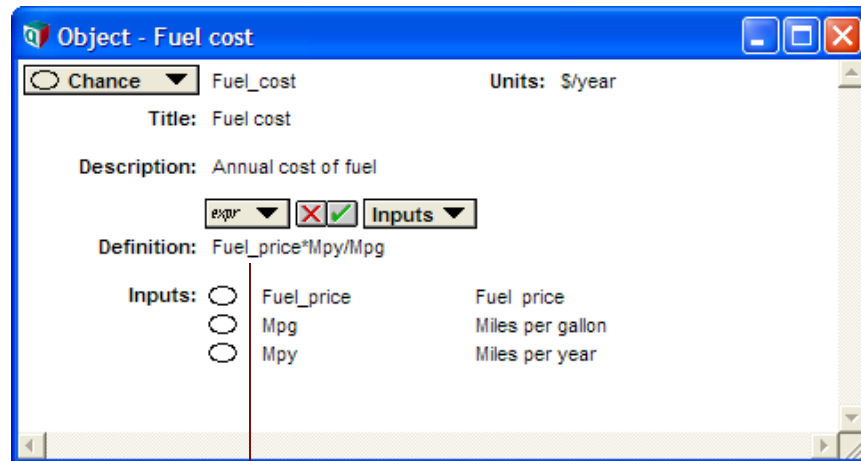
6. Type an asterisk (*****).

7. Select *Miles per year* from the Inputs popup menu.

8. Type a slash (/).

9. Select *Miles per gallon* from the Inputs popup menu.

10. Press *Alt+Enter* or click the check button ✓ to accept the definition. The Definition field should look like this (spaces between terms and operators are optional).

Object - Fuel cost

Chance ▼	Fuel_cost		Units: $/year

Title: Fuel cost

Description: Annual cost of fuel

expr ▼ ☒ ✓ Inputs ▼

Definition: Fuel_price*Mpy/Mpg

Inputs: ○ Fuel_price   Fuel price
○ Mpg   Miles per gallon
○ Mpy   Miles per year

Based on the definition you just entered, the value of *Fuel cost* is calculated by multiplying the values of *Fuel price* and *Miles per year*, and then dividing the result by the value of *Miles per gallon*.

11. Click the **Diagram** button to return to the influence diagram.

Note that *Fuel cost* is no longer diagonally shaded, indicating that it has a valid definition.

Diagram - Car cost moel

Miles per year → Fuel cost

Miles per gallon → Fuel cost

Fuel price → Fuel cost
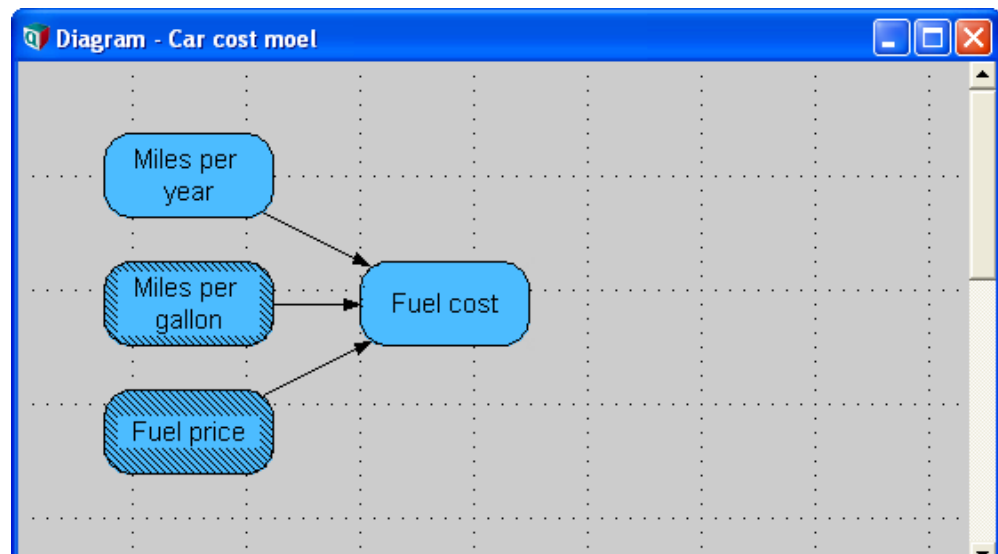
Entering attributes using the Attribute panel

Sometimes you may find it more convenient to view or edit attributes of a variable in the ***Attribute Panel*** as part of the Diagram view instead of using a separate Object Window.

In this section, you will enter data for the *Fuel price* variable in the **Attribute** panel.

1. Select the *Fuel price* node.

2. Click the key icon ♪ to display the **Attribute** panel.
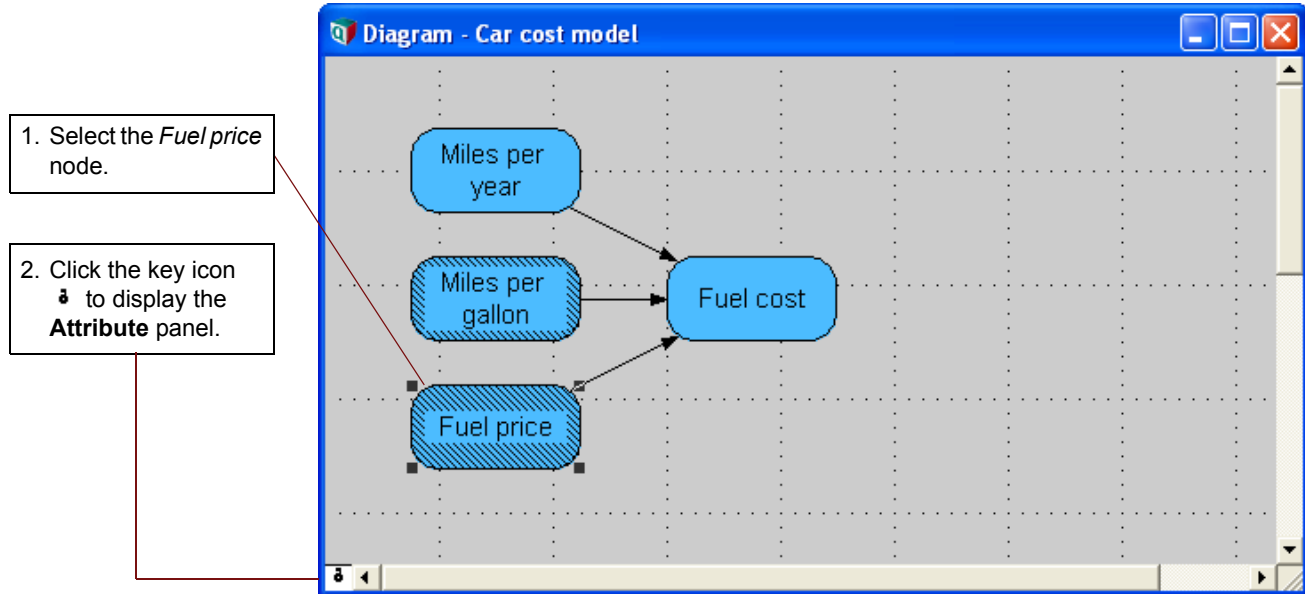
The Attribute panel appears below the diagram. You can use this view to enter or edit data for the currently selected variable in the influence diagram.

3. Click on the Attribute popup menu and select **Description**.

4. Enter the description as shown, then press *Alt-Enter*.

5. Select **Units** from the Attribute popup menu.

6. Type *US Dollars*, then press *Alt+Enter*.

Now that you have entered attributes for the *Fuel price* variable, you will enter its definition in the Attribute panel.

In this example we will assume that Fuel price has a fixed value of $3 per gallon.

Select *Definition* from the Attribute popup menu and enter *3*.

1. Select **Definition** from the Attribute popup menu.

   The warning icon ⚠ is there to remind you that the variable is not yet defined.

2. Enter *3* in the Attribute field.

   The *Fuel price* variable is now defined as an explicit value.

Note that *Fuel price* is no longer diagonally shaded, indicating that it has a valid definition.
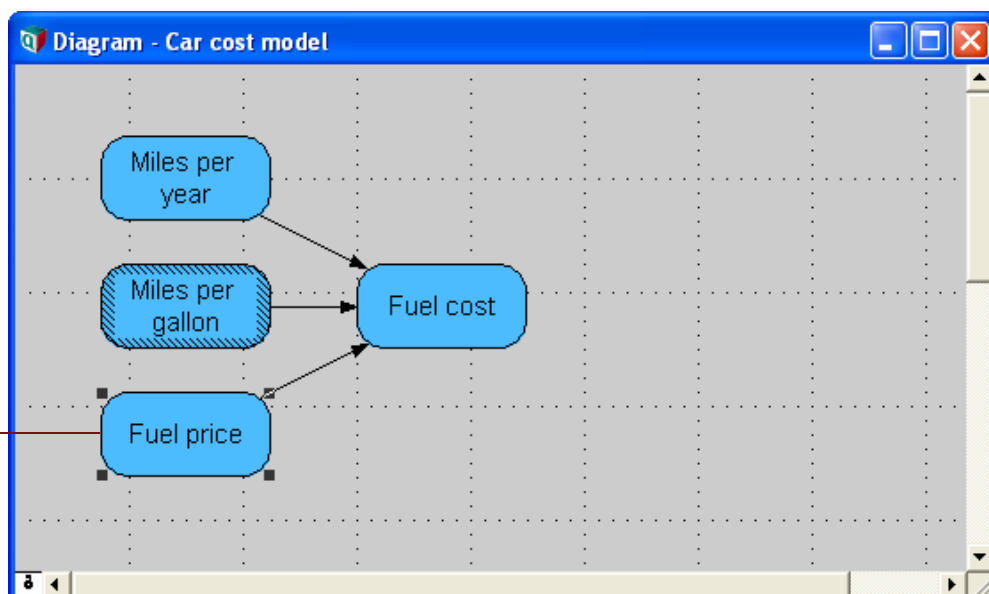
Defining a variable as a list

In this section you will enter attributes for the *Miles per gallon* variable and define it as a sequence of numbers. You will use this sequence later to perform a Parametric analysis to show how the *Fuel cost* is affected by *Miles per gallon.*

Using the Attribute panel, enter ***Average number of miles per gallon*** for the description and ***miles/gallon*** for the units.

1. Choose **Description** from the Attribute popup menu and enter: *Average number of miles per gallon.*
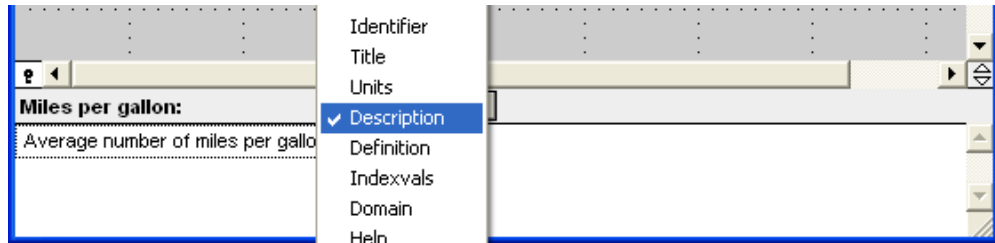
2. Choose **Units** from the popup menu and enter: *miles/gallon*
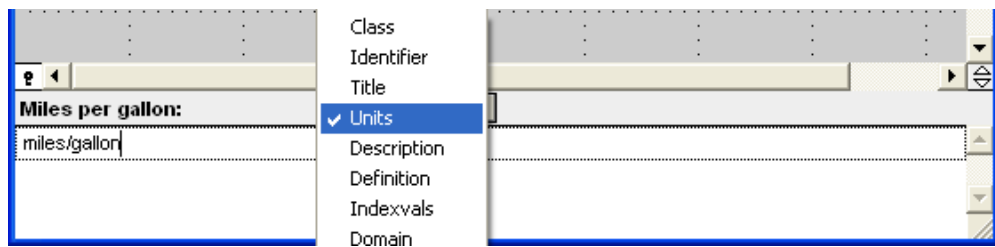
Now that you have entered attributes for the *Miles per gallon* variable, you will define it as a sequence of integers between 20 and 50, by increments of 10.

Select **Definition** from the Attribute popup menu. You will notice a second popup menu appears to the right. This is the Expression popup menu. Open the Expression popup menu and select **list**.

1. Select Definition from the Attribute popup menu.

The Expression popup menu appears next to the Attribute menu.
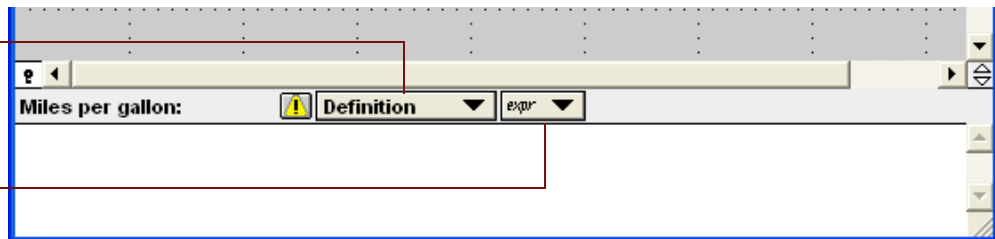
2. Choose **list** from the Expression popup menu
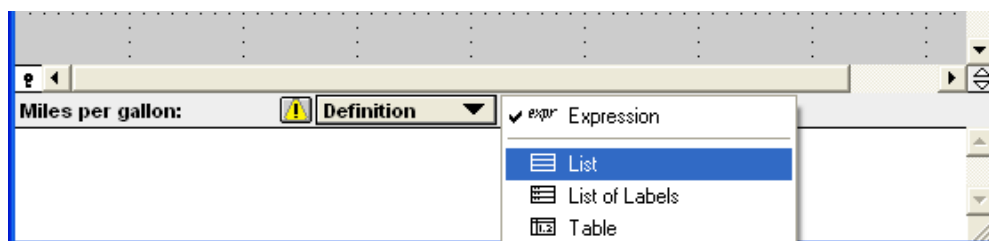
3. Type *20* in the first cell and press *Enter*

Analytica automatically sets the next value using the default increment of 1.
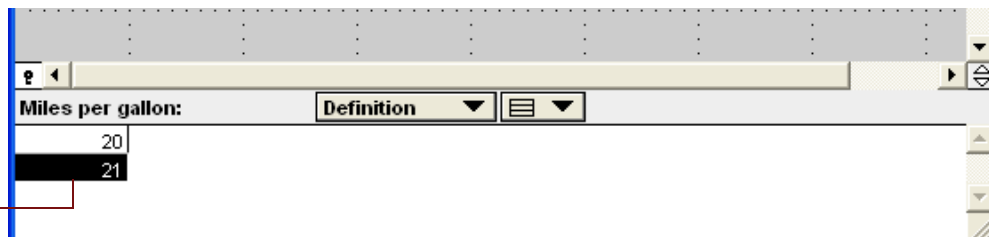
4. Change the second cell to *30* and press *Enter*.

Analytica automatically sets the next value using the increment implied by the first two values.
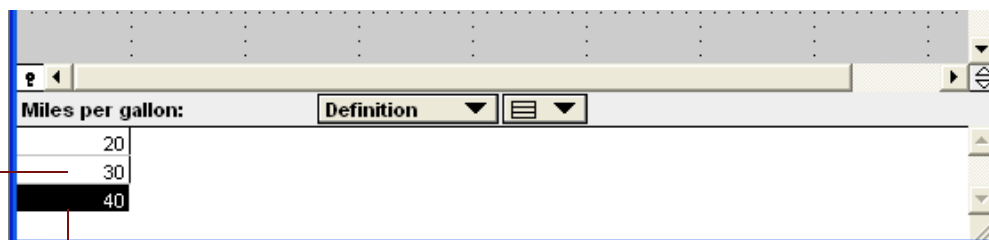
5. Simply press *Enter* one more time to complete the sequence.
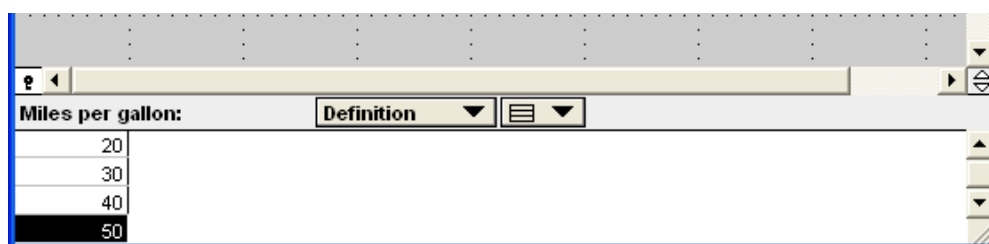
Although the auto-fill feature makes it convenient to enter a simple linear sequence, you are free to edit the values as you please.

Now you will use the *Expression view* to see the variable's definition in a different form. The Expression view shows the full syntax of the definition regardless of the type of variable.

6. Select **Expression** from the Expression popup menu

As an alternative to the list view, a list variable can be entered directly using square brackets as shown:
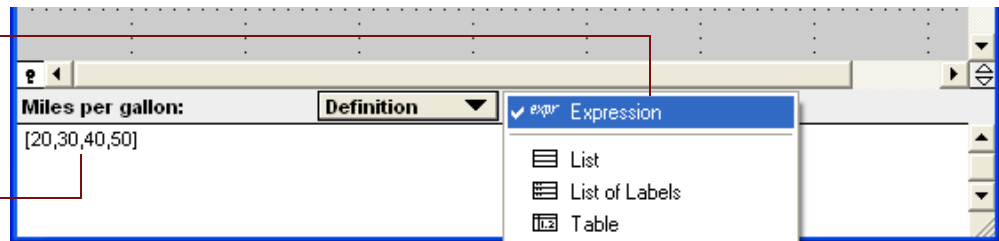
Viewing results in the Result window

Now that you have entered attributes and definitions for all variables it is time to see the result. The **Results button** calculates values for all selected variables.

Select the Fuel cost node and click the Results button ( ).

1. Select the *Fuel cost* variable.

2. Click the Results button ( )
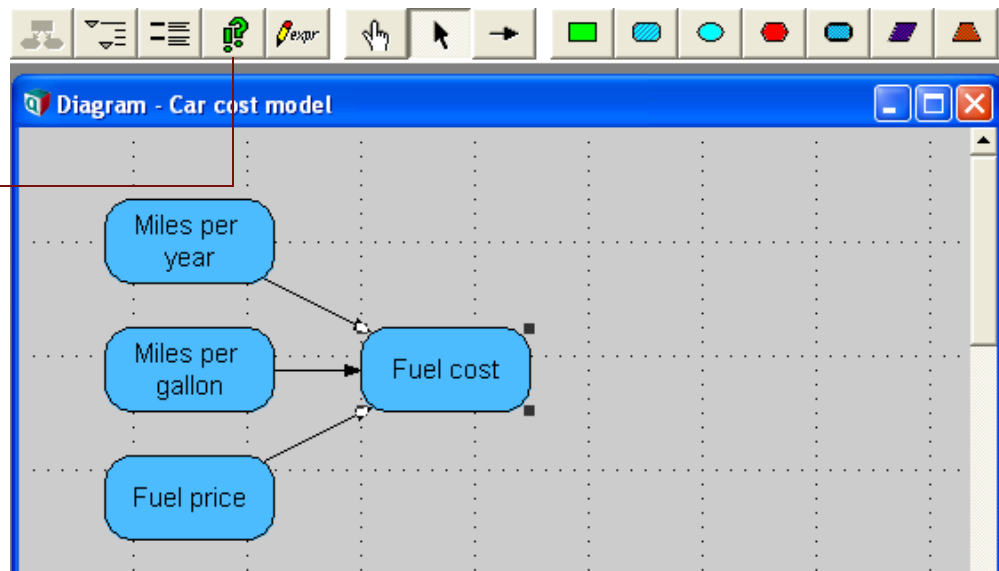
The result window appears. Icons in the upper left corner of the Result window control the view mode. Graph view ( ) is the default.

The result window appears in graph mode when the Graph icon is selected.
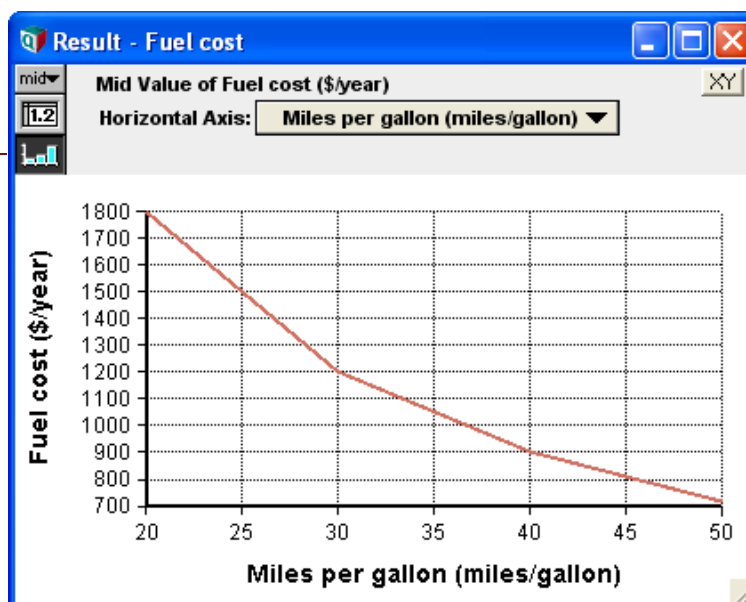
Tip   The default view for the Results window can be changed. See the "Preferences dialog" section in Chapter 4 of the *Analytica User Guide* for details.

Click the Table button (**1.2**) to view the results in tabular form.

Defining a variable using a built-in function

Analytica offers a wide array of built-in functions to simplify the process of defining variables. In this example we will take advantage of the **Sequence** function to re-define the *Miles per gallon* variable.

Let's suppose you want to change the sequence such that the interval between *Miles per gallon* values is 5 instead of 10.

Select the *Miles per gallon* variable. Make sure that the Attribute window is open and **Definition** is selected.

Next, open the Expression popup menu and select **Other** at the bottom of the list. This opens the **Object finder** where you will find a large collection of built-in functions. Analytica will prompt to confirm that you want to replace the current definition of the variable. Click **OK**.

The Result window appears in tabular form when the Table icon is selected

1. Select **Other** from the Expression popup menu.

2. Confirm that you are replacing the current definition by clicking **OK**.
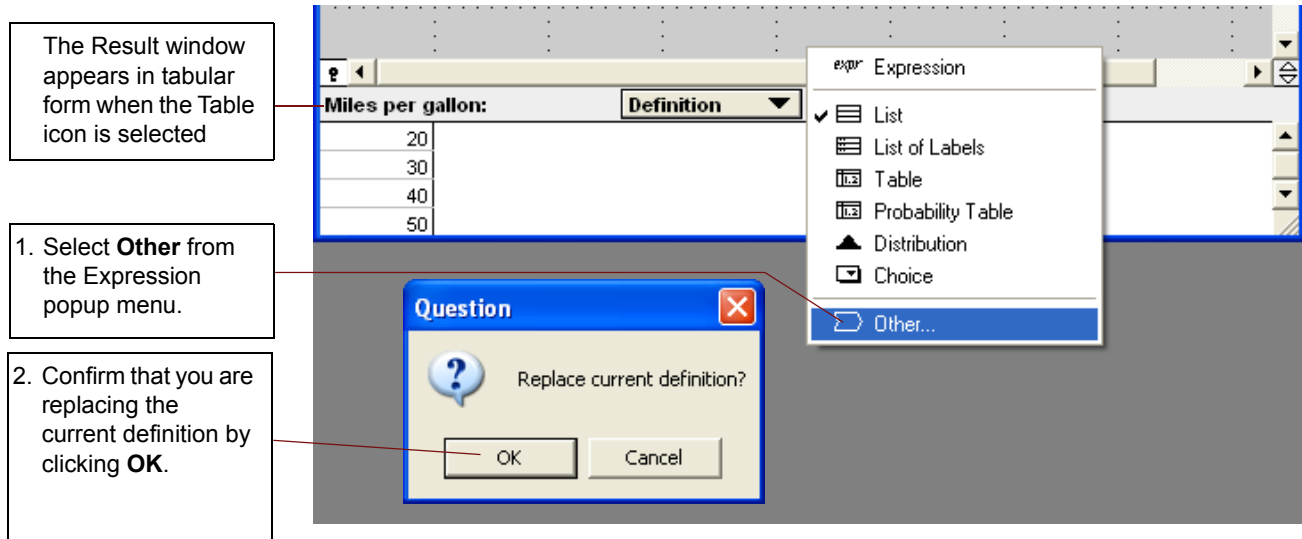
The **Sequence** function is contained in the Array library. Choose **Array** from the Library popup menu.

Scroll down the list and select **Sequence**.

Enter *20* for the *start* value, *50* for the *end* value, and *5* for the *step*.
(Leave the *strict* field blank.) Click **OK**.

3. Select **Array** from the Library popup menu.

4. Scroll down and choose the **Sequence** function.

5. Enter start value *20*, end value *50* and step value *5*.

6. Click **OK** to finish.
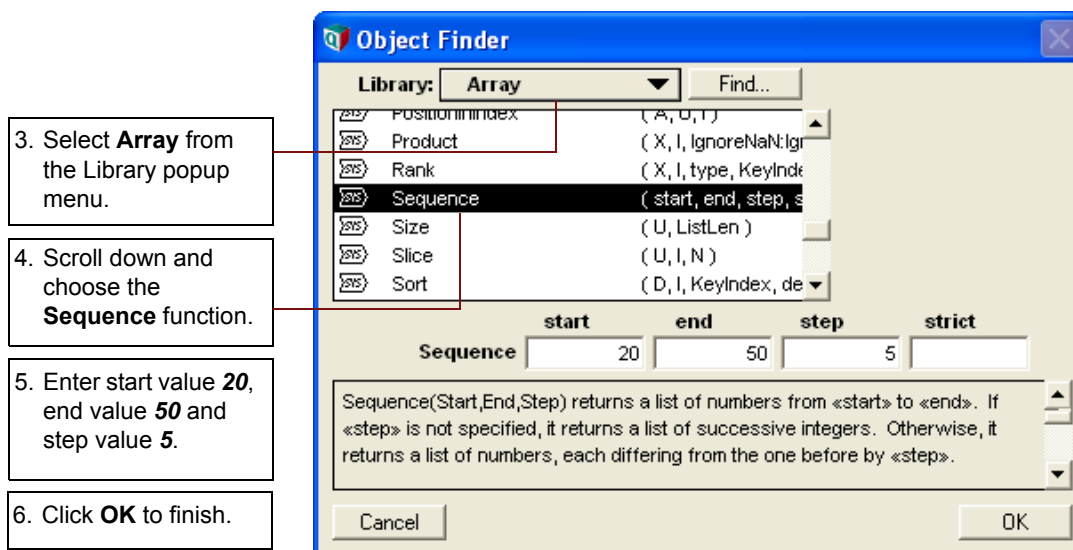
The new definition is displayed in the Attribute panel.

The Function button takes you back to the Object Finder window

Miles per gallon: Definition ▼   ▭ ▼

Sequence   (20, 50, 5)
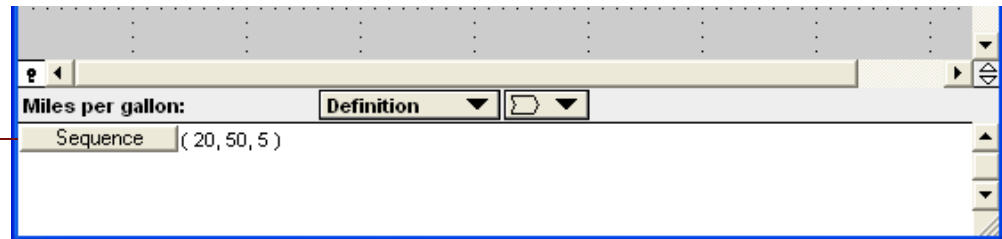
Select the *Fuel cost* variable and click the Results button ( ).

The result shows improved precision with the smaller *Miles per gallon* intervals.

7. Select Fuel Cost and click the Results button ( )

Toggle between table and graph views.

Result - Fuel cost

Mid Value of Fuel cost ($/year)

Miles per gallon (miles/gallon) ▼   ☐ Totals

20	1800
25	1440
30	1200
35	1029
40	900
45	800
50	720
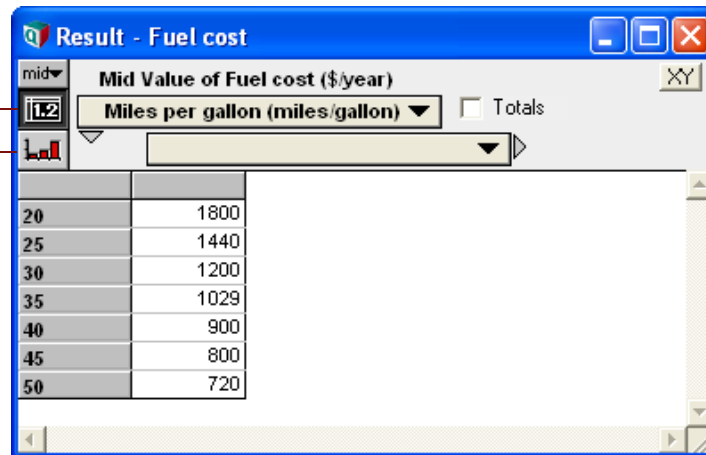
Saving your model

After you have created part or all of a model, you should save it. Select **Save** from the **File** menu (or press *control-s*). Because you previously saved your model, it is saved with the original name. You can quit Analytica by choosing **exit** from the **file** menu or closing the Analytica window.

Summary: Creating Models

In this chapter you have:

- Started a new model
- Created new variables
- Entered attributes for variables in two different ways:
  using the Objects window and the Attributes window
- Re-arranged variable nodes to improve the appearance of your influence diagram
- Drawn arrows to establish relationships between variables
- Defined variables as explicit values, functions and lists
- Used a built-in function to define a variable
- Displayed the result of a simple parametric analysis

In the next chapter, you will extend the *Car Cost* model to include tables of data.

Chapter 5

Working with Arrays (Tables)

This chapter shows you how to:

- Create and define an Index
- Define an array variable as a Table
- Understand principles of array abstraction when combining arrays in an expression
- Use conditional expressions and logical values when defining an array
- Define variables using Expression syntax
- Use local variables to simplify an expression
- Analyze a multi-dimensional result by pivoting indexes
- Reduce an array using subscripts
- Use array reducing functions such as **Sum()** and **NPV()**
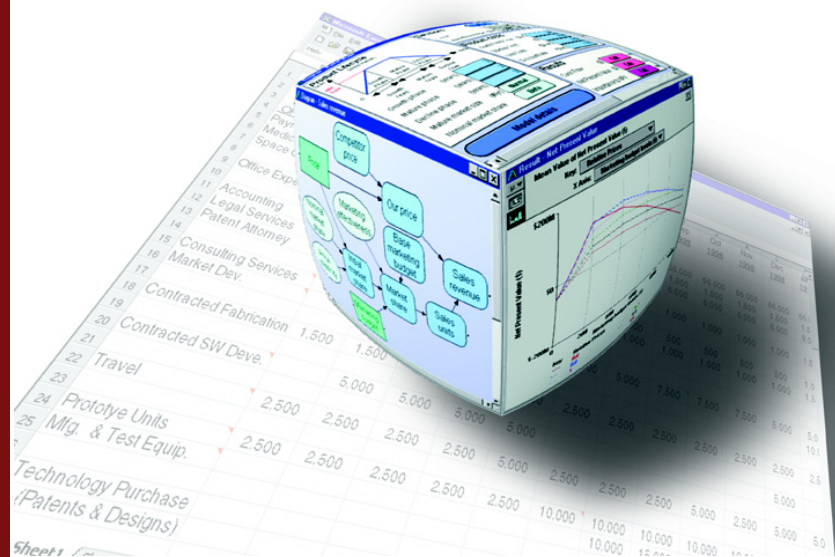
This chapter demonstrates *Intelligent Arrays*, one of the most powerful features of Analytica. An array can be generally defined as a variable to which multiple values are assigned simultaneously. For example, in the previous chapter you assigned a list of values to a single variable, **Miles per gallon.** This is an example of a simple one-dimensional array. When you defined **Fuel cost** using the **Miles per gallon** array variable as an input, it *also* became an array. This demonstrates the concept of *array abstraction*. Array variables can be used just like ordinary variables in expressions. Whenever you expand an index of the array to include more values, or even add an entirely new index along a new dimension, all dependent variables downstream will be extended automatically! *Intelligent arrays* allow you to scale your model without making any changes to the design.

This Chapter is rich in content and covers lots of ground. The example model is still somewhat simplified, but it is chosen to be complex enough to demonstrate as many important array concepts as possible. The workflow will be streamlined to allow you to concentrate on the ideas without getting bogged down with procedural details. Therefore, you should already be familiar with the basic mechanics of Analytica's user interface. The prerequisite skills include:

Creating a new model; Opening an existing model; Save; Save As... (See Chapter 1)

Creating and defining new variables; Entering attributes in Attribute or Object windows; Drawing influence arrows between nodes. (See Chapter 4)

Summarizing variables using Expression syntax

Although Analytica offers a convenient and visually intuitive interface with which to define variables, it is often helpful for users to be able to enter definitions directly using *Expression syntax*. Every variable type and every functional expression can be represented this way. In addition to its usefulness in building models, expression syntax is a convenient and efficient way to summarize examples in the documentation. This efficiency is evident in the following four-line summary of the **Car cost** model from the previous chapter:

```
Variable MPY := 12K
Variable MPG := Sequence(20,50,5)
Variable Fuel_price := 3
Variable Fuel_cost := MPY*Fuel_price/MPG
```

In order to expedite the process of building the model, we will summarize new variables in this format throughout the rest of this chapter. It is useful to be familiar with Expression syntax since you will also see it in User Guide and Wiki examples:

- The first term indicates the *Class* of the object being defined.
  There is a Class identifier for each type of node found in the node palette. These include:
  Decision, Variable, Chance, Objective, Module, Index, Constant and **Function**

- The second term contains the *Identifier* for the object.
  (Keep in mind that identifiers are not allowed to contain spaces.)

- A colon followed by a equal sign (`:=`) is referred to as the *assignment operator*.

- The expression to the right of the assignment operator defines the object.
  Conveniently, this portion of the summary line exactly matches the syntax that Analytica requires in the definition field. **In fact, you can copy this directly into Analytica if you wish**! For your convenience, the portions of the summary line that can be copied into the definition field will be highlighted in **blue**.

Fast Tony's Generic Wheels

In this chapter you will extend the **Car cost** model to address an important decision scenario: the purchase of a new car. There are three cars to choose from: *Standard*, *SUV* and *Hybrid*. There are also three finance options: *Cash purchase*, *Lease*, or *Loan*. As you can see, Fast Tony has presented you with an array of decision possibilities, a 3 x 3 array to be exact. Your objective is to determine the option with the lowest total cash outflow over a 24-month period, including fuel. (Maintenance costs are covered under warranty as long as Fast Tony's cousin, Greasy Tony, is on his feet that day.) Ownership equity will be represented as a positive cash flow on the last period.

Creating an index variable

Index variables define the categories along a dimension of an array. They are represented by index nodes (parallelogram shapes) in the influence diagram. In this example, your decision will be arrayed by **Car type** and **Finance Option** so you will need to define an index for each of these dimensions. Index variables are generally defined as a *list of labels* or a *list of values* representing categories in the array.

(For the finance options you will leave out "Loan" for now. It will be added later.)

```
Index Car_type := ['Standard','SUV','Hybrid']
Index Finance_option := ['Purchase','Lease']
```

For each of the index nodes listed above, do the following:

1. Choose the Edit tool

2. Drag the Index icon down to the diagram

3. Enter a title for the node

4. Open the definition field and select **List of labels** from the expression popup menu.

5. Enter labels as shown
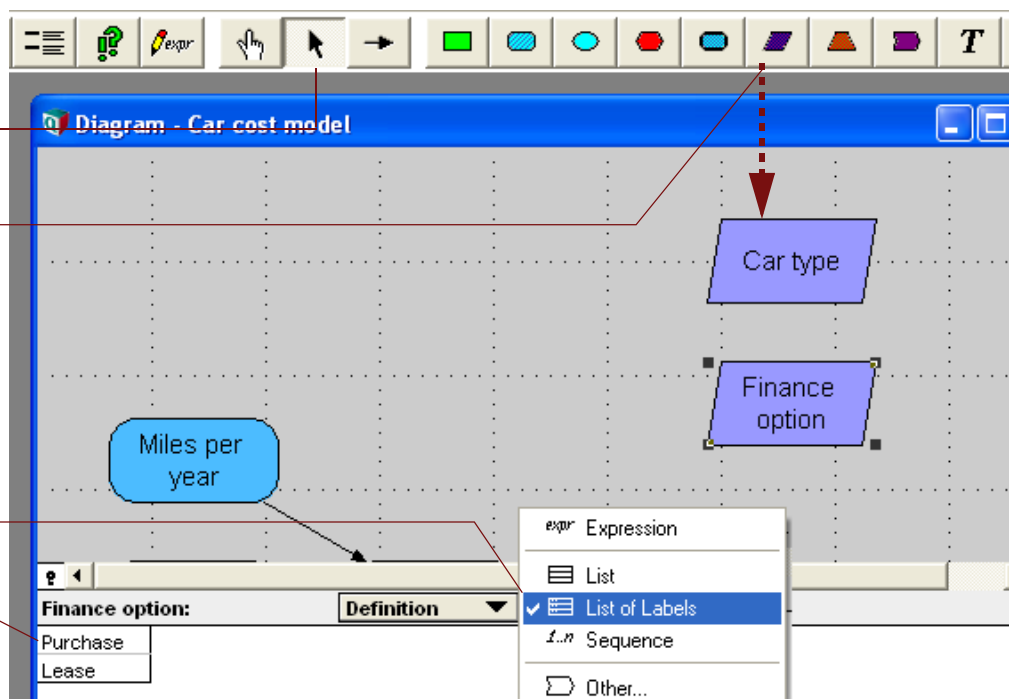
Tip As an alternative to selecting *List of Labels* in the Expression pop-up menu, you can select (*expr* Expression) and enter the definition directly using Expression syntax. Simply list the values separated by commas and encase them in square brackets [ ]. Note that all text values in Expression syntax must be enclosed in quotes (single or double).

Defining a variable as a table

Each of the cars has a unique purchase price. You will define **Car price** as a *Table* indexed by **Car type**. In this context, a "table" does not necessarily have two dimensions. It can have one (as in this case), two, three or more.

```
Variable Car_price := Table(Car_type)(22K,40K,24K)
```

1. Create a new variable titled **Car price**

2. Open the definition field, then open the Expression popup menu. Select *Table*.

3. The Indexes window appears. Select the **Car type** index and click the transfer button to move it to the selected Indexes list. Click **OK**.

After you close the Indexes window an *edit table* will appear.
Enter the following values for **Car price**:

Standard: *22K*

SUV: *40K*

Hybrid: *24K*

Click the Index button if you want to go back to the Indexes window.

Note that index categories appear as table headings.

Press **Tab** or **Shift-Tab** to move through the cells.

Tip You can draw influence arrows from index nodes to table variables to pre-populate the Indexes window with the chosen indexes. Influence arrows from index nodes are invisible by default but can be made visible by choosing the *Diagram* menu and selecting *Set diagram style*.

Next you will create a variable for lease payments. Fast Tony lists the following monthly payments for the two-year lease option:

Standard: ***$400***

SUV: ***$700***

Hybrid: ***$500***

Lease Payment is similar to the **Car price** variable. But this time you will enter the definition directly using expression syntax instead of using the edit table.

Variable Lease_payment := Table(Car_type)(400, 700, 500)

<table>
<tr>
<td>

1. Create a new variable titled **Lease payment**

2. Open the definition field and select *expr* Expression from the Expression popup menu.
(This will not change the definition of an existing variable.)

3. Enter the definition directly as shown

</td>
<td>

</td>
</tr>
</table>

As you can see, Table definitions follow a specific syntax. The **Table** declaration is followed by two lists enclosed in parentheses. The first list contains the indexes by which the table will be arrayed. (In this simple example there is only one index but there can be many.) The second list contains the values.

It is important for the values to be in the same order as the index categories. Recall that the Car type index was defined as:

Index Car_type := ['Standard','SUV','Hybrid']

which matches the respective order of values in the **Lease payment** variable. The same principle applies when there is more than one index. For example, a two dimensional array might look like:

Table (Index_abc, Index_123)
('a1', 'a2', 'a3', 'b1', 'b2', 'b3', 'c1', 'c2', 'c3')

It is generally easier to use an *edit table* when multiple dimensions are involved.

Finally, re-define the existing **Miles per gallon** variable as a table indexed by **Car type**. The MPG values for Standard, SUV and Hybrid are 28, 23, and 45 respectively.

Variable Mpg := Table(Car_type)(28, 23, 45)

Select the **Miles per gallon** variable.

Open the Definition field and select *expr* Expression from the Expression popup menu.

Enter the new definition as shown above (blue text).

Combining arrays

Now that you have established purchase and lease payments, you can combine them in one variable. This time the index will be **Finance Option**.

```
Variable Payments :=
Table (Finance_option)(Car_price, Lease_payment)
```

Create a new variable titled **Payments**.

Open the definition field and select *Table* from the Expression pop-up menu.

The Indexes window appears. Select the **Finance option** index and transfer it to selected indexes list. Click **OK**. An edit table will appear.

1. Enter *Car_price* next to **Purchase**

2. Enter *Lease_payment* next to **Lease**

3. Press *Alt-Enter*
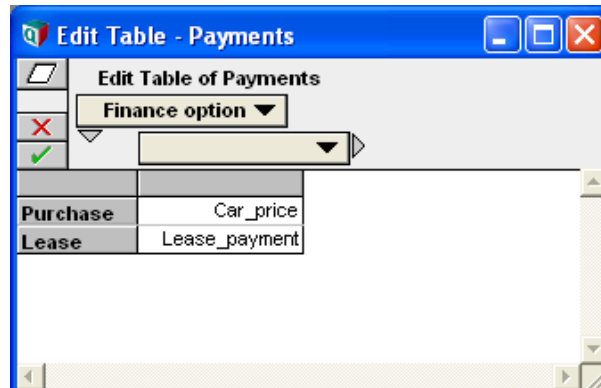
You may have noticed an important difference between this table and the previous tables. In the previous tables you entered number values into the table cells, but this time you are entering *expressions*! Expressions are valid entries for table cells. With very few exceptions they can be as complex as anything you might enter in a standard definition line, including formulas, logical expressions and conditional expressions. You will see more examples of this later in the chapter.

Now look at the result of this new variable.
Select the **Payments** variable and click the Results button ( )

Select the table view mode

Change vertical and horizontal indexes if necessary to pivot the table as shown.

Notice that the **Payments** array has two dimensions!

Something interesting has happened! Notice that the array contains *two* dimensions even though only one index is called out in the definition of the array:

```
Variable Payments :=
Table (Finance_option)(Car_price, Lease_payment)
```

Why is **Car type** included in the result when **Finance_option** is the only index included in the definition of the table?

The **Car type** index is included because it is an index of the input variables **Car price** and **Lease payment**.

You have just encountered one of the most powerful features of Analytica's array abstraction: The dimensions of arrayed input variables will automatically be incorporated into the resulting output of an expression. The general principle of array abstraction can be summarized as follows:

In Analytica, the result of an expression is arrayed by the *union set* of dimensions from all input variables, and the definition of the expression itself.

Adding a new dimension to an array

So far, the **Payments** variable is not very useful for comparing the purchase and lease options. Lease payments are paid every month instead of just once, and there is still ownership equity to consider. In order to represent payments as a stream of cash flows over time, you will need to add a new dimension to the **Payments** array:

Create and define a new index titled **Period**. Define it as a sequence of months from 0 to 24.

```
Index Period := Sequence(0,24)
```

Tip   There is a special shortcut syntax for sequences that have a step value of one: Type the starting value, followed by two periods (..), followed by the final value. For example, the **Period** index can be defined as: `Index Period := 0..24`

Next, you will edit the **Payments** definition to include the new index. Try using the edit table for this:

1. Select the **Payments** variable and open the definition field.

2. Select Table from the Expression pop-up menu.

3. Click the *Edit Table* button

There is a button in the top left corner of the edit table, labeled with a parallelogram. Click this button to open the Indexes window. Then add **Period** to the list of selected indexes.

1. Click the *Index button* in the top left corner of the edit table.

2. Select the **Period** index and add it to the Selected Indexes list

Note: By default, Analytica will copy current values to all new slices along the added index.

The edit table should now look like this:

The subset of an array corresponding to a singe index value is referred to as a *slice*. When you expanded the **Payments** array to include a new index, the original values (expressions in this case) were copied into all new slices along the **Period** index.

The **Payments** array is not yet accurate. Under the Purchase column you should have zero values for all periods except Period 0. Lease payments should be included in periods 1 through 24 but not Period 0. It would not take too much effort to edit all the cells to be the way you want them to be. The final definition of the array would look something like this:

```
Variable Payments := Table(Finance_option,Period)
(Car_price, 0, 0, 0, 0, ... 0, Lease_payment, Lease_payment,
Lease_payment...
```

Although this would work, it would not be a very efficient approach. In general, whenever you find yourself editing tables in the same way you would edit a spreadsheet, you are not benefiting from the advantages of array abstraction.

Using conditional expressions

Expressions can include conditional statements using If...then...else syntax. The condition can be based on a variable or an index value. This is a very useful way of defining array variables. For example, the cash flow for purchasers can be represented as:

```
If Period = 0 then Car_price else 0
```

Likewise, the cash flow for lease holders can be represented as:

```
If Period = 0 then 0 else Lease_payment
```

Specifying a condition based on **Period** will automatically add the **Period** index to the array. This would have been a better way to add a new dimension to the **Payments** variable, compared to the example above.

Let's revert the Payments variable to the way it was before adding the Period index:

```
Variable Payments := Table(Finance_option)(Car_price, Lease_payment)
```

As you did in the section above, Select the **Payments** variable, open the edit table, and click the Index button in the top left corner.

Select the **Period** index in the Selected Indexes list and transfer it back to left hand box. Click **OK**.

This time you will add the **Period** index by including it in a conditional expression. Edit the cells to include conditional expressions:

```
Variable Payments := Table(Finance_option)
(If Period = 0 then Car_price else 0,
If Period = 0 then 0 else Lease_payment)
```

Enter conditional expressions in the edit table cells as shown.
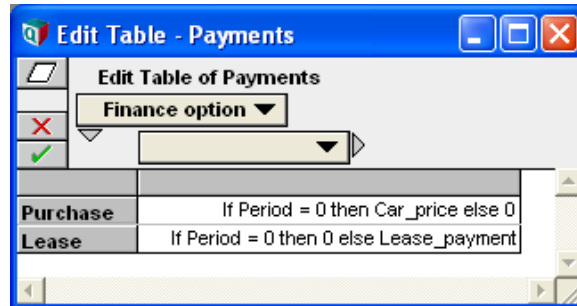
Let's take a step back to consider how many indexes are now included in the **Payments** array:

- **Finance_option** is called out specifically as a table index
- **Period** is the basis for a condition statement within an expression
- **Car_type** is an index of the **Car price** and **Lease payment** input variables.

Therefore, you can expect that **Payments** will be a three-dimensional array.

Conditional expressions can have nested IF...THEN...ELSE statements. In fact, it would be possible to define the **Payments** array using only one conditional expression:

```
Variable Payments :=
If Finance_Option = 'Purchase' then
if Period = 0 then Car_price else 0
else
if Period = 0 then 0 else Lease_payment
```

This would be equivalent to the table definition above.

Logical statements can also be used to specify a condition. They are enclosed in parentheses and include an equal sign. When used in a mathematical expression, a logical statement assumes a value of one if true, or zero if false. For example:

```
24 * (2 + 2 = 4) → 24
24 * (2 + 2 = 5) → 0
```

Viewing the results of a multi-dimensional array

Representing a multi-dimensional array on a two-dimensional screen presents a challenge. When viewing results in tabular form, Analytica will display only a two-dimensional slice of the data. You can change the orientation and selection of data by *pivoting* the table. To do this, select the desired row and column indexes from the row and column pop-up menus. Any leftover indexes become *slicer indexes* for which only slice of data (corresponding to a single index value) can be displayed at a time.

Select the Payments variable and click the Results button ( ).

Select the table view button ( ).

You can experiment by selecting different row and column indexes, and by changing the value of the slicer index.

The row index pop-up menu selects the index of rows.

The column index pop-up menu selects the index of columns.

The slicer menu button displays a pop-up menu of all slicer index values.

Slicer steppers change the value of the slicer index sequentially.

Now click on the graph view button ([graph icon])

The vertical axis of the graph represents data values. The graph indexes can be pivoted to select the desired horizontal axis, key, and slicer index. You can experiment with various graph configurations and slicer values.

The graph displays data for only one value of the slicer index.

The Horizontal Axis pop-up menu selects the index for the horizontal axis.

Key index categories are divided by color
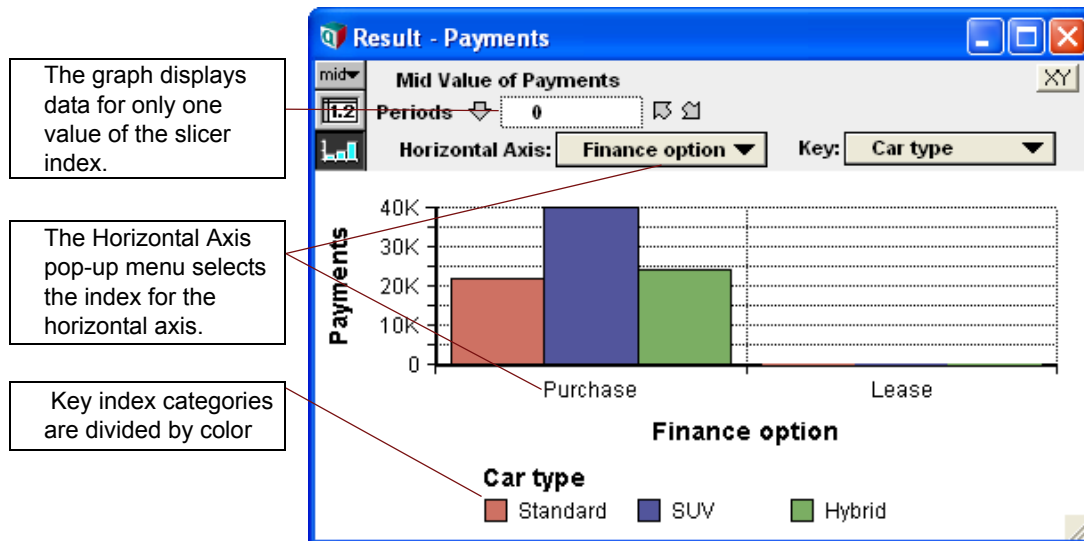
Tip You can explore various chart types and options by selecting *Graph Setup...* from the *Result* menu. For more details about graphs reference Chapter 2 of the User Guide.

Adding a new value to an index

Fast Tony's loan financing terms require 10% down and 2% per month for 60 months. His financing manager, Big Anthony, explains that the implied interest rate is 12%. There are severe penalties for delinquency but none of them are expressed in dollar values.

One of the advantages of *intelligent arrays* is that they are easily expandable. So far, you have set up cash flows for purchasers and lease holders. Adding a third financing option is a simple matter of adding a new value to the **Finance option** index.

> **Index Finance_option := ['Purchase','Lease','Loan']**

Select the *Finance option* index and open the Definition field. Select *List of Labels* from the Expression pop-up menu. You will see a list of current values in the definition field. Select the bottom value (**Lease**) and press the **Enter** key. A new box appears at the bottom of the list. By default the previous label is copied down. Change the new value to *Loan*.

1. Select List of Labels from the Expression pop-up menu.

2. Select the bottom category and press Enter

3. Enter **Loan** as the new value
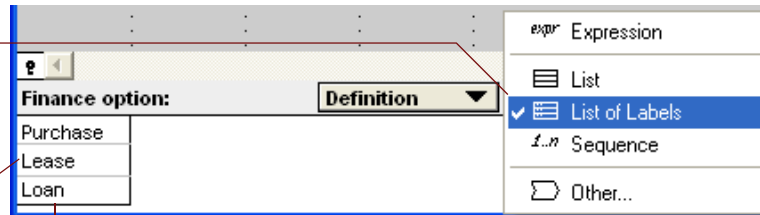
Now select the **Payments** variable and click the Results button ( ).

A new column appears with zero values by default.

1. Click the table view button

2. Pivot the table as shown

A new column appears for **Loan**

Now you will need to add an expression for loan payments in the edit table:

> **Car_price * (If Period = 0 then 10% else 2%)**

Select the **Payments** variable. Open the Definition field, then click the *Edit table* button.

Enter the new expression for Loan payments as shown.

Recall that the input variable **Car price** is indexed by **Car type**. Therefore, your new expression will be evaluated separately for the Standard car, the SUV and the Hybrid. The new expression also references the **Period** index, so it will be evaluated separately for each period as well.

Loan payments have been added for all values of **Car type** and **Period**.

Using local variables and indexes in an expression

Now that you have established all the cash outflows, you can start thinking about how much ownership equity you will have at the end of the 24-month period. Unavoidably you will need to calculate the remaining loan balance at that time. The financial dynamics of a typical loan include compound interest offset by fixed payment amounts. Although the formula for loan balances can be derived fairly easily, this section will focus on how to apply the formula rather than explaining the formula itself. Readers are invited to verify on their own if they wish:

Assuming that a constant amount is paid and nominal interest is compounded every month:

B_0 = Starting balance of the loan
`12%` = Nominal interest rate
`60`= The total number of periods required to pay off the loan
`Period` = A number from 0 to 24 representing the current period
`B(Period)`= The remaining loan balance for the current period

Let `A = (1 + 12% / 12)`
`B(Period) = `B_0` * (A^60 - A^Period) / (A^60 - 1)`

It is easy to verify that `B(0)=`B_0 and `B(60)=0` as required.

The quantity "`A`" appears several times in this formula so it makes sense to define it as an intermediate variable. To do the same thing in Analytica, you could define a new variable called `A` and use it as an input to the loan balance variable. But this would clutter your diagram with an unnecessary node.

Analytica allows *local* variables and indexes to be used temporarily within a definition. In this case you will define a local variable.

Create a new variable titled **Loan balance** and enter the following definition (copy blue text only):

```
Variable Loan_balance :=
var A := (1 + 12%/12);
Car_price * (1 - 10%) * (A^60 - A^Period)/(A^60 - 1)
```

Syntax rules for local objects are as follows:

- The instruction line starts with `var` or `index` to declare a local variable or index.
- The declaration is followed by an identifier. This can be anything you choose but it must not be identical to any other input contained in the expression.
- The *assignment operator* comes next (colon followed by an equal sign), followed by an expression defining the object.
- A semicolon (`;`) is required to end the instruction line. You may choose to have a visual break (Enter key) after the semicolon for appearance purposes but this is optional. Analytica only looks at semicolons to mark line breaks in code. Each local variable declaration must be on a separate code line.
- The final line of instruction contains the definition as usual. All declared local variables and indexes will be available to use in expressions.

Local objects are used temporarily and then forgotten after the expression is evaluated. Therefore, the same local identifier (i.e. "`A`") can be used in any number of nodes without conflict.

Create a new variable titled **Loan Balance** and enter definition as shown:
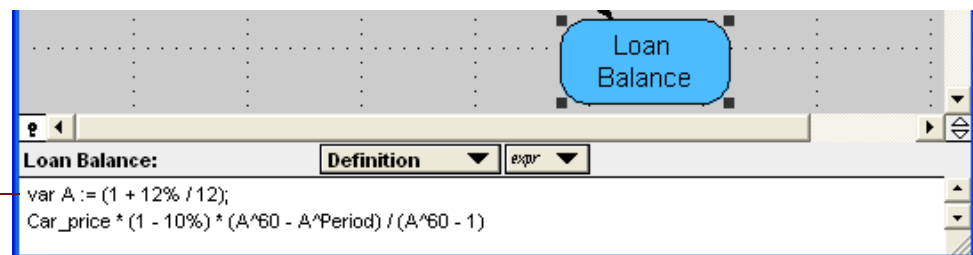
A final note about the expression for **Loan Balance**:

Although **Period** is an index, you have used it here as a variable in a mathematical expression. This is possible because **Period** is a list of numerical values. It would not have been possible to use **Car type** or **Finance option** in a mathematical formula since they are lists of labels (although they could be used in logical or conditional statements). In general, you can think of an index as a one-dimensional array that uses its own values as an index.

By the principles of array abstraction, you should expect **Loan Balance** to be a two-dimensional array indexed by **Car type** (an index of **Car price**) and **Period**.

Loan Balance is a two-dimensional array indexed by **Car type** and **Period**

Tip   The Future Value function **FV()** is a convenient way to calculate loan balances and annuity values without using formulas. Arguments in order are: Interest rate per period, Future period, Payments (negative for payouts), Initial value. For example, **Loan Balance** could be defined as:
`FV(1%, Period, -Car_price*2%, Car_price*90%)`
Reference User Guide Chapter 13.

Reducing an array using subscripts

Sometimes you may want to reference part of an array variable without using all of its available indexes. Subscripts allow you to referernce a *slice* of an array, effectively eliminating one or more dimensions. The subscript specifies indexes and their desired values for the slice. It is enclosed in square brackets placed immediately after the identifier for the array. For example:

```
Variable subscript_demo := Loan_balance[Period = 24]
```

This refers only to the **Loan Balance** values that correspond to Period 24. Recall that **Loan Balance** is a two-dimensional array indexed by **Period** and **Car Type**. The subscripted reference above will eliminate the **Period** dimension and return a one-dimensional array indexed only by **Car_type**.

The subscript has reduced a two-dimensional array to a one-dimensional array
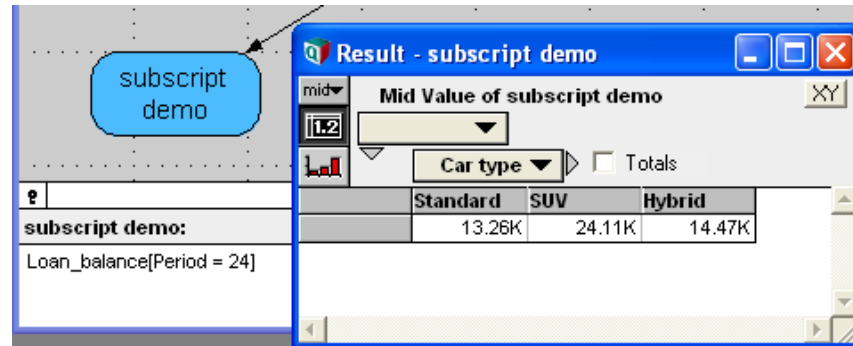
It is possible to eliminate multiple dimensions by including multiple indexes in the subscript, separated by commas. For example:

```
Variable subscript_demo :=
Loan_balance[Period = 24, Car_type = 'SUV']
```

The result of this expression will be a single value corresponding to the loan balance at Period 24 for the SUV: 24.11K

An Index value can also be selected according to its position in the index rather than the actual value. To make a positional reference, place the "at" symbol (@) in front of the index identifier. For example, 'SUV' is second in the order of values for the **Car type** index.

The following subscript references are equivalent:

```
Loan_balance[Car_type = 'SUV']
Loan_balance[@Car_type = 2]
```

The verbal terminology for this reference would be:
"**Loan_balance** sliced at position of **Car_type** equal to 2"

Completing cash flows

Now you can add a variable to represent the amount of ownership equity you will have after 24 months. Of course this will depend on the finance option.

Assume that the value of every car depreciates by 35% over two years.

- Cash purchasers' equity will be: `Car_price * (1 - 35%)`
- Lease holders' equity will be zero
- Loaners' equity will be the difference between car value and the loan balance at Period 24:
  `Car_price * (1 - 35%) - Loan_balance[Period = 24]`

You can define the **Equity** variable as a table indexed by **Finance option**. Each of the expressions above can be entered into the appropriate cell in the edit table:

```
Variable Equity := Table(Finance_option)
(Car_price*(1-35%),0,(Car_price*(1-35%)-Loan_balance[Period=24])
```

Alternatively, you can use conditional and logical statements as follows:

```
Variable Equity := If Finance_option = 'Lease' then 0 else
Car_price*(1-35%)-(Finance_option='Loan')* Loan_balance[Period=24]
```

Once again, consider how many dimensions your new variable will have. At first you might be tempted to list the indexes as follows:

Finance option (Specified as a table index or used in a conditional expression)

Car type (Index of the input variables **Car price** and **Loan balance**)

Period (Index of the input variable **Loan balance**)

But **Loan balance** is not an input to this expression; only a slice of it is used. The slice is arrayed only by **Car type**. Therefore, the Equity variable will be indexed by **Finance option** and **Car type** but not by **Period**.

The **Equity** variable has only two dimensions: **Car type** and **Finance option**. It refers to a single slice of the **Period** index.

Combining dissimilar arrays

Now that you have established payments and ownership equity, you can combine them into a single cash flow variable. **Payments** will be represented as negative cash flows. **Equity** will be represented as a positive cash flow on the last period.

Create a new variable titled **Cash flow** and enter the following definition:

```
Variable Cash_flow :=
If Period = 24 then Equity-Payments else -Payments
```

In this expression you have combined arrays with different numbers of dimensions:

Payments is indexed by **Car type**, **Finance option** and **Period.**

Equity is indexed only by **Car type** and **Finance option**.

There is no ambiguity here since `(Equity-Payments)` applies to a slice along the **Period** index. The slice is called out in the conditional statement.
Notice that `Equity` and `Payments[Period=24]` have the same dimensions.

There are other situations where you may want to use a scalar variable (not an array) as an input to an array. Similarly, you may want use an array as an input to another array that has more dimensions. In this situation Analytica will repeat the values of the smaller array for each slice along the new dimensions.

For example, suppose you want to incorporate monthly fuel cost into the **Cash flow** variable. Here again you are combining two dissimilar arrays:

Cash flow is indexed by **Car type**, **Finance option** and **Period**

Fuel cost is indexed only by **Car type**

Create a new variable titled **Cash flow with fuel** and define it as follows:

```
Variable Cash_flow_with_fuel :=
If Period = 0 then Cash_flow else Cash_flow - Fuel_cost/12
```

In this example, **Fuel cost** is applied to multiple slices along new dimensions:

• All **Period** values except Period 0

• All three **Finance option** categories.

Select **Cash_flow_with_fuel** and click the Results button (![icon]( ))

You will notice "Totals" check boxes next to the row and column index pop-up menus. Checking the box next to **Period** will allow you to see totals along that index.

1. Pivot the table as shown.

2. Check the "Totals" box next to the row index menu (**Period** index)

Array Reducing Functions

A function that reduces one or more indexes of an array is referred to as an *array reducing function*. The output of an array reducing function will have fewer dimensions than the input array. For example, the following expression will determine the sum of cash flows over all periods:

```
Variable Total cash flow := Sum(Cash_flow_with_fuel, Period)
```

The input array is indexed by **Car type**, **Finance option** and **Period**. The output will be a two-dimensional array indexed by **Car type** and **Finance option**. In this example the **Period** dimension has been reduced.

The **Period** index has been reduced by the Sum() function.

- The first argument of the **Sum()** function is the identifier of the variable to be operated on.
- This is followed by a list of indexes separated by commas. These are the indexes over which the function will operate.

In this example you have taken a sum over a single index.

Similar examples of array reducing functions include **Max()**, **Min()**, **Product()** and **Average()**.

Performing Net Present Value (NPV) analysis

If someone offered you a choice of receiving $100 today or receiving $100 two years from now, which would you choose? If you choose to receive it immediately, you can invest in a low risk bond with an annual return of, say, 6%. In two years the investment will be worth about $112. Clearly these two alternatives do not have equal value. Conversely, you would probably prefer to pay an obligation later rather than sooner.

Net present value (NPV) analysis is a way to compare various scenarios that involve inflows and outflows of money at different time periods. It converts a stream of cash flows to an equivalent scenario where all money is exchanged in a single transaction at the present time. The interest rate that can be achieved in a low risk investment is referred to as the *discount rate*. For example, given a discount rate of 6%, the NPV of $112 paid two years in the future would be $100.

By taking a simple sum of cash flows as in the example above, you have ignored the time value of money. A Net Present Value analysis is more useful in this case. The **NPV()** function in Analytica makes this easy. It is another example of an array reducing function.

Assume the discount rate is 0.5% per month.

Create an objective node titled **Cash flow NPV**.
(Objective nodes are represented by the hexagon shape on the node palette.)

Apply the following definition:

```
Objective Cash_flow_NPV := NPV(0.5%, Cash_flow_with_fuel, Period)
```

- The first argument of the NPV function is the discount rate per period.
- The second argument is the identifier for the variable containing cash flows.
- The third argument is the index containing period values.

The analysis is complete!

Summary: Working with arrays

In this chapter you have:

- Learned Expression syntax

- Created and defined index variables

- Created array variables as tables dimensioned along established indexes

- Expanded arrays by adding new indexes

- Expanded indexes by adding new index values

- Created examples that demonstrate the basic principle of array abstraction:
  The result of an expression is arrayed by the *union set* of dimensions from all input variables, and the definition of the expression itself.

- Used conditional and logical statements within an expression

- Established local variables in the definition field

- Analyzed a multi-dimensional result by pivoting indexes

- Reduced array variables using subscripts

- Applied array reducing functions to arrays

Chapter 6

Creating the Party Problem Model

This chapter shows you how to:

- Define a variable as a probability table
- Define a variable as a deterministic table
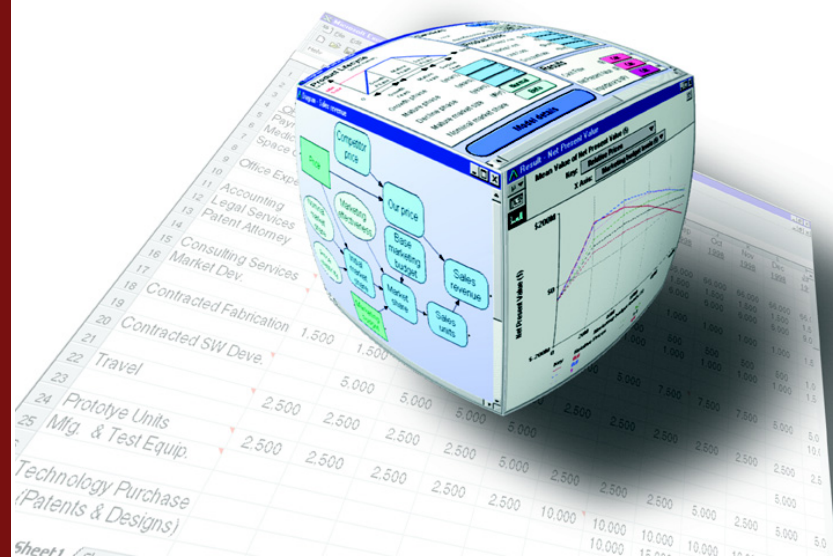
In this chapter you will create a new Analytica model called *Party Problem*. (For information about how to create a new model, see the beginning of Chapter 4, "Creating Models".) The **Party Problem** model illustrates the use of discrete probability distributions.
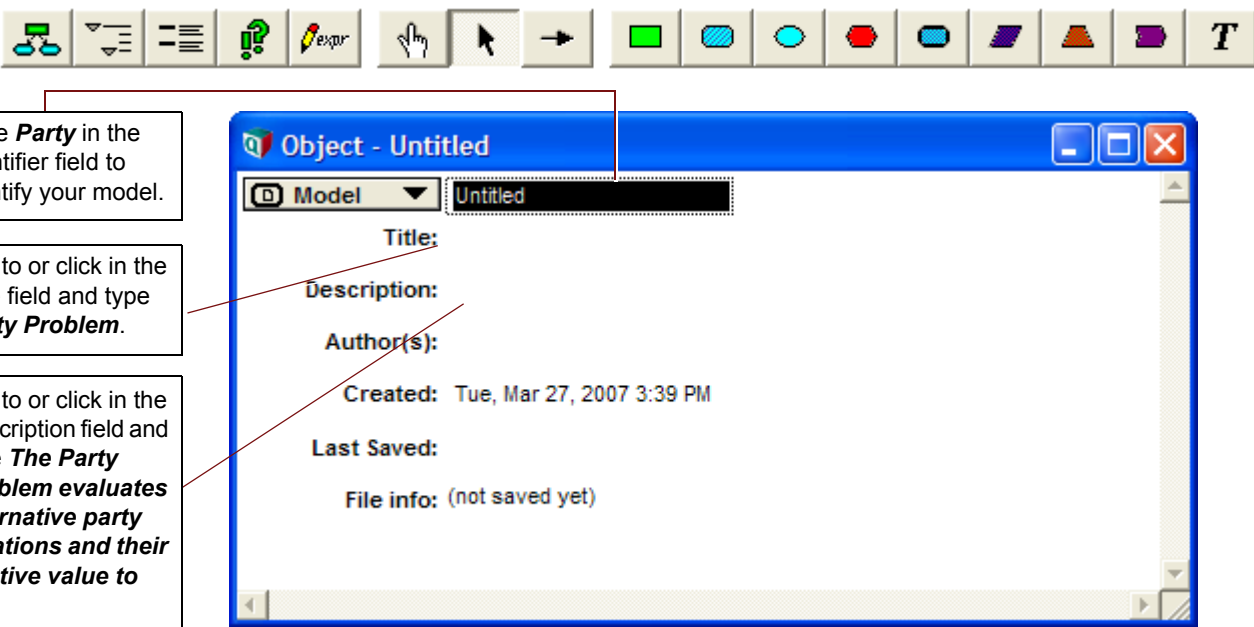
You should study this chapter if your models use discrete or conditional probabilities.

In many kinds of models, your variables can be described using probability distributions based on expert judgment or on empirical data. On those occasions when the outcomes are discrete or qualitative (for example, low, medium, or high), you might need to use **discrete probability** distributions.

In the **Party Problem** model, the key uncertain variable is weather: it could be sunny or rainy. The weather has an impact on the decision about the location of a party — indoors, on a porch, or outdoors.

Documenting the model

Specify the information shown in this model's **Object** window.

1. Type *Party* in the Identifier field to identify your model.

2. Tab to or click in the Title field and type *Party Problem*.

3. Tab to or click in the Description field and type *The Party Problem evaluates alternative party locations and their relative value to me*.

Object - Untitled	
Model ▼	Untitled
Title:	
Description:	
Author(s):	
Created:	Tue, Mar 27, 2007 3:39 PM
Last Saved:	
File info:	(not saved yet)

The **Object** window should now look similar to this:

4. Save the model by selecting **Save** from the **File** menu.

5. Click the **Diagram** button to bring the **Diagram** window to the front.
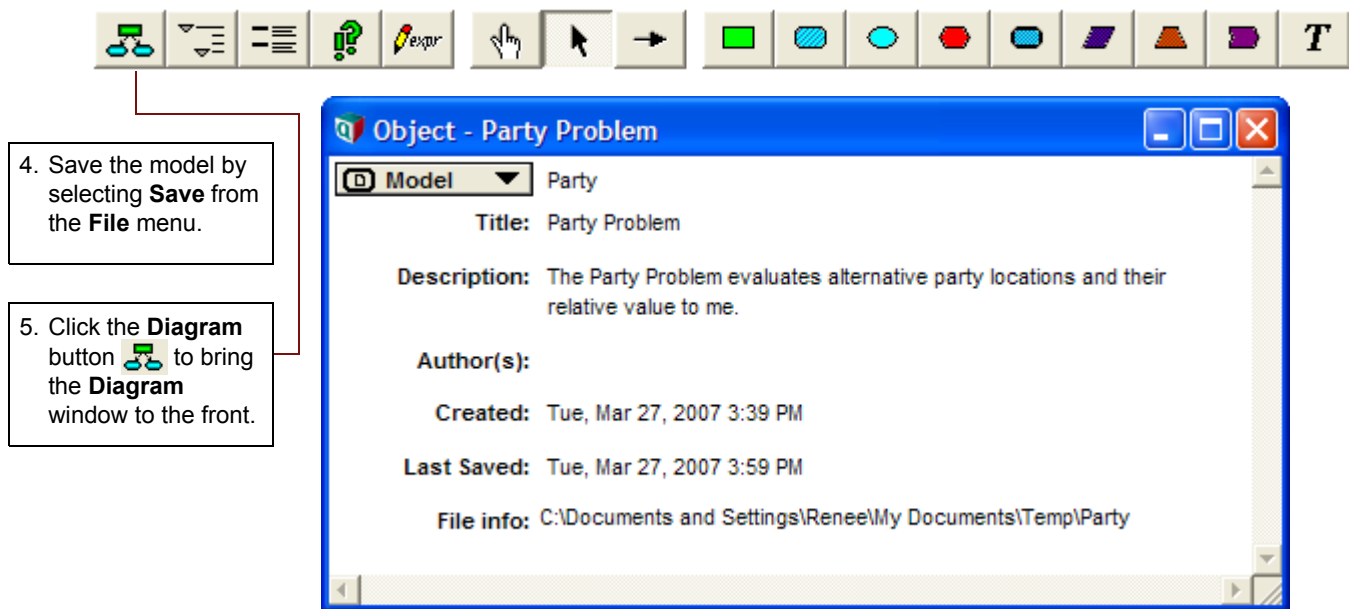
Creating the Party Location, Weather, and Utility variables

Using the techniques introduced in "Creating variable nodes" on page 61, you will create a **Party Location** decision variable, a **Weather** chance variable, and a **Utility** objective variable.

1. Create a decision node and title it *Party Location*.

2. Create a chance node and title it *Weather*.

3. Create an objective node and title it *Utility (value to me)*.
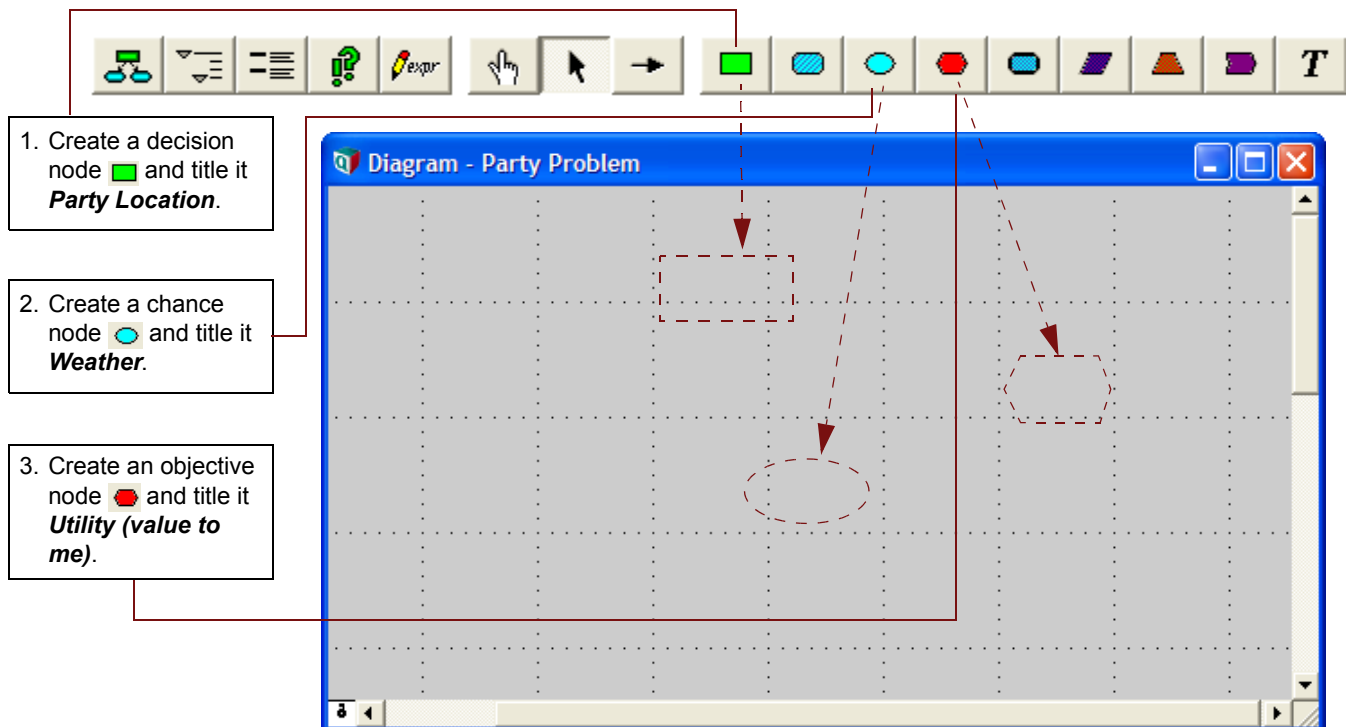
Your diagram should now look like this:
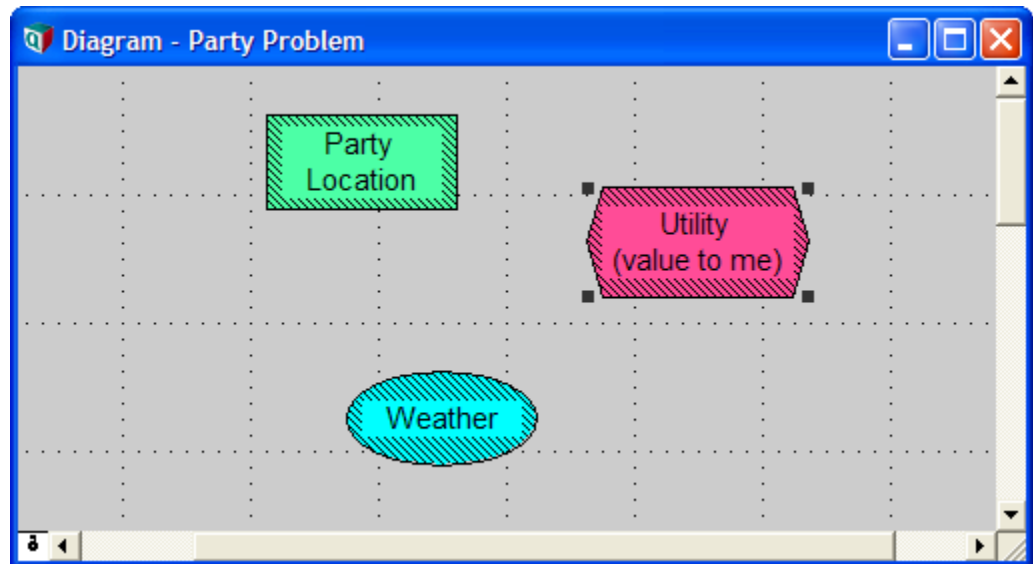
Drawing arrows between variables

In this section, you will draw arrows from *Party Location* and *Weather* to *Utility* using the techniques introduced in "Connecting multiple arrows" on page 70.

1. Select the **arrow** tool.

2. Select both *Party Location* and *Weather*.

3. Drag from either node onto *Utility*.
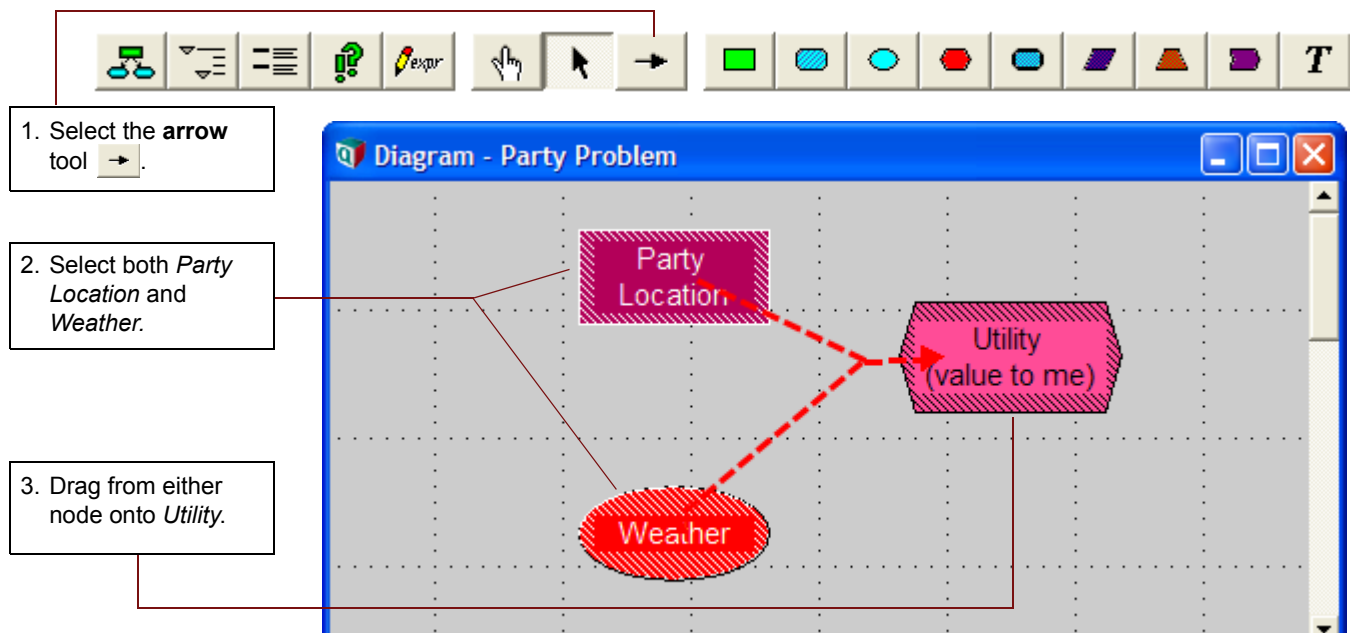
Your diagram should now look like this:
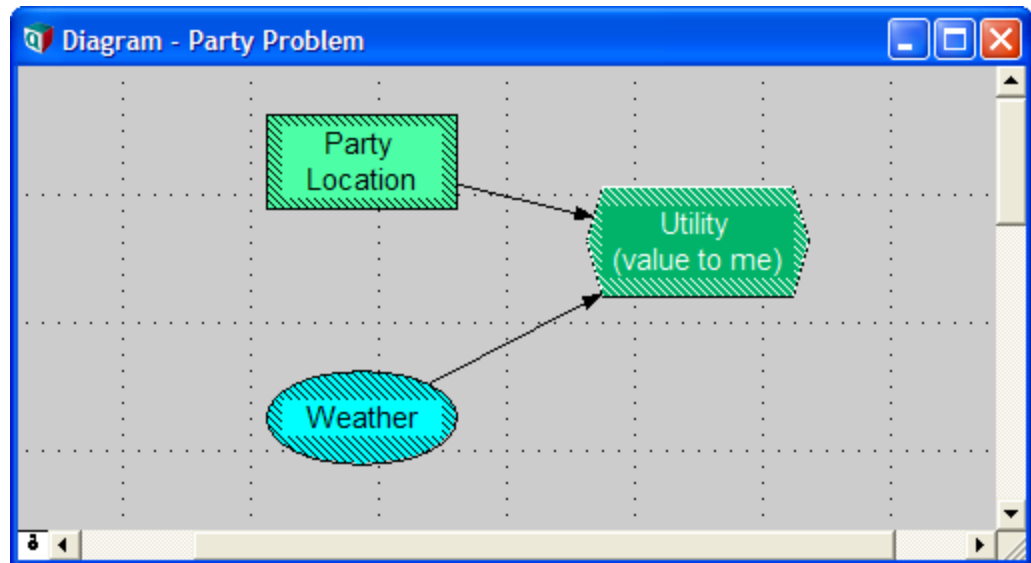
Defining Party Location as a list of labels

Consider that there are three possible locations where you could hold your party — indoors, on the porch, and outdoors. You will define *Party Location* as a list of labels identifying each location.

Party Location is used to index the *Utility (value to me)* objective node, so it is similar to the *Car Type* index variable created for the *Car Cost* model in the section "Creating an index variable" beginning on page 91. (*Party Location* is a decision variable, rather than an index variable, because it is directly under your control.)

1. Double-click *Party Location* to open its **Object** window.
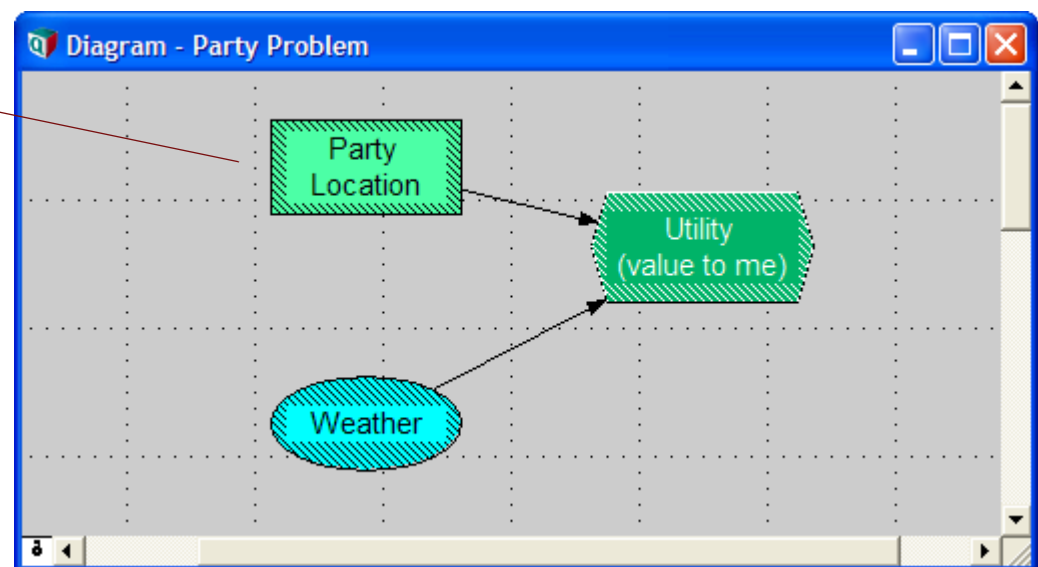
2. Click in the Description field and type *The party locations under consideration*.

3. Select **List of Labels** from the Expression popup menu.

4. Click in the first cell and type *Outdoors*; then press the *Enter* key.

5. In the next cell, type *Porch*; then press the *Enter* key.

6. In the next cell, type *Indoors*; then press *Alt+Enter* to accept the changes.

7. Click the **Diagram** button 🔗 to return to the diagram.
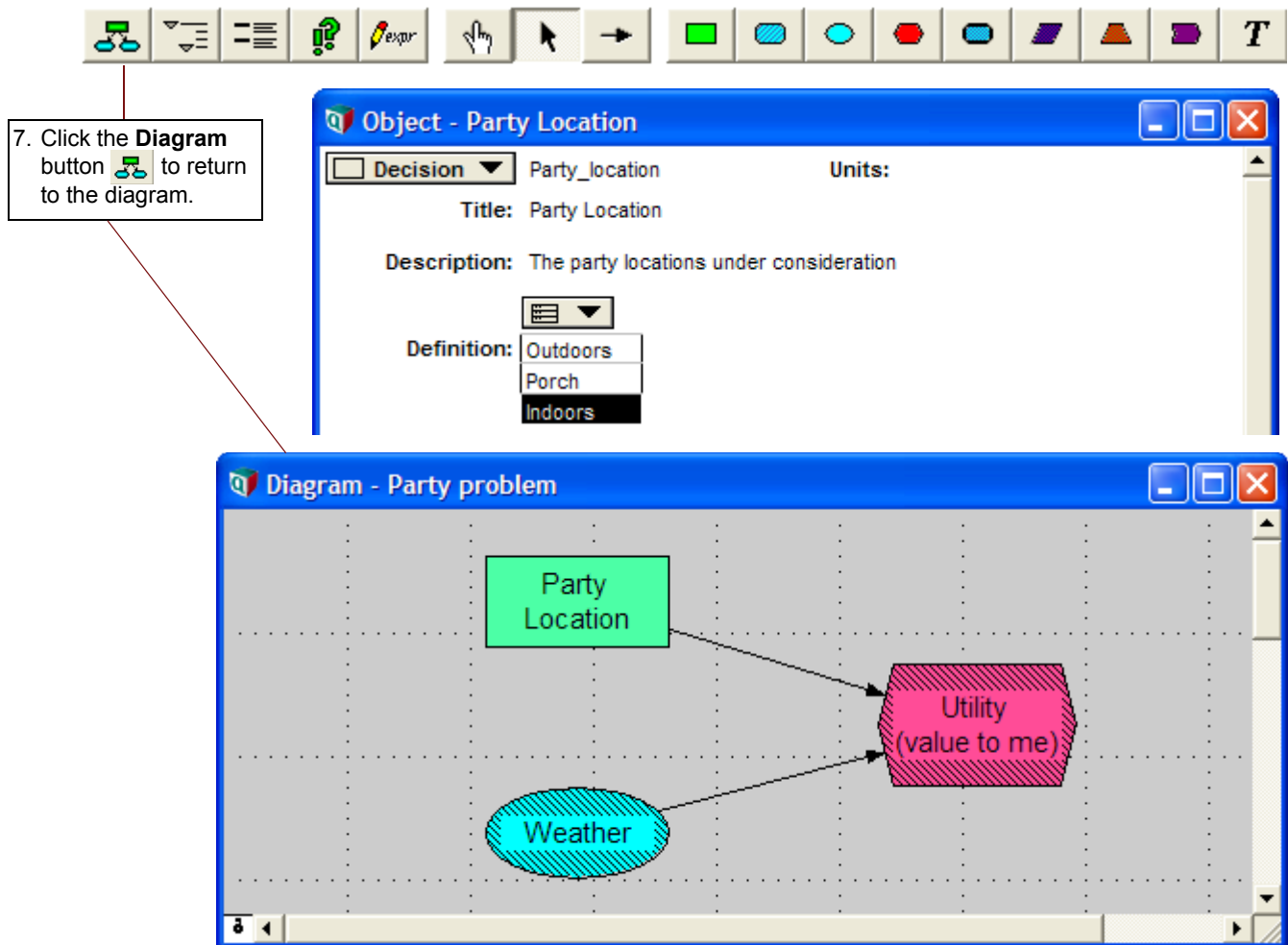
Defining Weather as a probability table

In this model, as in real life, weather is unpredictable.

In this section, you will characterize weather as having two possible discrete outcomes, either sunny or rainy. In addition, you will assign probabilities for each possible outcome. This is done by defining weather as a *probability table*. A probability table is Analytica's function for describing discrete probabilities.

1. Double-click *Weather* to open its **Object** window.
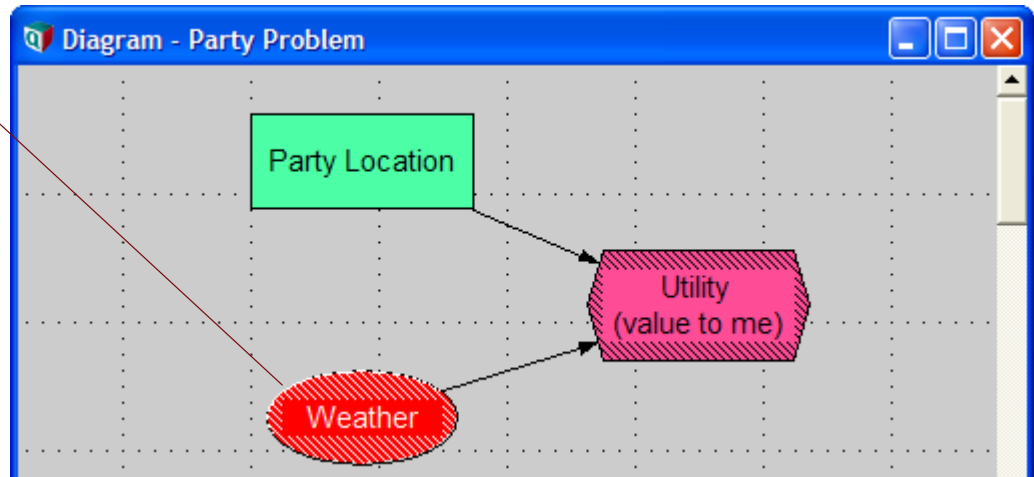
2. Click in the Description field and type *Weather outcomes and probabilities*.

3. Select **Probability Table** from the Expression popup menu.

The **Indexes** dialog box opens to confirm your choice of index for the table. *Weather (Self)* appears in the Selected Indexes list. *Self* indicates that the index — the possible outcomes of the discrete distribution — is contained within the probability table. *Self* is required as an index of a probability table.

4. Click the **OK** button.

An **Edit Table** window appears. In this table, the first column holds the outcomes and the second column holds their probabilities. You will enter the possible outcomes in the first column.

5. Click in the cell labeled **Item 1** to select the first label for Weather. Type *Sunny* into the cell, replacing **item 1**, and press the *Enter* key.
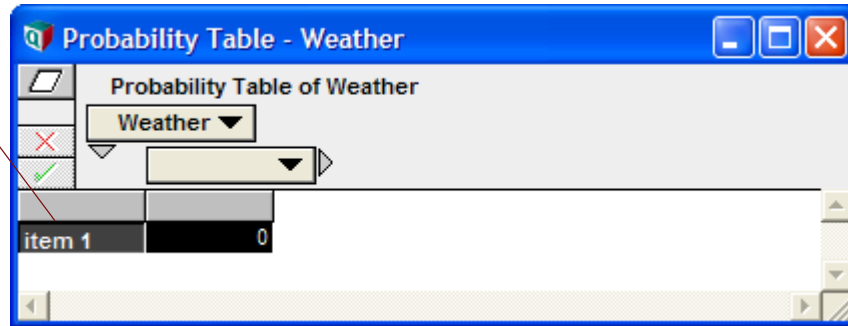
A second row should be added to the table, containing the same label as the first row, *Sunny*. If a second row does not appear, make sure you have the text for *Sunny* selected, and press the *Enter* key again.

6. A second row is added to the table. Type *Rainy* and press *Alt+Enter* to accept the entries.
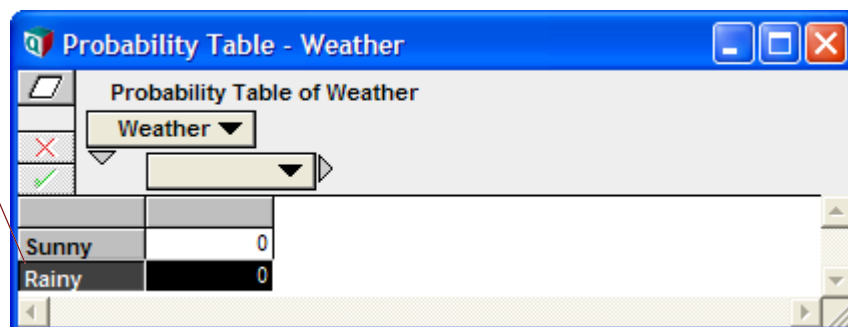
In the second column, you will enter the probabilities of the possible outcomes.

7. Enter the values *0.4* and *0.6* for the probabilities of sunshine and rain.

8. Click the check button ✔ to accept these entries.
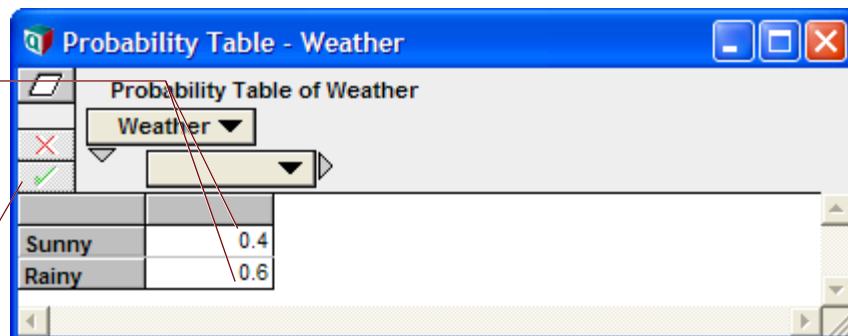
9. Click the **Result** button to evaluate *Weather*.
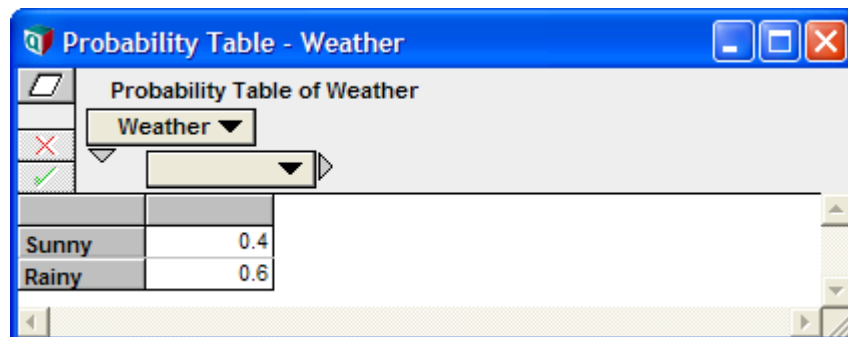
The mid value is *Rainy*, since more than 50% of the probability was assigned to *Rainy*.

10. Click the **Diagram** button to return to the **Diagram** window.

Result - Weather

mid▼   **Mid Value of Weather**   XY

Rainy
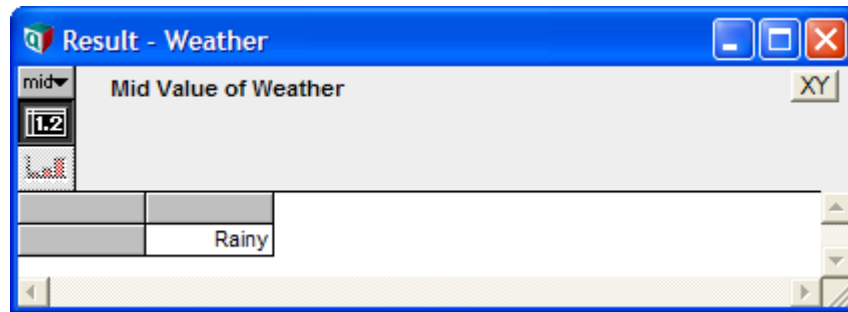
Defining Utility as a deterministic table

The utility of a party location decision depends on the outcome of the weather.

In this section, you will define *Utility* as a **deterministic table** (or **determtable**) using both *Party Location* and *Weather*.

A deterministic table appears similar to an edit table or probability table. At least one index of a determtable must be a discrete probabilistic variable (probability table). The result of evaluating a deterministic table takes into account the probability distribution described by the probability table index.

1. Double-click *Utility* to open its **Object** window.

Diagram - Party Problem

Party Location

Utility (value to me)
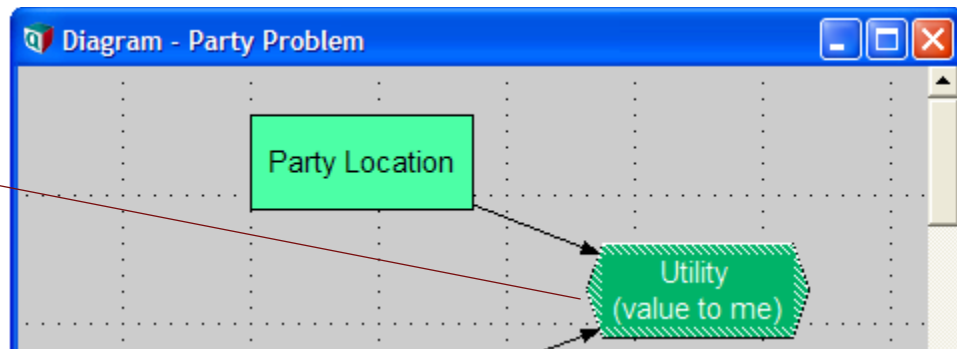
2. Enter the units and description as shown here.

3. Select **Other** from the Expression popup menu.

Object - Utility (value to me)

◯ Objective ▼   Utility_   Units: $

Title: Utility
        (value to me)

Description: The utility, or value to me, given a party location and weather outcome, assigned in dollars.

Definition:

✓ *expr* Expression

🔢 Table

🔢 Probability Table

▲ Distribution

▭ Other...

Inputs: rty location
                           eather
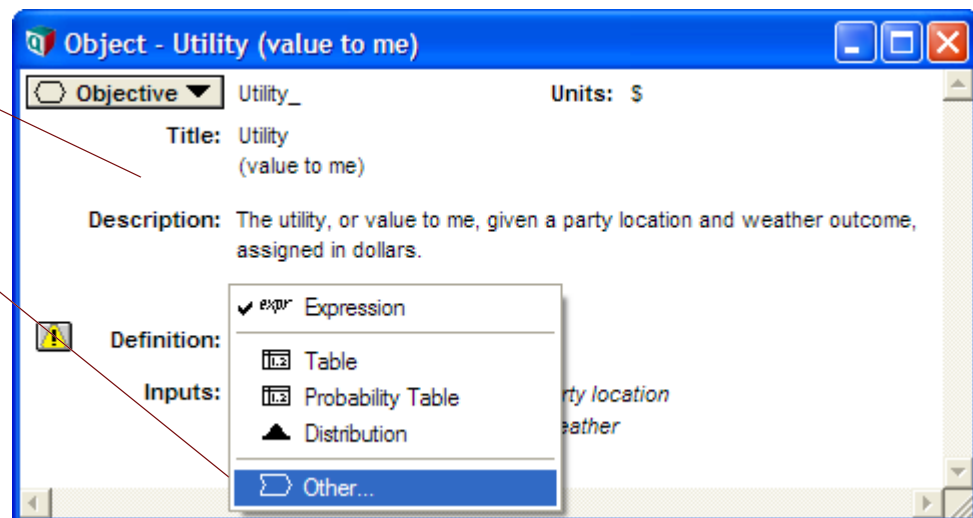
Because the *Utility* variable hasn't been defined yet, the first function in the first library is displayed. Libraries hold the functions available for you to use. For an overview of Analytica's libraries and their functions, see the section "Definition menu" in Appendix A of the *Analytica User Guide*.

You will select the **Determtable()** function, which is in the **Array** library.

4. Select **Array** from the Library popup menu.
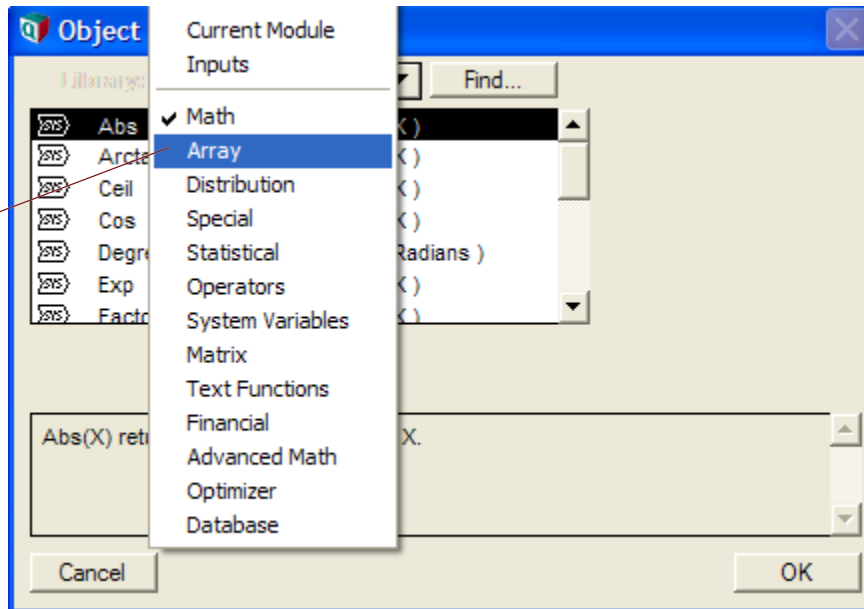
Tip   The **Object Finder** window gives a brief description of the selected function.

5. Scroll down the list and select **Determtable**.

6. To specify the indexes for the table, click the **Indexes** button.
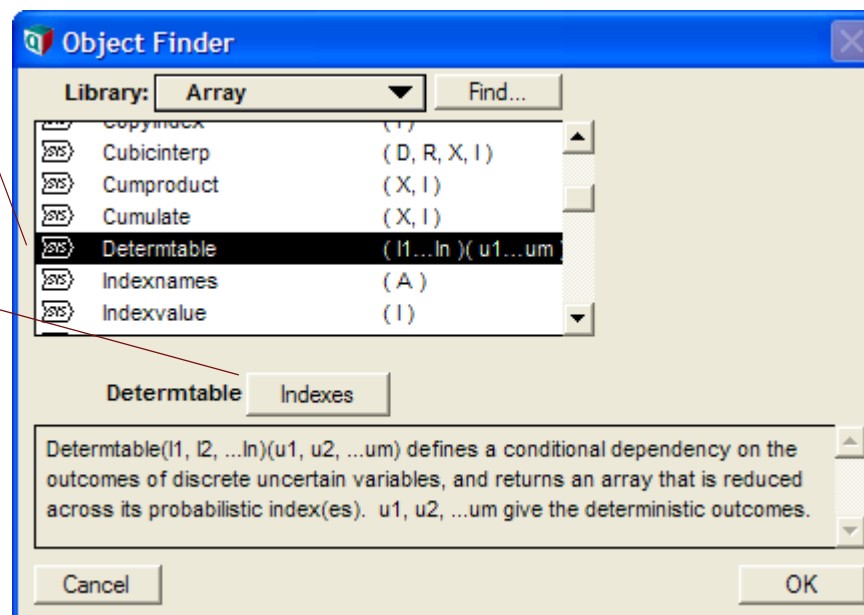
Party Location and *Weather* are already selected as indexes because you drew arrows from them to *Utility* and they are defined as lists.

7. Click the **OK** button to accept these indexes.
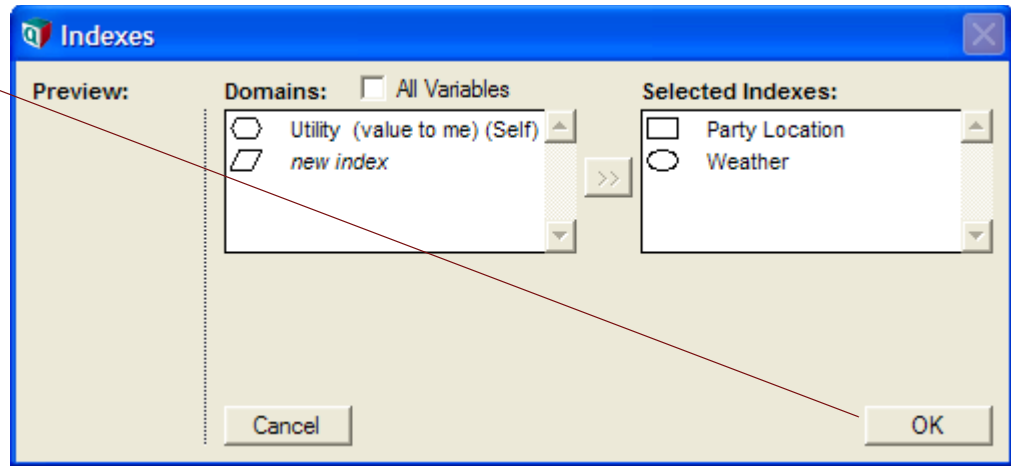
An **Edit Table** window opens for you to specify the dollar values for *Utility*.

You will give a value for each combination of party location and weather. These values describe the amount of money it is worth to you if you use a given party location and the weather turns out a certain way. Nominally, it is most valuable to have the party outdoors if it is sunny; it is least valuable if it is rainy and you are outdoors. Other values fall between these extremes.

8. Enter the values shown in the cells.

9. Click the check button ✔ to accept these entries.

10. Click the **Diagram** button 🖧 to return to the **Diagram** window.

Computing Utility

In this section, you will compute the value of *Utility*.

1. *Utility* should still be selected. Click the **Result** button 🔍 to compute the mid value.
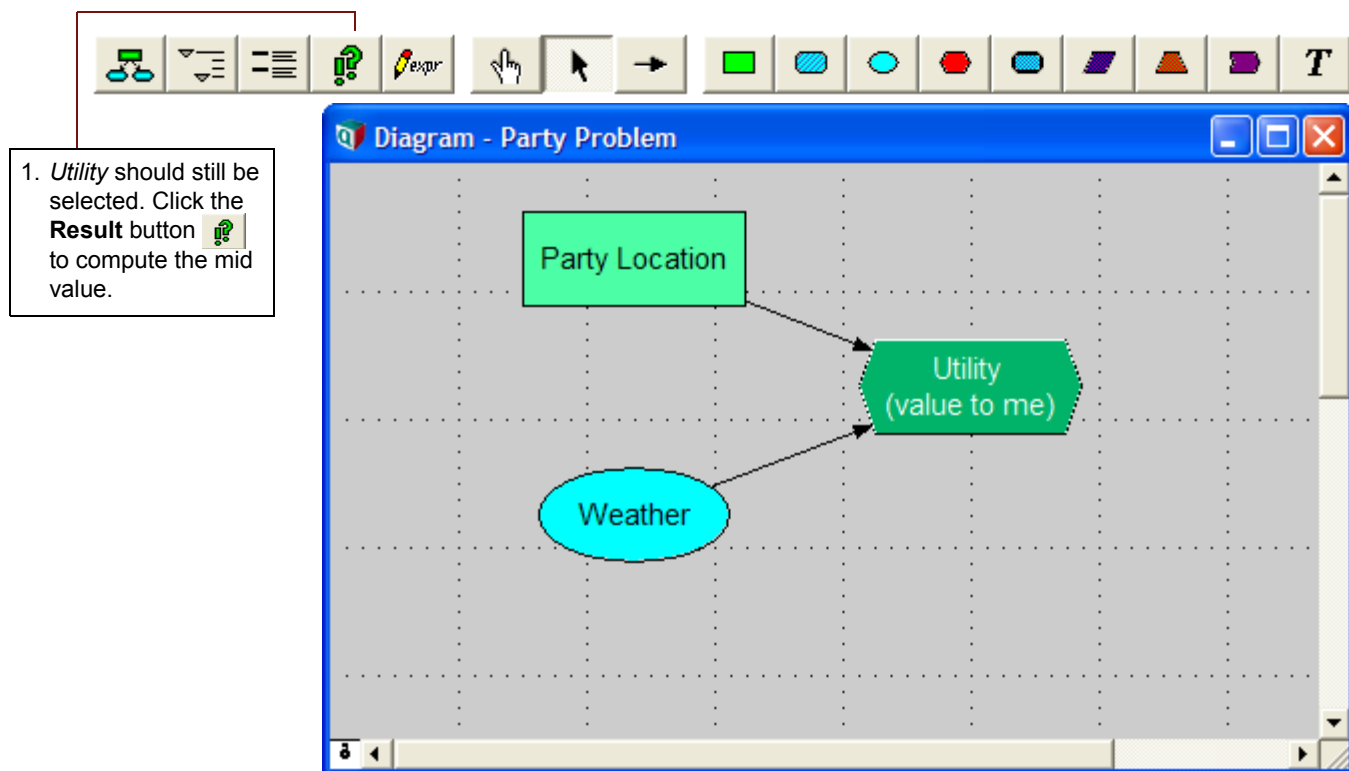
The utility of each party location corresponds to the values you just entered for *Rainy* in the *Utility* deterministic table, because the mid value of *Weather* is *Rainy*. With these outcomes, a deterministic assessment leads you to conclude that your best bet is to hold the party indoors.

2. Select **Mean Value** from the Uncertainty View popup menu.
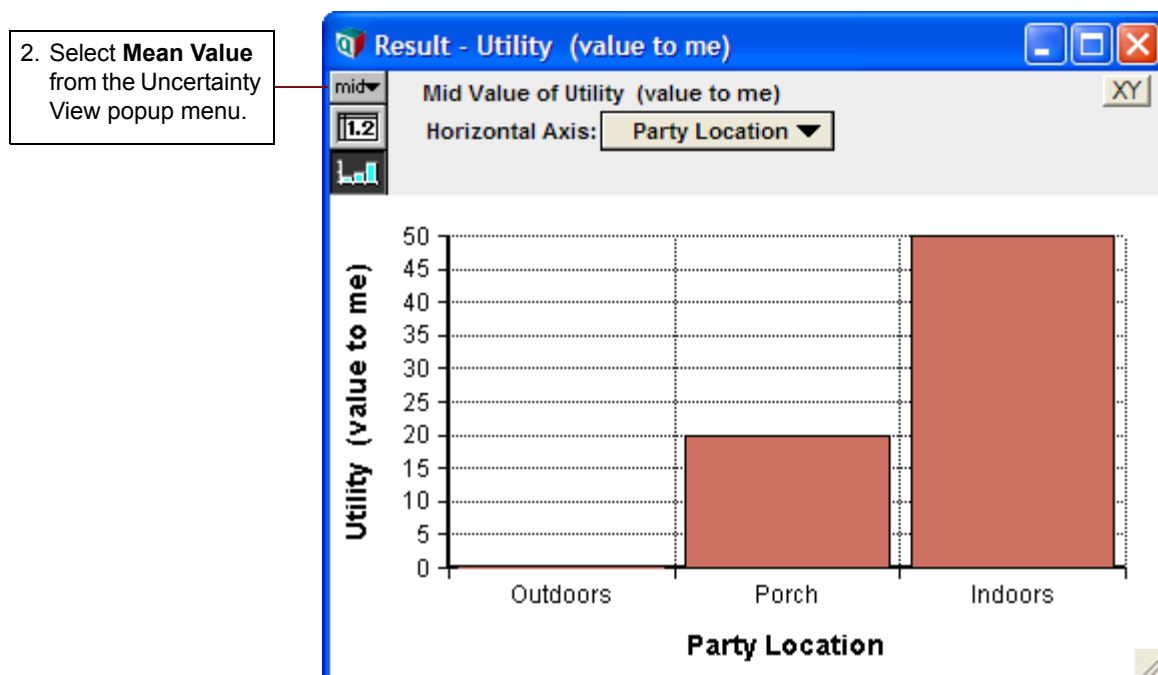
The mean value gives you a very different assessment: the porch gives you the greatest expected utility. The mean value as estimated by the sample is approximately 40 for outdoors, 48 for the porch, and 46 for indoors.
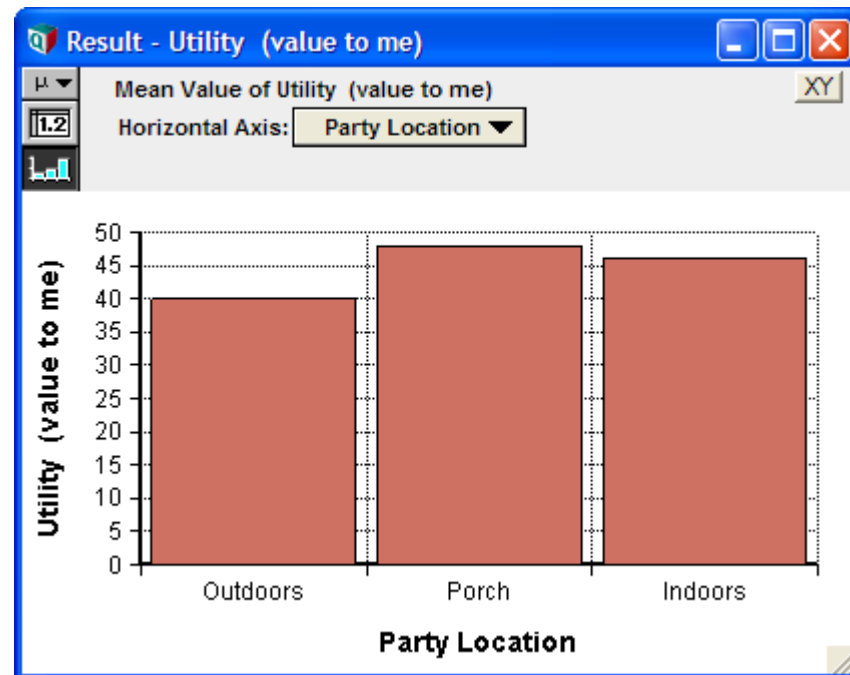
The exact expected utility for each party location can be calculated by multiplying the probabilities of the outcomes by the values of *Utility*:

```
Outdoors   = (100)*0.4 + (0)*0.6   = 40
Porch      = (90)*0.4 + (20)*0.6   = 48
Indoors    = (40)*0.4 + (50)*0.6   = 46
```

Note to those with a background in discrete modeling

Analytica simulates all probability distributions and calculates the expected (mean) value of a distribution by computing the average of the samples. For a discrete distribution, the computed mean converges, with increasing sample size, toward the value obtained by multiplying the probabilities by the discrete outcome values.

Creating the Party Problem model: summary

In this chapter, you have:

- Defined a variable as a probability table, a discrete distribution.
- Defined a variable as a deterministic table, a function that defines a conditional dependency on the outcome of a discrete uncertain variable.

Exercise

As an exercise, extend the model to examine how the utility of each party location varies as the probability of rain varies from 0 to 100%.

1. Create another chance node, titled **p**, the probability of sunshine. Define it as the range of probabilities from 0 to 100%, as **Sequence(0, 1, 0.5)**.
2. Redefine the probabilities for *Weather* as **p**, for sunny, and **(1-p)**, for rainy.
3. Recalculate the mean value of *Utility*. Display the result as a graph.
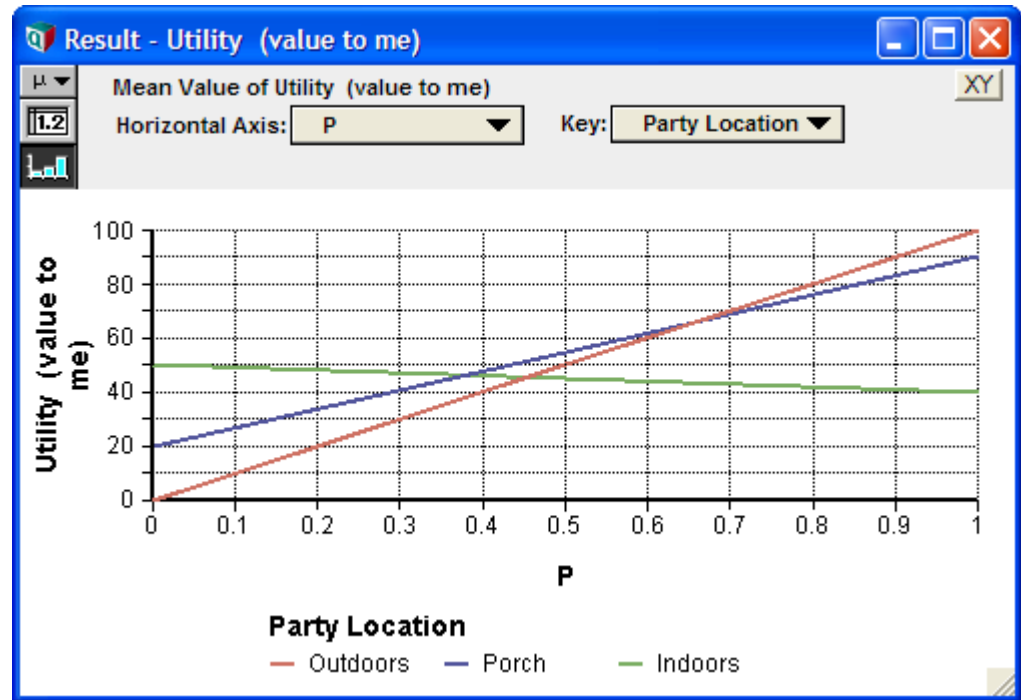
Chapter 7

Creating the Foxes and Hares Model

This chapter shows you how to:

- Use the **Dynamic()** function and the system variable *Time*
- Customize the appearance of variable nodes
- Organize a model using modules
- Make duplicates of nodes and modules
- Create aliases of variable nodes
- View the results of two variables simultaneously
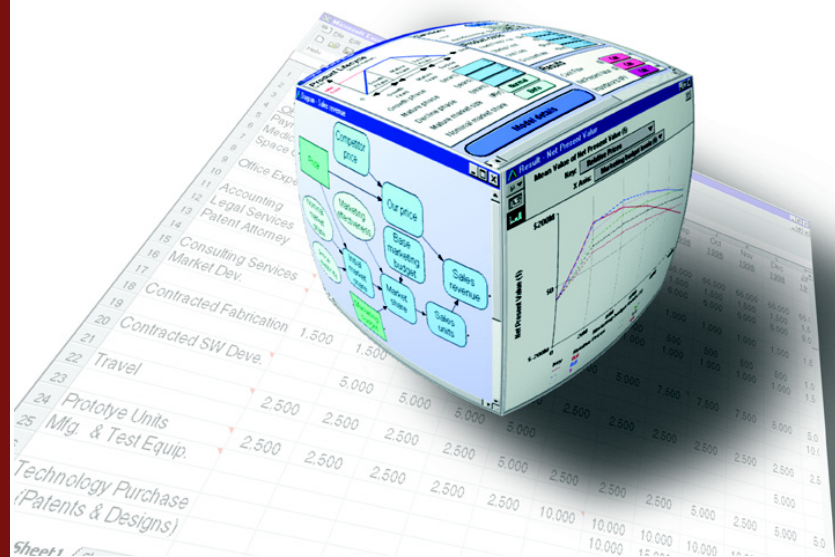
Before starting this chapter you should know how to perform the following actions which will no longer be described in detail:

Create a new model; Open an existing model; Save; Save As... (See Chapter 1)

Create and define new variables; Enter attributes in Attribute or Object windows; Draw influence arrows between nodes. (See Chapter 4)

You should also be familiar with Array variables and Index variables. (See Chapter 5)

Using the Time index

In this model you will set up a dynamic system to model a predator/prey relationship. The populations of foxes and hares will be set up as state variables: one-dimensional arrays indexed by Time.

Start by opening a new model and titling it *Foxes and Hares Tutorial*.

Create a general variable titled *Hare population*.

Using either the Attribute window or the Object window, place the cursor in the definition field and choose Table from the expression popup menu. The Indexes window will appear.

Notice that *Time* is listed as one of the indexes in the left hand window. You did not have to define this index before using it. *Time* is a system index in Analytica that is always available to you. Note that it is distinguished by the system variable icon ((SYS)). Although it is a system variable, the values of *Time* can be edited just like any other index.

Select the *Time* index and click the transfer button (>>) to select it as an active index in the right-hand window. Click **OK** to exit the Indexes window.

1. Create a new general variable titled *Hare population*

2. Open the Attribute (or Object) window, Select **Definition** in the Attribute popup window then select **Table** in the Expression popup menu

3. Select *Time* in the Indexes dialog box

4. Click the transfer button to move *Time* to the Selected Indexes window

Analytica will show an edit table for the Hare population variable. Notice that the default index values for *Time* are 0,1, and 2. You will need to have more than just three time points in your dynamic simulation.
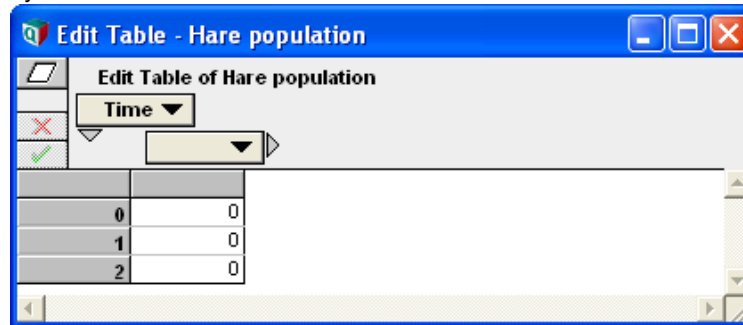
Go to the **Definition** menu and select **Edit time**. Click the Expression popup menu and select **Sequence**. Analytica will prompt to make sure you want to change the Time index to a Sequence. Click **OK**.

1. Select **Edit time** from the **Definition** menu.

2. Select **Sequence** from the Expression popup menu (just above the Definition field)
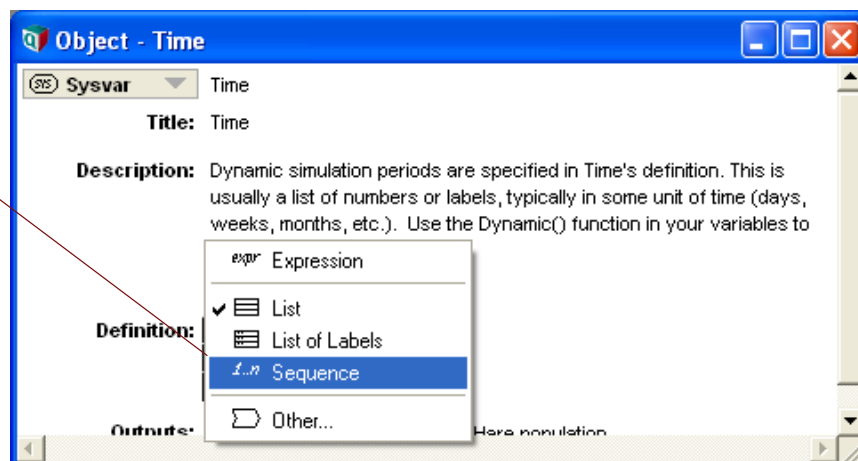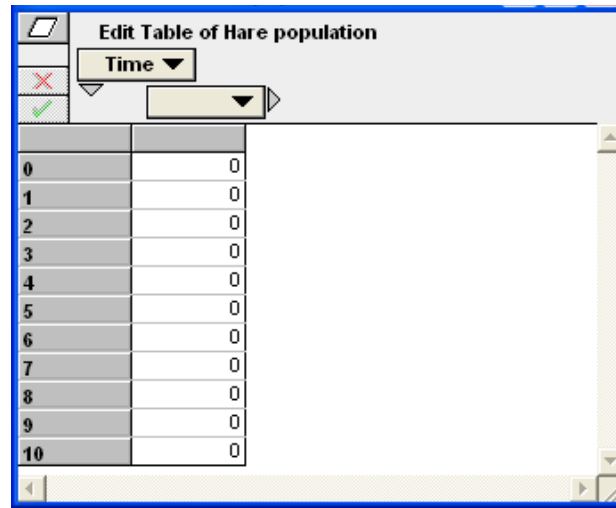
Edit the sequence by keeping 0 for the start value and entering 10 for the end value. Click **OK** to exit the Object finder window.

3. Set sequence to span from 0 to 10.
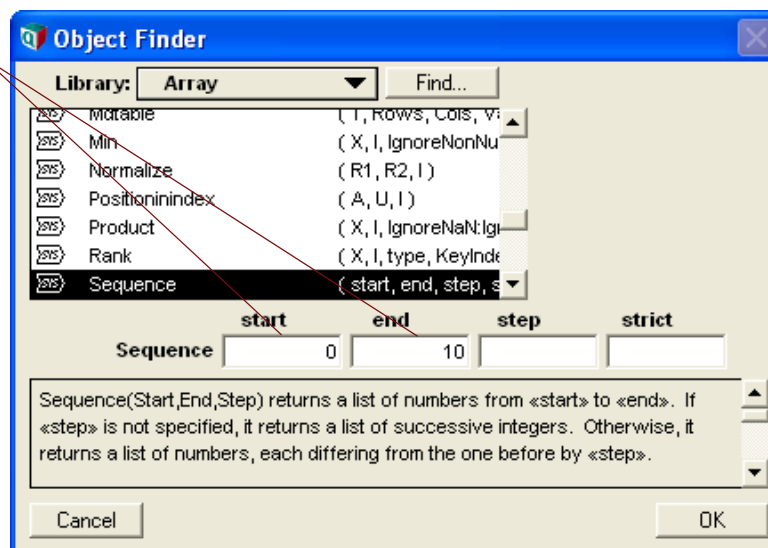
If the edit table for Hare population is not already open select the Hare population node and click on the Result button (🐞). Notice that our time index now goes from 0 to 10. We will leave the values of the Hare population zero the moment.

Edit Table of Hare population	
Time ▼	
▽	▼ ▷
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0

Setting up the Hare sub-model

Create three new general variables titled *Initial hare population*, *Hare fertility*, and *Hare births*. Draw influence arrows as follows:

- from *Hare fertility* to *Hare births*
- from *Initial hare population* to *Hare population*
- from *Hare population* to *Hare births*

Define *Initial hare population* as *500*. Change its Identifier to *Hare_zero*.

Define *Hare fertility* as *60%*. We will take this to mean that the number of newborn hares will be equal to 60% of the total population during the previous time period!

Define *Hare births* as *Hare_population * Hare_fertility*. This is the total number of births during a time period.

Create, define, and draw influence arrows for three new variables as instructed above.

Diagram - Foxes and Hares Tutorial

Initial hare population

Hare fertility

Hare population
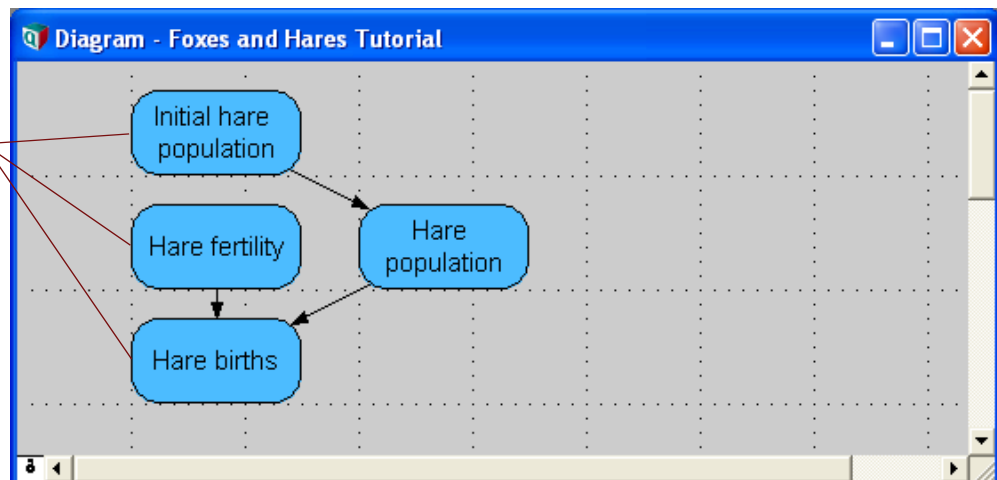
Hare births

Using the Dynamic() function

Now we consider how the hare population will develop over time. The hare population will be influenced by the number of hare births, which will be influenced by the hare population during the previous period. Ultimately the value of *Hare population* depends on the value of itself during the previous time period. We can address this situation using the **Dynamic()** function.

Using either the Attribute window or the Object window, open the Definition field of the *Hare population* variable. Click on the Expression popup menu and select ✔ *expr* **expression...** Analytica prompts to make sure you want to replace the current definition. Click **OK**.

Delete the existing table expression and replace it with the following:

Dynamic(hare_zero, self[time - 1] + hare_births[time - 1])

The first term of the Dynamic() expression specifies the initial value at the first *Time* value.	**self**, when used within a dynamic expression, refers to the same variable that is defined by the Dynamic expression. Circular references are avoided by specifying a *Time* value other than the current time step.	Square brackets placed after an array variable call a specific index value. In this case the index is *Time* and the specified index value is one less than the current time step.

The Dynamic() function can be entered directly as an expression or accessed as a built-in function (by selecting **Other...** in the expression popup menu). It is listed under the **Special** library heading.

Note that the Dynamic() function automatically incorporates the Time system variable. It was not necessary to initially define our variable as a table indexed by *Time*, although this step was included in the first section of this chapter to illustrate the *Time* index.

Select the *Hare population* node and click the Result button (🔳)
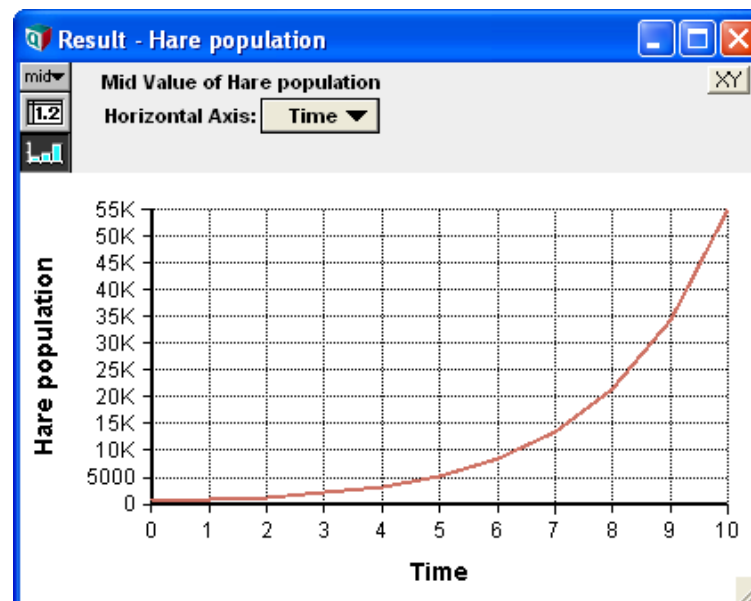
This is a simple exponential growth curve. It represents a rapidly multiplying population of immortal hares with no natural predators and an unlimited food supply. You will need to introduce a limiting factor to avoid this buried-in-bunnies scenario.

Completing the Hare sub-model

Quickly before it's too late, create four new general variables as follows:

- Title: *Hare carrying capacity*
  Identifier: *Hare_capacity*
  Definition: *1000*

- Title: *Hare natural death rate*
  Identifier: *Hare_ndr*
  Definition:
  *hare_fertility * hare_crowding*

- Title: *Hare crowding*
  Identifier: *Hare_crowding*
  Definition:
  hare_population / hare_capacity

- Title: *Hare deaths*
  Identifier: *hare_deaths*
  Definition:
  *Hare_population * hare_ndr*

Finally you need to change the definition of *Hare population* as follows:

```
Dynamic(hare_zero, self[time -1] + hare_births[time - 1]
- hare_deaths[time - 1])
```
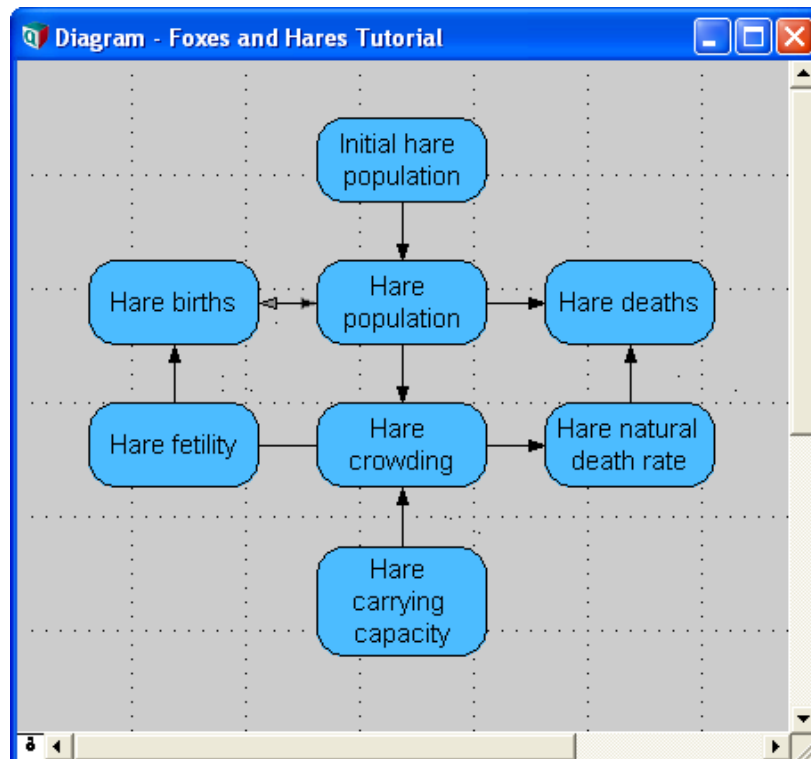

Now we have accounted for mortality and limited food supply by introducing a crowding factor. When the crowding factor is equal to one, the hare population has reached the carrying capacity for the habitat. At this point the death rate will be equal to the birth rate.

Select the Hare population node and click the Result button (🔲)

Introducing a crowding factor limits the hare population to the carrying capacity
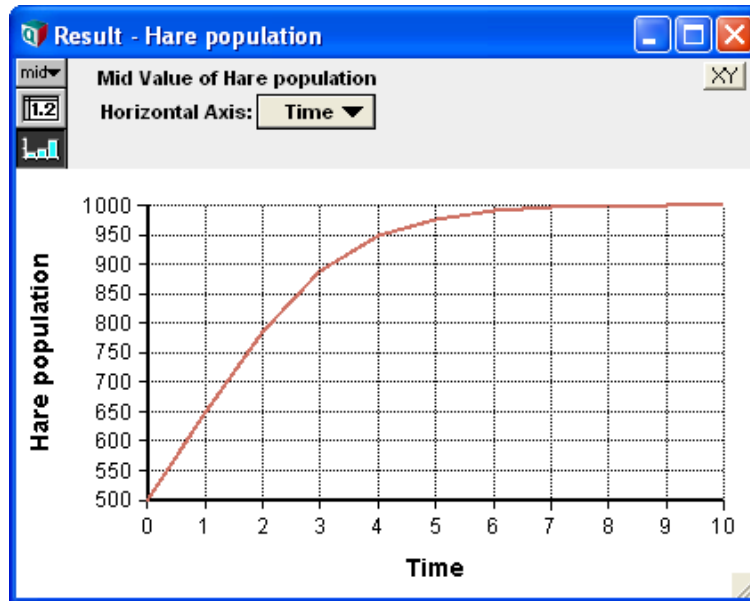
Using the color palette

In order to make an influence diagram easy to understand it is often useful to color code nodes according to function. The **color palette** is used for this purpose. You can access it in two ways:

- Right-click the node and select **Show/hide color palette** from the context menu
- Go to the **Diagram** menu and select **Show/hide color palette**

In the color palette you can select from standard colors or click **Other...** to select from the entire 24-bit RGB color space.

In this example we use color to categorize variables in the following way:

🟩	initial values
🟩	inflow/outflow
🟦	intermediate
🟨	stock

Use the color palette to code variables according to function as shown.
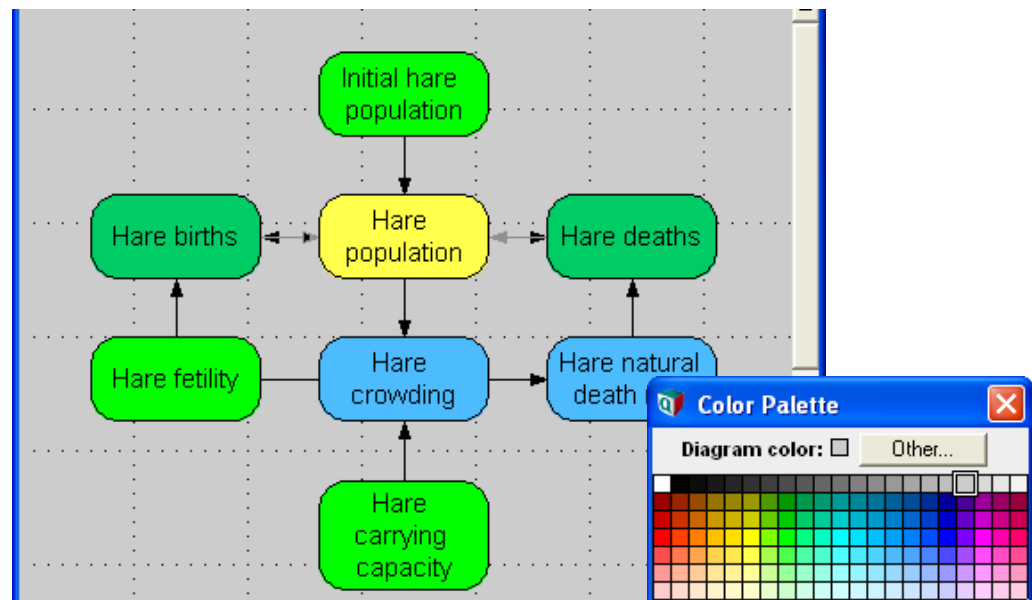
Tip The **node style** and **diagram style** windows are also useful for customizing the appearance of your diagram. They can be accessed in the same manner as the color palette.

Creating a module

Modules are an important feature of Analytica. They allow you to place a collection of nodes inside a single object, a module, which has the appearance of a single node. In this way, the top level of your diagram can be a simplified view showing influence arrows between modules. Each module can be opened as its own diagram window to show detailed interactions of the variables inside. Modules can even be placed within modules to create a nested hierarchy of detail.

Tip   If you have several diagram windows open at once you can click the Diagram window button (🖧) to return to the top level diagram at any time.

Creating a new module is just like creating a variable node. Drag the Module node from the node palette to the diagram window. A module looks similar to a node but it has a thick border around it and displays the title in bold print. Whereas double-clicking a variable node will open an Object window, double-clicking a module will open a new diagram window. Attributes for the module can be entered using the Attribute window at the bottom of the diagram. Your new module will have the following attributes:

- Title: ***Hare module***
  Identifier: ***Hare_module***
  Description: ***A dynamic sub-model of the hare population***

To populate your new module, start by making a multiple selection of all the nodes except for the module itself.

Reminder: To make a multiple selection hold down the Control or Shift keys, or drag a rectangular frame around the nodes you want to select.

Drag one of the selected nodes to the module and release.

1. Create a new module by dragging the Module icon down to the diagram space

2. Make a multiple selection of all variable nodes by dragging diagonally across the space. (Make sure that the module is not also selected)

3. Drag one of the selected nodes to the module and release
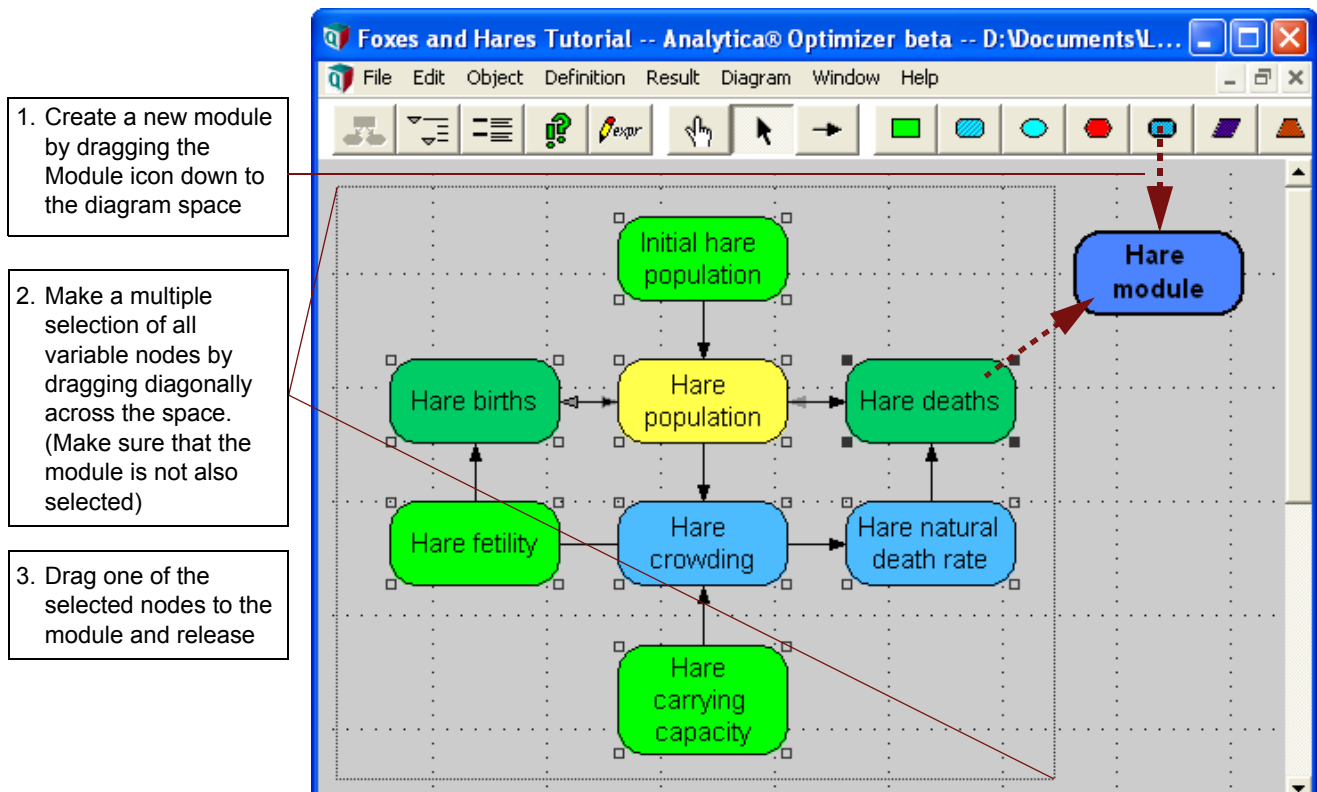
Your top level diagram should now contain just the module by itself. Double-click the module to see the variables in a separate window.

4. Double-click on the module to open a new diagram window
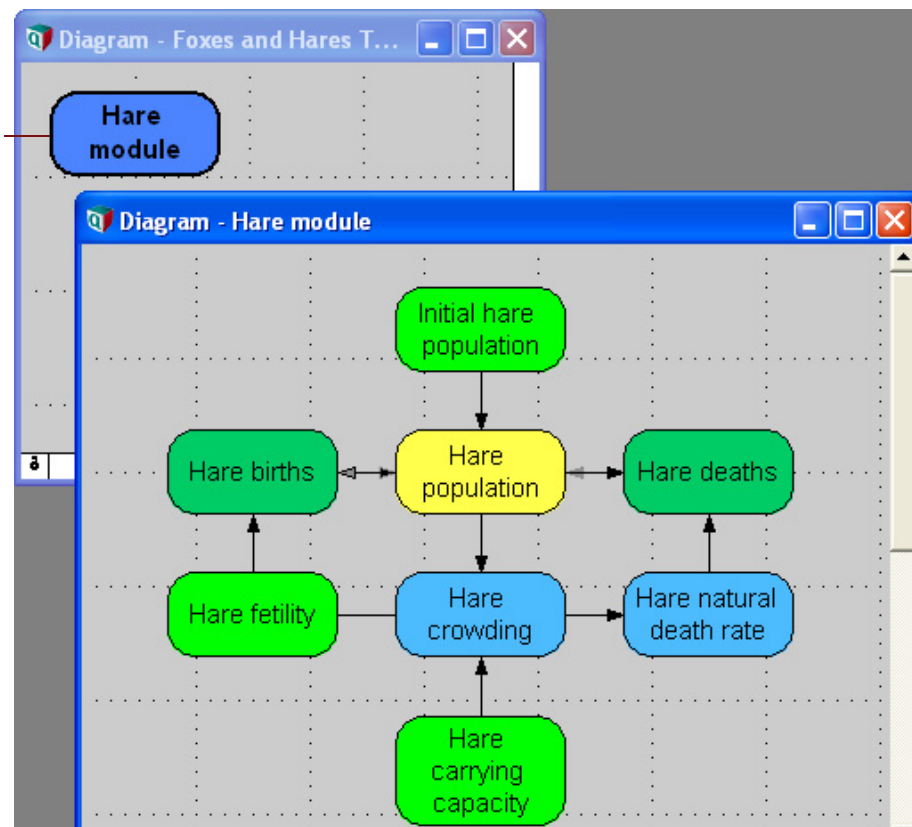
Tip If a single diagram window takes up the entire workspace press the maximize/un-maximize toggle (▣) near the upper right corner of the window.

Duplicating a module

Now you will make a similar sub-model for the fox population. Since the basic structure will be similar to the hare sub-model, you can save time by duplicating it automatically.

Select the hare module in the top level diagram window. Select **Duplicate nodes** from the **Edit** menu. A duplicate module will appear.

Edit the Title and Identifier of the duplicate to read *Fox module*

Edit the Description attribute to read *A dynamic sub-model of the Fox population*

1. After selecting the Hare module, go to the **Edit** menu and select **Duplicate node**

2. Re-title the duplicate as *Fox module*
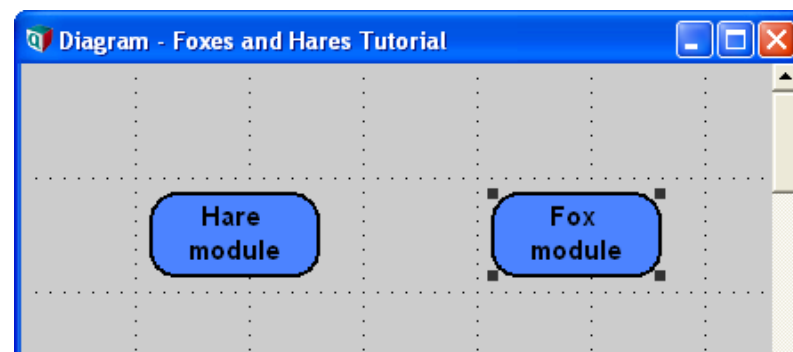
Double-click the *Fox module* to open its diagram window.

The diagram appears to be identical to the original but you will notice some important differences in the Identifiers. Open the Attribute window and check the identifier for the Hare population node. Notice it has been changed to **Hare_population1**. Analytica has appended the **1** to ensure that the identity of the new variable is distinct from the original. Different variables are allowed to have identical titles but they must always have different identifiers

Identifiers in functional expressions are modified accordingly. This ensures that the duplicate variables belong to a self-contained group that does not interfere with the original variables.

> Analytica appends a **1** to keep the identifiers of duplicated variables distinct

Hare population:	Identifier ▼
Hare_population1	

Completing the Fox sub-model

Edit the titles and identifiers in the fox module as follows:
(These edits mostly involve substituting the word Fox for Hare.)

- Title: *Fox population*
  Identifier: *Fox_population*

- Title: *Fox carrying capacity*
  Identifier: *Fox_capacity*

- Title: *Fox natural death rate*
  Identifier: *Fox_ndr*

- Title: *Fox births*
  Identifier: *Fox_births*

- Title: *Initial fox population*
  Identifier: *Fox_zero*

- Title: *Fox crowding*
  Identifier: *Fox_crowding*

- Title: *Fox deaths*
  Identifier: *Fox_deaths*

- Title: *Fox fertility*
  Identifier: *Fox_fertility*

Change the definition of input variables as follows:
- Initial Fox population: *120*
- Fox fertility rate: *35%*
- Fox carrying capacity: *200*

The fox population is now a self-contained sub-model.

Select the *Fox population* variable and click on the Result button (🖼).

> Due to a lower fertility rate, the fox population approaches its carrying capacity more slowly than the hare population.
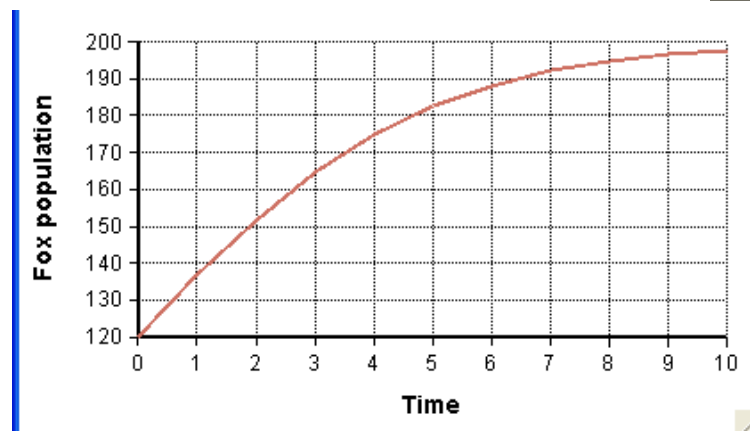
Creating aliases

Although it is useful to organize a model with modular structure, accessing variables inside modules can be less convenient than having them all in the same window. To address this problem Analytica allows you make an *alias* of a variable and place it anywhere you wish. The alias provides local access to the variable but it is merely a representation, not a duplicate, of the original node.

In this case it will be convenient to access the *Hare population* and *Fox population* nodes on the parent diagram without having to open the modules.

Click the Diagram button (🖧) to open the top level diagram. Open the *Fox module* and arrange the windows so that both are visible.

Select the *Fox population* variable. Go to the **Object** window and select **Make alias**. The alias looks similar to the original node but the title will be shown in italics. Drag the *Fox population* alias to the parent diagram.

1. Arrange windows so that the *Fox module* diagram and its parent diagram are both visible.

2. Select the *Fox population* variable and select **Make alias** from the **Object** window.

3. Drag the *Fox population alias* to the parent diagram window.

Note that Alias nodes always show titles in italics.
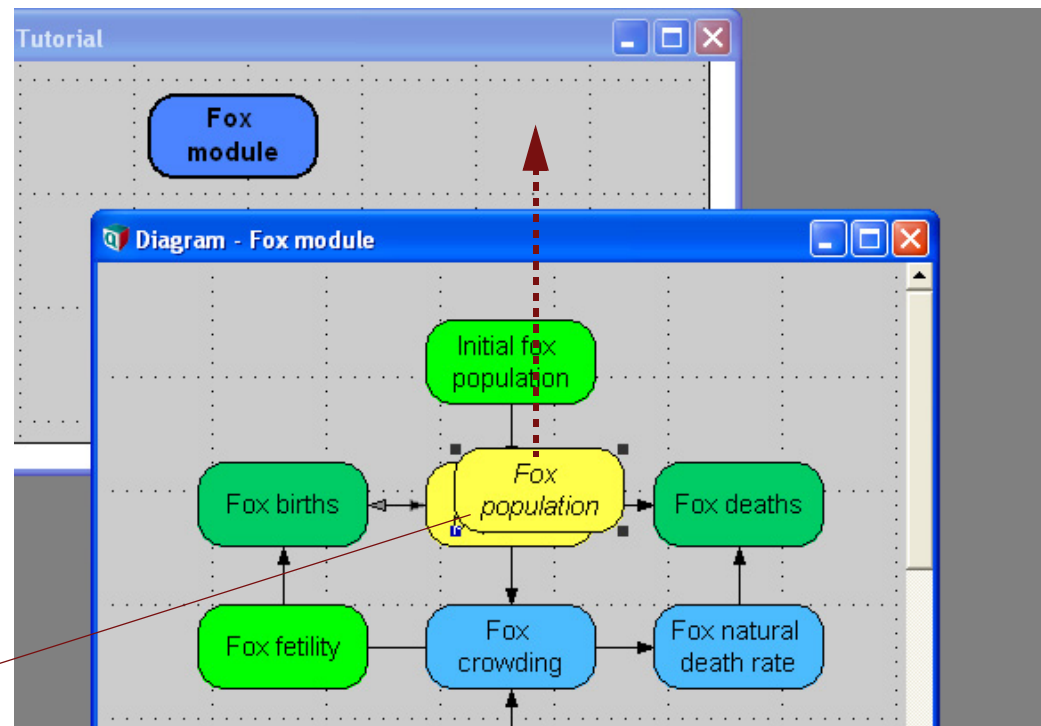
Repeat the same procedure to create a *Hare population* alias on the top diagram.

4. Create an alias for the Hare population variable in the same manner.

Influence arrows are automatically drawn
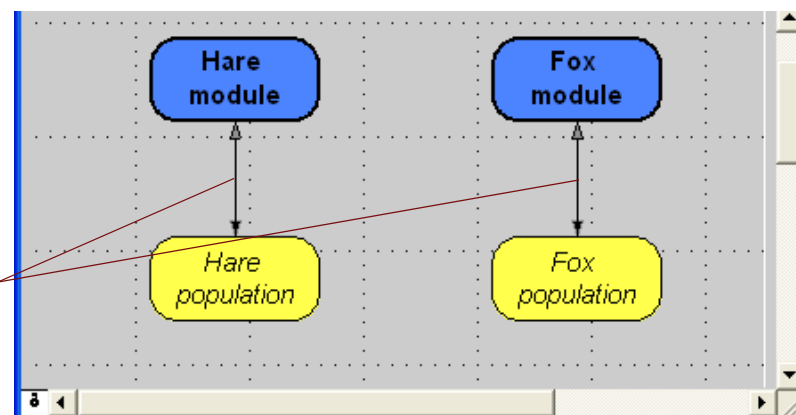
Drawing influence arrows across modules

Now you are ready to introduce predatory interaction! Create a new general variable titled *Predations*. This will represent the number of hares eaten by foxes in a given time period. The predation level is assumed to be proportional to the product of fox and hare populations.

Draw influence arrows from the *Hare population* alias to *Predations*, and from the *Fox population* alias to *Predations*. Enter the following expression to define the new variable: *Hare_population * Fox_population / 600*

1. Create a new general variable titled *Predations*.

2. Draw influence arrows from the *Hare population* and *Fox population* aliases to the *Predations* variable.

3 Define *Predations* as: *Hare_population *Fox_population/600*
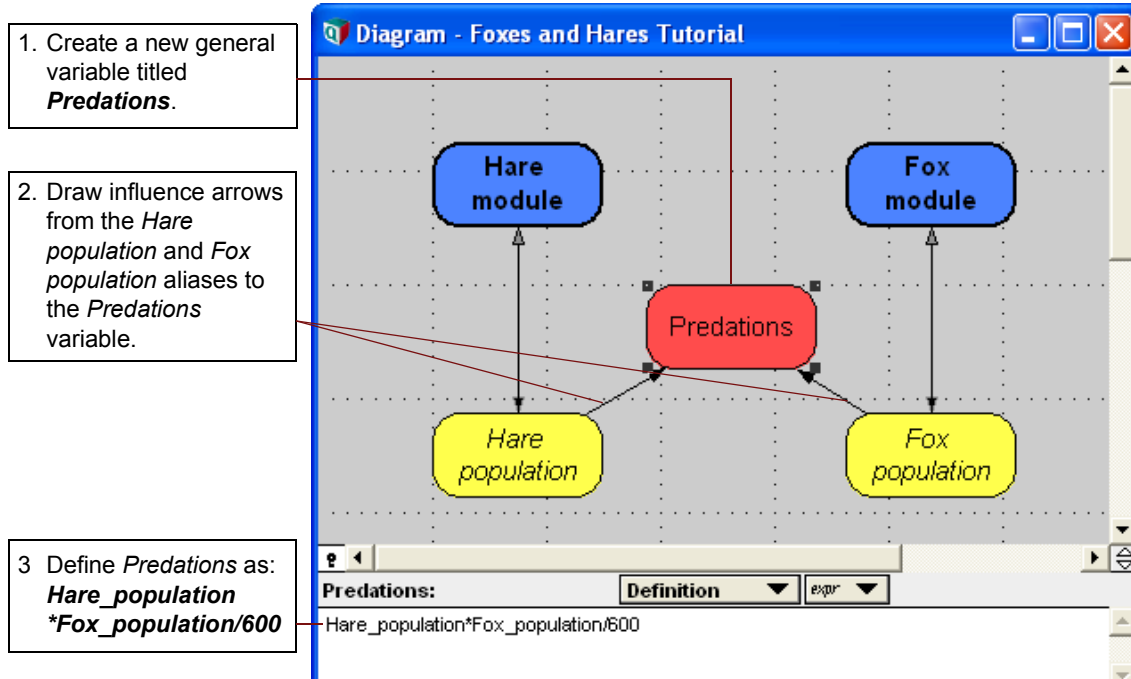
Unfortunately for the hares, each predation will count as a death for their team. Open the *Hare module* and arrange windows so that the *Hare module* diagram and its parent diagram are both visible. Draw an influence arrow from *Predations* to *Hare deaths*.

4. Open the *Hare module* and arrange window so that the *Hare module* diagram and its parent diagram are both visible.

5 Draw an influence arrow from the *Predations* variable to the *Hare deaths* variable.
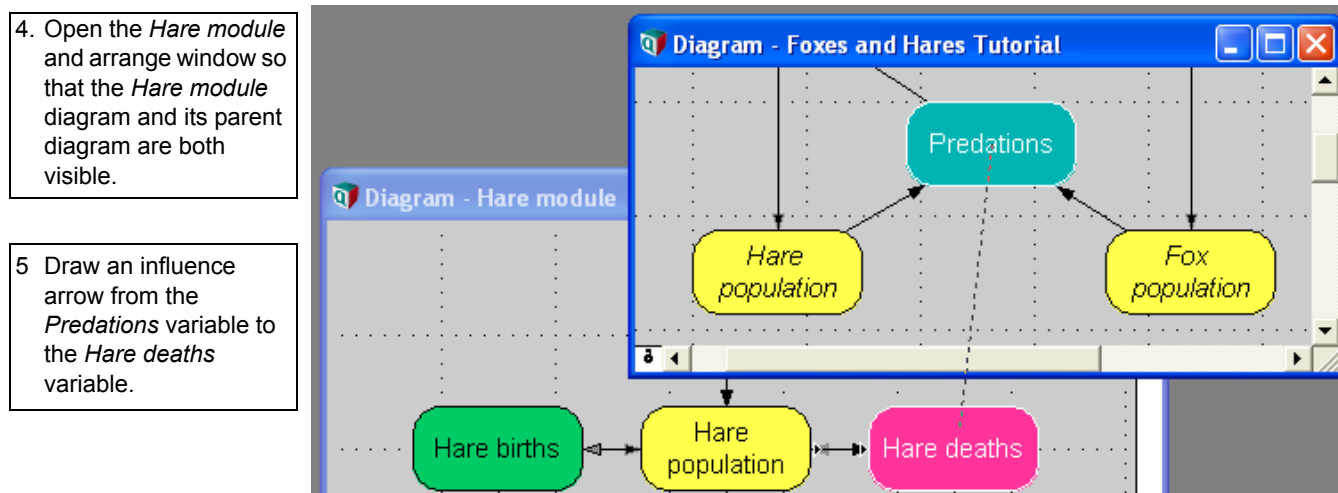
Re-define Hare deaths as: `Hare_population * Hare_ndr` **+ Predations**

6. Edit definition of Hare
deaths to read:
*Hare_population ***
Hare_ndr +
Predations

| Hare deaths: | Definition ▼ | expr ▼ | ✕ ✓ | Inputs ▼ |

Hare_population*Hare_rip + Predations|

if you look closely at the *Hare population* and *Hare deaths* nodes you will notice small arrowheads on the margins. These indicate influence relationships with other nodes outside the module.

Arrowheads indicate
influences directed
outside the module.
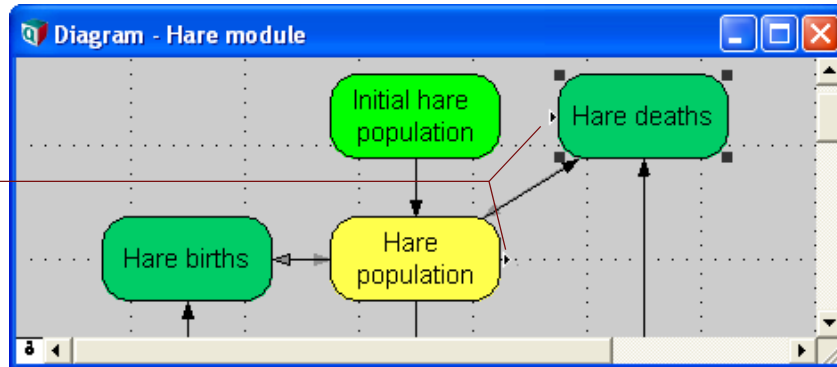
Predations are a food source for fox population and have a positive effect on their numbers. This can be modeled by making the *Fox carrying capacity* proportional to the *Predations* variable.

Re-define *Fox carrying capacity* as: **Predations*1.6**

7. Re-define Fox carrying
capacity directly by
typing
*Predations * 1.6*
into the Definition field

Tip Analytica will allow you to use a variable in an expression before drawing its influence arrow. The variable will not be listed in the Inputs popup menu so you must type the exact identity correctly. Influence arrows will automatically be drawn after you finish editing the expression.

Viewing multiple results simultaneously

Your model is now complete. Click the Diagram button (🔲) to open the top diagram. As you already know, you can view the results of the fox and hare populations separately by selecting either one of the population aliases and clicking the Results button (🔳). But there is an easy way to combine both variables on the same graph.

Select both alias nodes simultaneously (holding down the *Control* or *Shift* keys while selecting). Now when you click the Results button a dialog box will appear asking if you want to compare more than one result. Click **OK**. Analytica will automatically create a *comparison node*. A comparison node is essentially a list variable with the variables to be compared as elements of the list.

The default title in this case will be *Compare1*.

1. Select the *Hare population* and *Fox population* aliases simultaneously.

2. Click the Results button (🔳) in the tools palette.

3. Click **OK** to confirm you want to compare more than one result.

A new comparison node is automatically generated.
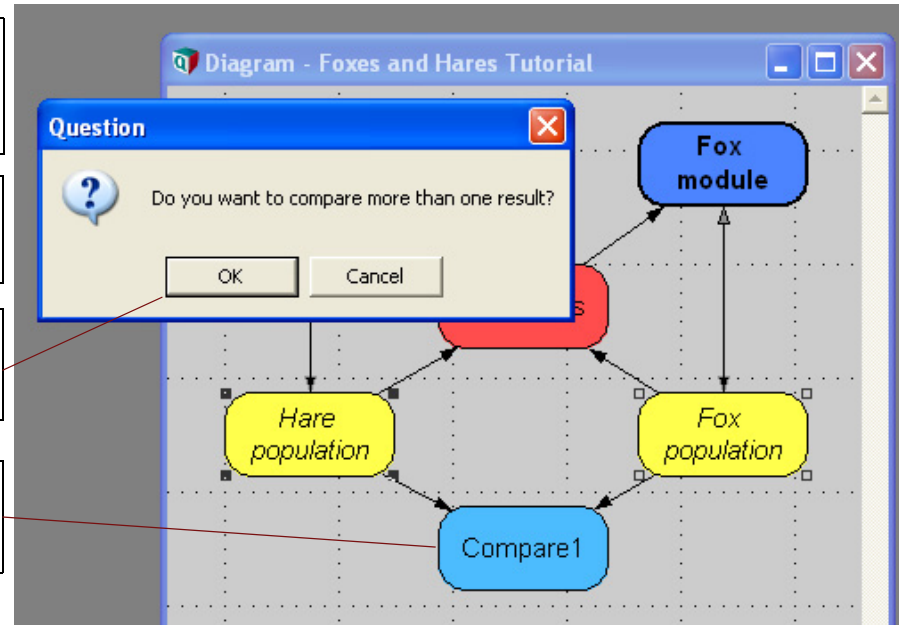
The result window for *Compare1* will contain information from both of the selected nodes. Choose Time as the horizontal axis in the Result window of Compare1. You can now see the result of the predatory interaction.
I

4. Set **Time** as the horizontal axis

The graph shows results for Fox and Hare populations simultaneously
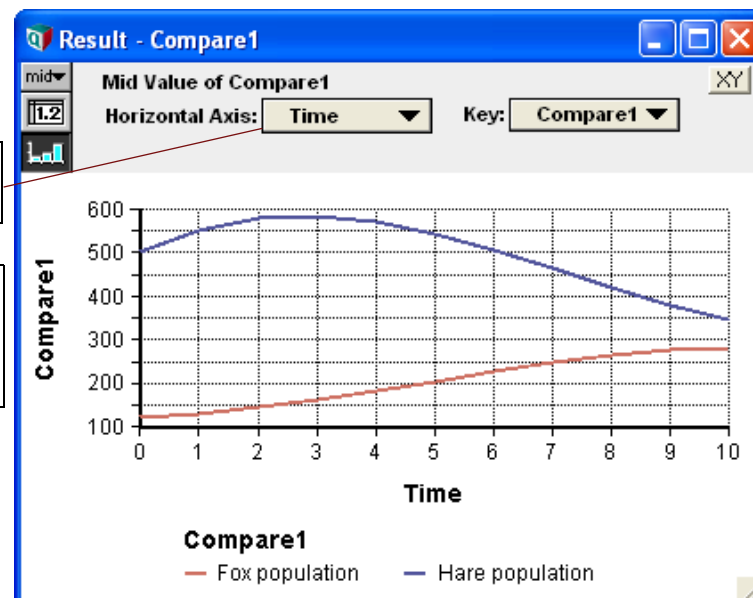
Foxes and Hares Act III

How does the Fox/Hare drama end? You can extend the simulation by simply adding more values to the *Time* index.

Go to the **Definition** menu and select **Edit time**. Extend the sequence to go from *0* to *100*. Click **OK** to exit the Object Finder.

1. Select **Edit time** from the **Definition** menu.

2. Click the **Sequence** button in the Definition field.

3. Set end step to *100*
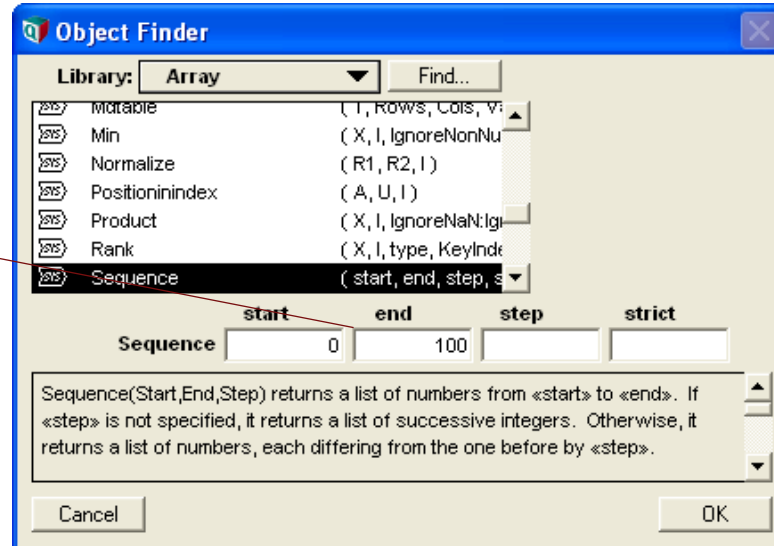
Select the *Compare1* node and click the Results button ( ) again.

4. Select the Compare1 node and click the Results button ( )

The populations eventually settle down to a new equilibrium
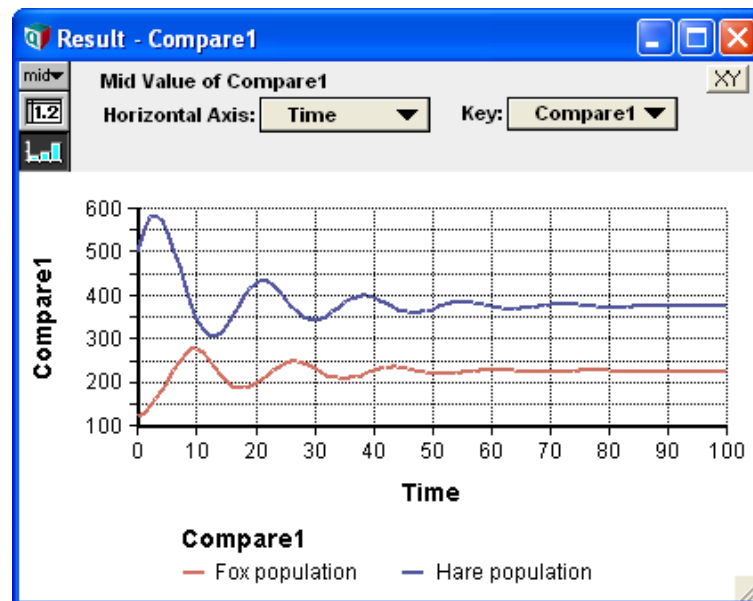
Now you can experiment by changing input values to see how they impact the result.

If you would like to see a more complete Foxes and Hares example, open **Foxes and Hares.ana** in the **Tutorial Models** folder.

Summary: Creating the Foxes and Hares Model

In this chapter you have:

- Used the Dynamic() function to model a feedback loop in a dynamic system.
- Edited the system variable *Time* which is used as an index in the Dynamic() function
- Customized the appearance of nodes using the Color palette
- Grouped nodes into Modules
- Used the Duplicate command to create a new module with similar structure to the original
- Created aliases for convenient access to nodes contained within modules
- Established influence relationships between modes in different diagram windows
- Created a comparison node to view the results of two variables simultaneously.

Chapter 8   *On Your Own*

This chapter describes the example models and libraries that are provided with Analytica.

Chapter **8** *On Your Own*
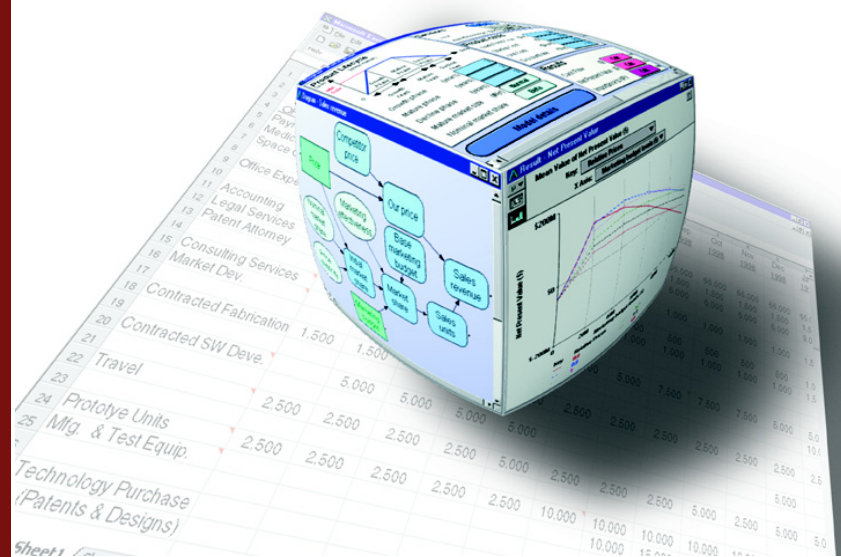
Congratulations on completing the Analytica Tutorial. You are now ready to begin creating your own models.

For more detailed information on Analytica, see the *Analytica User Guide*. It is a reference on all aspects of Analytica, including descriptions of all available functions.

Within the Analytica folder are folders titled **Example Models** and **Libraries**, which contain a variety of Analytica models, including the examples illustrated in the *Analytica User Guide*. These resources are useful to include when building your own models. Many of the example models were created by users just like you. These models contain a wealth of ideas on using Analytica for practical applications. You should investigate these examples to see some of the different ways in which models can be constructed.

If you create models that you feel would be helpful or interesting to others, please send them to us for inclusion in a future Example Models folder; see the end of this chapter.

The *Example Models* folder is subdivided into these folders:

- Business Examples
- Data Analysis
- Decision Analysis
- Dynamic Models
- Engineering
- Function Examples
- Optimizer Examples
- Risk Analysis
- User Guide Examples

Business Examples

Bond Model	This model takes typical bond purchase inputs (purchase price, par value, interest rate, and life to maturity) and calculates bond cash flows, current yield, and yield to maturity.
Breakeven Analysis	This model is an example of a breakeven analysis of a set of revenue levels, when the fixed expenses are set at one amount and the variable expenses are a constant fraction of revenue.
Expected R&D Project Value	This model evaluates and compares the expected commercialization value of multiple proposed R&D projects.
Financial Statement Templates	This model contains a complete set of standard financial statements: a profit and loss statement, balance sheet, and cash flow statement. It provides a step-by-step guide to using these templates to generate your own financial statements. You can enter values into the existing template and modify the variable definitions to reflect your own accounting standards.
Market Model	This model explores a market for a new product, and the pricing and advertising budget decisions involved. This example also shows the use of "forms" for receiving input and presenting output for users of the model.
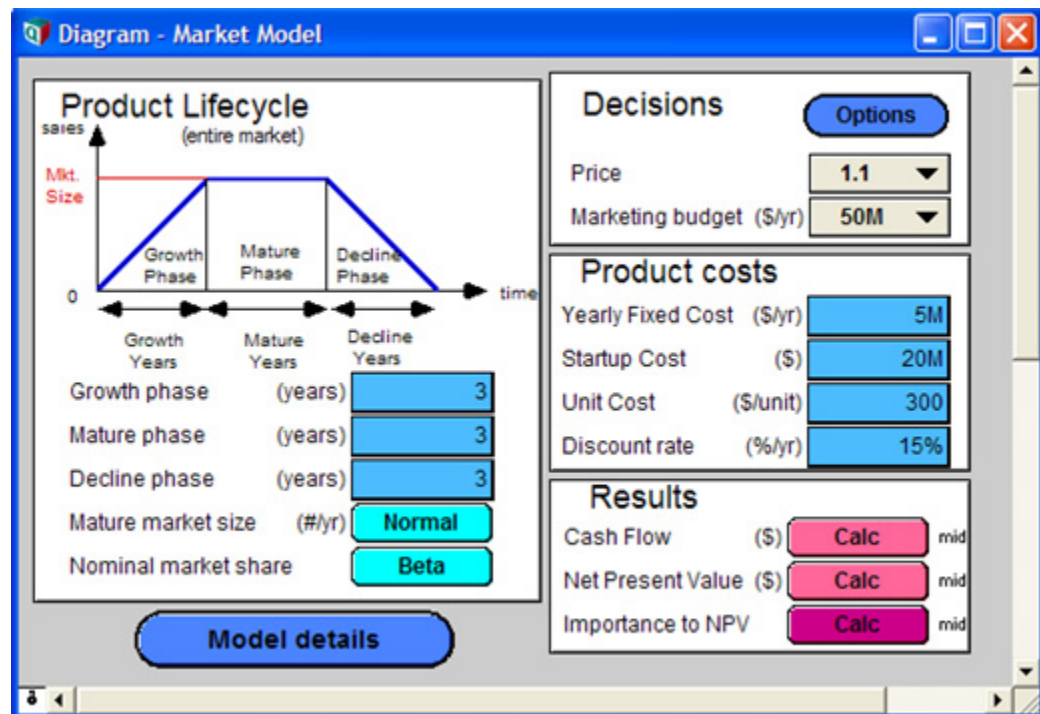
Plan_Schedule_Control	This model takes input data for activity paths required to complete a project, and calculates various outputs describing the critical path, timing, and costs for project completion.
Project Portfolio Planner	This model evaluates and prioritizes a portfolio of projects based on either the estimated net present value or a multi-attribute score, based on strategy fit, staff development, the generation of public goodwill, and estimated net revenue.
Sales Effectiveness	This model evaluates the effects of unit price on salesmen head count and production capacity. The model contains an example of taking user estimates of uncertainty in a standard high-medium-low form, and transforming those inputs into a continuous distribution for propagation through the model.
	Derived from *Principles of Systems* by Jay W. Forrester, 1968, ISBN 0-915299-87-9.
Subscriber Pricing	This model determines the amount of revenue needed on a monthly basis from each subscriber of a service to just meet the weighted average cost of capital of the firm from the service release date to the end of the study horizon. In other words, it calculates the monthly unit revenue rate

required from each subscriber of a service to give a return on investment at the end of the study horizon that is equal to the weighted average cost of capital of the firm.

Data Analysis

Kmeans Clustering

This model shows an example of scatter plots in Analytica. A **K**-means clustering algorithm (where **K** is the number of clusters) is applied to some random data to partition points into groups (clusters) of similar points.

This model also demonstrates the *Iterate* function.

Moving Average Example

This is a simple model that shows how to compute the moving average for a data stream. It defines a **Moving Average** function you can use.

Multidimensional Scaling

This model performs multidimensional scaling. It takes as input **N**, which is the dimensionality of the problem, and **Distances**, which is an **NxN** symmetric matrix of distances (or dissimilarities). It calculates and outputs a two-dimensional set of **N** points **XY** (or separately as **Xcoord** and **Ycoord**) that best approximates the spatial layout of points that could generate the input distances.

Reference: *Multivariate Analysis* by K.V. Mardia, J.T. Kent, and J.M. Bibby, Academic Press, London, 1979, Section 14.2.2, page 400. Model supplied by Michael L. Thompson.

Principle Components

Principal components analysis (PCA) is a technique used to reduce multidimensional data sets to lower dimensions for analysis. PCA involves computing the eigenvalue decomposition or singular value decomposition of a data set.

This model shows how to find the principle components in a uses an eigenvalue decomposition to compute the principle components of the covariance matrix of historical stock prices.

Regression Examples

This model demonstrates the use of generalized linear regression by best fit curves of various function forms to a set of (x,y) points. It includes:

- Linear regression
- Quadratic regression
- Polynomial regression
- Discrete Fourier series
- Regression with redundant basis
- Regression using a large arbitrary collection of terms (useful in the situation where you do not have any reason to prefer one functional form over another)
- An auto-regressive series

Decision Analysis

This folder includes models that illustrate the discipline of decision analysis.

Two Branch Party Tree The author of this model wants to throw a party, and can't decide where to throw it. This model shows how to model a two-branch decision tree in Analytica.
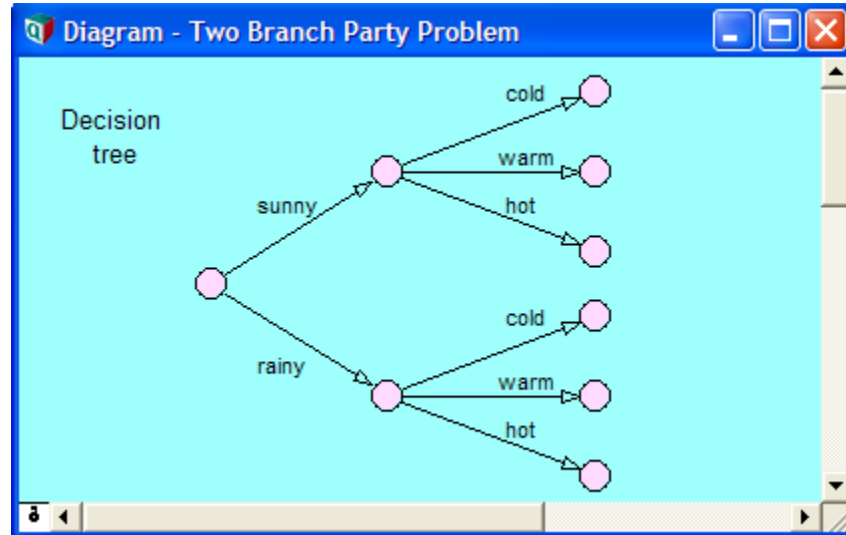
Beta Updating This model uses the beta distribution for the Bayesian update of beliefs about the probability that a coin will come up heads.

Biotech R&D Portfolio This multi-project R&D evaluation models a typical R&D decision problem that might be faced by a biogenetic company.

Diversification Illustration This model is an example of a Blitzogram™, which is one way to display the effect of diversifying over a growing set of investments. The example from the model is taken from "Blizograms - Inter- active Histograms" in *Informs Transactions on Education*, Vol. 1., No. 2 (Jan. 2001) by Sam Sav- age. For further information, consult:

- "Beat The Odds: Understand Uncertainty" at
  http://www.optimizemag.com/issue/002/financial.htm
- Sam Savage's web site: http://drsamsavage.com

Gibbs Sampling in Bayesian Network This model solves a Bayesian network using the Gibbs sampling method, also referred to as Sto- chastic Simulation. It is an instance of Markov Chain Monte Carlo simulation. This implementation runs multiple simulations simultaneously. You can specify observations for any subset of vari- ables in the model (using the pull-down menus), and compute the posterior probabilities for any of the other variables.

For more on this technique, see S. L. Lauritzen and D. J. Spiegelhalter, "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems," Journal Royal Statistical Society Series B50:2, 1984, p. 157-224.

LEV R&D Strategy This example models R&D decision analysis for investment strategy among several choices of powerplants for a low emissions vehicle (LEV).

Multi-attribute Utility Analysis This model is an example of a multi-attribute utility analysis for cars, showing how to analyze an array of cars across an array of attributes, where different drivers assign differing weights to the importance of each attribute.

Newton-Raphson Method This model implements the Newton-Raphson (or simply Newton's) method, one of the most pow- erful and well-known numerical methods for finding the root of f(x)=0.

Nonsymmetric Tree This model uses decision tree terminology to provide an example asymmetric decision tree in Analytica.

Party With Forecast   This model presents a problem facing a party host. In the face of uncertain weather, what is the best location to hold a party? The value the host assigns to the party is a function of both the location chosen and the weather outcome.

This model augments the basic party model in order to show the value of imperfect information — in this case, a weather forecast — using Bayesian updating.

Supply and Demand   This model calculates the required supply level to maximize profit when the profit function is asymmetric around the average demand value.

Tornado Diagrams   A tornado diagram is a common tool used to depict the sensitivity of a result to changes in selected variables. The fundamental analysis behind a tornado diagram consists of varying only one input variable at a time, keeping all other variables at their nominal values. Typically, a low and a high value are selected for each input, and the output variable is computed while only one variable varies at a time. This example model shows two methods for selecting high and low values: 1. By varying all inputs by the same relative amount, e.g., low=90% of nominal, high=110% of nominal, or 2. By varying all inputs between two given fractiles. This only makes sense if your inputs are uncertain variables. Example: low=10% fractile, high=90% fractile, nominal=50% fractile.
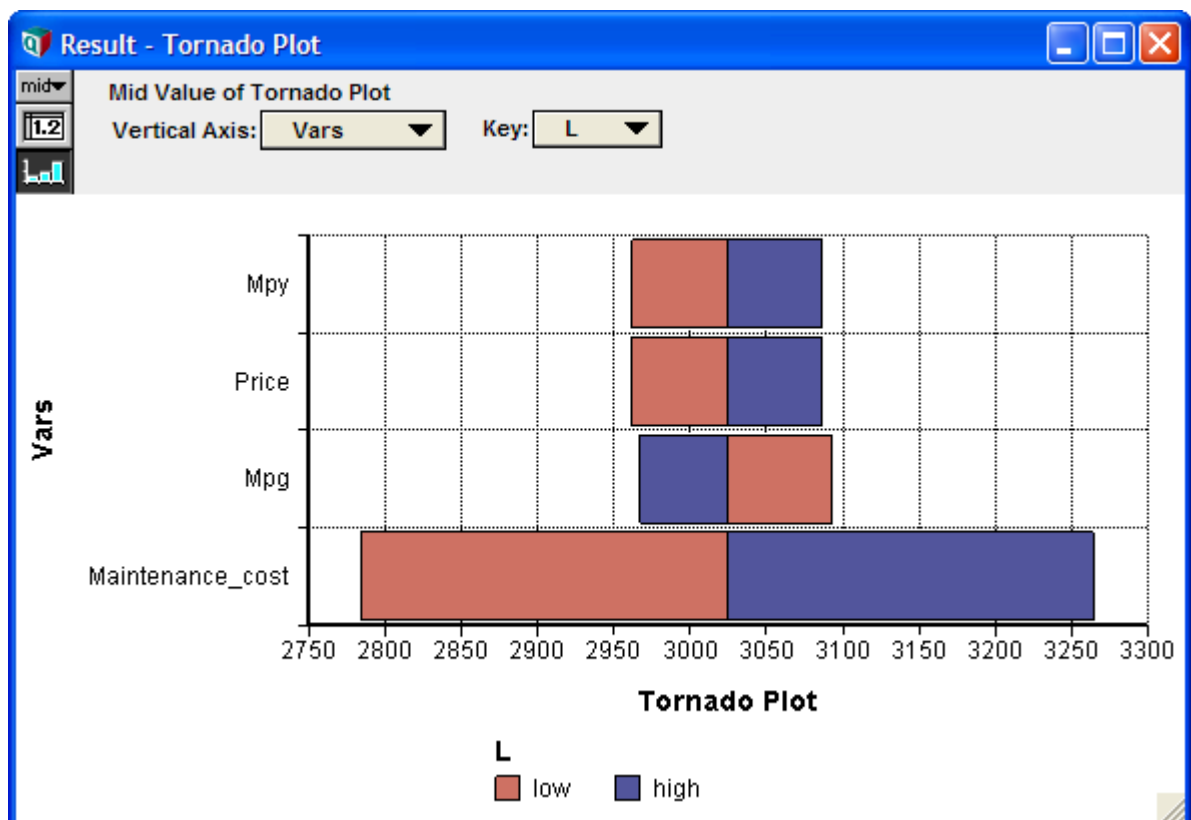
Upgrade Decision   This model represents a decision often faced in today's world: which technology to purchase now, in the face of uncertain future products and prices.
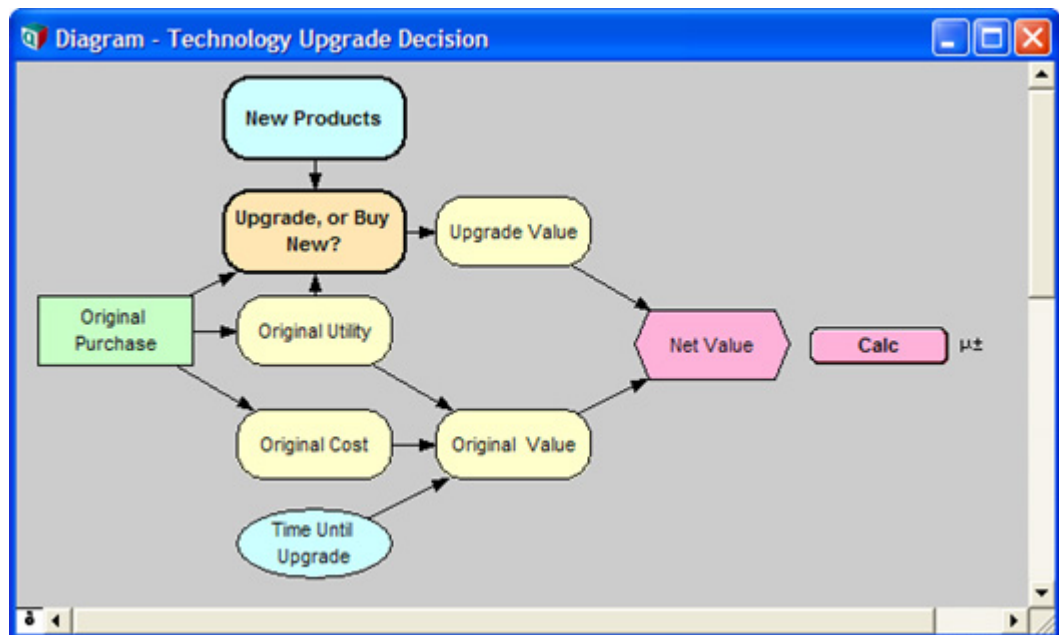
Dynamic Models

This folder includes models that use Analytica's **Dynamic()** function for performing dynamic simulation (modeling with cyclic dependencies).

Leveling   This example levels staff efforts over time according to staff available, computing both the work done over time and idle time.

Markov Chain   This model demonstrates how to simulate a Markov process using dynamic time. The example estimates the number of hospital patients over time, modeled as a Markov process.

Mass-Spring-Damper   This model simulates a typical free mass-spring-damper system. The term "free system" means that there is no time-dependent driving force or displacement acting on the mass. Ordinarily solutions to such a system are determined from a set of homogeneous second-order differential equations accompanied by the appropriate initial conditions. In this model, the kinematic variables (displacement, velocity, and acceleration) are related to the typical kinematic equations, and the dynamic variables (spring force and damper force) are related to the acceleration and the system mass by Newton's second law. You input the various initial state conditions (spring constant, damper constant, mass, initial displacement, and initial velocity) and the run time of the model. The graphical solutions generated by this dynamic model are comparable to the solutions determined by the corresponding differential equations.

Projectile Motion   An example demonstrating how to use the system variable `Time` and the **Dynamic()** function to model time-variant behavior — in this case, the motion of a projectile.
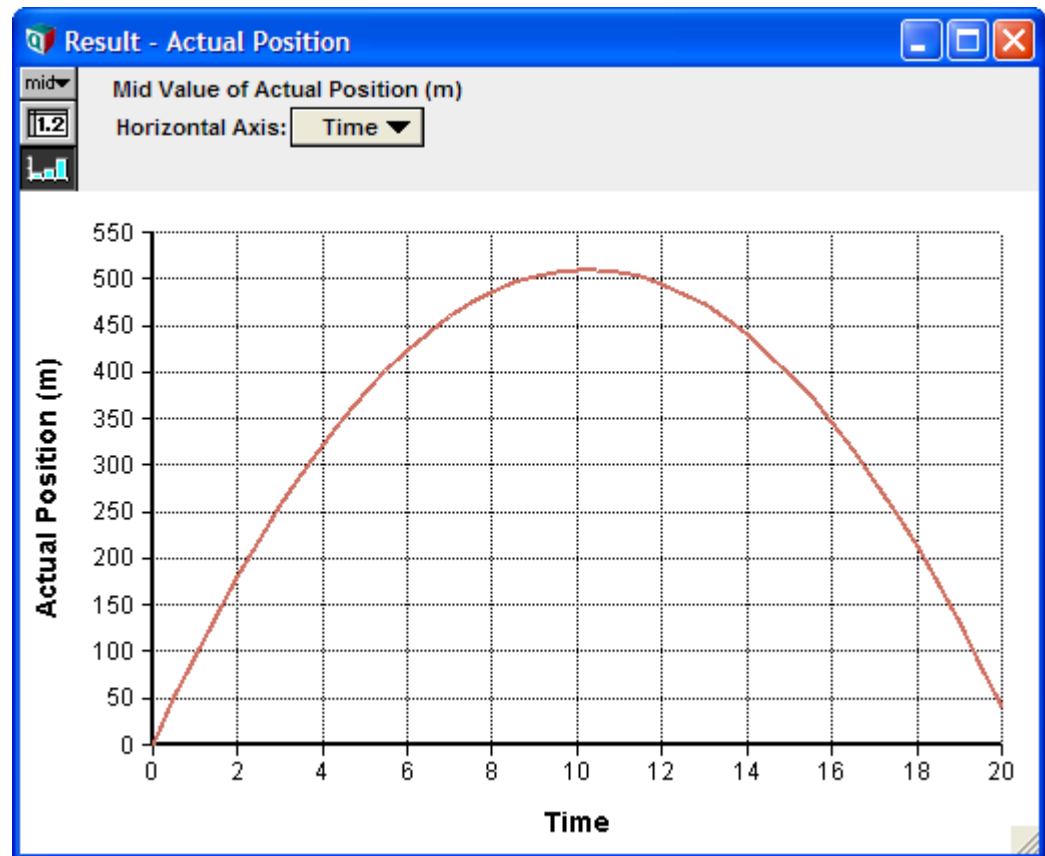
Unequal time steps   This model gives an example of a dynamic variable that calculates growth over time, where *Time* is defined with unequal time steps. It is an example of exponential or linear growth or decay — that is, a dynamic variable whose values in one time period are exponentially or linearly dependent on the values in the previous time period.

Engineering Examples

Adaptive Filter   This model curve fits noisy time-sequence data using an adaptive filter.

Antenna Gain   This model calculates the expected gain of an antenna looking at two different satellites.

Failure Analysis   This model provides a system simulation demonstrating a failure analysis with both parallel (bulbs) and series (bulbs and switch) components. The model shows the use of a **Determtable** instead of nested `if...then` statements to assess the state of the system. Both the switch and the bulbs use an exponential function to assess the probability of failure.

Function Examples

This folder contains examples illustrating a variety of Analytica's functions and modeling techniques.

Assignment from Button   This model demonstrates how you can use a button to copy a computed result from one node into the edit table in another node. Analytica Enterprise users can create buttons by dragging a button node from the tool bar onto the diagram.

Autocorrelation   This model calculates the auto-correlation coefficients of noisy time sequence data.

Choice and Determtables	This model shows that when `Choice` nodes are indexed by "self," you can use **Determtable** functions to propagate the selected choice. This is cleaner than some other methods of using `Choice` outputs.
Correlated Distributions	This model reorders a group of probabilistic variables' samples so that they mimic a desired correlation structure as closely as possible.
	For more information on this method see R.L. Iman and W.J. Conover, "A distribution free approach to inducing rank correlation among input variables," Commun. Statist.-Simula. Computa. (Marcel Dekker, Inc.), 11(3), 1982, 311-334.
Correlated Normals	This model demonstrates a method for creating two normal distributions with a specified correlation between them. The two resulting unit normals can be transformed to have any mean and standard deviation. From E.M. Scheuer and D.S. Stoller, "On the generation of Normal Random Vectors," *Technometrics*, 4:278-281, 1962.
DBWrite Example	This model demonstrates how you can write data from an Analytica model to a relational database using ODBC. This model requires Analytica Enterprise; refer to Chapter 22 of the *Analytica User Guide*.
Discrete Sampling	This model demonstrates how to generate a distribution from a discrete sample of numbers.
Extracting Diagonal	This model demonstrates how to extract a diagonal from a matrix.
Lookup Reindexing	This model demonstrates a simple re-indexing operation, essentially how to look up a value from another table. This is shown by the *Salary by person* node, and demonstrates how Excel's **VLOOKUP** function is performed in Analytica expressions.
Sample Size Input Node	On occasion, you might want to provide an input node on your form for the sample size system variable, so that a user can adjust the number of samples directly from your form, rather than having to bring up the **Uncertainty Options** dialog. Because you cannot select the `sampleSize` system variable, it is not possible to do this from the Analytica menus. This module provides a way to create this input node — just select **Add Module** and then **Embed**, and you can drag the input node in the module to the form where you would like it to appear. After you do that, you can delete this module, which will then be empty.
Sorting People by Height	This example sorts an index (*People*) by a table of values (*Heights*), and then uses the sorted index to created a sorted table of values (*Sorted heights*).
Subset of Array	This model creates a subset array out of a larger array based on a decision criterion.
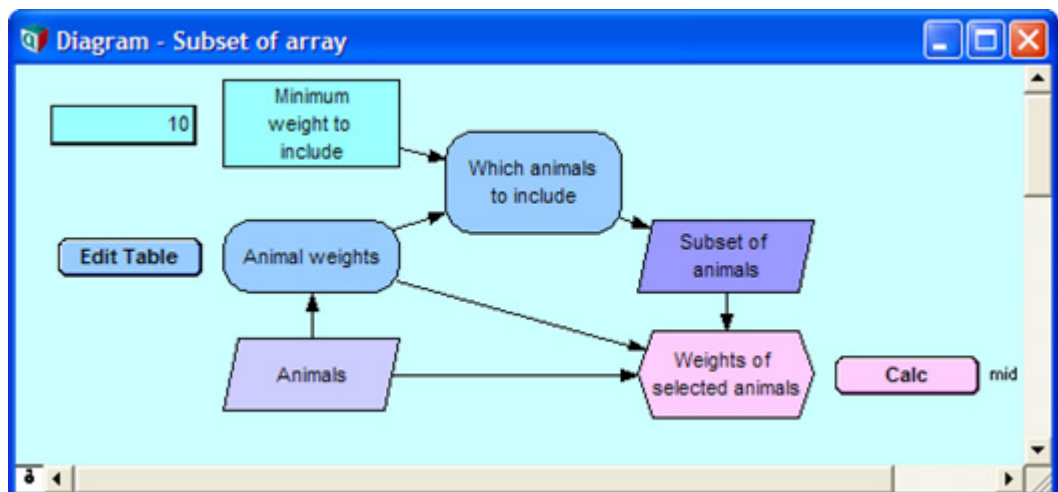
Swapping y and x-index	This model swaps a computed or one-dimensional table value with its index, thereby making the computed value an index.
Use of MDTable	This model demonstrates the use of the **MDTable** function, which converts records in table form into multi-dimensional arrays. Multi-dimensional arrays are often useful for visualizing large sets

of records. By allowing data to be viewed either as a graph or in a pivot table, the geometric relationship between records often becomes immediately evident.

Optimizer Examples

This folder contains examples of how to use Analytica with Optimizer. These models are fully functional only if you have purchased the Optimizer license along with Analytica.

Airline NLP   This model gives examples of nonlinear programming optimization and Intelligent Arrays. The examples are:

- Simple airline decision problem to select the number of planes and fares to maximize profit; parametric analysis with respect to demand.
- The same problem with uncertainty, to maximize expected profit.
- The same problem with uncertainty, to maximize profit, given value of uncertain variables
- A dynamic model optimization over multiple years
- A dynamic analysis with optimization in each year

Asset allocation   Given many possible investments with varying risk-versus-return trade-offs, one can often reduce risk through diversification. In the best case, investing in two assets with identical expected appreciation, **r**, but which are perfectly anti-correlated in their co-variation yields the expected rate of return with no risk. The more general problem is to select a portfolio that both maximizes return and minimizes risk. There are several possible formulations for this, and this model explores three:

1.   Minimize variance (risk) subject to a given expected return.
2.   Maximize return subject to a given variance (risk)
3.   Balance risk and return by maximizing expected utility, given a person's level of risk aversion.

Automobile Production   This model is a linear programming example, taken from "Quick Review of Linear Programming," *Management Science Techniques for Consultants,* by M.A. Trick (1996).

Its objectives show all the varies results from a linear program:

- The optimal solution
- The value of the objective function at the optimum
- The solution status
- The reduced costs (dual values for the variables) at the optimal solution
- The slack or surplus values for the constraints at the optimal solution
- The shadow prices or dual values for the constraints at the optimal solution
- The range over which the objective function coefficient can vary in the linear program without changing the optimal solution
- The range over which a right-hand-side coefficient can vary without changing the dual value (shadow price) of the optimal solution

Big Mac Attack   This model addresses the issue of meeting one's daily dietary requirements at McDonalds. The objective can either be to minimize cost, total caloric intake, or total carbohydrates.

The model allows you to solve this problem to result in a *Continuous*, *Integer*, or *Binary* solution.

Selecting *Continuous* results in a computationally easier problem, but less realistic answer, since you cannot order 4.35 Big Macs.

Capital Investment   This model is an example of capital budgeting for four possible projects, where the objective is to decide which projects to choose in order to maximize the total return.

NLP with Jacobian   This model demonstrates the use of the Jacobian and gradient in a nonlinear optimization with constraints. When a Jacobian is available analytically, it can accelerate the optimization convergence.

The example is a simple geometric problem. Given a set of intersecting circles (where the intersection of all the circles is not empty), which point of those contained in all the circles is closest to a target point?

This model also features a node, *Abstractable NLP*, that is an example of a general mechanism for defining NLPs in an array-abstractable manner.

Optimal can dimensions This model is a simple example of using the NLP Optimizer in Analytica Optimizer. The model computes the optimal dimensions for a cylindrical can that must hold a given volume. The optimal can has the minimal surface area (and thus, uses the minimum material).

Optimal Production Allocation This model is a simple example of linear programming in Analytica Optimizer.

An integrated circuit manufacturer produces several different IC products (chips). Each chip is created by a sequence of processes, each carried out by a different machine. Every chip must pass through every process, but the time required for each process depends on the product. Some products require a lot of time at Process1, while other require very little time there but more time at other processes. The company's objective is to determine how much of each product to produce to maximize profit without exceeding the capacity of each process.

Problems with Local Optima Nonlinear optimization problems often contain local optima as well as global optima. Ideally, we hope that an optimization algorithm would always find the global optima, but no algorithm can guarantee this in the general case. Local minima generally appear to have all the characteristics of a global optima, so that when the Optimizer has found a local minima, it usually terminates the search.

The optional **guess** parameter to **NlpDefine()** provides a way to seed the search to the general area of where you think the global optima might be. When the Optimizer converges to a local optima, it is likely to be in the vicinity of the guess. So by trying a variety of guesses, you might be able to locate a set of local minima. Using the best of those might increase your chances of finding the true global optima.

This model demonstrates this method.

Production Planning LP This model is an example of production planning linear optimization. A company manufactures four versions of the same product and in the final part of the manufacturing process there are assembly, polishing, and packing operations. For each version, the time required for these operations is different, as is the profit per unit sold. How many of each variant should the company make per year and what is the associated profit?

Solve using NLP This model demonstrates how an nonlinear programming formulation can be used to solve a nonlinear equation. In this case, the equation is encoded as a constraint (this can be generalized to a system of nonlinear constraints), and the objective function is ignored (constant).

Two Mines Model This model is another production example. The *Two Mines Company* owns two different mines that produce an ore which, after being crushed, is graded into three classes: high, medium and low grade. The company has contracted to provide a smelting plant with 12 tons of high-grade, 8 tons of medium-grade, and 24 tons of low-grade ore per week. The two mines have different operating characteristics, in terms of cost to operate and production of each type of ore. How many days per week should each mine be operated to fulfill the smelting plant contract?

Risk Analysis

This folder contains applications relating to the field of risk analysis.

Seat belt safety This model compares the value of various policies for restraints on occupants of automobiles.

Txc This model demonstrates risk/benefit analysis, in this case regarding the benefits of reducing the emissions of fictitious air pollutant TXC.
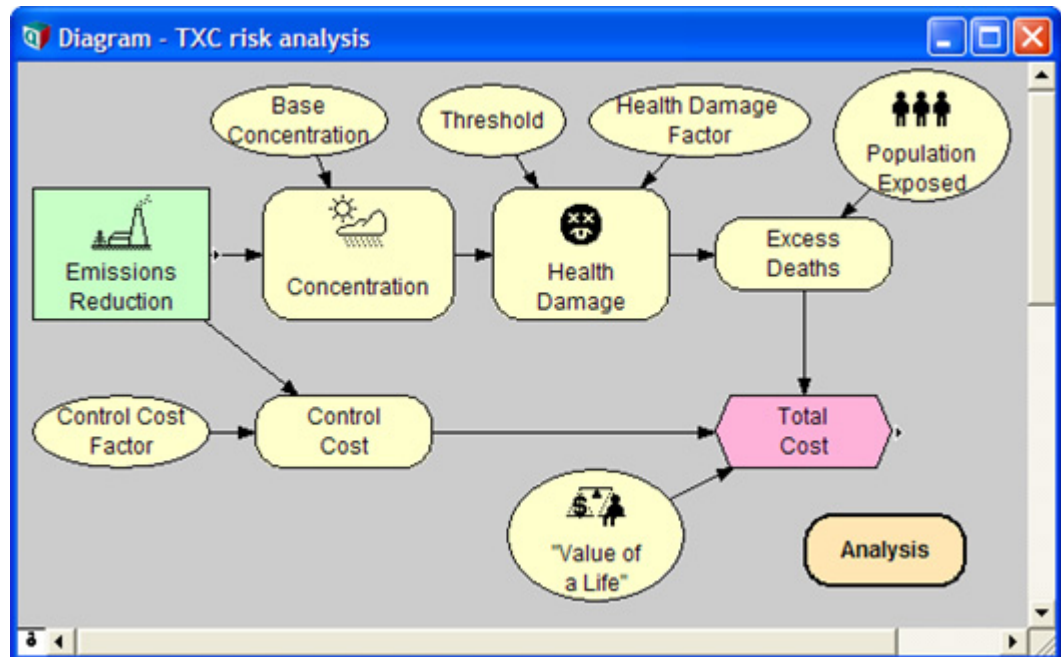
User Guide Examples

This folder contains the examples that are given in the *Analytica User Guide*.

Analyzing Unc & Sens   The examples in this model demonstrate Analytica's tools for analyzing the uncertainty of variables, relationships between uncertain variables, and sensitivity of outputs to changes in inputs. These include statistical functions and sensitivity analysis functions.

This model is used in the *Analytica User Guide*, Chapter 16, "Statistics, Sensitivity, and Uncertainty Analysis."

Array Examples   The examples in this model demonstrate the basics of working with multidimensional arrays.

This model is used in the *Analytica User Guide*, Chapter 11, "Arrays and Indexes."

Array Function Examples   The examples in this model demonstrate many more of Analytica's built-in array functions.

This model is used in the *Analytica User Guide*, Chapter 12, "More Array Functions."

Continuous Distributions   A continuous distribution is one that is defined for a continuous variable — that is, for a real-valued variable. The examples in this model demonstrate Analytica's built-in functions that create or modify continuous distributions.

This model is used in the *Analytica User Guide*, Chapter 15, "Probability Distributions."

Discrete Distributions   A discrete distribution is a probability distribution for a variable that can result only in certain, discrete outcomes. The examples in this model demonstrate Analytica's built-in functions that create or evaluate discrete distributions.

This model is used in the *Analytica User Guide*, Chapter 15, "Probability Distributions."

Expression Examples   The examples in this model demonstrate the building blocks for creating and editing variable definitions — expressions, standard operators, and mathematical functions.

This model is used in the *Analytica User Guide*, Chapter 10, "Using Expressions."

Input and Output Nodes   This model is used in the *Analytica User Guide*, Chapter 9, "Creating Interfaces for End Users."

Dynamic Models

A dynamic model is a model with one or more dynamic variables — that is, variables that can change over time. These models illustrate various uses of the **Dynamic()** function.

These models are used in the *Analytica User Guide*, Chapter 17, "Dynamic Simulation."

Dynamic & Dependencies
This model is a dynamic model that finds the downward velocity and position of a dropped object over a six second time period.

Dynamic & Uncertainty
This model shows three ways to use uncertainty with the **Dynamic()** function. In the first case, uncertainty samples are calculated once, at the initial time period. In the other two cases, new uncertainty samples are created for each time period (i.e., the values are re-sampled).

Dynamic Example 1
This model is the most simple dynamic model, with one variable that changes over time. This example finds the gasoline price for each of five years, assuming a 5% growth rate.

Dynamic Example 2
This model is a slight increase in complexity over **Dynamic Example 1**. This model still uses one variable that changes over time. However, instead of assuming a fixed inflation rate, this example, looks at the price with three different inflation rates for comparison.

Libraries

The libraries in this folder contain functions that can be added to your model and used similarly to Analytica's built-in functions. These libraries can be added to your model; see Chapter 19 of the *Analytica User Guide* for information on how to add a library to a model.

Bayes Function
This library contains **Posterior()**, a function for calculating posterior probabilities using Bayes' Theorem.

Complex Library
This is a library of functions for working with complex numbers. It contains functions for basic arithmetic, polar representations of complex numbers, scalar functions for finding complex roots, logs, exponents, matrix functions, and trigonometric functions. Addition, subtraction, and scalar multiply are performed with the usual operators. Complex multiplication and complex division require the use of explicit functions.

Complex numbers as seen by users of this library should always be in the Euclidean complex form, such as `a + bi`, where `a` is the real part and `b` is the imaginary part.

Concatenation
This library contains functions to make concatenation more convenient. Functions **Concat3()** through **Concat10()** are generalizations of the built-in **Concat()** function which concatenate from 3 to 10 arrays in a single call (the built-in **Concat()** function concatenates two arrays). **ConcatRows()** concatenates all the rows of a single array.

Data Statistics Library
This library contains functions for calculating statistical quantities for a list of numbers over an explicit index other than **Run**, such as the index used for the statistics results in Analytica: the mean, variance, standard deviation, kurtosis, skewness, fractiles, covariance, correlation, frequency, etc.

Distribution Density
The functions in this library return the probability densities (for continuous distributions) and probabilities (for discreet distributions) for the standard distribution functions that are built into Analytica. It also includes Cumalative distribution functions for continuous distributions and some discrete distributions, and inverse cumalative functions for most continuous distributions.

Distribution Variations
This library contains various functions for defining standard disributions using different sets of parameters.

Expand Index
This model contains the function **Change Index**. When this function is given an array indexed by one index, it returns an array indexed by another index.

This model contains an example in which **Change Index** is used to combine cashflows over two different time periods into a single cashflow over a single time period.

Financial Library
This model contains a variety of corporate finance functions: Black-Scholes Option Values (**CallOption**, **PutOption**), Capital Asset Pricing Model (**CAPM**), Miles/Ezzell Adjusted Cost of Capital (**CostCapME**), Modigliani/Miller Adjusted Cost of Capital (**CostCapMM**), Present Value of Perpe-

tuity (**PVperp**), Present Value of Growing Perpetuity (**PVgperp**), and Weighted Average Cost of Capital (**WACC**).

Flat File Library This library provides functions for writing data to and from flat files, particularly between two-dimensional tables and comma-separated value (CSV) files.

Garbage Bin Library This library provides a *Recycle Bin* for your model (including a recycle bin icon). To use it, simply drag your discarded objects into the recycle bin module.

Unlike deletion, items in your recycle bin can be retrieved, and the **Undo** command (*Control+Z*) works for items dragged into the bin. If an item you put into the bin has dependents outside the bin, it shows arrows from the bin — a signal that you should either retrieve the item, since it is being used, or also drag its dependent(s) into the bin.

To delete the items in the bin permanently, open the bin, select all its contents (with *Control+A*), and delete them (press *Delete* key).

Linked List Library This library contains routines for manipulating linked lists.

The simplest linked list is just **NULL** — a linked list with zero elements.

Any other linked list is a reference to a record indexed by **Linked_List** (an index defined in this library) with each cell containing a reference, to the element, and a pointer to the remainder of the list.

Linked lists are created and manipulated using functions in this library:

- Use the function **LL_Push()** to build the list.
- After a list is built, the easiest way to use and view it is to convert it back to an array using **LL_to_RArray()**. This reverses the order of items in the linked list (which has the last item "pushed" into the linked list as the first item in the list) so that the array has the items ordered the same as when they were added to the list.
- The function **LL_to_Array()** returns an array with the items ordered the same way as the linked list (the last entered item is first in the list/array).
- Other functions provide the first item in the list, the Nth item in the list, the list length, and allow you to remove (pop) the first item in the list.

Multivariate Distributions Library This library contains functions for creating several multivariate distributions:

- Gaussian
- Dirichlet
- BiNormal and Multinormal
- Uniform Spherical and MultiUniform
- Sample covariance and Sample Correlation
- Functions for correlating distributions and results

ODBC-Library This library provides additional functionality (**ValList**, **InsertRecSql**, **WriteTableSql**) for using ODBC access to databases. Note that using ODBC requires Analytica Enterprise; refer to Chapter 22 of the *Analytica User Guide*.

Profiling Library Use this library to see which variables and functions are taking most of the computation time when running your model. Consult the model's description for an explanation of how to use the library.

Tip The ODBC and Profiling libraries require Analytica Enterprise, or ADE. They do not work with other versions of Analytica.

Optimization Functions Library This library contains functions for optimization and equation solving, using a Newton-Raphson style search.

Summary

If you have created a model that other Analytica users would benefit from, please send it to us for possible inclusion in future versions of the *Example Models* folder. Send your well-constructed, thoroughly documented models to:

support@lumina.com

Also, if you experience any problems with the example models, or if you feel that they need to be changed in some way, please contact us at the above email address.

You can also submit example models to the Analytica wiki, a web application that allows member of the Analytica community to collaborate and share information. Go to:
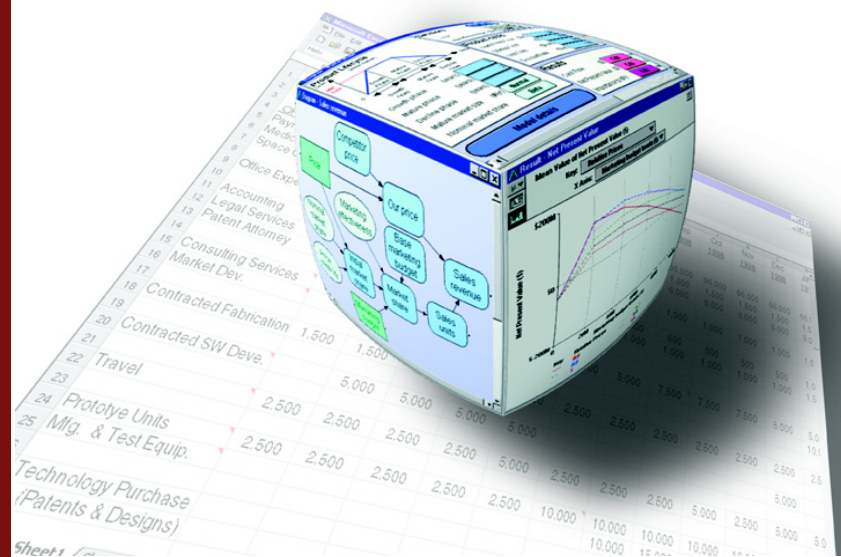
http://lumina.com/wiki/index.php/Example_Models

Here you can view example models submitted by others and to submit your own examples.

Go to the Analytica wiki home page (http://lumina.com/wiki/) to find all sorts of information on Analytica including additional information on Analytica functions, frequently asked questions (FAQs) and their answers, guides to modeling, what's new in Analytica, and much more.

Glossary

This glossary includes a compilation of terms specific to Analytica as well as statistical terms used in this manual.

Glossary

Array	A collection of values that can be viewed as one or more tables. An array has one or more dimensions; each dimension is identified by an index.
Arrow, influence arrow	Influence arrows (or arrows) from one variable node to another indicate that the origin node affects (influences) the destination node. If the nodes depict variables, the origin variable is an input to the destination variable, and the destination variable is an output of the origin variable.
Arrow tool	The tool for drawing arrows between nodes.
Attribute	A property of an object, such as its title, description, definition, value, or inputs.
Attribute panel	An auxiliary window pane that can open below an influence diagram. Use it to rapidly examine one attribute at a time of any variable, function, or module.
Browse tool	A tool for examining the structure and assumptions of a model, with limited ability to make changes to the model.
Chance variable	An uncertain variable that cannot be directly controlled by the decision maker. It is usually defined by a probability distribution. A chance variable is represented by an oval node.
Class	Analytica objects are organized into the following classes: module, attribute, function, and decision, chance, objective, index, and general variables.
Cumulative probability distribution	A graphical representation of a probability distribution. It plots the cumulative probability that the actual value of the uncertain variable **x** is less than or equal to each possible value of **x**. The cumulative probability distribution is a display option in the Uncertainty View popup menu in a **Result** window.
Decision variable	A variable that the decision maker can control directly. A decision variable is represented by a rectangular node.
Definition	A specification for computing a variable's value. The definition can be a number, a mathematical expression, a list of values, a table, or a probability distribution.
Description	Text explaining what the object represents in the system being modeled. The description is limited to 32,000 characters in length.
Deterministic table	A function that gives the value of a variable conditional on the values of its input variables, where the inputs are all discrete variables.
Deterministic value	See *Mid, Mid value*.
Domain	The possible outcomes for a variable defined as a probability table.
Edit table	A definition that is an array (table) is also called an edit table because it can be edited.
Edit tool	A tool for creating or changing a model. Use it to move, resize, and edit nodes, and to expose the arrow tool and node palette.
Expression	A formula that can contain any combination of numbers, variables, functions, distributions, and operators, such as `0.5`, `a-b`, or `Min(x)`, combined according to the Analytica language syntax.
Expression type	Expression types include expression, list (of expressions or numbers), list of labels (text strings), table, probability table, and distribution. You select an expression type using the Expression popup menu, which appears above the **Definition** field. Note that any definition, regardless of expression type, can be viewed as an expression.
General variable	Any type of variable; useful when the variable type is unknown. The general variable typically represents a deterministic or functional dependency.
Graph	Format for displaying a multidimensional result. To view a result as a graph, click the **Graph** button. See also *Table*.
Identifier	A short name for an object. A variable's identifier is used to refer to the variable in mathematical expressions in definitions of other variables. An identifier must start with a letter, have no more than 20 characters, and contain only letters, numbers, and the underscore characer (_) (used instead of a space). Compare to *Title*.
Importance analysis	Shows the effect the uncertainty of one or more input variables has on the uncertainty of an output variable. Importance is defined as the rank-order correlation between the sample of output

Glossary

values and the sample for each uncertain input. This is a robust measure of the uncertain contribution because it is insensitive to extreme values and skewed distributions.

Index	Identifies a dimension of an array (table). An index is usually a variable defined as a list, list of labels, or sequence.
Indexes	Plural of index, indicates a set of index variables that define the dimensions of a table (in an edit table or value).
Index variable	A class of variable, defined as a list, list of labels, or sequence, that is used to identify the dimensions of a table, for example, in an edit table. An index variable is represented by a parallelogram node. See also *Edit table*.
Influence arrow	See *Arrow, influence arrow*.
Influence diagram	A graphical representation of a model, consisting of nodes (variables) and arrows (relationships between variables).
Input	An input of a variable **x** is a variable that has an arrow drawn to **x**, or appears in the definition of **x**. See also *Output*.
Input arrowhead	Shows that a node has one or more inputs from outside its module. This arrowhead is located on the left side of a node. Press the arrowhead for a popup menu of input variables.
List	A type of expression, consisting of an ordered set of numbers or expressions, available in the Expression popup menu. A list is often used to define index and decision variables.
List of labels	A type of expression, consisting of an ordered set of text labels, available in the Expression popup menu. A list of labels is often used to define index and decision variables.
Mean	The average or expected value.
Median	The middle number or value when the data values are ranked in order of size, i.e., the middle data point.
Mid, Mid value	A calculation of the variable's value assuming all uncertain inputs are fixed at their median values.
Model	A module, or a hierarchy of modules; the main, or root, module at the top of the module hierarchy. Between sessions, a model is stored in an Analytica document file.
Module	A collection of related objects, including variables, functions, and other modules, organized as a separate influence diagram. A module is represented by a rounded rectangular node with a thick outline.
Node	A box (rectangular, oval, or any other shape) that represents a variable in an influence diagram. Different node shapes are used to represent different types of variables.
Normal distribution	The bell-shaped curve, or Gaussian distribution.
Object Finder	A dialog box used to browse and edit the functions and variables available in a model.
Object window	List of the attributes for an object (variable, function, or module), including its class, identifier, title, and description.
Objective variable	A variable that evaluates the overall value or desirability of possible outcomes. The objective can be measured as cost, value, or utility. A purpose of most models is to find the decision or decisions that optimize the objective — for example, minimizing cost or maximizing expected utility. Most decision models contain a single objective node, although the objective can be composed of several sub-objectives. An objective variable is represented by a hexagonal node.
Output	An output of a variable **x** is a variable that has an arrow drawn from **x**, or whose definition refers to **x**. See also *Input*.
Output arrowhead	Shows that a node has one or more outputs outside its module. This arrowhead is located on the right side of the node. Press the arrowhead for a popup menu of the output variables.
Parametric analysis	A type of sensitivity analysis in which you specify a set of alternative values for one or more inputs, and examine the effect on selected model output variables. See also *Sensitivity analysis*.
Probabilistic variable	A variable that is uncertain and is defined with a probability distribution.

Glossary

Probability bands	The bands that capture a certain portion of the total probability for a variable. For example, the 5% and 95% probability bands contain 90% of the total probability, while the 50% probability band corresponds to the median value. By default, the 5%, 25%, 50%, 75%, and 95% probability bands are shown. These bands are also referred to as *confidence intervals* or *fractiles*. Probability bands are a display option in the Uncertainty View popup menu on a **Result** window.
Probability density function (PDF)	A graphical representation of a probability distribution that plots the probability density against the value of the variable. The probability density at each value of **x** is the relative probability that **x** is at or near that value. The probability density function can be displayed for continuous, but not discrete, variables. It is a display option in the Uncertainty View popup menu on a **Result** window.
Probability distribution	Describes the relative likelihood of a variable having different possible values.
Probability table	A table for defining a chance variable with a set of outcomes and a discrete probability distribution (numerical probability for each outcome). If the variable depends on (that is, is conditioned by) other discrete variables, each of these conditioning variables gives an additional dimension to the table, so you can specify the probability distribution conditional on the value of each conditioning variable.
Self	A keyword referring to the index of a table that is indexed by itself. `self` refers to the alternative values of the variable defined by the table.
Sensitivity analysis	A group of methods to identify and compare the effects of various input variables to a model on a selected output. Example methods of sensitivity analysis are importance analysis and parametric analysis. See also *Parametric analysis*.
Standard deviation	Reflects the amount of spread or dispersion in an uncertainty distribution. It is the square root of the variance.
Table	A two-dimensional view of an array-valued definition or result. The array can have more than two dimensions, but only two can be seen at one time. A table is a type of expression available in the Expression popup menu. A definition that is a table is also called an *edit table*. In the **Result** window, click the **table** button to select the table view of an array-valued result. See also *Graph*.
Title	The full name of an Analytica object, briefly describing what the object represents. The title of a variable, function, or module is displayed in its node, in window titles, and in object lists. It is limited to 255 characters in length. The title can contain any characters, including spaces and punctuation. Compare to *Identifier*.
Uniform distribution	A distribution representing an equal chance of occurrence for any value between the lower and upper values.
Uncertainty View options	An uncertain result can be displayed as a *mid value*, *mean*, statistics, *probability bands*, *probability density*, *cumulative probability*, or sample. Select the option to display with the popup menu in the top left corner of the **Result** window or in the **Result** menu.
Units	The increments of measurement for a variable. Units are used to annotate tables and graphs, but are not used in evaluating a variable.
Value	A variable's value attribute is its mid value, computed by assuming that all uncertain inputs are fixed at their median values. It can be a scalar (single) number, a table of numbers, or a probability distribution.
Variable	An object that has a value, which can be a text string, a number, or a table. Classes of variable include decision, chance, and objective.

Analytica Windows and Dialogs

Diagram Window:
Inputs and Outputs

Diagram Window:
Influence Diagram

Result Window — Graph View

Object Window

Object Finder

Result Window — Table View

Diagram Style Dialog

Node Style Dialog

Number Format Dialog

Graph Setup Dialog

Uncertainty Setup Dialog

Preferences Dialog

Attributes Dialog

Outline Window

Find Dialog

Quick Reference

The Tool Bar

Parent Diagram, Outline, Object, Result, Definition, Browse tool, Edit tool, Arrow tool, Decision node, Variable node, Chance node, Objective node, Module node, **Index node**, Constant node, Function node, Text box, Button node

It displays the node palette when you select the edit tool or arrow tool.

Numerical Formats (Output)

Format	Description	Example
Suffix	letter denotes order of magnitude, such as M for 10^{-6} (see table below)	12.35K
Exponent	scientific exponential	1.235e+004
Fixed Point	fixed decimal point	12345.68
Integer	fixed point with no decimals	12346
Percent	percentage	1234568%
Date	text date	12 Jan 2008
Boolean	true or false	True

Suffix format

Power of 10	Suffix	Prefix	Power of 10	Suffix	Prefix
			10^{-2}	%	percent
10^3	K	Kilo	10^{-3}	m	milli
10^6	M	Mega or Million	10^{-6}	μ	micro (mu)
10^9	G	Giga	10^{-9}	n	nano
10^{12}	T	Tera or Trillion	10^{-12}	p	pico
10^{15}	Q	Quad	10^{-15}	f	femto