Vensim®

Ventana® Simulation Environment

DSS
Professional
PLE Plus
PLE

with

Causal Tracing®
Reality Check®
and
SyntheSim?

User's Guide
Version 5

Patent and Trademark Notices

Table of Contents

1 Introduction

About Vensim

Vensim is a visual modeling tool that allows you to conceptualize, document, simulate, analyze, and optimize models of dynamic systems. Vensim provides a simple and flexible way of building simulation models from causal loop or stock and flow diagrams.

By connecting words with arrows, relationships among system variables are entered and recorded as causal connections. This information is used by the Equation Editor to help you form a complete simulation model. You can analyze your model throughout the building process, looking at the causes and uses of a variable, and also at the loops involving the variable. When you have built a model that can be simulated, Vensim lets you thoroughly explore the behavior of the model.

About this User's Guide

This User's Guide will show you the main features of the Vensim simulation software, introducing Vensim in a hands-on environment where you can examine existing models, and construct your own causal loop diagrams, stock and flow diagrams, and simulation models. Advanced features of Vensim, such as sensitivity testing, subscripting (arrays), and optimization, are presented using existing simulation models to speed learning. All of the models in this Guide (with the exception of those used in Chapters 3, 14 and 18) are presented with all the structure and all the equations you need so that you can build the models yourself.

This Guide is common across all Vensim configurations. In some cases functionality that is not in your Vensim configuration will be discussed. We have marked this discussion as clearly as possible. The following table shows which chapters apply to which configurations:

Chapter	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
PLE	?	?	?	?	?	?	?	?	?	?		?	?	?		?			?	?
PLE Plus	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?			?	?
Standard	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?			?	?
Professional	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
DSS	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

For new users of Vensim, this Guide is designed to be read and worked through from beginning to end. For experienced users of Vensim Chapters 3, 7 and especially 13 are recommended as they describe added and changed functionality.

How this Guide is Organized

This Guide is broadly divided in three parts. The first three chapters are intended to get you started using the software. Chapters 4 through 10 cover the mechanics of building models with Vensim: how to draw diagrams, add equations, simulate and analyze models, and display output. Chapters 11 through 18 demonstrate some advanced operations in Vensim using existing sample models (which you can also build).

Chapter 1 provides an overview of this Guide and of Vensim including instructions for installing Vensim.

Chapter 2 introduces you to the Vensim User Interface. This chapter provides an overview of Vensim's functionality, along with information on the Sketch tools, Analysis tools, and Control windows.

Chapter 3 provides hands-on experience simulating and analyzing an existing model.

Chapter 4 introduces you to the construction and use of causal loop diagrams. Structural analysis of diagrams using Analysis tools is also described.

Chapter 5 covers building stock and flow (Level and Rate) diagrams.

Chapter 6 steps you through the construction of a simulation model of the growth of a population. This problem helps you learn the mechanics of building, simulating, and analyzing models with Vensim.

Chapter 7 demonstrates the inclusion of functions in models and also shows how to detect and correct errors that occur during simulation.

Chapter 8 describes how to create and use Lookups. These are functions that relate an input to an output by drawing a graph of the relationship. They are also called Lookup Tables and sometimes just Tables.

Chapter 9 develops a model with multiple views, allowing you to split a model into different sectors. Vensim PLE users should skip this chapter.

Chapter 10 shows you how to customize output graphs from the Analysis tools. The Custom Graph and Table editors, with which you can create customized graphical and tabular output for multiple variables are also described.

Chapter 11 explains how to use models as games, or "flight simulators", where you can step progressively through time while making decisions at each time. Vensim PLE users should skip this chapter.

Chapter 12 shows how to make use of Input Output Controls and Navigation Links to make a model easier and more fun to use and present to others.

Chapter 13 presents SyntheSim and shows how it can be used to work with and understand models. The SyntheSim functionality is new to Vensim 5.

Chapter 14 covers the Reality Check feature in Vensim which allows you to build model validity tests.

Chapter 15 provides an example of Monte Carlo sensitivity testing. You set parameters with uncertainty values and then run sensitivity analysis to determine uncertainties in particular variables over the simulation time. Vensim PLE users should skip this chapter.

Chapter 16 describes how to use data in models. Data variables are defined which access exogenous time series and drive model behavior. This chapter covers importing data from text files and from spreadsheets. Not all functionality is available in Vensim PLE.

Chapter 17 introduces subscripted models (subscript or array variables in a model) and provides both simple and more complex examples. This chapter is applicable to Vensim Professional and DSS only.

Chapter 18 covers two types of model optimization. The first type is calibration of model constants while fitting model output to an external data series. The second is policy optimization, finding the best model parameters to maximize or minimize payoff variables. This Chapter is applicable to Vensim Professional and DSS only.

Chapter 19 gives directions and hints for making your model available to others using the Vensim Model Reader and other Vensim configurations.

Chapter 20 is an introduction to using reference modes as a way of conceptualizing feedback models and also as a means of testing and developing mental integration skills.

The Appendix provides information about resources available for learning more about building and using dynamic models.

Finally, Support and Licenses, provides information on getting support and training for Vensim as well as a reference copy of the license agreements.

Style Conventions

To differentiate among the many elements in Vensim, this Guide and the rest of the documentation follow some style conventions:

? Names of files stored on disk and their extensions are shown in *italic* (e.g. *project.mdl*). The names of datasets are shown in *italic* without the *.vdf* extension (e.g. *baserun*).

? Names of variables and equations in a Vensim model are in *`Italic Courier font`*, (e.g., *`Population`*).

? Names of Vensim menu items, controls, buttons, tools, toolbars, and names in dialog boxes are all capitalized, (e.g., Control Panel), and usually appear in **bold font** if the name refers to an object in Vensim that you will be selecting or acting on (e.g., press the **Simulate** button).

? Actions that you need to perform use a triangular bullet (e.g.,

✍ Click the **Open Model** button and choose the model *wfinv.mdl*).

Important Notes

About Directories

All of the models discusses in this Manual are contained in the *guide* subdirectory of the directory into which you installed sample models. This will typically be *c:\Program Files\Vensim\models*. On the Macintosh it will normally be the *Models* Folder in the *Vensim* Folder on your hard disk. However, you can install Vensim and the models in any location you choose so we will name directories using the path starting with *guide* as in *guide\chap07\complete*. On the Macintosh this just means the *complete* folder in the *chap07* folder in the *guide* folder.

When you are working with your own models we strongly recommend that you store them in a directory that is not a subdirectory of Vensim. For the purposes of this Guide saving your work in subdirectories of the *guide* directory is perfectly reasonable.

About Screen Shots

There are some differences between the appearance of Vensim PLE, PLE Plus, Standard, Professional and DSS and it is also possible to make changes to Toolsets. Most of the pictures of screens in this Guide were taken using Vensim DSS and default Toolsets. If you see something that looks somewhat different from your version of Vensim it is not a problem.

About the Mouse

Windows computers have left and right mouse buttons, but Macintosh computers have only one mouse button. Vensim makes use of the left and right buttons on PCs as described below. Macintosh users will need to use their mouse button and the Ctrl key or Apple key (for right button clicks) as described below.

Left Button

The left button is used to perform almost all operations in Vensim, such as selecting a menu, clicking a button, dragging graphs or sketch objects. Whenever the Guide requires a mouse click without mention of left or right, perform a click with the left button. Macintosh users should click the only button.

Right Button

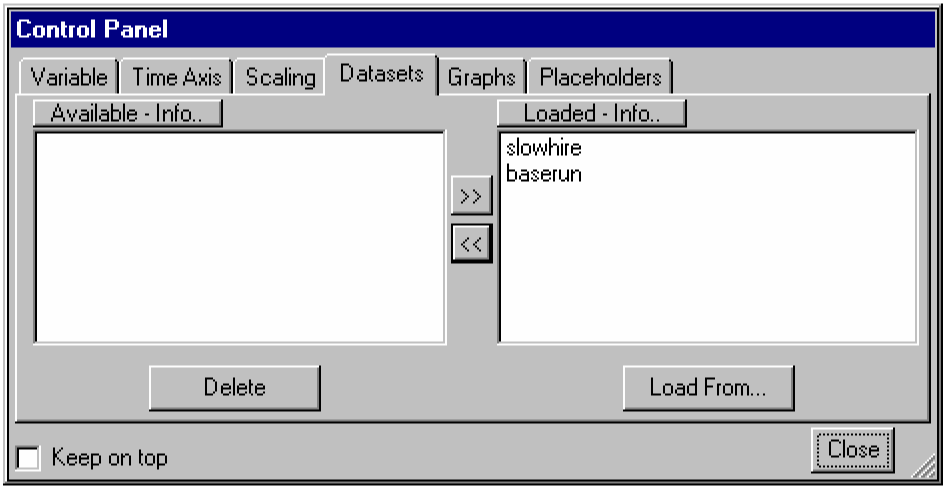
The right button is used to set options for Sketch tools, Analysis tools, and Sketch objects and also for positioning and zooming on Sketches. When a right button click is required, click the right button on PCs. Macintosh users need to click the mouse button while holding down the Ctrl key or the Apple key (Ctrl + Click or ✍ + Click). To scroll a sketch drag with the right mouse button or hold down the Ctlr key and drag with the left mouse button. To zoom drag with the right mouse button while holding down the shift key.

Mouse Wheel

On computers equipped with a Mouse Wheel you can use this to scroll windows up and down. For horizontal scrolling hold down the shift key and move the mouse wheel. Holding down the control key will allow you to zoom in and out of a sketch.

About Tab Dialog Boxes

Tab Dialogs are special dialog boxes common in Windows 95and later. These dialog boxes simplify controls by separating information into different "folders" with tabs. You can switch between folders by clicking on the appropriate tab. Examples of tab dialog boxes are the Simulation Control, the Equation Editor, and the Control Panel (shown below):



In this picture, the **Datasets** tab has been selected showing two loaded simulation runs. The **Variable**, **Time Axis**, **Scaling**, and other controls can be selected by clicking on the tab desired.

Installing Vensim

To install Vensim you need to get and then start the installation program. You can get the installation program from the CD or from our website http://www.vensim.com.

The Vensim CD

The Vensim CD contains the installation programs for all Vensim configurations for both Windows and Macintosh. The label of the CD will show the version number. Though installers for all configurations are contained, you will only be able to install the configuration you have a registration code for as described below.

Downloading Vensim

You can download Vensim from our website http://www.vensim.com. Your purchase of a Vensim license includes one year of free electronic updates, thereafter you will need to pay the maintenance subscription fee to receive updates and technical support. You can check our website to see what versions are available. The menu item Help>Newer Released will also tell you if a more recent version is available. The direct link for downloading Vensim is http://www.vensim.com/cgi-bin/download.exe . When you enter this you will be asked to enter your registration code (see below). The registration code identifies the product you have as well the date through which you have continued maintenance.

Once you enter your registration code you will be able to choose between the versions of Vensim available to you. You should choose the most recent available version unless there is a specific version you need.

To downloadVensim PLE for educational use go to http://www.vensim.com/freedownload.html.

Windows: The Windows installer is broken into a number of relatively small files. The first of these files has a name that depends on the product (for example, *vendss32.exe* for Vensim DSS). The remaining files are labeled *disk2.vip*, *disk3.vip* and so on. When downloading, be sure to save all the files in the same directory on your computer and it is very important that you do not change the names of any files except the first. The contents of the different files are clearly marked on the download page. The first disk contains the actual program, *disk2.vip* contains the sample models and *disk3.vip* through *disk6.vip* contain the electronic documentation. Vensim DSS has an additional *disk7.vip* containing the Vensim DLL and support files.

If you only want to install the updated program you can just download the first file.

NOTE Do not try to open the *.vip* files. The will be used during the installation process but cannot be opened individually. Even if you only want to install the contents of one of them you will need the first (*.exe*) file.

Macintosh: There are separate installers for the program and sample models on the Macintosh. The program installer is specific to the product configuration while the models are common.

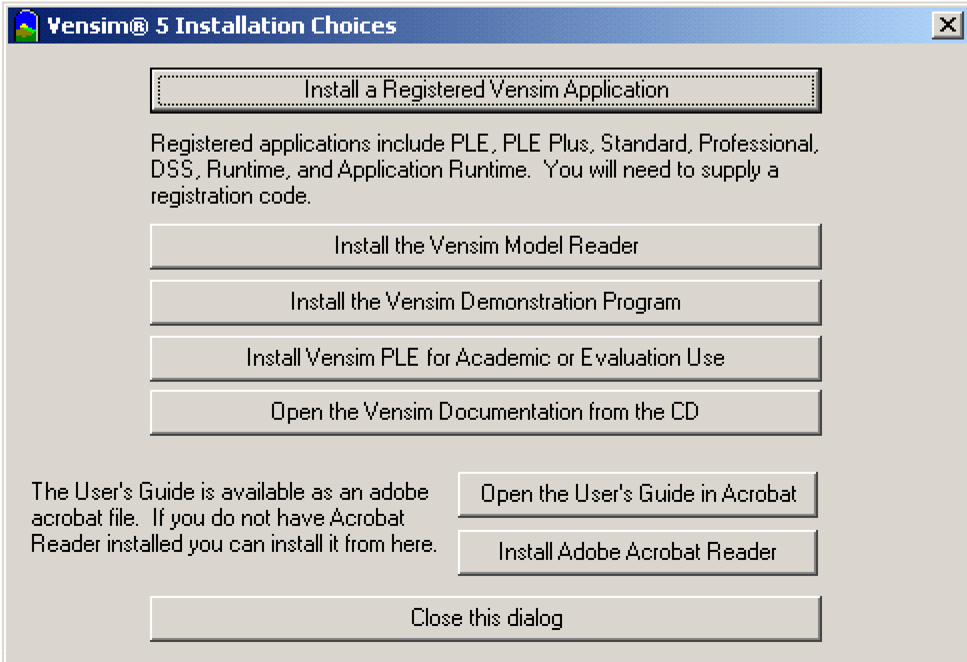
The downloadable Macintosh files are binhex files and need to be converted to Macintosh program files. This will most likely be done automatically by your web browser, if not there are a number of utilities that do this conversion.

Browser based help files: The help files are available as *.htm* files on the Vensim CD. You can drag these on to your hard drive if you want. To use them open your Web Browser then open the file *vensim.htm.* You can also view the help files online at http://www.vensim.com/documentation/vensim.htm.

Start the Installation Program

From CD

If you have a CD just insert it into your computer. Under Windows the Installation Choices dialog should automatically open:



If this dialog does not open double click on the program file *setup.exe* contained on the CD.

From the Installation Choices dialog select the program you want to install. If you have a registration code, click on **Install a Registered Vensim Application** and then enter the registration code to start the installer.

On the Macintosh just open the CD folder and look for the name of the installer you want. You should install the program first, then the models, and then the help file.

From Download

If you downloaded the Vensim installation program double click on the first file (for example, *vendss32.exe* for Vensim DSS) you downloaded. This will be in the directory you chose when your web browser queried you for the location to save to. (It is important to remember where you saved it to).

If you are on the Macintosh and the download files are still binhex (.hqx) files you will need to convert this to an application. There are a number of programs that do this.

License Agreement

Before you can install Vensim you will need to agree to the terms of a license agreement. For your convenience this licensee agreement is repeated at the end of this Manual. If you agree to the terms of the license agreement indicate this and continue with the installation. If you do not agree to the terms of the license agreement you may return the software for a refund of whatever license fees you have paid.

Registration Code

Vensim DSS, Professional, Standard, PLE Plus and PLE for commercial use require a Registration Code. Vensim PLE for educational or evaluation use do not require a registration code. If you do not have a Registration Code you will need to Install Vensim PLE for educational or evaluation use. On the Macintosh you can use the registration code EDU for educational use and EVAL for evaluation purposes when installing Vensim PLE. On Windows the install program will have radio buttons for these choices.

The Registration Code is a series of letters, digits and dashes. If you purchased your license on-line or elected to receive your license electronically, your Registration Code will be sent to you via email. Otherwise it will be printed on a license certificate or on a label placed on the back of the CD sleeve or Jewel Case. In any case you will see something like:

```
Registration Code: ABCDE-FGHIJ-KLMNO-PRQS
Company: Ventana Systems, Inc.
Product: DSS
Serial#: 0
```

Enter the Registration Code as it appears. It is not case sensitive and you may substitute spaces for the dashes. Your company name must also match the company name shown below your registration code (again this is not case sensitive). If you received the code via email, it is easiest to copy it from the email and paste it in. If you make a mistake entering the code, you will be asked to edit the code you entered. Double check to be sure it is identical with the one you received.

If your company name is incorrect please contact us.

NOTE If you are installing Vensim PLE On the Macintosh you will need to enter the registration code **EDU** for educational use and **EVAL** for evaluation purposes. On windows these choices are identified with Radio buttons.

Installation Directory

You can choose the directory or folder into which you want to install Vensim. For Windows this defaults to the program files directory (normally *c:\Program Files\Vensim*). On the Macintosh a *Vensim* folder will be created on the hard disk. However, you can choose to install Vensim anywhere that you want. When we refer to directories in this Manual they are subdirectories of the directory in which Vensim has been installed.

Other Vensim Configurations

Vensim Professional and DSS will install to a program named *vensim.exe*. Thus if you upgrade from Vensim Professional to Vensim DSS the installation will simply replace Vensim Professional. Vensim PLE and PLE plus, on the other hand, install to different names. Vensim PLE installs as an executable named *venple.exe* (*Vensim PLE* on the Macintosh) and Vensim PLE Plus as an executable named *venplep.exe* (*Vensim PLE Plus* on the Macintosh). Therefore, you can install Vensim PLE or PLE Plus and another Vensim configuration in the same directory without any conflicts. The support files installed for all configurations are identical, though some configurations install additional files not needed by other configurations. Vensim also stores a limited amount of configuration in the files *vensim.ini* (Professional and DSS), *venple.ini* (PLE) and *venplep.ini* (PLE Plus).

The long and short of this is that installing Vensim PLE along with another Vensim configuration in the same place will not cause any problems.

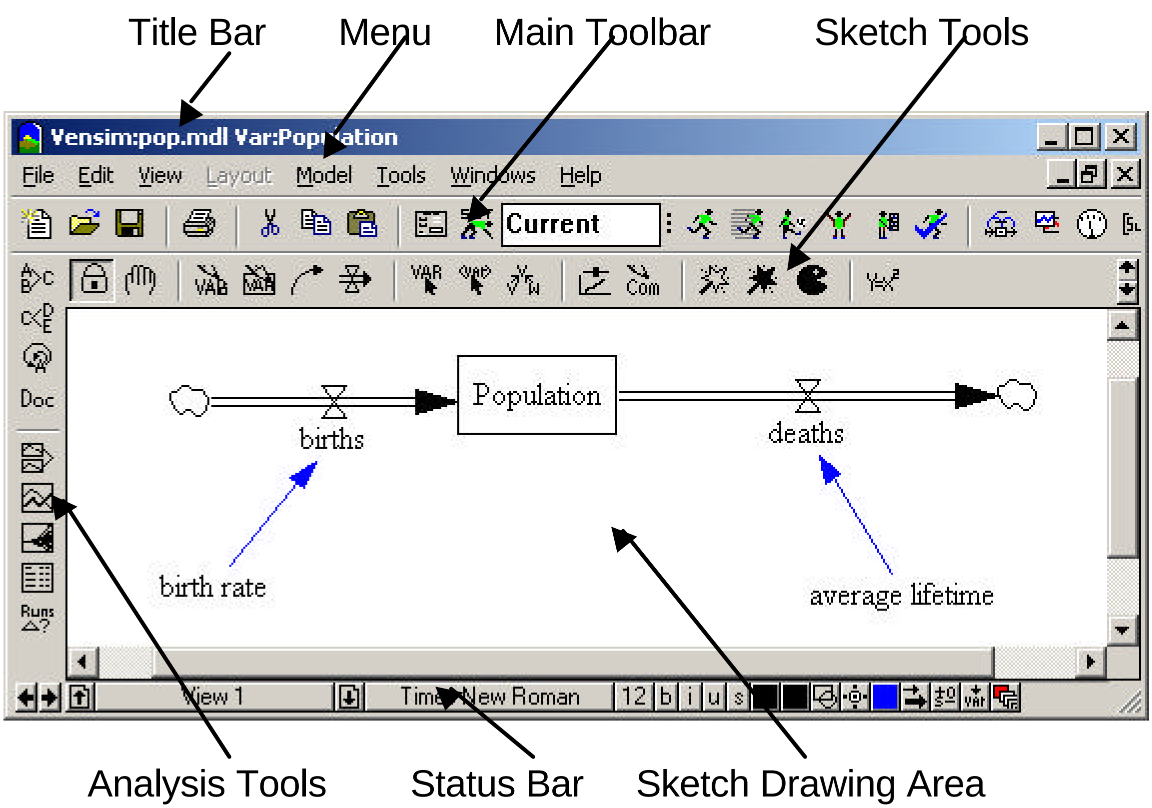
Other Resources

As you work with Vensim, you have several valuable resources at your fingertips. The Pullout Reference card shows the names and operations of buttons found in the main Vensim window. This User's Guide is complemented by the *Modeling Guide* and the *Reference Manual*. The *Modeling Guide* describes the development of a number of different dynamics models and goes into some more advanced modeling techniques. The *Reference Manual* provides in-depth coverage of all the features and operation of Vensim. There is a *Vensim DSS Reference Supplement* that documents capabilities unique to Vensim DSS. All of these manuals are available as on-line help, which facilitates searches on particular topics.

2 The Vensim User Interface

Main Features

Vensim uses an interface that can be thought of as a workbench and a set of tools. The main Vensim window is the Workbench, which always includes the Title Bar, the Menu, the Toolbar, and the Analysis tools. When Vensim has a model open (as shown below), the Sketch tools and the Status Bar also appear.



Title Bar

The Title Bar shows two important items: the model that is open (e.g., *Sales.mdl*) and the Workbench Variable (e.g., `sales force productivity`).

The Workbench Variable is any variable in the model that you have selected and for which you want more information, such as the dynamic behavior of that variable. The Workbench Variable is selected clicking on a variable or by using the Variable Selection Control in the Control Panel ("Control Panel" later in this Chapter).

Menu

Many operations in Vensim can be performed from the menu.

? The **File** menu contains common functions such as Open Model, Save, Print, etc.

? The **Edit** menu allows you to copy and paste selected portions of your model. You can also search for a variable in your model.

? The **View** menu has options for manipulating the sketch of the model and for viewing a model as text-only (available only in Vensim Professional and DSS).

? The **Layout** menu allows you to manipulate the position and size of elements in the sketch.

? The **Model** menu provides access to the Simulation Control and the Time Bounds dialogs, the model checking features, and importing and exporting datasets.

? The **Tools** menu sets Vensim's global options and allows you to manipulate Analysis tools and Sketch tools as well as set global options. In Vensim PLE and PLE Plus there is an **Options** menu rather than a **Tool** menu.

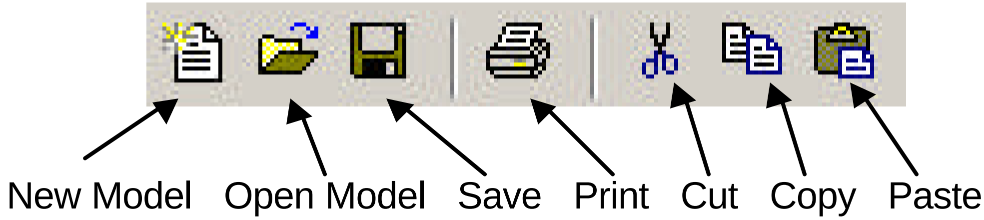
? The **Windows** menu enables you to switch among different open windows.

? The **Help** menu provides access to the on-line help system.

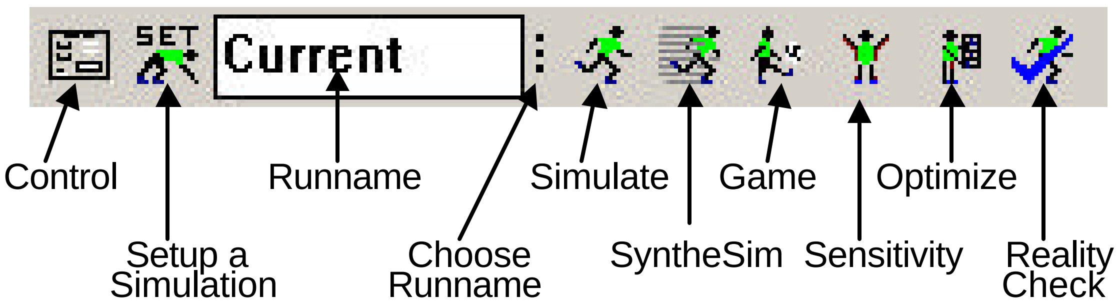
Menus are context sensitive and the commands apply to whichever window currently is active. The most commonly used menu commands also have shortcut keys and can be performed from the toolbar described below.

Toolbar

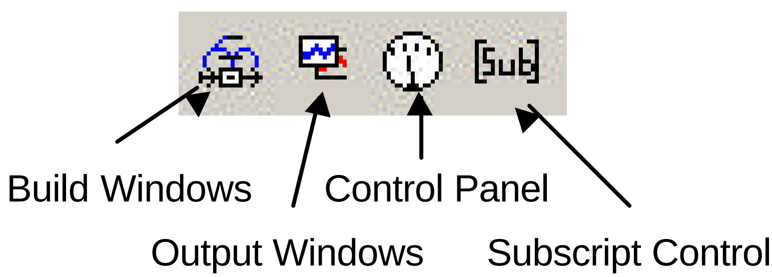
The Toolbar provides buttons for some of the most commonly used menu items and simulation features. The first set of buttons access some File and Edit menu items.



The next several buttons and the **Runname** editing box are used for simulating models.



The last few buttons access the window classes. Click on a button to bring forward that type of window or circulate through windows of that type.



The exact appearance of the toolbar depends on your Vensim configuration. Some configurations have fewer entries than shown above. For example the Vensim PLE toolbar appears as:

The Window Classes

Vensim contains several types or classes of windows:

1. Build Windows are used for constructing new models, or for modifying, navigating, and simulating existing models. In Vensim Standard, Professional and DSS several models can be open at once, each in its own Build Window.
2. Output Windows are created by Vensim's Analysis tools, and include graphs, tables and lists.
3. Control Windows include the Control Panel, a tab dialog box used to control Vensim's internal settings, and the Subscript Control, used for defining and selecting subscripts in Vensim Professional and DSS.

Moving Among Window Classes

When a window is first selected or created, that window moves to the top and is active while all other windows become inactive. You can only work in the active window. Four different methods allow you to move between the window classes:

1. Click on the appropriate window button on the Toolbar.
2. Press Ctrl + Shift + Tab to cycle among the window classes.
3. From the Windows menu, select Pop Build Forward, Pop Output Forward, Control Panel, or Subscript Control.
4. Use the mouse and click on the appropriate window (this only works if the window is visible).

The last method works particularly well for the Build Window, which is the largest and is not usually covered when other windows are active.

Moving Among Windows within a Class

There can be multiple output windows open and, in the more advanced configurations multiple build windows open. Four methods enable you to cycle through the open windows within a class:

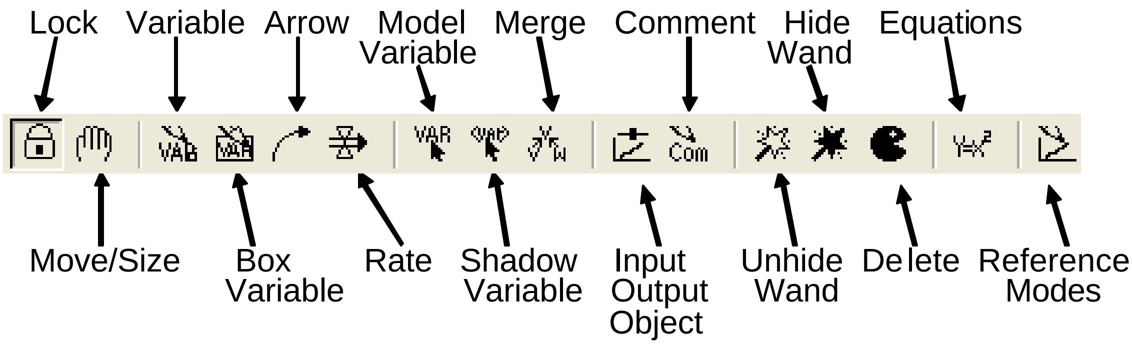
1. Click repeatedly on the window class button.
2. Press Ctrl + Tab.
3. From the **Windows** menu, select **Output Window List** for the Output windows, or click on the desired Build window showing at the bottom of the **Windows** menu.
4. Use the mouse and click on the appropriate window (this only works if the window is visible).

The Build Window

Build Windows are used to create models in Vensim. By default, they open with the Sketch tools for sketching the structure of the model and for writing equations. The Status Bar provides buttons for modifying the sketch. Except in PLE models can be built from several different sketches or sketch views. Each sketch view shows a part of the model, much like each page in a book tells part of a story. In Vensim Professional and DSS, the Build Window can be switched to a Text Editor for building and editing text-based models. The Status Bar then switches to a text-editing version.

Sketch Tools

Sketch tools are grouped into a Sketch toolset. Vensim PLE and , PLE Plus have only a built in Sketch toolset, but the other configurations allow you to choose and modify your Sketch toolsets by adding, moving and changing the actions of the different tools.. Customized toolsets can be saved to files and reopened for later use. The built in Sketch toolset (*default.sts*) contains most of the Sketch tools needed for building models.



Vensim PLE and PLE Plus do not contain the Model Variable, Merge, Unhide Wand or Hide Wand tools.

For the other configurations the Sketch tools can be configured by clicking with the right mouse button on the tool and changing its options. If you change a tool's configuration, you will be asked whether you want to save the Sketch toolset when exiting Vensim. Clicking **Yes** overwrites the old toolset. Clicking **No** keeps the old toolset (you will lose your changes). Clicking **Cancel** allows you to use the **Tools** menu to save the toolset with a new name before exiting. There is no limit to the number of toolsets you save, but most users find it simplest to use one sketch toolset configured to their needs.

The Sketch tools in the built in Sketch toolset are:

? **Lock** — sketch is locked. Pointer can select sketch objects and the Workbench Variable but cannot move sketch objects.

? **Move/Size —** move, sizes and selects sketch objects: variables, arrows, etc.

? **Variable —** creates variables (Constants, Auxiliaries and Data).

? **Box Variable —** create variables with a box shape (used for Levels or Stocks).

? **Arrow** — creates straight or curved arrows.

? **Rate** — creates Rate (or flow) construct, consisting of perpendicular arrows, a valve and, if necessary, sources and sinks (clouds).

? **Model Variable** — adds an existing model variable and the causes of that variable to the sketch view.

? **Shadow Variable** — adds an existing model variable to the sketch view as a shadow variable (without adding its causes).

? **Merge** — merges two variables into a single variable, merges Levels onto existing clouds, merges Arrows onto a variable to split an Arrow, and performs other operations.

? **Input Output Object**— adds input Sliders and output graphs and tables to the sketch.

? **Sketch Comment** — adds comments and pictures to the sketch.

? **Unhide Wand** — unhides (makes visible) variables in a sketch view.

? **Hide Wand** — hides variables in a sketch view.

? **Delete** — deletes structure, variables in the model, and comments in a sketch.

? **Equations —** creates and edits model equations using the Equation Editor.

? **Referemce Modes —** use to draw and edit reference models (Chapter 20).

To build a model, first select a Sketch tool by clicking on it with the mouse. You can also select a tool by pressing a character on the regular keyboard (*not* the numeric keypad). Use 1 for the first tool, 2 for the second and so on (0 is the 10th, Q the 11th, W the 12th and so on). Note that this only works when the Build window is active.

Move the mouse to the sketch view and click once with the left mouse button to apply the tool (for Arrows and Rates, first click once, then move the mouse and click once again).

NOTE Sketch tool selection is sticky. That is, the Sketch tool you chose stays active until you choose another — just keep applying it to the sketch.

Status Bar

The Status Bar shows the state of the sketch and objects in the sketch. The Status Bar contains buttons for changing the state of selected objects, and moving to another View.

A number of sketch attributes can be controlled from the Status Bar, including:

? Change characteristics on selected variables; font type, size, bold, italic, underline, strikethrough.

? Set the hide level.

? Variable color, box color, surround shape, text position, arrow color, arrow width, arrow polarity etc.

When using the Text Editor (Vensim Professional and DSS), the Status Bar changes to reflect text editing operations.

Simulating

In addition to building models, you can use the build window to perform simulation tasks. Most importantly, you can enter Simulation Setup mode and SyntheSim mode from the build window. In Simulation Setup mode all model Constants and Lookups will be highlighted. By clicking on them you can make temporary changes to the values to be used for a simulation. In SyntheSim mode each model Constant will have a slider attached that you can use to adjust values. More details can be found in Chapter 13. You enter Simulation Setup and SyntheSim mode by clicking on the associated icon in the Toolbar.

Output Windows

Output Windows are generated by clicking on an Analysis tool. The Analysis tools gather information from the model and display the information in a window as a diagram, graph, or text, depending on the particular tool. Dozens of these windows can be open simultaneously, and a particular window can be closed individually by clicking the **Close** button in the top left or top right corner, or all windows can be closed at once using the menu item **Windows>Close All Output**.

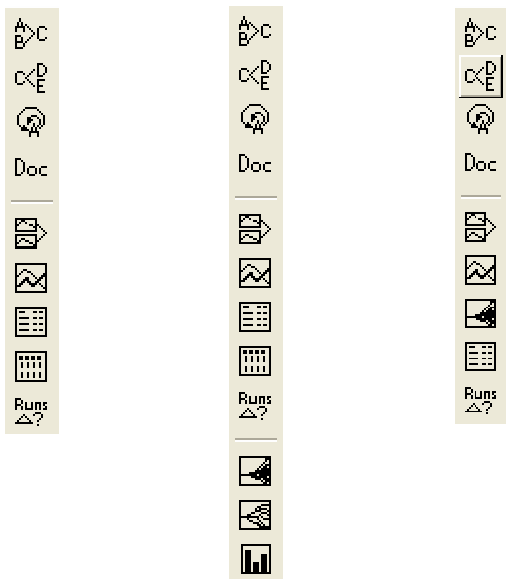
Analysis Tools

The Analysis tools are used to show information about the Workbench Variable, either its place or value in the model, or its behavior from simulation datasets. Analysis tools are grouped into toolsets. In Vensim PLE and PLE Plus you can only use the built in toolset. In the other configurations the Analysis toolsets can be modified The built in toolset (also named *default1.vts*) and the more complete toolset *default2.vts* contain many of the Analysis tools needed for investigating models.

Except in Vensim PLE and PLE Plus the Analysis tools can be configured to show different things about the Workbench Variable. To configure a tool, click on the tool with the right mouse button and change its options. Tools can also be added to a toolset. As with Sketch toolsets, if you make changes you will be prompted to save the toolset when exiting Vensim. Several different Analysis toolsets are supplied with Vensim, and can be opened from the menu **Tools>Analysis Toolset>Open**.

The following toolsets are built in.

PLE/Reader   Plus   DSS/Pro/Std



A description of the function of the tools follows below. The tools shown below are from the toolset *default2.vts* which contains more tools than the built in toolset. In Vensim PLE and PLE Plus only the built in tools shown above are available. The Statistics tool and Text Editor tool are not available in Vensim Standard and the Venapp Editor tool is only available in Vensim DSS.

Note that you can use Ctrl+1 to activate the first too, Ctrl+2 the second and so on. This will work for the first 10 tools (Ctrl+0 is the tenth).

Structural Analysis Tools

 **Causes Tree —** creates a tree-type graphical representation showing the causes of the Workbench Variable.

 **Uses Tree —** create a tree-type graphical representation showing the uses of the Workbench Variable.

 **Loops —** displays a list of all feedback loops passing through the Workbench Variable.

Document — reviews equations, definitions, units of measure, and selected values for the Workbench Variable.

Dataset Analysis Tools

 **Causes Strip Graph —** displays simple graphs in a strip, allowing you to trace causality by showing the direct causes (as shown) of the Workbench Variable.

 **Graph —** displays behavior in a larger graph than the **Strip Graph**, and contains different options for output than the **Strip Graph**.

 **Sensitivity Graph —** creates a sensitivity graph of one variable and its range of uncertainty generated from sensitivity testing.

 **Bar Graph —** creates a bar graph of a variable at a specific time, or displays a histogram of variables over all times or across sensitivity simulations at a time.

 **Table —** generates a table of values for the Workbench Variable.

 **Table Running Down —** table with time running down.

Runs Compare — compares all Lookups and Constants in the first loaded dataset to those in the second loaded dataset.

 **Statistics —** provides summary statistics on the Workbench Variable and its causes or uses.

Other Tools

 **Units Check** — provides an alternative way to access the units checking feature.

 **Equation Editor —** provides an alternative way to access the equation for the Workbench Variable.

Venapp Editor‡ — supports the visual editing of Venapps.

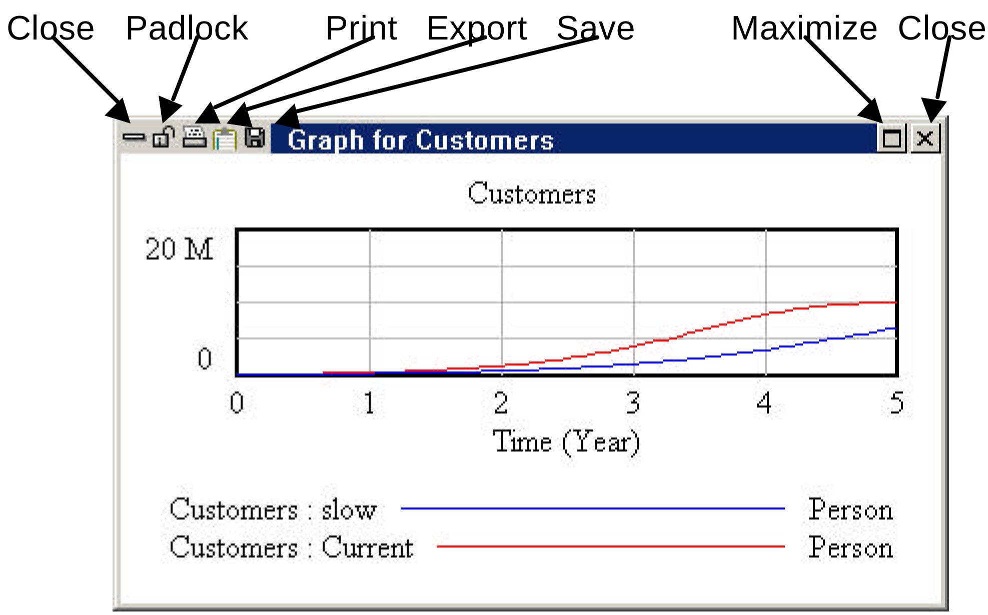
 **Text Editor —** a general purpose text editor. As shown, it is configured to edit *.vgd* files).

The **Tree Diagram**, the **Strip Graph**, the **Sensitivity Graph**, the **Table** and the **Statistics** tools can all be configured to show either causes or uses of the Workbench Variable.

Analysis Tool Output

Clicking on an Analysis tool generates a new window with formatted output, except for the Table and Document tools, which add information to any existing Table or Document Output window. The output of a tool remains on screen until you remove it, and *is not* updated as changes are made to the model. The only exception to this is tool output that is imbedded into a sketch using the Input Output object as described in Chapter 12.

An example of Analysis tool output is the graph displayed below. Descriptions of the buttons common to all Output windows are given below.



? If you change a model or make a new dataset, you can delete the old output easily and quickly by clicking one of the **Close** buttons located in the top left or top right corner or pressing the **Del** key.

? You can delete all Output windows by selecting the menu item **Windows>Close All Output**.

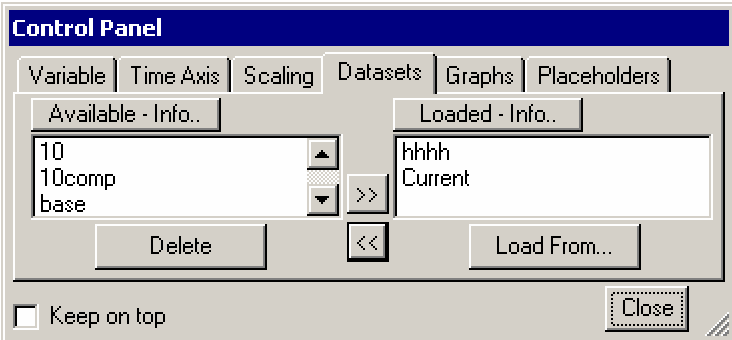
? You can prevent an Output window from being closed by clicking on the **Padlock** button in the top left-hand corner to lock the window. Clicking on the **Padlock** again will unlock the window.

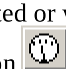
? You can permanently save information in an Output window by either clicking the **Save** button (to save to a file) or the **Export** button (to save to the clipboard for pasting into another application) while the Output window is active.

? If you remove the output, you can reproduce it easily by invoking the tool that generated it again (unless you have changed things in the model or set special Constant or Lookup table values).

Analysis tool output is easy to create and easy to get rid of. Analysis tools do not create information, but put existing information into a more useful and digestible form.

The Control Panel



The Control Panel allows you to change internal settings that govern the operation of Vensim, such as which Workbench Variable is selected or what Datasets are loaded. Open the Control Panel by clicking on the **Control Panel** button  on the Toolbar or by selecting the menu item **Windows>Control Panel**. The Control Panel groups controls in six tabbed folders (five in Vensim PLE and PLE Plus). Select a particular control by clicking on the appropriate tab at the top of the window.

? **Variable** allows you to choose a variable in your model and select it as the Workbench Variable.

? **Time Axis** allows you to change or focus the period of time over which Analysis tools operate.

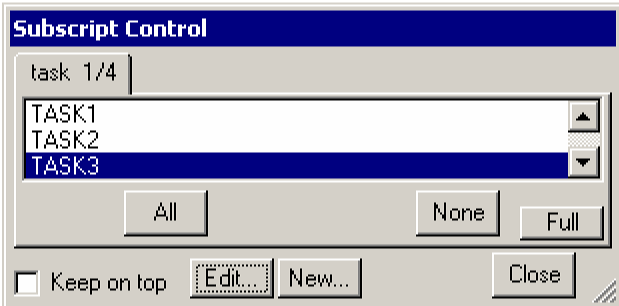
? **Scaling** enables you to change the scales of output graphs.

? **Datasets** allows you to manipulate the stored datasets (runs).

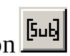
? **Graphs** brings up the Custom Graph Control.

? **Placeholders** is a control that sets Placeholder Values (Not PLE or PLE Plus, see the Reference Manual).

Subscript Control



The Subscript Control is used in Vensim Professional and DSS to create, edit and select elements for Subscript Ranges in order to focus the operation of the Analysis tools. For example, for a model with variables subscripted by task, selecting just one task (*TASK3*) will focus the Analysis tools to show information only about the one task (*TASK3*). Subscripts are selected and deselected by clicking on them, or by clicking the buttons **All** or **None**.

To open the Subscript Control click on the **Subscript** Control button  on the toolbar. You can define new Subscript Ranges by clicking on the **New...** button and edit the Subscript Range for the current tab by clicking on the **Edit...** button.

3 A Hands-On Example

Modeling with Vensim

The following steps are typical for building and using Vensim models.

- ? Construct a model or open an existing model.
- ? Examine the structure using the structural Analysis tools (Tree Diagrams.).
- ? Simulate the model moving around model parameters to see how it responds.
- ? Examine interesting behavior in more detail using the dataset Analysis tools (Graphs and Tables).
- ? Perform controlled simulation experiments and refine the model.
- ? Present the model and its behavior to your audience using SyntheSim results, Analysis tool output customer Graphs and Tables.

Constructing, examining, and modifying models should follow an iterative approach. Starting from simple models with few feedback loops and little detail allows the quick construction of a working simulation model. The working model can then be modified and improved as necessary to show the desired level of detail and complexity.

Vensim has a unique approach to displaying simulation output, allowing you to instantly see simulation results for all variables on the screen. During simulation, dynamic behavior is stored for all variables in the model. You can select the any variable of interest and click on the appropriate Analysis tool to display more detailed results.

The Workforce Inventory Example

In this Chapter you will work through the mechanics of using Vensim with a workforce inventory model. This is a simple, but quite valuable model to study. It demonstrates how the interaction of inventory management policies and hiring practices can lead to instabilities in production. It also demonstrates the somewhat counterintuitive result that being more aggressive about hiring and laying people off can actually lead to a more stable workforce. The workforce inventory model is developed in Chapter 2 of the Modeling Guide.

Starting Vensim

Windows (95, 98, NT 4.0, 2000 or XP)

- ✍ Click on the **Start** button then **Programs>Vensim>Vensim XXX** (where XXX is one of PLE, PLE Plus, Professional or DSS).

Macintosh

✍ Double click on the **Vensim icon**.

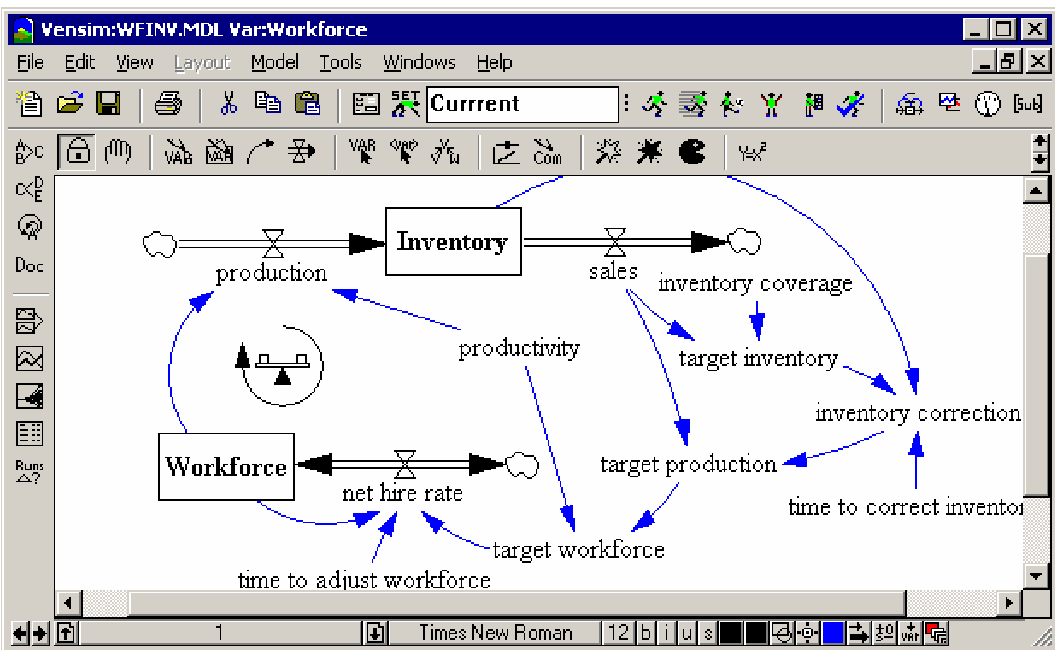
Vensim will open with a new (empty) model, or the last model you were working with. We could start developing our model here, but instead we will open and simulate an existing model.

Opening the Model

✍ Select the menu item **File>Open Model…**, *or* click on the **Open Model** button  on the Toolbar.

✍ Open the model *wfinv.mdl* located in the directory *guide\chap03* (normally the full path is *c:\Program Files\Vensim\models\guide\chap03* but we will omit the earlier part).

Vensim will load the Workforce/Inventory model and the screen should appear as below.



This model describes the dynamic behavior of a manufacturing plant that carries inventory. The Title Bar displays the model that is loaded (*wfinv.mdl*) and the Workbench Variable (`Workforce`). We can see that the variable `Workforce` also appears in the sketch. The Workbench Variable is any variable in the model that we are currently interested in focusing on. We can change the focus any time we want by clicking on another variable.

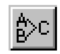
✍ The Sketch **Lock** tool should be selected by default. Place the mouse cursor over the box in the sketch that says `Inventory`, then click on it.

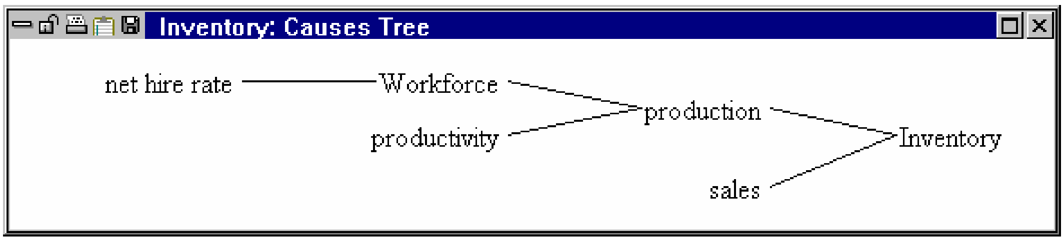
We see that the Workbench Variable (on the Title Bar) changes from `Workforce` to `Inventory`.

Examining Structure

The workforce/inventory model presented is relatively simple, although it may look confusing if you are not familiar with the conventions of stock and flow diagramming. In this visual representation, arrows imply cause and effect: the variable at the tail of the arrow causes the variable at the head of the arrow (to change). For example, `production` is caused by `Workforce` and also by `productivity`.

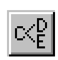
We can investigate the structure of this model with the structural Analysis tools. We will get answers only about structure, not about dynamic behavior of the model (that comes next when we simulate the model and use the dataset Analysis tools).

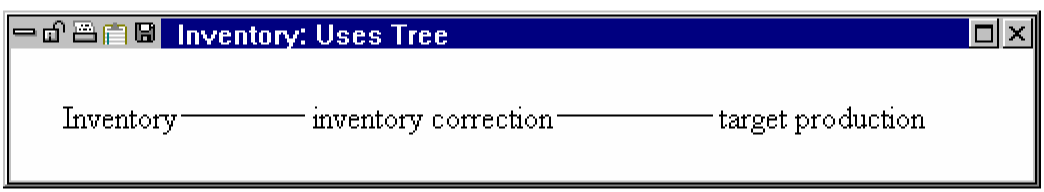
✍ Click on the top Analysis tool, the **Causes Tree Diagram**  and an Output window opens:



We see that the Workbench Variable, `Inventory`, is on the right and everything that causes it to change (up to 2 connections distant) is on the left.

✍ Click the **Close** button  in the upper left corner, or the **Close** button  in the upper right corner, or press the Del key, to close the Tree Diagram.

✍ Click on the **Uses Tree Diagram** Analysis tool  and an Output window opens:

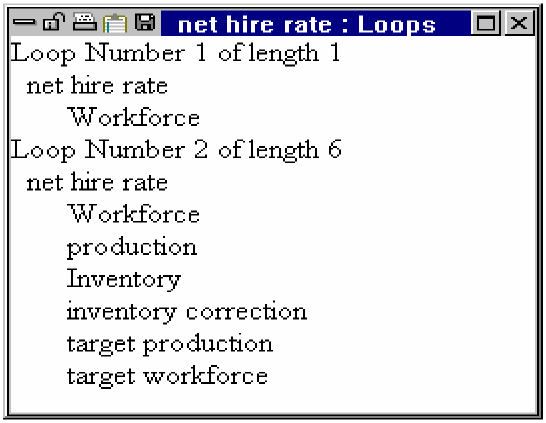


Now we can see the Workbench Variable on the left and where it is used in the model (what *it* causes to change, up to 2 connections distant) on the right. Note that these Tree Diagrams simply present information from the model in a different manner. We can observe all the causal connections by examining the sketch, but trees present only part of a model and can be easier to understand.

✍ Click the **Close** button or press the Del key to close the Tree Diagram.

✍ Place the mouse cursor pointer on the variable `net hire rate` that appears in the sketch, and then click to select it as the Workbench Variable.

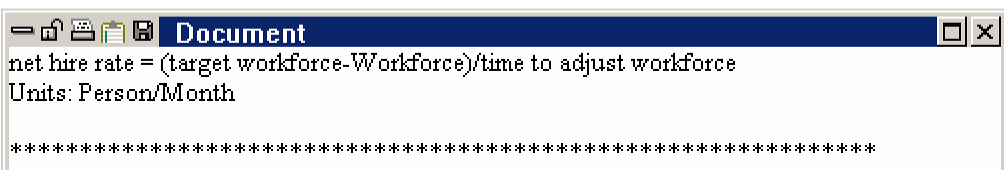
✍ Click on the **Loops** Analysis tool 



An Output window opens that displays all variables in all feedback loops (two) that pass through the Workbench Variable (*`net hire rate`*).

✍ Click on the **Document** Analysis tool 

The output of this tool will depend on the configuration of Vensim you are using. For Vensim PLE and PLE Plus the document tool provides documentation on the entire model, displaying all model equations in a simple text format. For the other configurations this information is displayed only for the current Workbench variable.



An Output window opens that displays the equation underlying the Workbench Variable (*`net hire rate`*) and the units of measurement.

✍ Select the menu item Windows>Close All Output.

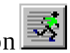
This closes all the Output windows that have been created.

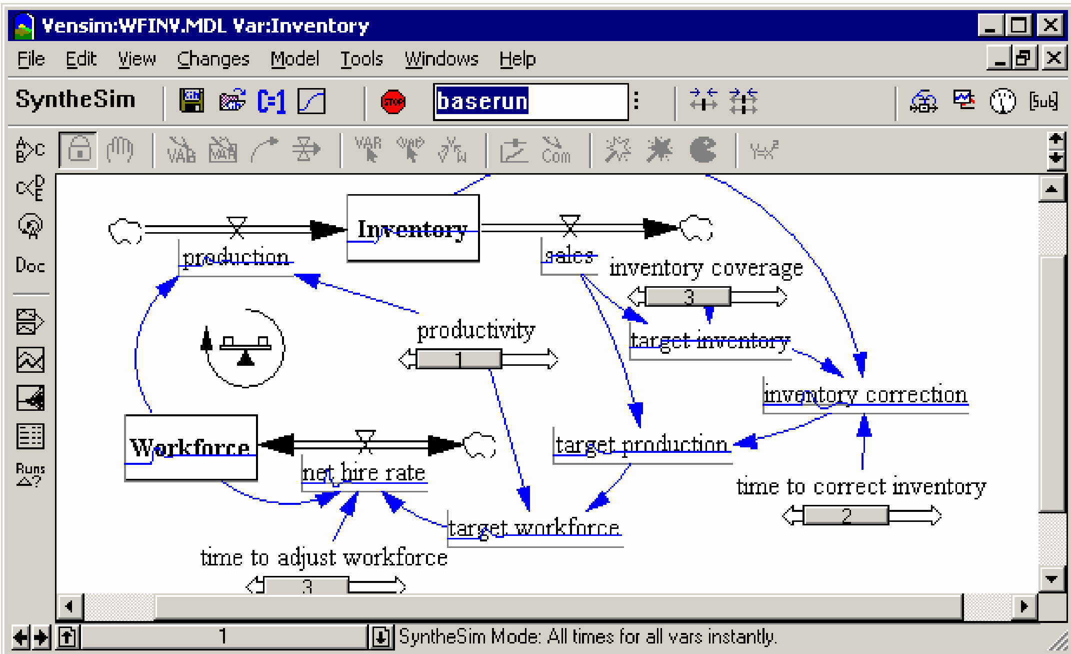
Simulating the Model

Now we would like to examine the dynamic behavior of the model. We want to look at the behavior of variables in the model, such as the amount of *`Inventory`* over time. To do this, first we need to simulate the model. The easiest way to simulate models is using the Toolbar. To access some of the more advanced options for setting up simulations, you can use the Simulation Control as described in Chapter 8 of the Reference Manual.

✍ Double click on the simulation **Runname** editing box on the Toolbar 

to highlight the default name *Current* (or click once and drag over the name *Current*), then type in the name *baserun*. This is the name of the dataset that holds all the simulation output values for behavior of variables when we make a simulation run.

✍ Click on the **SyntheSim** button  on the toolbar. Vensim will change to SyntheSim mode, and you will see:



For each variable there is either a graph superimposed on top or a slider placed below. The sliders are attached to Constants (variables that take on a single value for all times) and thumbnail time graphs are shown for the remaining variables. If you position the mouse over a variable name and wait a larger popup graph will appear.

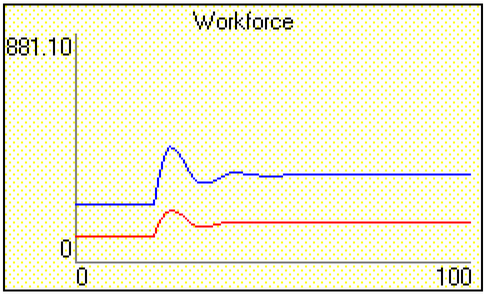
✍ Double click on the simulation **Runname** editing box on the Toolbar again and replace the name *baserun* with *experiment*. Now when you make changes they will be recorded in the dataset *experiment* while the dataset *baserun* will remain unchanged.

✍ Using the mouse drag the slider below `productivity` back and forth. As you move the slider the model will simulate and the results will display in blue, with the results from *baserun* being shown in red. The simulation is done quickly enough that the graphs should update almost instantly.

Examining Behavior

While the graphs are displayed for all the model variables it is often useful to get more detailed, and bigger, output.

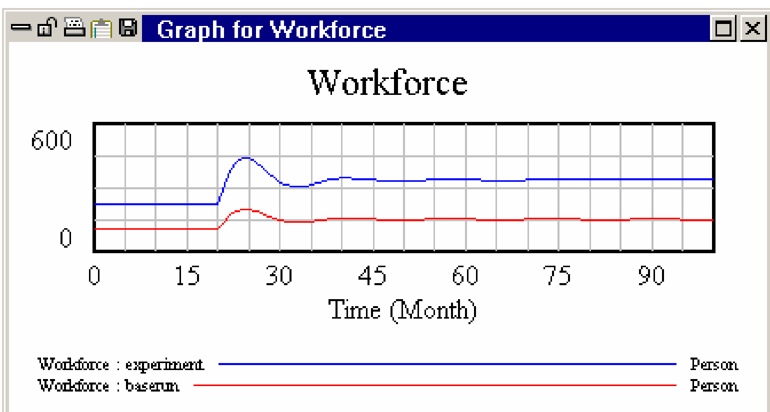
✍ Position the mouse over `Workforce` and leave it there. A graph should pop up just below workforce.



This is the same as the graph shown on top of the variable except that it is bigger, and has labels on the axis. The graphs are made the same so that they can easily be related to the thumbnail graphs showing on the diagram.

✍ Click on `Workforce` to select it into the Workbench.

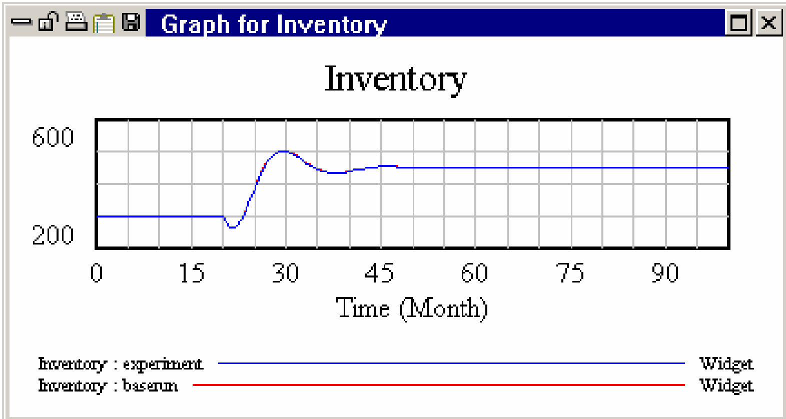
✍ Click on the **Graph** Analysis tool 



Two things are worth noting about the above graph. First both runs show a pattern of behavior known as damped oscillation. Second, the two runs are identical except for scaling.

✍ Close the graph by clicking on the **Close** button or pressing Del.

✍ Click on the variable `Inventory` appearing in the sketch and then click on the **Graph** tool 



We see a graph of `Inventory` with oscillating behavior similar to `workforce`, although `Inventory` starts out by declining before increasing in value. More importantly, there is only one graph visible. Let's look at a table of the actual values for inventory.

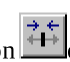
✍ Click on the **Table** tool 

Table

Time (Month)	20	21	22	23
"Inventory" Runs:	experiment	baserun		
Inventory	300	265.25	266.80	294.93
: baserun	300	265.25	266.80	294.93

✍ Use the bottom scrollbar of the Output window (or the left and right arrow keys) to look through the values for `Inventory`.

Only one graph line is visible for `Inventory` because the values are identical for both runs. Changing `productivity` impacts only `Workforce`, `target workforce` and `net hire rate`. This is quite clear just by looking at the model diagram as you drag the slider and occurs because `productivity` really just scales the number of people required to produce one item.

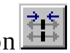
✍ Click on the **Reset Slider** button  or press the **Home** key to return the value of `productivity` to its original setting.

✍ Repeat the above experimentation process with each of the three remaining constants. Drag the sliders observing behavior and then bring up more detailed graphs when you see something interesting.

The things to be looking for when you are evaluating behavior are the period of oscillation, the extent by which variables change and the degree of damping. The period of oscillation is the time from one peak to the next on the time graphs. Damping is the decrease in amplitude that occurs from peak to peak, where amplitude is the distance along the y axis from the eventual value the variable settles to. It should be easy to see that `productivity` and `inventory coverage` do not significantly change these while `time to adjust workforce` and `time to correct inventory` do.

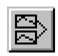
Causal Tracing

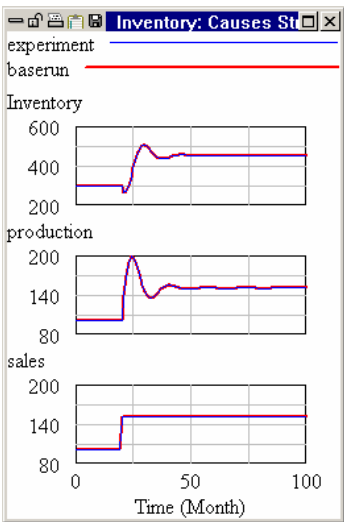
Just as you looked at the causes of `Inventory` by using the **Causes Tree Diagram** Analysis tool you can also look at graphs of behavior of the variables that cause `Inventory` to change.

✍ Click on the **Reset All** button  or use the key combination Ctrl+Home.

This resets all model constant to their original value. You will see only a single graph line on each variable.

✍ Click on `Inventory` to select it into the workbench.

✍ Click on the **Causes Strip** Analysis tool 



A strip graph is generated that shows the Workbench Variable (`Inventory`) at the top, and all the variables that directly cause `Inventory` to change below it (`production` and `sales`). The two runs are currently the same, so the graph lines lie on top of one another. You should be able to see two colors in each graph line, the **Causes Strip** tool uses both color and line thickness to distinguish runs.

Notice something very interesting in this graph. `Inventory` has oscillating behavior which then is damped out and becomes stable. `Inventory` is being changed by both `production` and `sales` but *only* `production` is oscillating. `Sales` does not have the oscillating pattern of behavior contained in `Inventory` and `production`. Therefore we will look into `production` and not `sales` to understand the source of this oscillation.

Causal Tracing is a quick and powerful tool that helps us determine *what* portions of a model are causing *which* types of behavior. The **Causes** and **Uses Tree Diagrams** and the **Table** tool can all be used for Causal Tracing but the most commonly used tool is the **Causes Strip** tool and we will use that to investigate the sources of oscillation in this model.

Lets find out which feedback loops in the model are causing the oscillating behavior.

✍ Click on `production` appearing in the **Causes Strip** to select it as the Workbench Variable, then click on the **Causes Strip** tool.

✍ Click on `Workforce` in the **Causes Strip** that has just been displayed, then click on the **Causes Strip** tool.

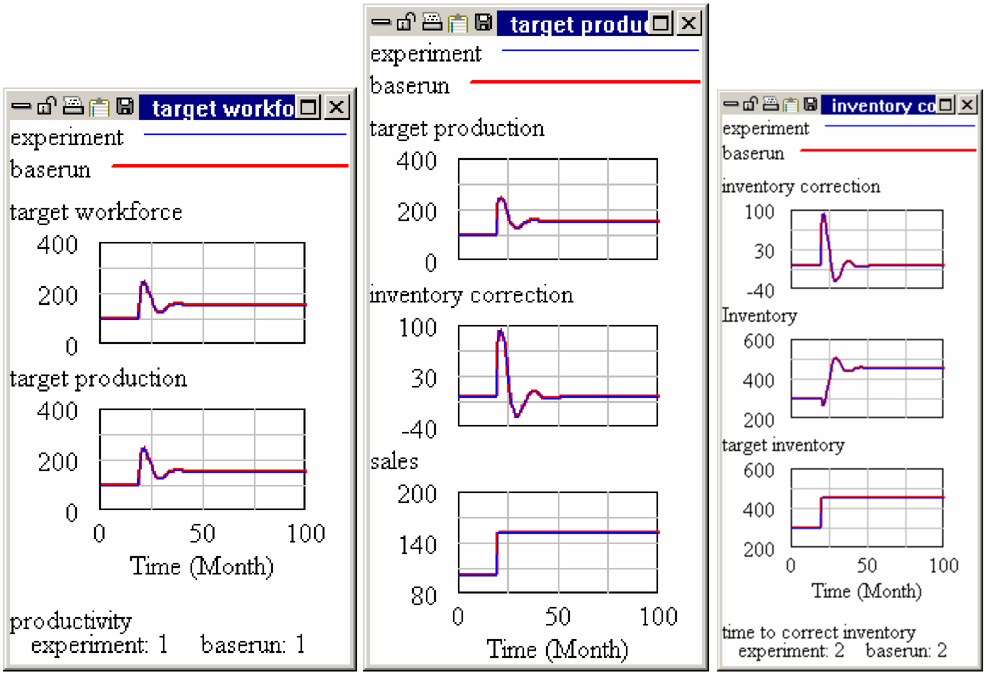
✍ Click on `net hire rate` then click on the **Causes Strip** tool.

The three strip graphs are displayed below. Note how the oscillation is traveling through all these variables.

✍ Click on `target workforce` then click on the **Causes Strip** tool.

✍ Click on `target production` then click on the **Causes Strip** tool.

✍ Click on `inventory correction` then click on the **Causes Strip** tool.



The last two graphs show similar behavior. The **Causes Strip** for `target production` shows that the oscillation is coming from `inventory correction`, not from `sales`. In the `inventory correction` graph, we see that `Inventory` is causing the oscillation, not `target inventory`.

We know that the oscillations follow a path back to `Inventory` and do not go through the variable `sales`. Let's look back at the sketch to get a feel for what is happening.

✍ Click on the **Build Windows** button  on the Toolbar.

This brings the Build window to the front and pushes the Output windows to the back.

With your eyes, trace the feedback loop that the oscillations have followed, from `Inventory` to `production` to `Workforce` to `net hire rate` to `target workforce` to `target production` to `inventory correction` and back to `Inventory`.

Look at the variable `target production`. Note how the oscillations travel through the feedback loop to `Inventory`, not through `sales`. The variable `sales` is a Constant with a STEP function. `sales` causes other variables to change, but nothing causes it to change. `sales` is not part of any feedback loop. The variable `sales` imparts the sudden change to the Level `Inventory` (through a step increase in `sales`). The system structure (the negative feedback loop) then tries to correct `Inventory` and sets up the oscillation at some particular frequency. This is very much like a rocking chair that will rock back and forth in response to a push in one direction.

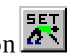
✍ Select the menu item Windows>Close All Output.

Individual Simulation Experiments

So far you have used the SyntheSim capabilities of Vensim to explore model behavior and this is an extremely efficient way to gain insights. There is a more traditional way to explore behavior, and this involves a setup step followed by a simulation step for each simulation made. This approach has the advantage of being very methodical, so that results are easily replicated by other people. This is also the only practical approach to dealing with very large models for which take more than a few seconds to simulate.

✍ Click on the **Stop Simulating** button  on the Toolbar.

The graphs and sliders will disappear. Vensim is now in the same state it was in when you first opened the model.

✍ Click on the **Set Up Simulation** button  on the Toolbar.

Some of the variable names in the sketch will appear with yellow text on a blue background. These are Constant variables that do not change during simulation; we can set them to a different value before we simulate and see the effect the changes have on model behavior.

✍ Click on the variable ***`time to adjust workforce`*** that appears yellow/blue on the sketch.

An editing box will open: 

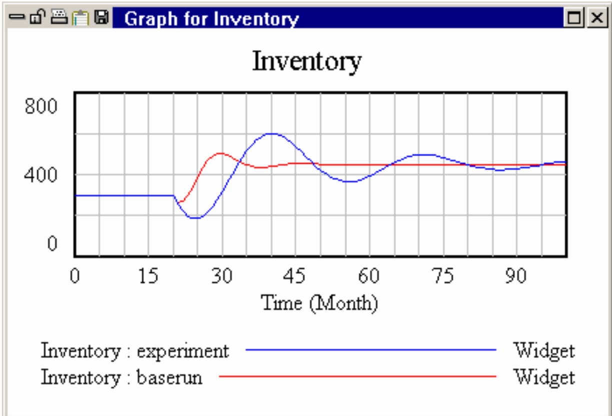
We will try an experiment where we slow the rate at which we hire new workers (and layoff current workers), to see if that removes the oscillation. Ideally we would like to see a smooth increase from our old inventory (and workforce) levels to the new levels.

✍ Type the number 12 into the editing box to replace the number 3, then press the Enter key.

This will change the ***`time to adjust workforce`*** from 3 months to 12 months.

✍ Click on the **Simulate** button  on the Toolbar, the model will simulate and store the values for the dataset *experiment*.

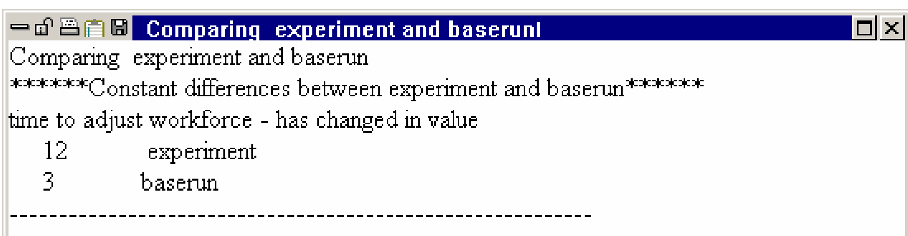
✍ Click on the variable ***`Inventory`***, then click on the **Graph** Analysis tool 



Here we see the results of two experiments: *baserun* with the original value (3) for the variable ***`time to adjust workforce`***, and *experiment* with the modified value (12) for ***`time to adjust workforce`***. The results show that slower hiring and firing practices actually increase the size of the oscillation, and make the oscillations last longer.

To see what the differences in the constants were for each run:

✍ Click on the **Runs Compare** Analysis tool 

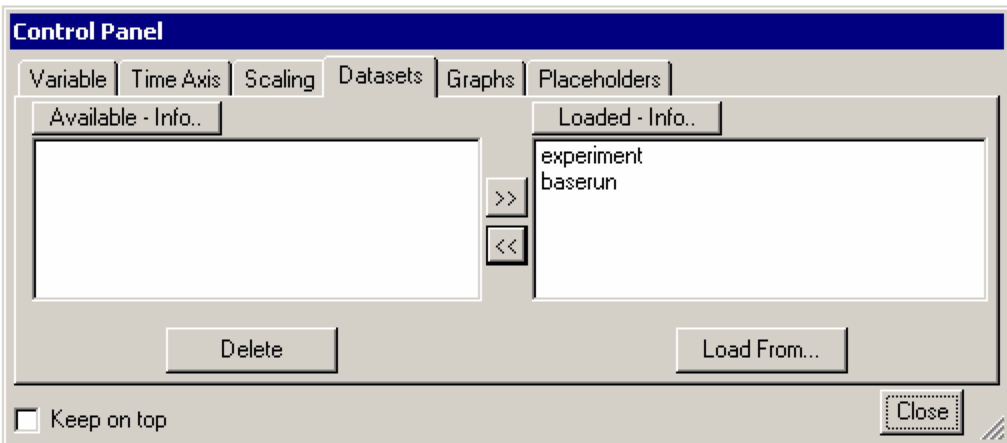


The **Runs Compare** tool lists all Constant and Lookup differences in the first two loaded simulation datasets. We have two datasets loaded (*baserun* and *experiment*) and the only difference is the value for the variable ***`time to adjust workforce`*** (3 and 12).

Making a Custom Graph

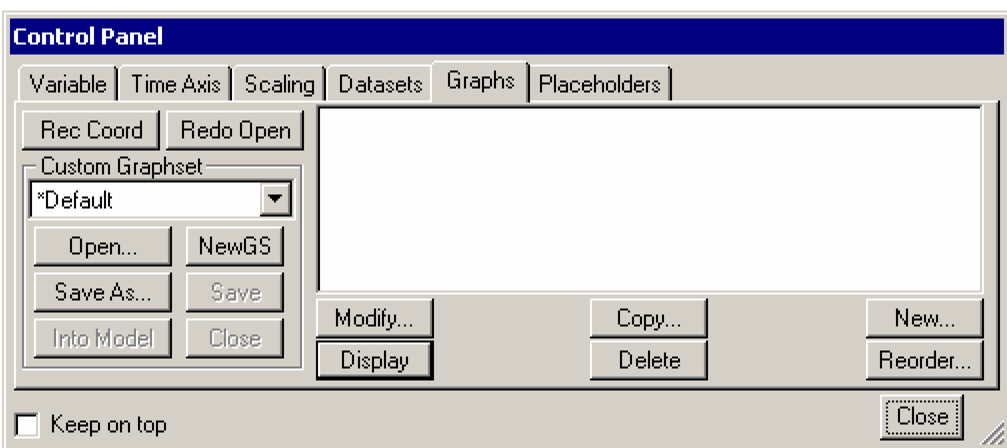
Sometimes you will want to see all of the important variables together in one graph. Graphs generated using the Analysis tools display behavior for the Workbench Variable. Using Custom Graphs, you can display the desired variables, dataset runs, style and formatting in one graph. **Custom Graphs** are created from the Graph Control located in the Control Panel.

✍ Click on the **Control Panel** button  on the Toolbar to select the Control Panel. Click the tab for **Datasets**.

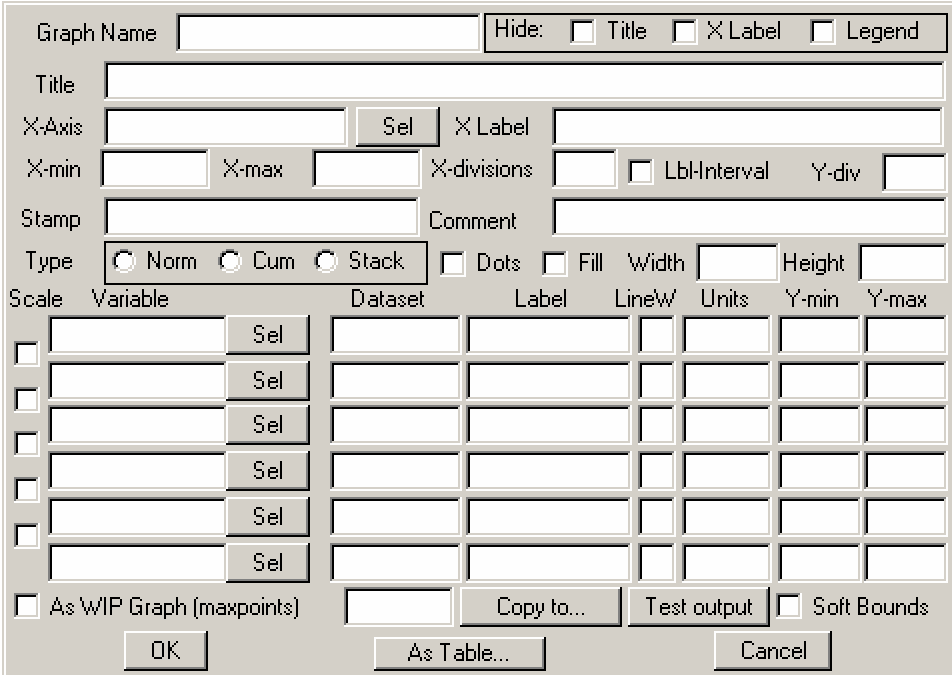


✍ Unload *baserun* by double clicking on the run name baserun in the **Loaded** runs box.

✍ Click on the tab **Graphs** in the Control Panel.

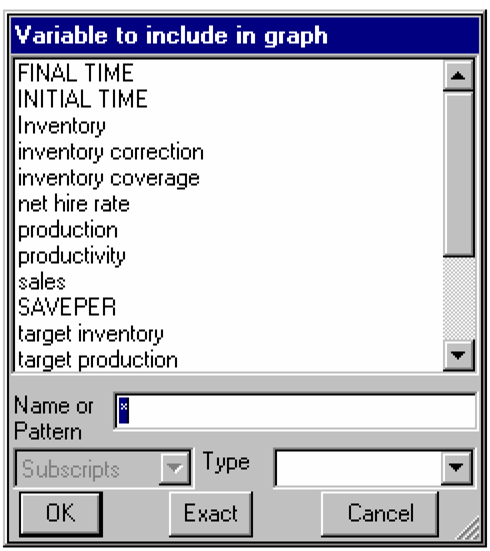


✍ Click the button **New…**. The Custom Graph Editor opens with the cursor positioned at the graph **Title** editing box.



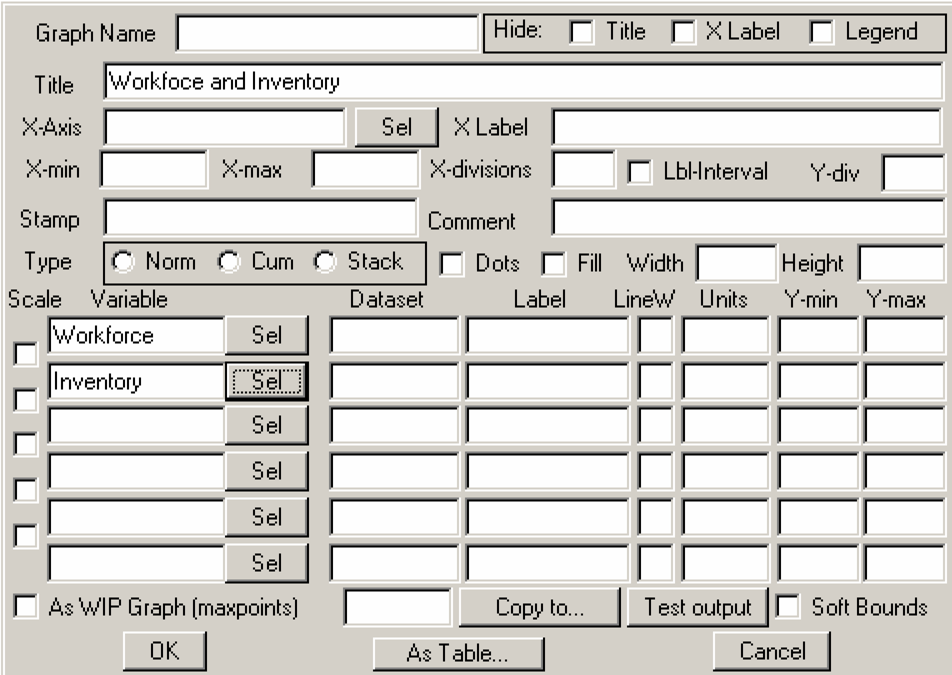
✍ Type the name Workforce and Inventory into the **Title** editing box.

✍ Using the mouse, move to the **Variable** boxes on the left side of the graph editor and click on the top button labeled **Sel**. A variable selection dialog box appears.

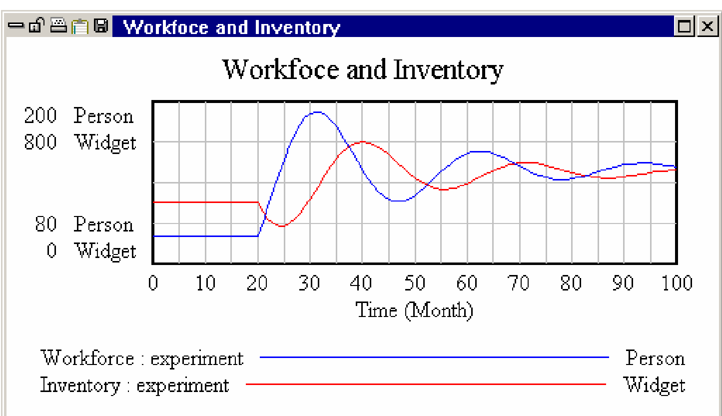


✍ Move the scrollbar down the list and double click on `Workforce`.

- Using the mouse, click on the second button down labeled **Sel**. A variable selection dialog box appears, move the scrollbar down the list and double click on `Inventory` (or single click and click **OK** to close the variable selection dialog).

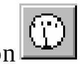


- Click the **OK** button to close the Custom Graph Editor.
- Click the button **Display** in the Graph Control to show the Custom Graph.



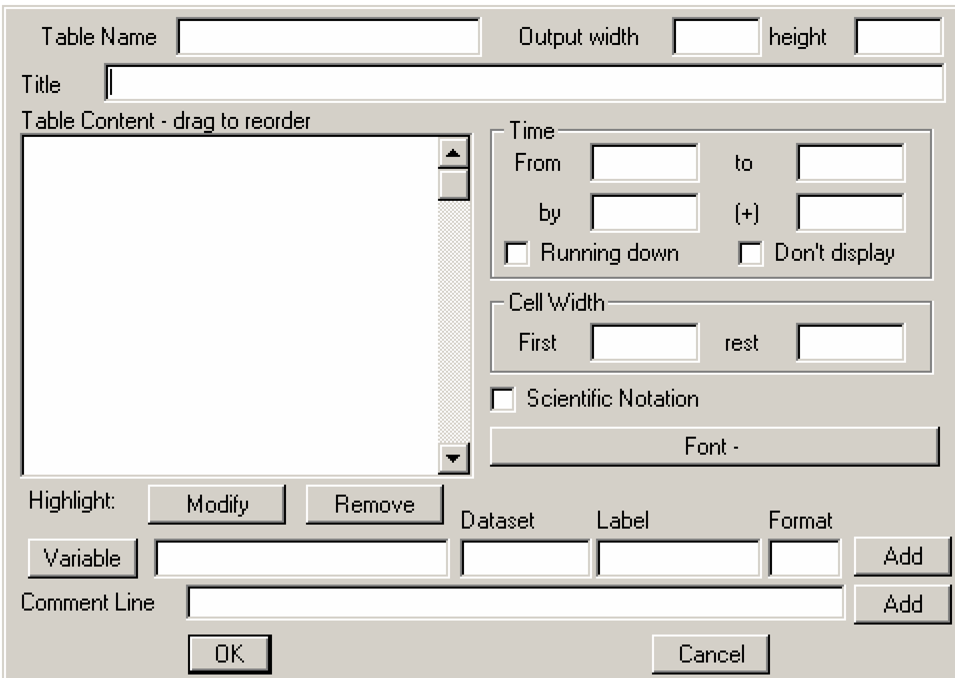
Making a Custom Table

Custom Tables allow you to look at tabular output from different variables at different times.

✍ Click on the **Control Panel** button  on the Toolbar to select the Control Panel. Click the tab for **Graphs**.

✍ Click the **New** button.

✍ In the graph dialog that opens click on the **As Table...** button at the bottom. The Custom Table Editor will open.

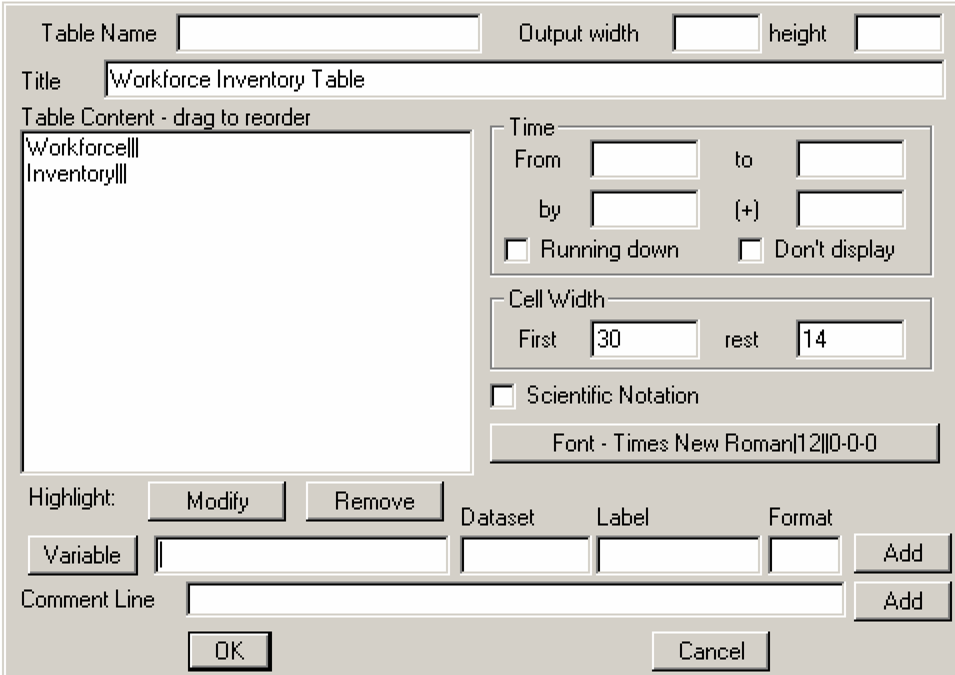


✍ Type in the title Workforce Inventory Table.

✍ Click on the **Variable** button and select *`Workforce`* then click on the **Add** button at the right.

✍ Click on the **Variable** button and select *`Inventory`* then click on the **Add** button at the right.

The dialog should look like:

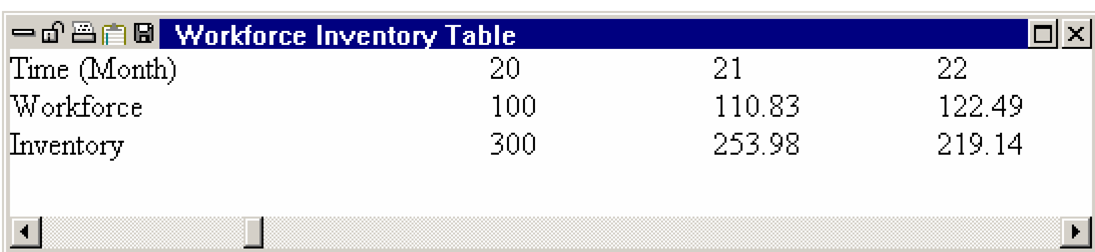


✍ Click on **OK**

The list of available graphs and tables will be updated. Notice that the name will appear as WORKFORCE_INVENTORY_TABL - the final E is truncated. You can also type in the name you would like to see in the list in the **Table Name** editing box.

✍ In the control panel double click **WORKFORCE_INVENTORY_TABL** in the list.

You will see the output



Summary

You have now worked through the use of a very simple model in Vensim. The techniques used to do this are the foundation of model analysis with Vensim. Even with very complicated models these analysis tools have tremendous power to help you understand and debug the models you are working on. The next six chapters focus on techniques for building models. After that we will return to more analysis and reporting topics that build from the basics covered in this Chapter.

4 Causal Loop Diagramming

Vensim Models

This chapter describes causal loop. Causal loop diagrams are called that because each link has a causal interpretation. An arrow going from A to B indicates that A causes B. Causal loop diagrams can be very helpful in conceptualizing and communicating structures. Many people find causal loop diagramming to be very helpful even when no simulation model is created, while others feel they can be harmful if done in isolation. This chapter is primarily about technique, and is useful to work through even if you choose not to build any additional causal loop diagrams. Causal loop diagrams are also often called influence diagrams.

Causal loop diagrams do not show accumulations (levels or stocks) in a system. Construction of stock and flow diagrams is covered in Chapter 5. However, even if you intend to build only stock and flow diagrams, we recommend that you start with this chapter as many of the basic drawing mechanics are the same and they are covered in more detail here.

It is important to note that causal loop diagrams and stock and flow diagrams are not simulation models. Simulation models, like the one used in Chapter 3, attach algebraic relationships to all the variables appearing in a diagram. In Chapter 6, "Building a Simulation Model" we describe how to create a simulation model. If you are using Vensim Professional or DSS you can skip building model diagrams altogether enter equations directly. Almost all people, however, find it easier to build up models diagrammatically.

Drawing Sketches

When a Sketch Tool is selected, that tool remains active until you select another tool. A single click (press and release) with the mouse button applies the tool to the sketch.

The **Lock** tool  provides the standard mouse cursor. The Lock tool can be used for selecting sketch objects (they highlight black) and for changing options. Sketch objects cannot be moved with the Lock tool. Tip — you can select the Lock tool by pressing the Esc key, or the keyboard number 1.

The **Move/Size** tool  is used for moving sketch objects around, including resizing variables and boxes, and reshaping arrows. The other sketch tools also allow you to move objects.

Variable sketch tools (**Variable**  **Box Variable**  and other variable tools you may configure) and the default setting for the **Rates** tool  bring up editing boxes (for naming the Variable or Rate) when applied to the sketch. The **Sketch Comment** tool  brings up a dialog box.

The **Arrow** tool  starts an arrow. To do this make a single click (press and release) of the mouse button on the middle of the starting word, then finish with another single click on the middle of the ending word. Arrows (curved) can take one intermediate point on a sketch with an extra mouse click.

NOTE Do not try to draw arrows by clicking and holding the mouse button down while dragging the mouse. This will just move the word you are starting from. The same applies for rates.

Objects in a sketch can have their appearance changed by clicking on them with the right mouse button, which brings up an options dialog box.

Mouse Tips

? If a mouse button click is called for without mention of left or right, use the left button (Macintosh, use the only button).

? If a mouse right button click is called for, with the Macintosh, hold down the Control key or the Apple key and click (Ctrl + Click or ✍ + Click).

Constructing a Causal Loop Diagram

This section describes the construction of a causal loop diagram of a construction project. A central concept is the amount of work to do that is left in the project. Much of our diagram will center on this concept. First we will construct a diagram with one view that describes some essential elements of getting a project done. A view is a single sketch of your model, like a single page of a book. Your model can contain multiple views. Later, we will add another view that incorporates more of our knowledge of the system.

Project Model (project.mdl)

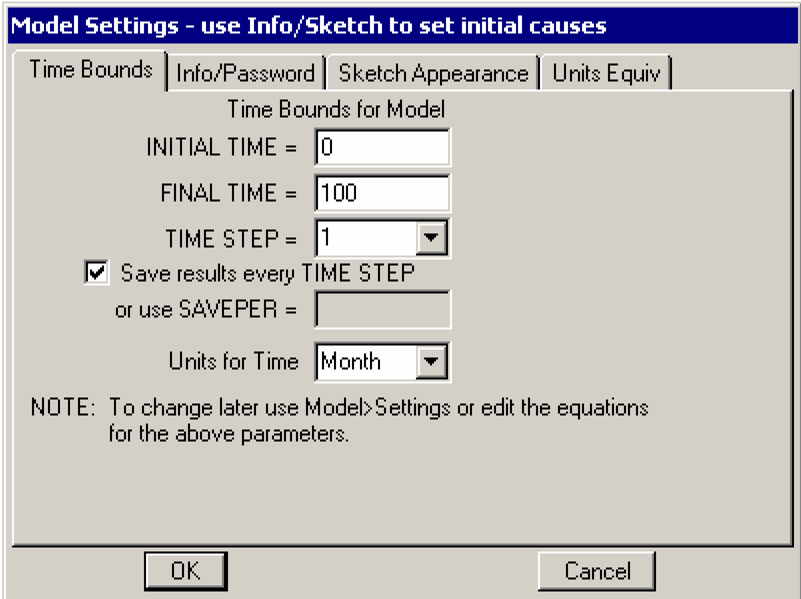
This model describes the competing feedback loops in a project (think of it as a construction project though the concepts are generic to all projects). The causal loops show the relationship between the amount of *`Work To Do`*, *`overtime hours required`*, and the effect of overtime on both the amount of *`work done`* and also *`fatigue`*. The first cut at this model assumes a constant size of workforce.

✍ Start Vensim

Vensim will open with the last model you worked on active.

✍ Select the menu item **File>New Model** or click the **New Model** button on the Toolbar.

The Model Settings dialog box opens:



✍ Click **OK** to accept the default values

A causal loop diagram does not use the Time Bounds, but a simulation model needs Time Bounds. All Vensim models (including diagrams) have Time Bounds even though they may not be used.

✍ Click the **Save** button on the Toolbar. Select the directory *guide\chap04* then type in the name *project* and click the **Save** button in the dialog box.

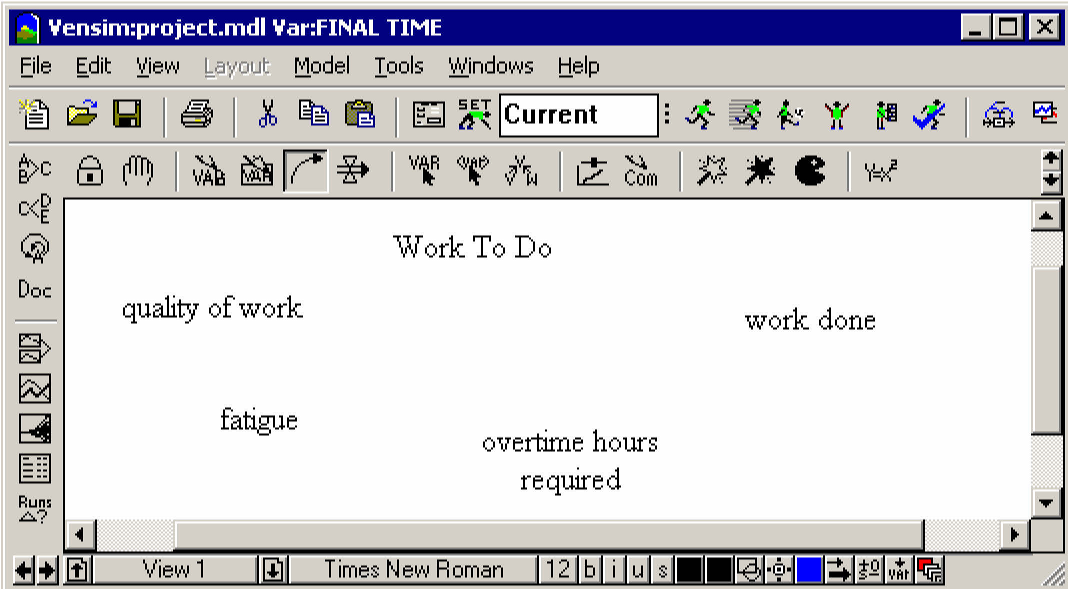
Adding Variables

✍ Click with the mouse button on the **Variable** tool 

(or press the keyboard number 3 above the letter keys — not the numerical keypad).

✍ Click in the middle top of the sketch and type `Work To Do` in the editing box, then press Enter.

✍ Click again on the sketch and continue filling out the diagram with the variables shown below.



Moving Sketch Objects

✍ Select the **Move/Size** tool by clicking on it (you can also press the keyboard number 2). Move the mouse directly over a variable. Press down and hold the mouse button then drag the mouse. A box will move to show you the new position for the variable. Release the mouse button and the variable will move to the new position.

You can also move and reposition objects using other sketch tools.

✍ Select the **Variable** tool again (click on it, or press the keyboard number 3). Move the cursor directly over a variable. Press down and hold the mouse button then drag the variable to a new position.

✍ Return the variables to the positions shown in the diagram above.

Now that we have laid out some important variables we will show their causal influences.

Adding Arrows

✍ Select the **Arrow** tool  by clicking on it (or press the keyboard number 5). Click once on `Work To Do`. Be sure to let the mouse button up without moving the mouse! Move the cursor to `overtime hours required` and click again. A straight arrow will join the two variables.

✍ Click once on `overtime hours required` then move the cursor to `work done` and click again. A straight arrow will join the two variables.

Handles

Handles are the little circles that appear in the middle of arrows in Vensim sketches, and at the corner of boxes and clear boxes, in the middle of rates and elsewhere. These handles allow you to resize or move things around. These handles appear when first entering variables, when creating arrows, and any time the **Move/Size** tool is selected. Handles can be manipulated by the **Move/Size** tool or any other sketch tool except **Lock**.

✍ Select the **Move/Size** tool to turn handles on.

Curved Arrows

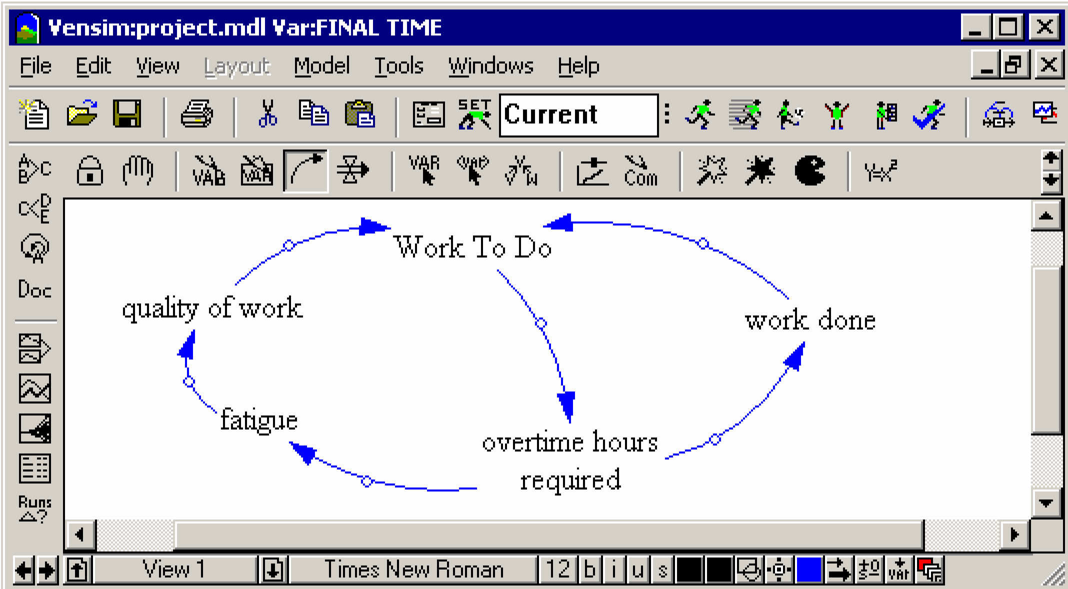
One Way:

✍ Position the pointer on top of the handle in the middle of the straight arrow from *`overtime hours required`* to *`work done`*. Press and hold the mouse button down, then drag the mouse (and arrow) a down to make a curved arrow.

Another Way:

✍ Select the **Arrow** tool. Click once on *`work done`*, then move the cursor to a blank portion of sketch just above and right of *`Work To Do`* and click once, then move the cursor onto *`Work To Do`* and click once again. A curved arrow will join the two variables. You can move this arrow by dragging the handle (with the **Arrow** tool or the **Move/Size** tool).

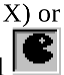
✍ Continue joining variables with curved arrows, according to the diagram below, by either making straight arrows and moving the handle to curve them, or by making a single intermediate click on the sketch.



Editing Variables

✍ To edit a variable name, click on it with the **Variable** tool to open the editing box, then type in a new name.

Deleting Variables

If you want to delete a variable from the model you can use either **Edit>Cut** (Ctrl + X) or press the Del key on the keyboard (both of which open a prompt dialog) or use the **Delete** tool  (which deletes from the model with no prompt).

Note that if you press the Del key or use **Edit>Cut** you will, except in PLE and PLE Plus, have the option to **Remove from this view but do not change model structure**. If you do this the variable may not appear in any sketch but will still be part of the model. Use caution in selecting this option.

✍ Select the **Variable** tool and click on the sketch, then type in the name *`temporary`* and press Enter.

✍ Select the **Delete** tool and click on the variable *`temporary`*.

Undo and Redo

If you make a mistake while creating a model, you can use the menu item **Edit>Undo** and **Edit>Redo** commands to step backward or forward. Ctrl+Z is the same as **Edit>Undo** and Ctrl+Y is the same as **Edit>Redo**. The undo/redo history has multiple levels for most editing changes.

Saving Your Model

✍ Click the **Save** button or select the menu item **File>Save** or press Ctrl + S. Save the model in the directory *guide\chap04* with a name such as *project*.

Models can be saved in text format, the default, with the file extension *.mdl*. Models can also be saved in binary format using the file extension *.vmf.* Binary format models can be used with the Vensim Model Reader. They also open more quickly than text format models though this is rarely an issue except with very large models. None of the models in this guide are big enough for this speed difference to be noticeable.

Modifying diagrams

Sketch objects have options which you can change. These options allow you to customize your sketch. Two different methods are used to change sketch options:

Click with the right mouse button on a sketch object (for the Macintosh, Ctrl + Click)

Select the sketch object (variable, arrow, etc.) then use the Status Bar to change the options or attributes of the selected object.

Selecting Sketch Objects

Several methods allow you to select single or multiple sketch objects.

? Click on a single object with the **Move/Size** tool.

? Select multiple objects by holding the mouse button down and then dragging the **Move/Size** (or **Lock**) tool over a region of sketch.

? Select multiple objects by holding down the Shift key and clicking on each object with the **Move/Size** tool.

? Select the whole sketch with **Edit>Select all** (or Ctrl + A).

? Deselect objects which are selected by holding down the Shift key and clicking on each object with the **Move/Size** tool.

? Deselect all objects by clicking on a blank region of sketch (outside of the selection rectangle).

Sketch Layout

Vensim includes menu commands to help you lay out your sketch in a tidy manner. These commands allow you to resize sketch objects to default values, line up objects by position with a "last-selected" object, size objects to the last-selected, and more.

We shall tidy up the diagram, centering most variables on *`Work To Do`*.

✍ Select the **Move/Size** tool by clicking on it or pressing keyboard number 2.

✍ Click once on *`overtime hours required`* then hold the Shift key down and click once on *`Work To Do`*. Select menu **Layout>Center on LastSel**.

`overtime hours required` will move to line up on the center of `Work To Do`.

✍ Click once on `quality of work` then hold the Shift key down and click once on `Work To Do`. Select menu **Layout>Vertical on LastSel**.

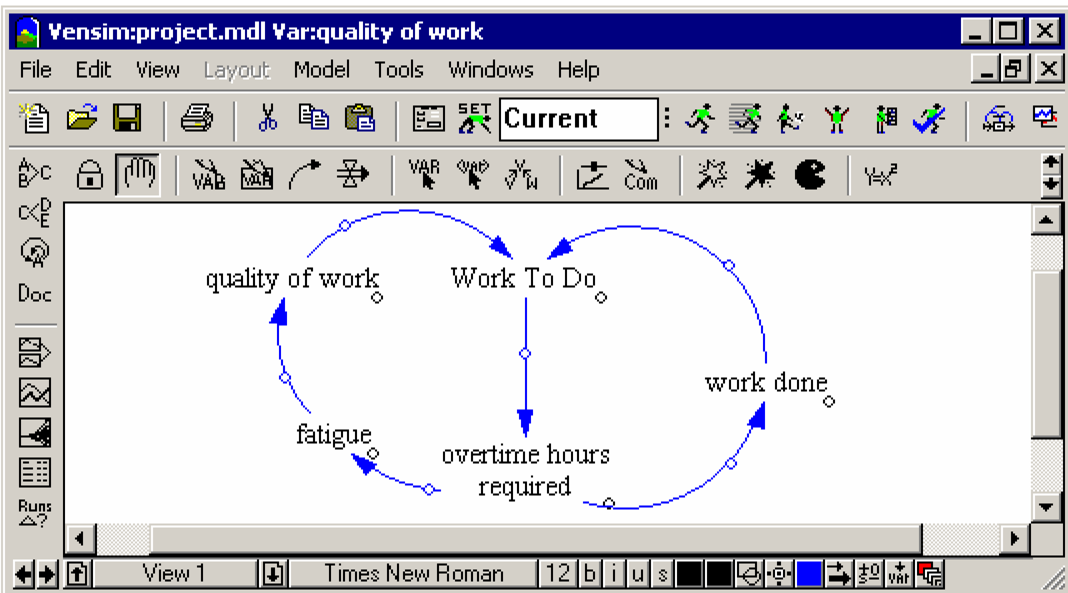
✍ Click once on `fatigue` then hold the Shift key down and click once on `quality of work`. Select menu **Layout>Center on LastSel**.

✍ Click once on `fatigue` then hold the Shift key down and click once on `overtime hours required`. Select menu **Layout>Vertical on LastSel**.

✍ Drag `work done` to right of and halfway between `Work To Do` and `overtime hours required`.

✍ Move the arrows to make neat curves resembling a circle (see below).

Your sketch should look something like:

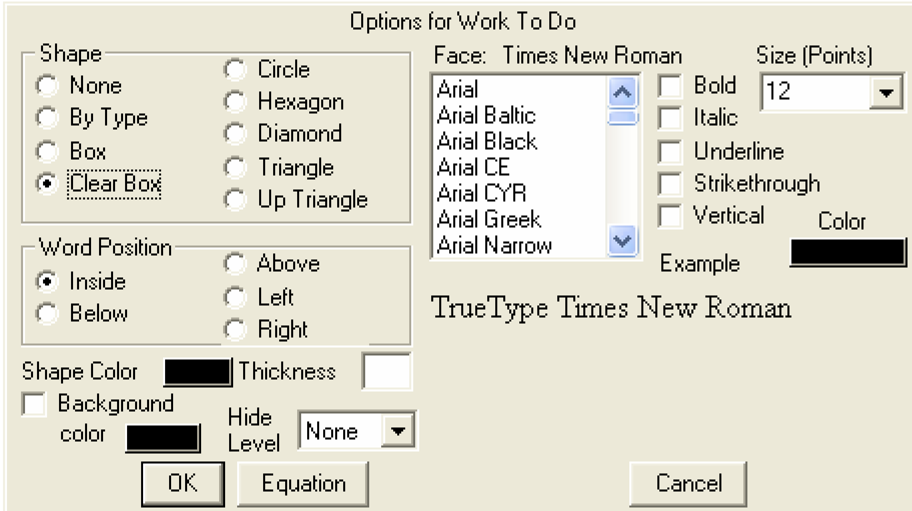


Sketch Options

Variables

✍ Select the **Lock** tool. Use the right mouse button single click on the variable `Work To Do`. For the Macintosh, hold the Control key down and click with the mouse button (Ctrl + Click).

An options dialog box will open.



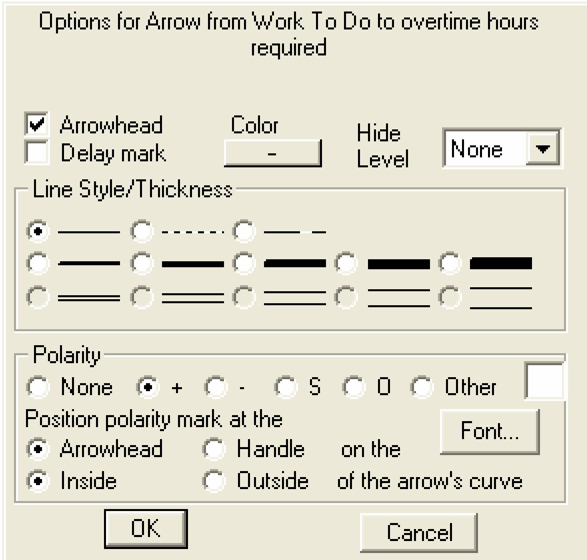
✍ Change the font (e.g., to Arial), the size, color, or anything else, then click **OK**.

Note that in the options dialog the **Word Position** option only applies if the variable has a **Shape** selected (anything but **None**).

✍ Select menu **Edit>Select all** or press Ctrl + A. Click on the font size button on the Status Bar at the bottom of the window (probably reads **12**) and choose a bigger size, say 14. Click outside of the highlighted box.

Arrows

✍ Click with the right mouse button on the arrowhead of the arrow from *`Work To Do`* to *`overtime hours required`*, an options dialog box will open:

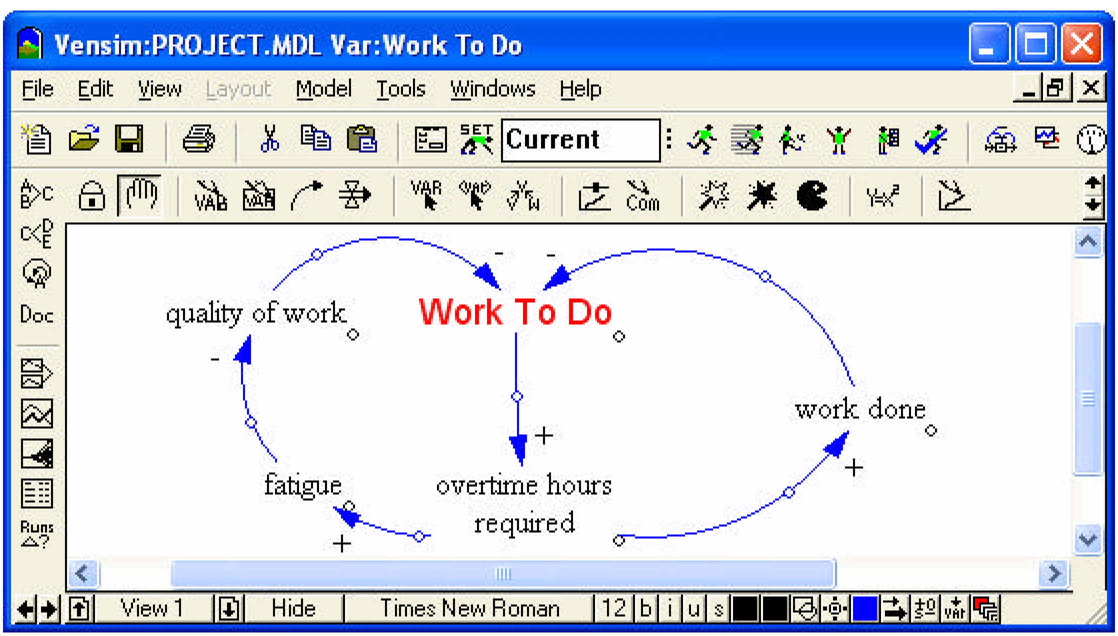


As *Work To Do* increases, *overtime hours required* also increase, a positive causality.

✍ Select + (under **Polarity**) and **Outside** (of the arrow's curve) then click **OK**.

The polarity (+) is, by default, attached to the head of the arrow inside the curve.

✍ Continue changing the polarity of arrows according to the figure below, selecting **Outside** of the arrow's curve for each arrow:



Now you will highlight the positive feedback loop with thicker and colored arrows.

✍ Select the **Move/Size** tool if it is not already selected.

✍ Click once on the arrowhead of the arrow from *`Work To Do`* to *`overtime hours required`*. Now hold down the Shift key and click on the other arrowheads of arrows from:

? overtime hours required to fatigue

? fatigue to quality of work

? quality of work to Work To Do

This will highlight all the handles and show a dotted box around the perimeter of the selected arrows.

✍ Release the Shift key.

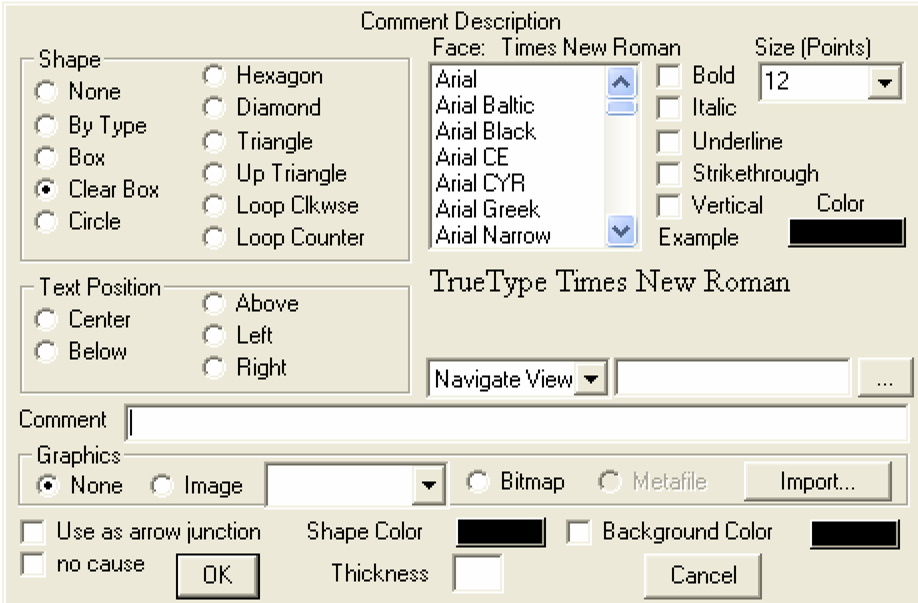
✍ On the Status Bar, look for the button with two arrows of different widths . Click on it and choose the fifth line from the top. All arrows highlighted will increase in width.

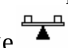
✍ Click on the **color** button just to the left of arrow width (probably colored blue) and choose a different color (e.g., red). Now click somewhere on the diagram outside of the dotted box to unselect the arrows.

Adding Comments and Graphics

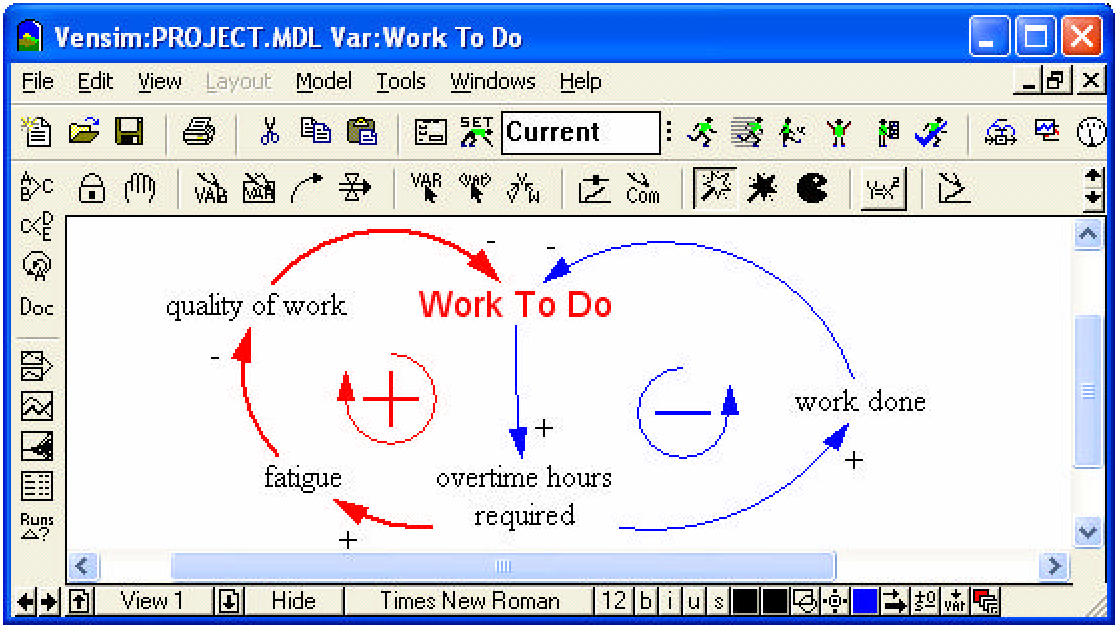
✍ If you need to make some room at the top of your diagram, select the **Move/Size** tool, then choose menu **Edit>Select all** (or Ctrl + A), then using the cursor, drag the whole diagram lower on the view to make room for the title.

✍ Select the **Sketch Comment** tool. Click at the top of your sketch to add a title; the Comment dialog box will open.



- Type in a title for your sketch (e.g., Work To Do Project Model). Choose a font, size, color, shape and word position for your comment, then click **OK**.
- Still with the **Sketch Comment** tool, click on the sketch in the center of the left hand loop. Click on the dropdown arrow in the **Graphics** field **Image** box and choose the positive sign (+) or the snowball image , then from the **Shape** field choose **Loop Clkwse** (clockwise).
- Click on the Black button in the middle right just below the word **Color** and click on red in the color palette that appears. Repeat this with the button labeled **Shape color**, and click **OK**. If you need to, reposition the loop image and resize the loop by dragging its handle. Note that you can also include a bitmap or metafile from the clipboard.
- Click on the sketch in the center of the right hand loop. Click on the dropdown arrow in the **Image** box and choose the negative sign (—) or the balance image , then from the **Shape** field choose **Loop Counter** (counter clockwise), set the two colors to blue and click **OK**. If you need to, reposition the loop image and resize the loop by dragging its handle.

Your diagram should now look similar to the figure below:



Refining the Model

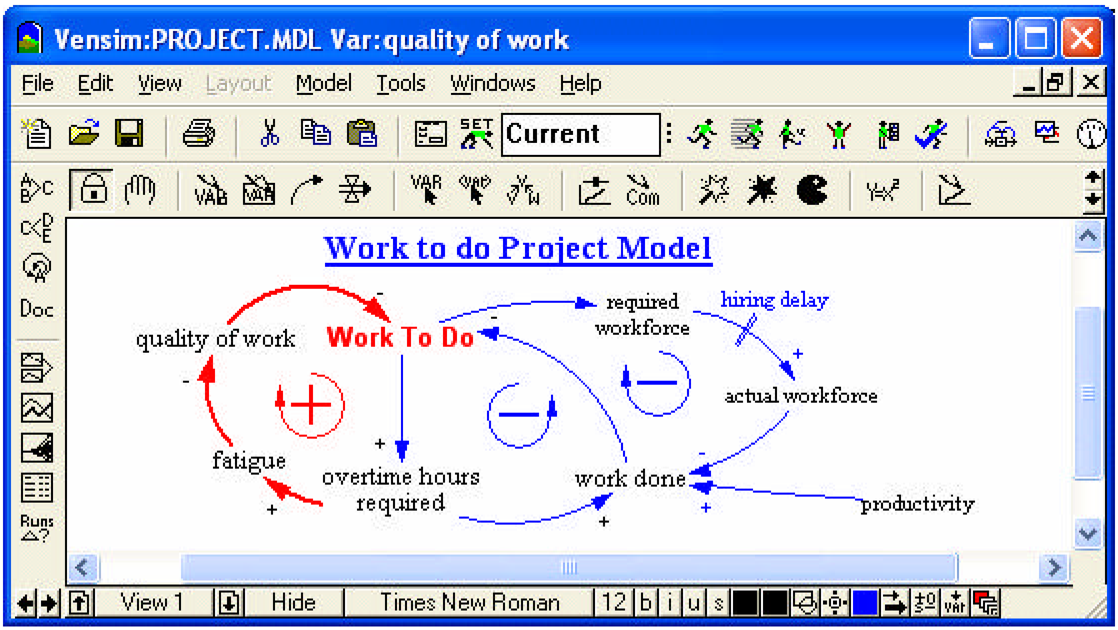
We see that `overtime hours required` increases the `work done`, which then decreases `Work To Do` (a negative feedback loop). However, `overtime hours required` also drives the positive feedback loop that increases `Work To Do`. To make this model more realistic, let us assume that we can change the size of the workforce through hiring and layoffs. Hiring workers will reduce the need for overtime and therefore reduce the fatigue/Work To Do spiral.

An Additional Feedback Loop

- Select the **Move/Size** tool and move the variable `work done` lower on the sketch. Reshape the arrows to and from `work done`, and move the negative loop symbol, as shown in the diagram below.
- Select the **Variable** tool. Click on the sketch and type in the new variables `required workforce`, `actual workforce` and `productivity` (pressing Enter after each variable) as shown in the diagram below.
- Select the **Arrow** tool and connect the variables as shown in the diagram below.
- Add polarities to the arrows. For the arrow from `required workforce` to `actual workforce` click on the Checkbox **Delay Marking**.
- Select the **Comment** tool and click to the right of the delay marking added above. Type in the phrase *hiring delay*, select blue as the **Color** and click on **OK**. You may need to move this somewhat to get the result shown below.

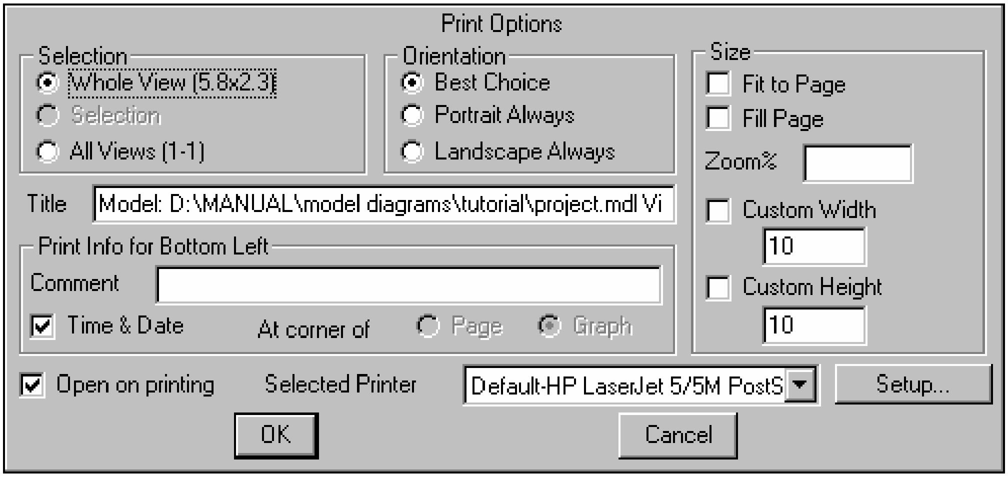
✍ Select the **Move/Size** tool, click once on the negative feedback symbol to highlight it, then choose **Edit>Copy** (or Ctrl + C). Select **Edit>Paste** (or Ctrl + V), then click on **OK** or press the Enter key in the dialog to choose **Replicate**. The new loop image gets pasted on top of the first image. Drag the copied feedback loop from the old location and place it in the center of the new feedback loop. Click on it with the right mouse button (Macintosh: Ctrl + click) then under **Shape** choose **Loop Clkwse** then click **OK**.

Your diagram will finally look something like this:



Printing and Exporting the Sketch

The sketch can be printed by clicking the **Print** button or by selecting the menu item **File>Print** while in the Build window.



The Print Options dialog gives a number of options, the more important ones are:

? **Selection** — print whole view, or print selected (portion of view), or print all views

? **Orientation** — portrait or landscape orientation

? **Size** — fit to page will fit your view onto a single page.

? **Title** — this will appear on the top of the printed page.

The sketch can be exported to the clipboard for use in other applications by using **Edit>Select all**, or selecting a group of variables with the **Lock** tool, then selecting **Edit>Copy** (Ctrl + C). This exports the sketch information to the clipboard as a metafile, which can then be pasted into other applications.

See Chapter 16 in the Reference Manual for more details on printing.

Structural Analysis of Diagrams

Analysis Tools

Vensim Analysis tools fall into two broad classes: tools for structural analysis, and tools for dataset analysis. Structural tools allow you to investigate the model structure; dataset tools allow you to investigate simulation datasets to determine the behavior of variables. In this section, we will analyze the structure of our model. Structural Analysis tools include the **Tree Diagram** tool (**Causes Tree** and **Uses Tree**), the **Loops** tool, and the **Document** tool.

Analysis tools almost always work by generating information about the Workbench Variable. You select the Workbench Variable by one of two methods. The easiest method is to click on the variable, wherever it appears. The variable is usually somewhere in one of the sketches, unless the model is text-based. You can also click on a variable in an Output window, such as a **Tree Diagram** or a **Strip Graph**. The second way to select a variable into the workbench is to click the **Control Panel** button to open the Control Panel, select the tab **Variable** to open the Variable Selection Control, then choose the variable from the list. The Workbench Variable always appears on the title bar of the model.

NOTE If you activate an Analysis tool that requires a simulation dataset, and you have no simulation dataset loaded, you will see the message "No runs are loaded. Please load runs". This tells you that you need to run a simulation. Chapter 5 will describe how to build a simulation model.

EITHER

- ✍ Use the model *project.mdl* that you just built,

OR

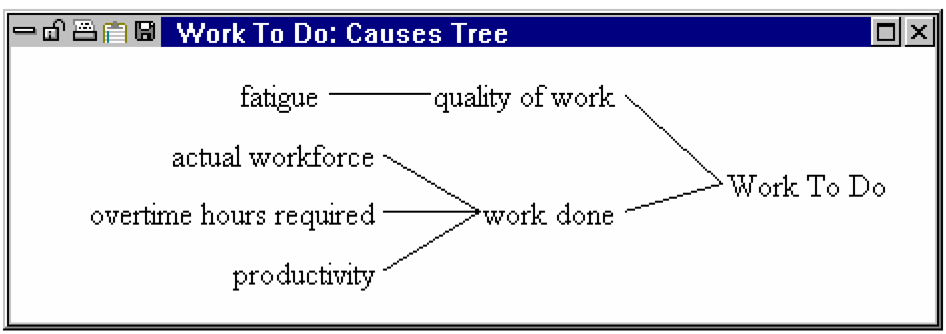
- ✍ Open the model *project.mdl* in the *guide\chap04\complete* directory.
- ✍ Select the **Lock** tool. Now move to the variable `Work To Do` and click on it to select it as the Workbench Variable. The title bar should look like:

Causal Tracing® with Trees

Causal Tracing is a powerful tool for moving through a model tracing what causes something to change. Causal Tracing Analysis tools can be configured to show the causes of a variable or the uses of a variable (the opposite direction to causes).

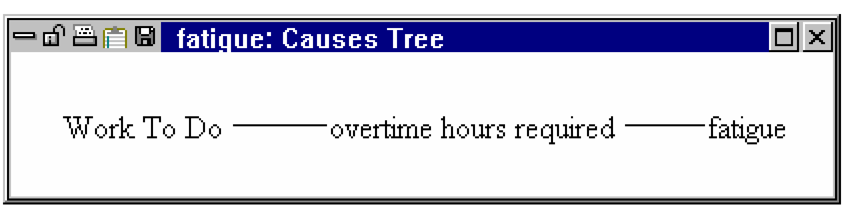
Causes Tree Diagram

- ✍ Click on the **Causes Tree** tool. We see the causes of `Work To Do`:



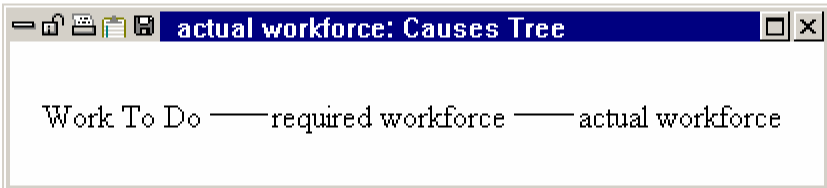
We can trace through the diagram looking at what causes any particular variable.

- ✍ Click on `fatigue` appearing in the tree diagram window then click on the **Causes Tree** tool again:



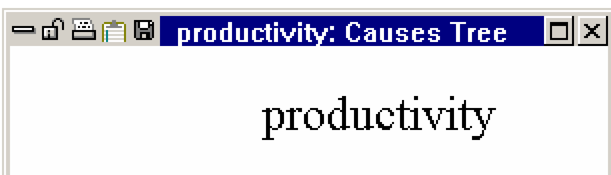
We can see that `fatigue` is caused by `overtime hours required` and by `Work To Do`. Now we have traced all the way around one feedback loop, starting and finishing at `Work To Do`. Let us look at what causes `actual workforce`.

✍ Click on `actual workforce` appearing in the first tree diagram or on the sketch then click on the **Causes Tree** tool:



We have traced another causal loop, from `Work To Do` through `actual workforce` and back to `Work To Do`.

✍ Click on `productivity` either in one of the previous tree diagrams or on the model sketch and then click on the **Causes Tree** tool:



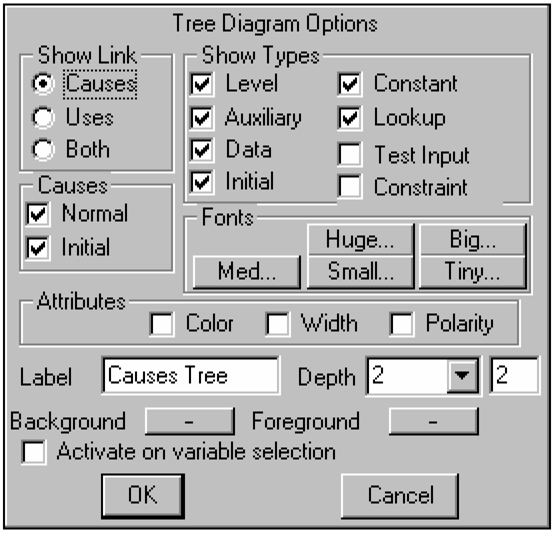
There are no causes of `productivity`; it is a Constant or Exogenous in this model.

NOTE You can get back to the sketch you built by either clicking on it, deleting the tree diagrams or clicking on the **Build Windows** button  or using the Shift+Ctrl+Tab key combination. If you don't delete the tree diagrams you can make them visible again by clicking on the **Output Window** button  and circulate between them by clicking again on this button or using the Ctrl+Tab key combination.

Analysis Tool Options (Not PLE or PLE Plus)

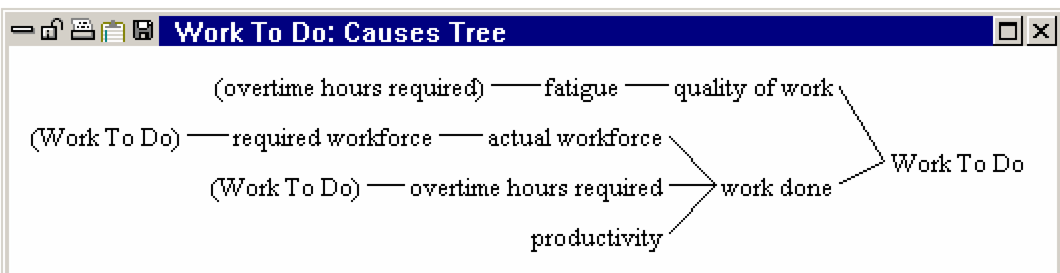
The analysis toolset for Vensim PLE and PLE Plus is fixed. In the other Vensim configurations the Analysis tools have options which cause them to display different information about the model. To look deeper into the model, you can set the tree diagram to different depths.

✍ Click using the right mouse button (or Ctrl + Click) on the **Causes Tree** tool. The Tree Diagram Options dialog opens:



✍ Click on the drop-down arrow in the **Depth** box and select 6, then click on **OK**.

✍ Click on the variable `Work To Do`. Click on the **Causes Tree** tool . You should see the causes of `Work To Do` to a depth of up to 6 causes:



Note that both `Work To Do` and `overtime hours required` are enclosed in parentheses and terminate the diagram before a depth of six causes are reached. The parentheses indicate that this variable appears somewhere else on the diagram, and therefore that there is a feedback loop within this tree diagram.

Automatic Tool Activation (Not PLE or PLE Plus)

When tracing a lot of structure, it can be useful to have the Analysis tool automatically activate whenever you select a new variable as the Workbench Variable.

✍ Click using the right mouse button (or Ctrl + Click) on the **Causes Tree** tool.

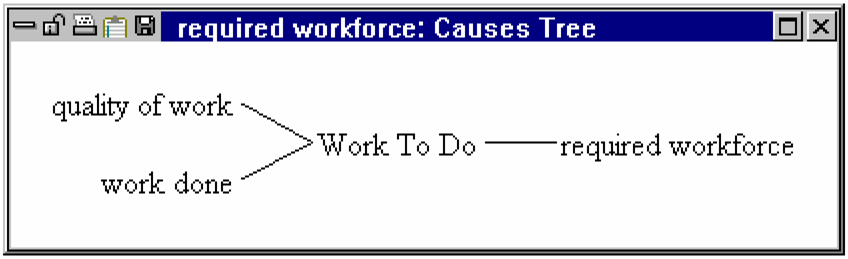
The Tree Diagram Options dialog opens (see above).

✍ Click on the drop-down arrow in the **Depth** box and select 2.

✍ Click on the checkbox for **Activate on variable selection**, then click on **OK**.

✍ Click on the variable `required workforce`, appearing in the tree diagram.

A **Causes Tree** is automatically generated.



✍ Click using the right mouse button (or Ctrl + Click) on the **Causes Tree** tool.

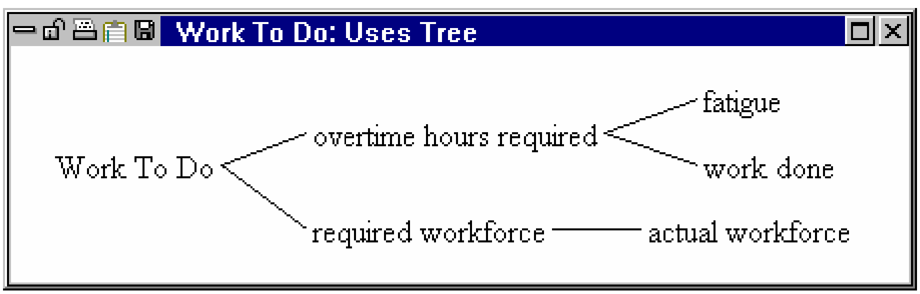
✍ Click on the checkbox to uncheck **Activate on variable selection**, then click on **OK**.

Uses Tree Diagram

Now let us look at a **Uses Tree** diagram.

✍ Click on `Work To Do` to select it as the Workbench Variable (check the title bar).

✍ Click on the **Uses Tree** tool.



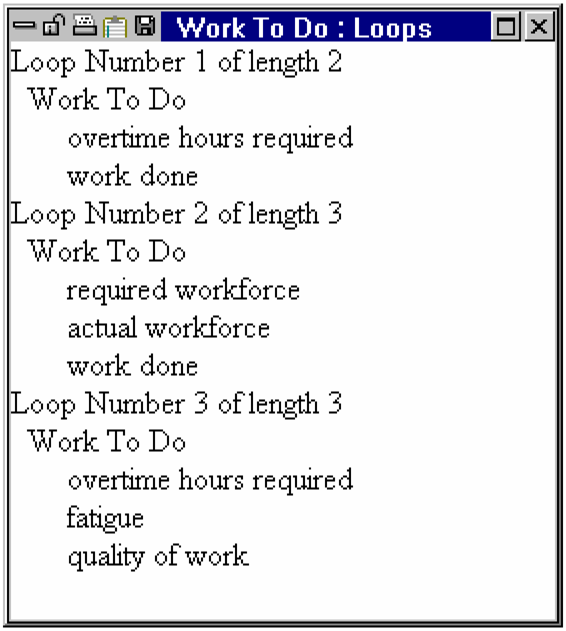
Uses are the opposite of causes, you see where in the model `Work To Do` is used.

Loops Tool

Now let us look at a tool that finds feedback loops for you: the Loops tool.

✍ Make sure `Work To Do` is still the Workbench Variable (look at the title bar).

✍ Click on the **Loops** tool. `Work To Do` is involved in three feedback loops:

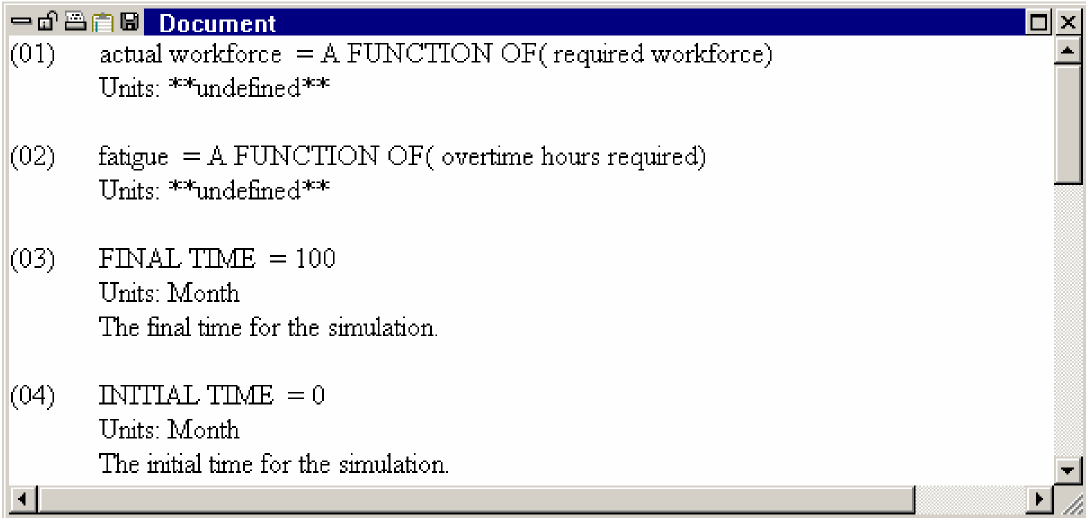


Document Tool

This tool provides a text-based description of some aspect of your model. In Vensim PLE and PLE Plus the document tool will display information about all model variables. In the other configuration you can set the options to display different things about the model, but the default is to document only the Workbench variable.

PLE and PLE PLUS

✍ Click on the **Document** tool.



Document

(01) actual workforce = A FUNCTION OF(required workforce)
Units: **undefined**

(02) fatigue = A FUNCTION OF(overtime hours required)
Units: **undefined**

(03) FINAL TIME = 100
Units: Month
The final time for the simulation.

(04) INITIAL TIME = 0
Units: Month
The initial time for the simulation.

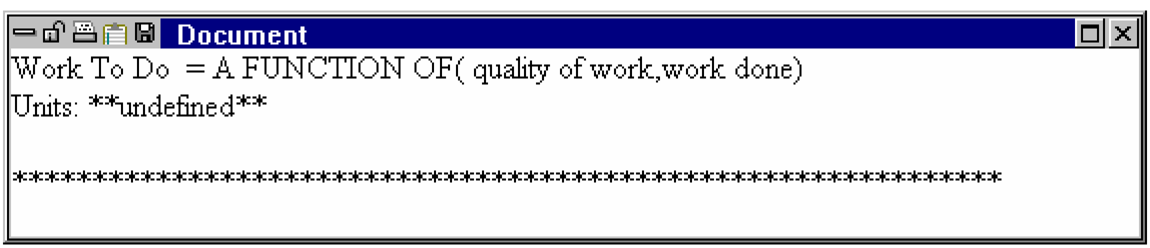
The different model variables are numbered and displayed in alphabetical order. You will notice that the variables ***FINAL TIME***, ***INITIAL TIME***, ***SAVEPER*** and ***TIMESTEP*** also appear in the document output even though you did not add these to the model. These four variables are used to control simulations and are part of every model - even models that have only diagrams.

Other Configurations

First, we will examine the Workbench Variable:

✍ Make sure ***Work To Do*** is still the Workbench Variable (look at the title bar).

✍ Click on the **Document** tool.



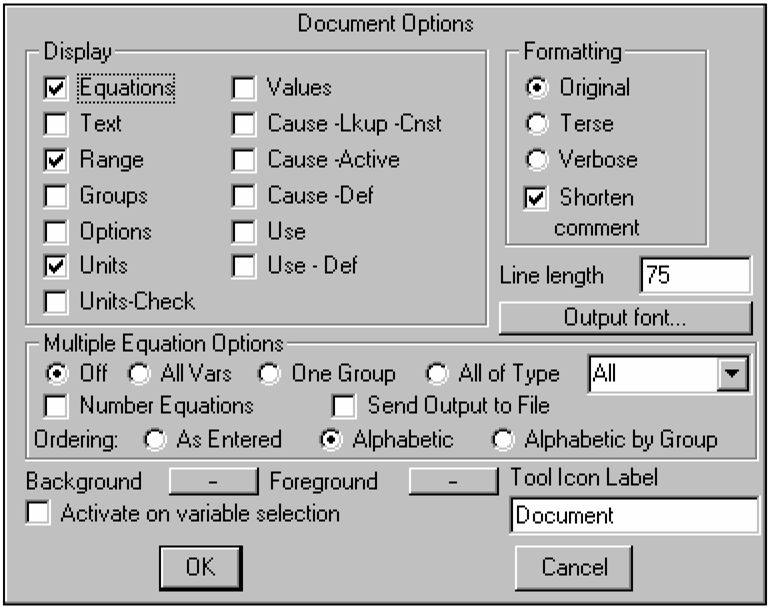
Document

Work To Do = A FUNCTION OF(quality of work,work done)
Units: **undefined**

****

The output displays a text description of the equations and definition of the variable ***Work To Do***. Next, let us document the entire model by changing the options for the **Document** tool:

✍ Use the right mouse button (or Ctrl + Click) on the **Document** tool.



- ✍ In the **Display** field, uncheck the check boxes **Range** and **Units**. In the **Multiple Equation Options** field, click the option button **All Vars**. Click on **OK**.
- ✍ Click on the **Document** tool.

A description of your entire model is generated. If you kept the document Output window (from the exercise above) open, the output is added to this window. If you closed it, a new window is generated containing the information. Note that the time bounds for the model are included, even though this is not a simulation model.

Saving Analysis Toolsets Not PLE or PLE Plus

NOTE When you close Vensim, you will see a message asking you "Do you want to save the current toolset" **Yes/No/Cancel**.

- ✍ Click **No** if you want to keep the original default toolset (*default.vts*).
- ✍ If you want to save the toolset with the settings you have made, you should click **Cancel**, then select menu item **Tools>Analysis Toolset>Save As**, and type in a new toolset name, then click **Save**.

5 Stock and Flow Diagrams

Stock and flow (or Level and Rate) diagrams are ways of representing the structure of a system with more detailed information than is shown in a causal loop diagram. Stocks (Levels) are fundamental to generating behavior in a system; flows (Rates) cause stocks to change. Stock and flow diagrams are the most common first step in building a simulation model because they help define types of variables that are important in causing behavior. We will construct a diagram describing the relationships among awareness of a product and the number of customers and potential customers.

Building a diagram (*customer.mdl*)

✍ Start Vensim.

✍ Click on the **New Model** button (or select menu **File>New Model…**) and click **OK** in the Model Settings dialog box to accept the default values.

✍ Click the **Save** button on the Toolbar. Select the directory *guide\chap05* then save as *customer* or some other name of your choice.

Entering Levels

Levels are also known as stocks, accumulations, or state variables. Levels change their values by accumulating or integrating rates. This means that the values of Levels change continuously over time even when the rates are changing discontinuously. Rates, also known as flows, change the value of levels. The value of a rate is not dependent on previous values of that rate; instead the levels in a system, along with exogenous influences, determine the values of rates. Intermediate concepts or calculations are known as auxiliaries and, like rates, can change immediately in response to changes in levels or exogenous influences.

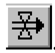
When constructing a Level and Rate diagram, consider what variables accumulate over a period of time. Another way to think about this: if Time slowed down to zero for your system, what variables would still be nonzero? For example, in the system where you pour water into a glass, the water contained in the glass is the Level. If you froze time, the pouring (a Rate) would stop, but you would still see a quantity of water in the glass (a Level). Once you know what levels you need, enter them first and then connect the rates and auxiliaries. Model building tends to be iterative. Don't try to get everything right the first time; you can always change things later on.

For the problem we are working here the levels are `Potential Customers` and `Customers`.

✍ Select the **Box Variable** tool  and click once on the diagram. Type in `Potential Customers` and press Enter.

✍ With the **Box Variable** tool still selected, click on the diagram approximately 3 inches (7 cm) to the right of `Potential Customers`, type in `Customers`, and press Enter.

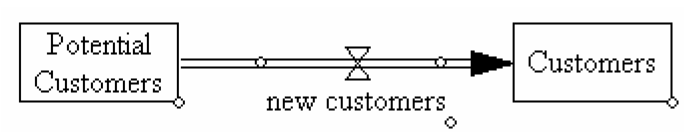
Creating Rates

✍ Select the **Rate** tool . Click once (click and release the mouse button) on top of `Potential Customers`, then move the cursor on top of `Customers` and click once again. Type the name `new customers` and press Enter.

The Rate has a single arrowhead, indicating the direction that material can flow (the Rate can only increase the Level). This is only a diagram, in a simulation model the equation governs the direction that material can flow. However, we can use the diagram to indicate whether the flow is intended to be one way or two way.

✍ Select the **Move/Size** tool.

When you do this you will see that the pipes have small circles in the middle of them.



✍ Right Click (or Ctrl+Click) on the small circle to the left of the valve (). Check the **Arrowhead** checkbox. Click **OK**.

You will see a two way flow.

This diagram would imply that `new customers` could be positive or negative. We do not want that here.

✍ Select **Edit>Undo Options** from the menu or use the Ctrl+Z key equivalents to undo the last change. The flow will return to one direction.

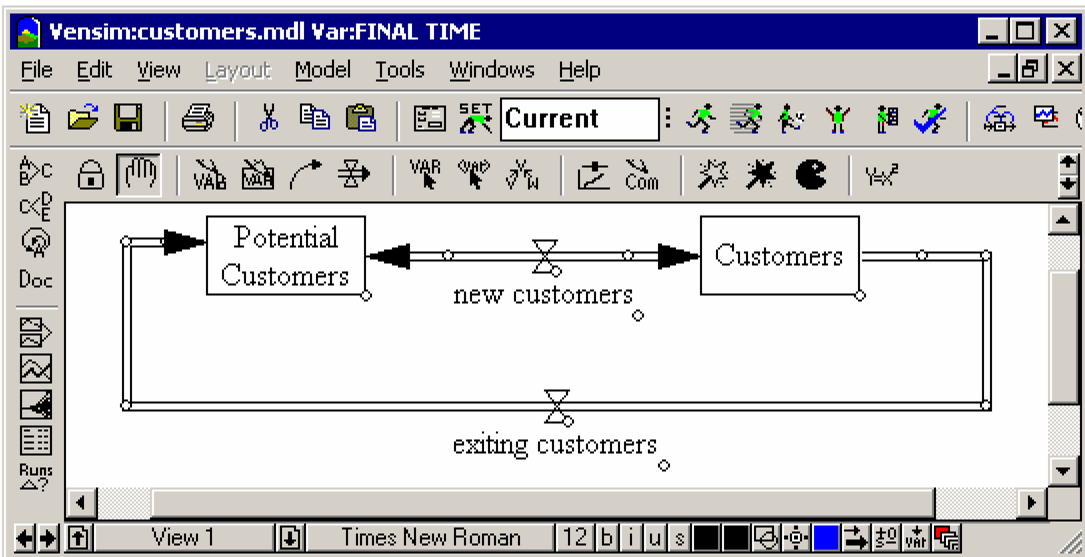
Configuring the Rate Tool (not PLE or PLE Plus)

You can change the behavior of the **Rate** tool to always start with two way flows by Right-Clicking on the **Rate** tool and unchecking the **One Way Flow** checkbox. For most models, however, two way flows are relatively infrequent and it is easier just to change them when they occur as demonstrated above.

Bending Rate Pipes

✍ Click once on `Customers`, then press and hold the Shift key down and move the cursor about half an inch (1 cm) to the right of `Customers` and click once. Continue holding down the Shift key for all further clicks. Move directly down about an inch and a half (3.5 cm) (below and right of `Customers`) then click once. Move directly left to 1 inch below and left of `Potential Customers` and click once. Move directly up just left of `Potential Customers` and click once. Move on to `Potential Customers` and click once. Release the shift key. Type the Rate name *`exiting customers`* and press Enter.

Your diagram should look like the one below:



Adding Auxiliaries and Arrows

✍ Select the **Variable** tool. Click once just to the left of and below *`new customers`* then type the name *`time to become customer`* and press Enter.

✍ Click once just right and below *`exiting customers`* and type the name *`product life`* and press Enter.

✍ Select the **Arrow** tool. Click once on *`Potential Customers`*, then once on a blank part of the sketch above and between *`Potential Customers`* and *`new customers`*, then once on the valve attached to *`new customers`*.

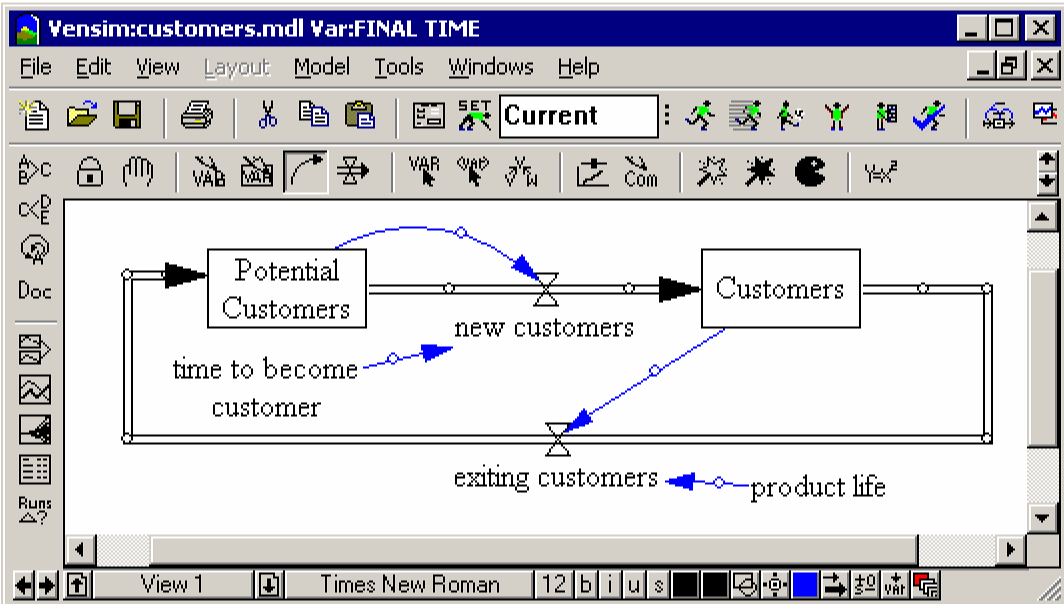
A curved arrow will join the Level and the Rate valve. Vensim allows you to connect arrows to either the Rate name or the Rate valve. The Rate name and valve are structurally the same.

✍ Click once on *`time to become customer`* then once on the Rate name *`new customers`*.

✍ Click once on `Customers` then once on the valve attached to the Rate `exiting customers`.

✍ Click once on `product life` then once on the Rate name `exiting customers`.

Your diagram should now look like the one below:



More Structure

✍ Select the **Box Variable** tool and click once on the diagram approximately 2 inches (5 cm) above the Level `Potential Customers`. Type in the name `Awareness` and press Enter.

✍ Select the **Rate** tool. Click once on an empty part of the diagram about 2 inches (5 cm) left of the Level `Awareness`. Move the cursor over to `Awareness` and click once on this Level. Type in the variable name `gaining awareness` in the editing box then press Enter.

A Rate named `gaining awareness` will be constructed between the Level and a cloud. The cloud defines the limits of the model; we do not care where the material comes from, or what happens if the material goes into a cloud.

✍ Click once on the Level `Awareness`. Move the cursor to an empty part of the diagram about 2 inches to the right of `Awareness` then click once. Type in the Rate name `forgetting` in the editing box, and then press Enter.

✍ Select the **Variable** tool. Click on the sketch below and left of the Rate `gaining awareness` and type the name `advertising effectiveness` but **do not** press Enter (leave the editing box open).

✍ Click again on the sketch just below *advertising effectiveness* and type *customer referral*.

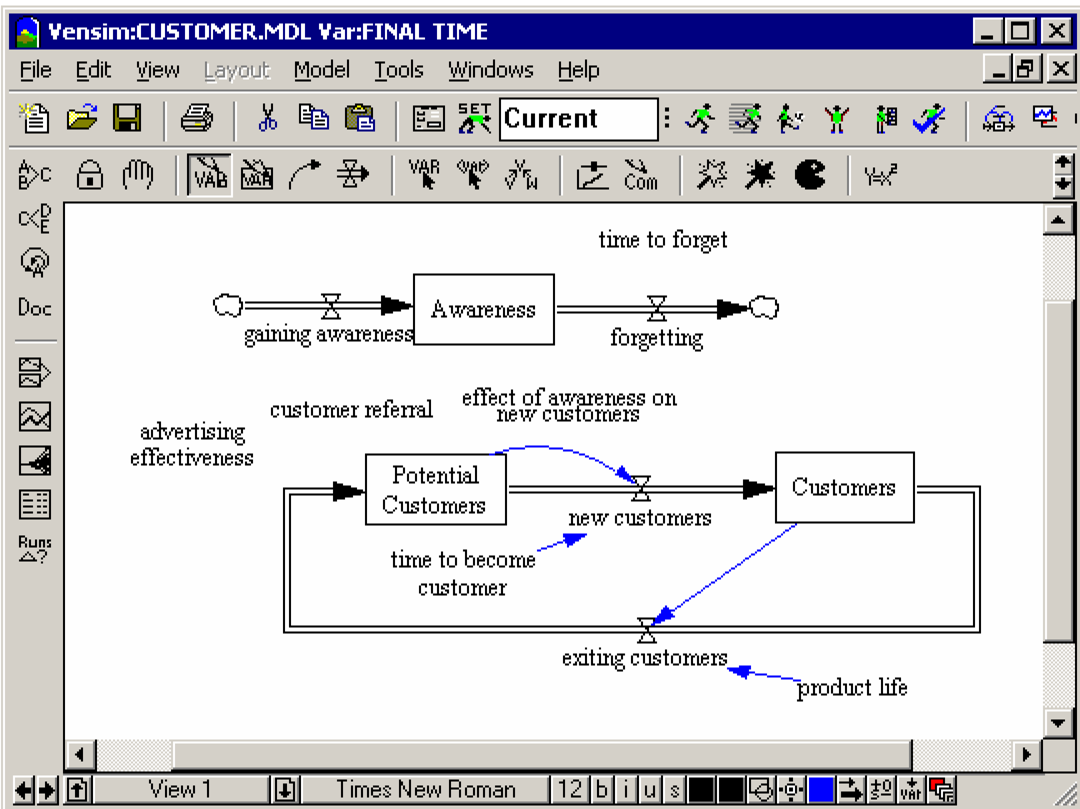
✍ Click again on the sketch just above the Rate *forgetting* and type *time to forget*.

Note how you can enter multiple variables by clicking on the sketch *without* pressing the Enter key.

✍ Click again on the sketch between *Awareness* and *new customers* and type *effect of awareness on new customers* and press Enter.

✍ If you are running out of space, you can move the whole structure around the page. Choose the menu item **Edit>Select All** (or press Ctrl + A) then move the structure around the page (you cannot do this if the **Lock** tool is selected, use the current Sketch tool or the **Move/Size** tool). Click outside the structure to unselect it.

Your sketch should now look like this:



✍ Select the **Arrow** tool. Click once on *advertising effectiveness* then once on *gaining awareness*.

✍ Click once on customer referral then once on gaining awareness.

✍ Click once on `Awareness` then once on a blank part of the sketch above and between `Awareness` and `forgetting`, then once on the valve attached to `forgetting`.

✍ Click once on `time to forget` and then once on the valve for `forgetting`.

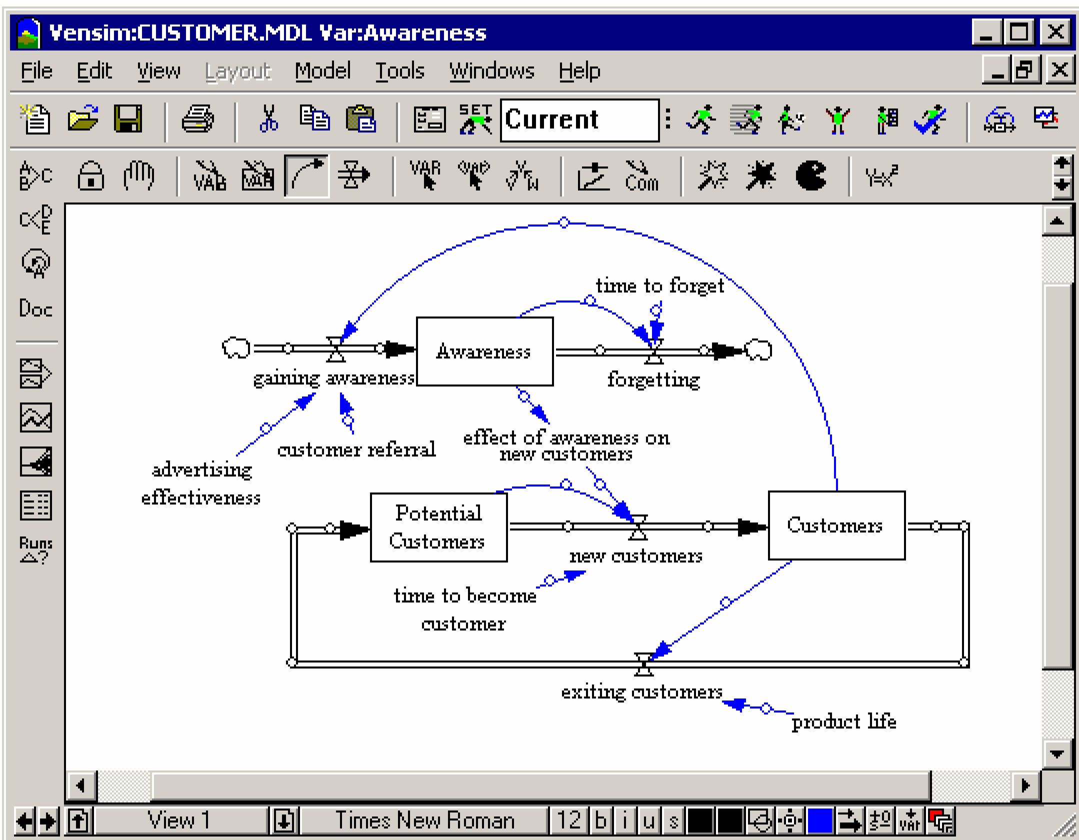
✍ Click once on Awareness then once on effect of awareness on new customers.

✍ Click once on effect of awareness on new customers then once on the valve for new customers.

✍ Click once on `Customers` then once on a blank part of the sketch a little above the cloud for the Rate `forgetting`, then once on `gaining awareness`.

✍ Move variables around and drag the arrow handles with the **Move/size** tool if you need to make the sketch neater.

Your sketch should now be complete and look like this:



✍ Click the **Save** button on the Toolbar to save your work.

Customizing Diagrams

Diagrams can be customized in many different ways. The standard practice for Vensim diagrams is to show levels (stocks) as a box, with the name inside the box. Rates are shown with the Rate valve explicitly named, although sometimes a Rate will be unnamed (show only a valve). Auxiliaries, constants, lookups, data (exogenous) variables, etc. are shown as a simple name. This section describes some options for customizing diagrams.

Sketch Options

✍ Select the **Move/Size** tool. Click on the handle of the Level *`Awareness`* (little circle at the bottom right corner of the box) and drag it until the box is a little larger.

✍ Click once on the Level *`Potential Customers`* to select it, then hold the Shift key down and click once on *`Customers`* and once on *`Awareness`*. Select menu **Layout>Size to LastSel**.

✍ Select menu item **Edit>Select all** (or Ctrl + A). Click on the button for setting color of arrows on the Status Bar (find this button by holding the cursor over each button until the name "set color on selected arrows" appears). Choose a color (e.g., red) then click on the sketch outside of the selection box.

Note that both arrows and rates are changed. If you want to change arrows and rates separately, you need to select each type individually. You can do this with the menu items **Edit>Or Select>Information Arrows** and **Edit>Or Select>Rates** or by holding down the Shift key and clicking on each of the rates or information arrows individually.

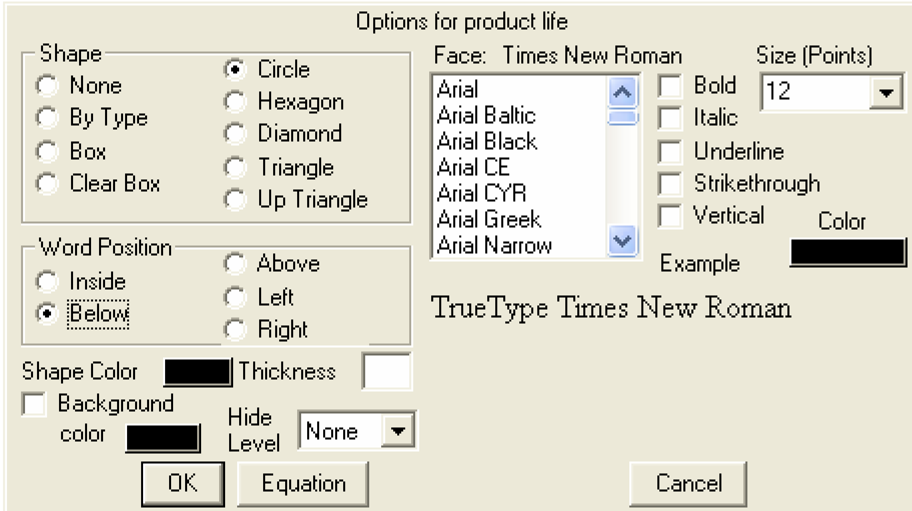
✍ Click on *`Awareness`* then hold the shift key down and click on *`Potential Customers`* and *`Customers`*. On the Status Bar, click on **b** to select bold (toggles to **B**), and if you wish, change the color of the variables and the box color (also located on the Status Bar).

✍ Press the Esc key or keyboard number 1 to return to the **Lock** tool (or click on the **Lock** tool).

Variable Shapes

We can change the shape of an individual variable by clicking on it with the right mouse button, then selecting a new shape in the Options dialog box.

✍ Click with the right mouse button on the variable *`product life`*.



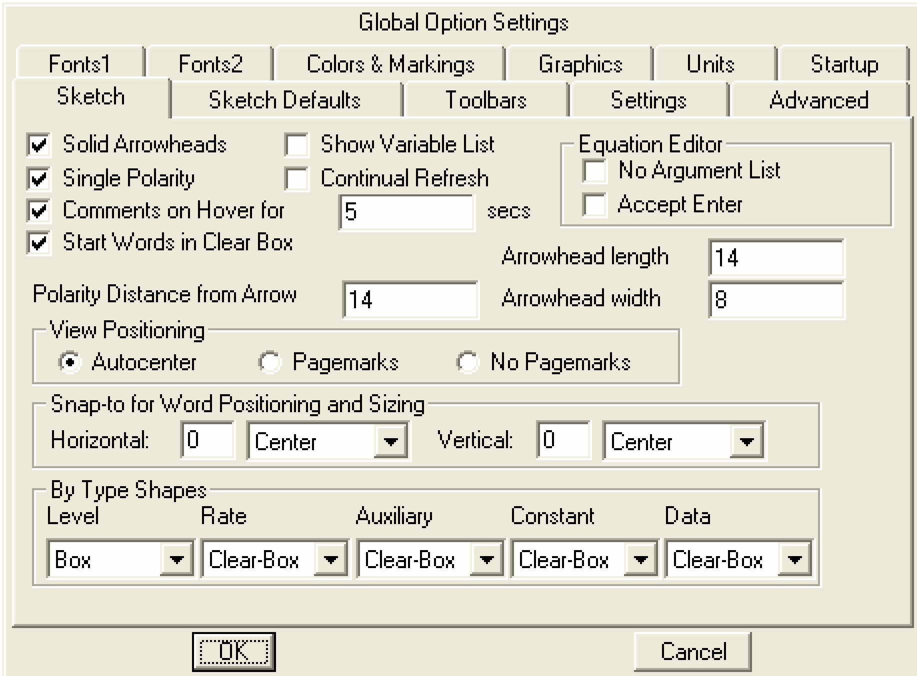
✍ In the Options dialog box, click on the option button for **Circle** in the **Shape** field, click on the button for **Below** in the **Word Position** field then click **OK**.

An alternative way to do this is to click on the variable with the **Lock** or **Move/size** tool to highlight it, then select a shape from the **Surround shape** button on the Status Bar.

Default Sketch Options (Not PLE or PLE Plus)

In the Options dialog (**Tools>Options**), there is a button for **Shape** labeled **By Type**. If you select this, the resulting shape comes from the settings in the Global Options dialog. Let us look at, but not change, the sketch defaults.

✍ Select menu **Tools>Options…** then click on the tab **Sketch**



Note the **By Type Shapes** field with drop-down boxes for the different variable types in Vensim.

✍ Click the button **Cancel**.

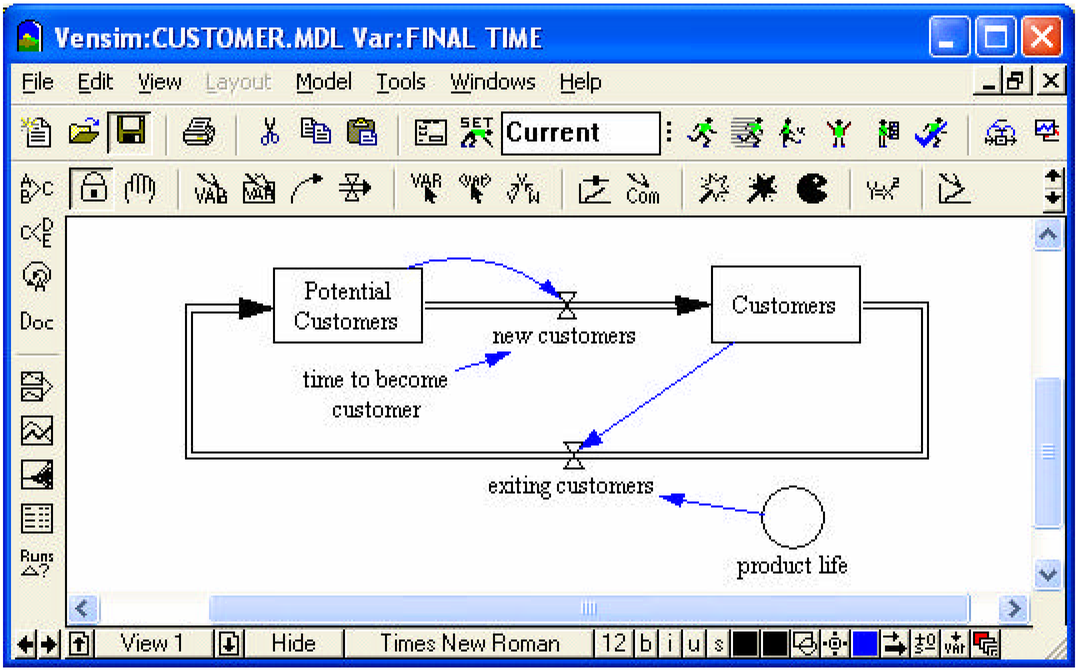
You can also set shape to **By Type** by selecting a variable using the **Lock** tool, then clicking on the **Surround Shape** button  and clicking on the second choice from the top .

Hiding Sketch Elements (Not PLE or PLE Plus)

Let’s customize the diagram and investigate some of the view menu capabilities.

✍ Select the **Hide Wand** tool . Click on the Level Awareness, click on forgetting, gaining awareness, time to forget, effect of awareness on new customers, advertising effectiveness, customer referral, and click on the two clouds.

Each element will be hidden from view, until your structure looks like this:



✍ Select **View>Show Hidden>Depth 1**, or press down arrow on the keyboard.

The word "Hide" on the status bar along the bottom will change to "Depth 1."

A check also appears beside the menu item to show it is active. The sketch elements hidden with the **Hide Wand** will pop into view.

✍ Select the **Unhide Wand** (the white wand). Click on each of the elements that was hidden.

✍ Select **View>Show Hidden>None** to uncheck it, or press up arrow key on the keyboard.

Now all your sketch elements should be showing. If you have missed any, they will be hidden. Repeat the above process again to make them visible.

NOTE You can set hide levels from 1 to 16 and use the up and down arrow keys to unhide and hide successive levels.

Resizing the View

✍ Select View>Zoom>200%.

The sketch will zoom to 200 percent.

✍ Select View>Zoom>100%.

The sketch will zoom back to the original size. Zooming simply focuses in or out of the sketch, making everything larger or smaller. You can also zoom by holding down the Shift and Ctrl keys and dragging the mouse up and down over the view.

Rescaling the View

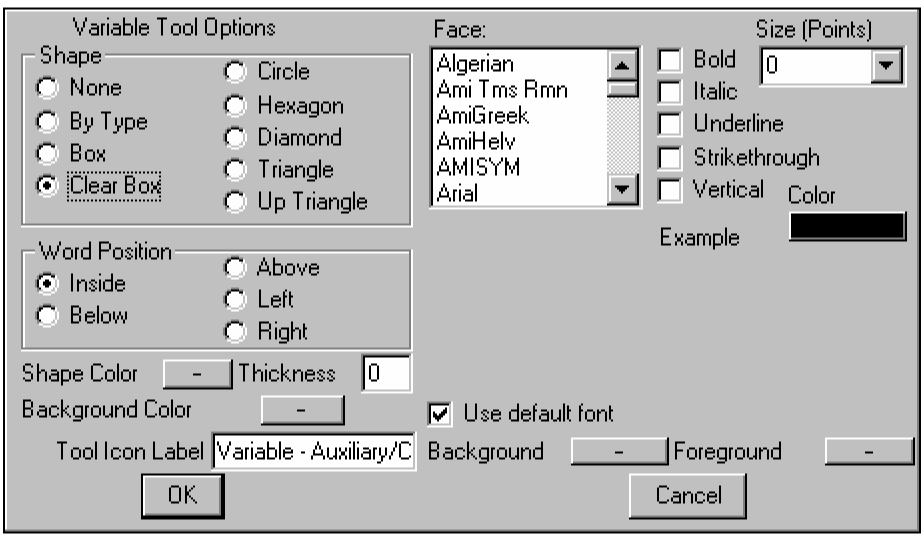
✍ Select **View>Rescale**. Type 120 into the editing box under Horizontal and 140 into the editing box under Vertical and click **OK**.

By default rescaling changes the scale (distance) between sketch objects, rescaling does not change the size of any sketch object. Note that we cannot rescale to the original size by selecting or typing 100%. We need to shrink the scale, in this case the X scale to 83% and the Y scale to 71%.

Modifying the Sketch Tools (Not PLE or PLE Plus)

We have been using the default Sketch tools to do our diagramming. Each of these tools can be modified, and the whole set of tools in use can also be changed (see Chapter 13 of the Reference Manual for details). The most common modification of the Sketch tools will be to apply different variable shapes.

✍ Click with the right mouse button on the **Variable** tool; an options dialog opens.



This dialog box is almost the same as the Variable Options dialog used above, but has a few additional entries. The only difference between the **Variable** tool and the **Box Variable** tool is the selection of shape. The Variable tool has a default of **Shape: Clear Box**. This tool creates a variable placed in a clear box with a sizing handle. You can use the sizing handle to break up long names onto a number of lines.

Sometimes you might prefer to have just one line variables. This is easily accomplished by selecting **Shape: None**.

✍ Click on the **OK** button to close the dialog.

Saving Sketch Toolsets

NOTE When you close Vensim, you will see a message asking you "Do you want to save the current toolset" **Yes/No/Cancel**.

✍ Click **No** if you want to keep the original default sketch toolset (*default.sts*).

If you want to save the toolset with the settings you have made, you should click **Cancel**, then select menu item **Tools>Sketch Toolset>Save As**, and type in a new toolset name, then click **Save** .

6 Building a Simulation Model

A Population Model

This chapter features a simulation model of rabbit population. The modeling process starts with sketching a model, then writing equations and specifying numerical quantities. Next, the model is simulated with simulation output automatically saved as a dataset. Finally, the simulation data can be examined with Analysis tools to discover the dynamic behavior of variables in the model.

Normal model construction follows a pattern of create, examine, and recreate, iterating until your model meets your requirements. Debugging (making a model simulate properly) and model analysis (investigating output behavior) both play a part in refining the model. Reality Check is another technology to aid in the construction and refinement of models and is described in Chapter 14.

The behavior of a simulation model in Vensim is solely determined by the *equations* that govern the relationships between different variables. We will list those equations in full for the simulation models developed in this *Guide*. The diagram of a model (causal loop or stock and flow) is a picture of the relationships between variables. Vensim enforces consistency of the diagram and model equations, but information can be omitted or hidden in diagrams. When you are building a simulation model, make sure the equations match what is in this manual. If there is a discrepancy in the appearance of the diagrams, it may be because you have inadvertently hidden something or set an option that influences the appearance of model diagrams.

Vensim Conventions

Naming

Model diagrams should be clearly presented to facilitate building, analysis, and presentation. Most of the models in this manual follow certain naming conventions that we recommend, though you may choose otherwise if you wish.

Levels have initial letters capitalized; e.g., *`Population`*

Rates, **auxiliaries**, **constants**, **lookups**, **data variables**, and other variable types are all lower case; e.g., *`average lifetime`*

Sketch

Levels or stocks are entered with the **Box Variable** tool. When using the **Box Variable** tool, the variable is designated as a Level. When you open the Equation Editor you will see that variables added with the **Box Variable** tool have type Level. You can change type in the Equation Editor, or make variables without boxes have type Level in the Equation Editor but this can cause confusion and is not recommended except for special purposes.

Rates are usually entered with the **Rate** tool. By default rates are added with a name and one arrowhead to indicate a flow in the direction you drew the rate. You can leave a rate unnamed by pressing the Esc key when the edit box for the rate name appears. You can add an arrowhead to the other end of the Rate by right-clicking on the handle (with the **Move/Size** tool selected) and checking the arrowhead box to indicate a two-way flow. Except in PLE and PLE Plus you can also change the default behavior by Right-Clicking on the **Rate** tool to open the options and unchecking **Query Valve Name** or **One Way Flow**.

IMPORTANT NOTE The presence or absence of an arrowhead on a Rate has no effect on the equation for that Rate in a simulation model. The equation for a Rate could allow it to decrease a Level, even though a single arrowhead indicates on the sketch that the Rate increases that Level. It is the equation that determines whether the Rate behave properly.

Constants, Auxiliaries (and Lookups, Data, and other variables) are usually entered with the **Variable** tool as words in a clear box or with shape **None**. Some diagramming conventions give Auxiliaries and Constants a Circle shape (usually with the name appearing below), but this adds clutter to a diagram without increasing the information conveyed. Having different shapes for each variable type displayed does increase the information conveyed, but no standard conventions exist for doing this and the visual clutter can be extensive. For building simulation models we recommend, and will use in this documentation, diagrams that contain Levels in boxes, Rates with bowtie valves and all other variables unadorned.

That said, except for PLE and PLE Plus you can add more tools to the Sketch toolset and change the behavior of the tools so that the **Variable** Sketch tool will generate shapes such as **Circles** or **Diamonds**.

Sketching the Rabbit Model (pop.mdl)

✍ Start Vensim.

✍ Click the **New Model** button, or select the menu item **File>New Model…**

✍ In the Model Settings dialog (Time Bounds tab) type 30 for FINAL TIME, type (or select from the drop down box) 0.125 for TIME STEP. Click on the dropdown box for **Units for Time**, and select **Year**. Click on **OK** (or press Enter).

✍ Select the **Box Variable** tool and click somewhere in the middle of the sketch. Type the name *`Population`*, and press the Enter key.

✍ Select the **Rate** tool. Click once (single click and release of the mouse button) about 2 inches (5 cm) to the left of the Level *`Population`*, then move the cursor on top of *`Population`* and click once again. Type the name *`births`*, and press Enter.

✍ Click once on the Level *`Population`* then move the cursor about 2 inches (5 cm) right and click again. Type the name *`deaths`*, and press Enter.

✍ Select the **Variable** tool. Click on the sketch below *`births`*, type *`birth rate`* and press Enter. Click on the sketch below *`deaths`*, type *`average lifetime`* and press Enter.

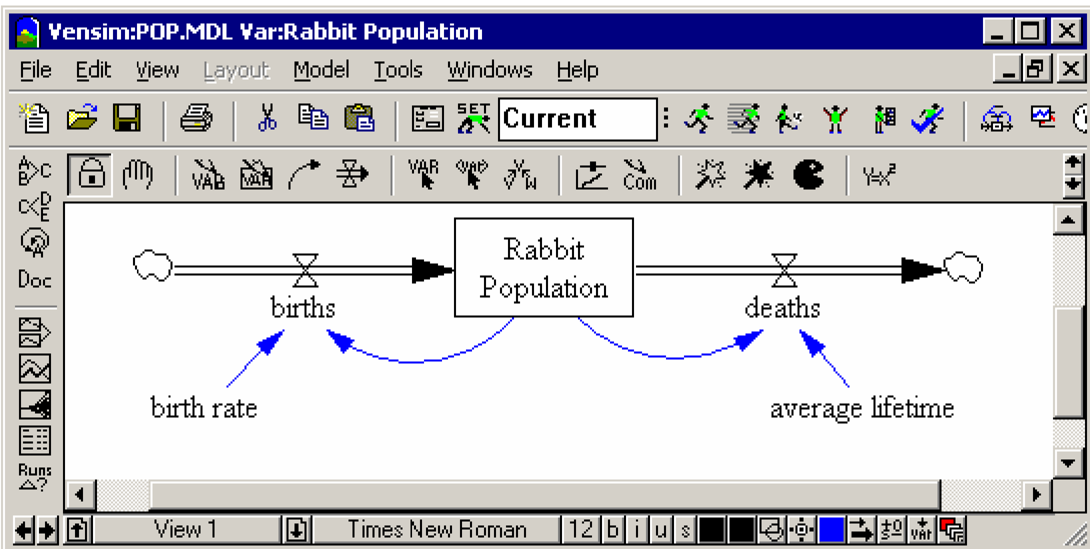
✍ Select the **Arrow** tool, click once on `birth rate` then once on `births`. Click once on `average lifetime`, then once on `deaths`.

✍ Click once on `Population`, then once on the sketch a little below and left of `Population`, then once on `births`.

✍ Click once on `Population`, then once on the sketch a little below and right of `Population`, then once on `deaths`.

✍ Click the **Save** button and save your model in the directory *guide\chap06*. Name your model (we call it *pop.mdl*).

The structure of the Population model is now complete, as shown in the figure below. A positive feedback loop from `Population` to `births` increases `Population`, and a negative feedback loop from `deaths` decreases `Population`.



Writing Equations

The model is now structurally complete. However, if you were to try to simulate it you would get a message saying that the model has errors and can't be simulated. In order to simulate, it needs a set of equations that describe each relationship. These equations are simple algebraic expressions, defining one variable in terms of others that are causally connected. For example:

```
births = Population * birth rate
```

Looking at the sketch view, `birth rate` has no causes; it is a Constant in the model. This Constant has a numerical value which we will fill in later.

We will fill in units of measurement for each equation we enter. Units allow us to check for dimensional consistency among all the equations. Dimensional consistency is important as a formal check of correct model structure. We will use the menu item **Model>Units Check** (Ctrl + U) to check the whole model after we have added all the equations.

✍ Click on the **Equations** tool.

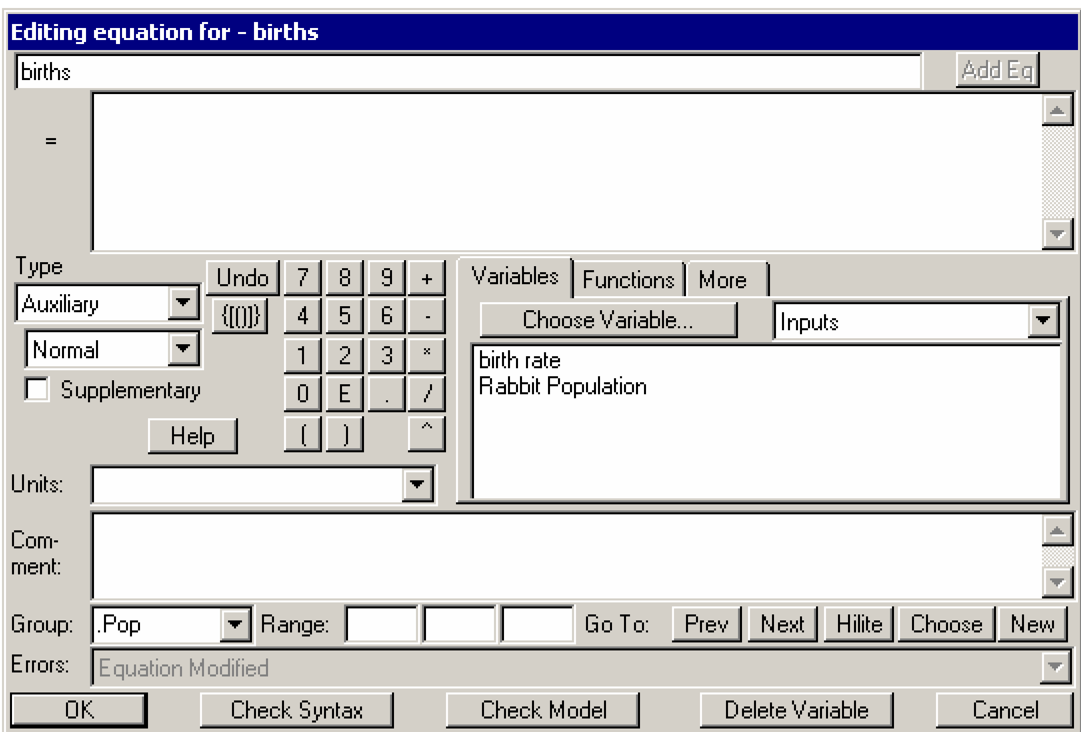
All the variables in the model will turn black. The highlights are used as a visible checklist of completeness. The highlights indicate which variables still require equations or have incomplete equations. As you complete the equations for each of the variables, the highlights will disappear. The menu item **Model>Check** (Ctrl + T) or the **Check Model** button in the Equation Editor also checks and displays what remains to be done.

✍ Click on the variable ***births***.

The Equation Editor will open. The exact appearance of the Equation Editor will depend on which Vensim configuration you are using. Vensim PLE and PLE Plus have a simplified Equation Editor, though the main elements are the same.

Variable Type: Auxiliary

The top of the editor has the name of the variable we clicked on: ***births***. The dropdown list box on the left shows the type of variable: **Auxiliary**. Vensim considers rates and auxiliaries to be the same type of variable. Click on the dropdown arrow to see the other types. Make sure that Auxiliary is still selected when you leave the list. Put the cursor in the equation editing box (next to the = sign).



✍ Complete the equation for births as below (in the editing box)

EITHER

✍ By typing Population * birth rate

OR

✍ Click on the variable *Population* in the **Variables** list (in the middle of the Equation Editor), then type the * symbol (or click on it in the Equation Editor keypad), then click on *birth rate* in the **Variables** list.

Spaces and new lines can be added to the equation for increased clarity, but are not necessary. To add a new line hold down the Ctrl key and press Enter. Note that you can't use a new line inside of a variable name.

Now we will add the units of measurement for *births*.

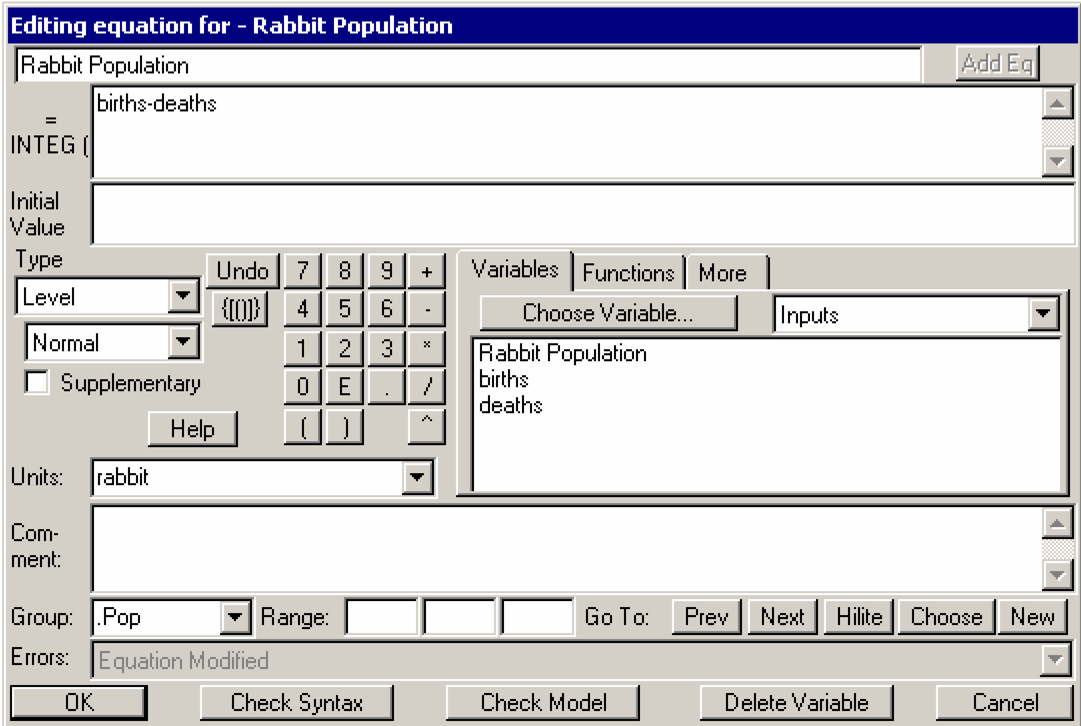
✍ Type in the units of measurement *rabbit/Year* in the **Units:** box. This indicates that we measure the rate of *births* in rabbit per Year. Click on **OK** or press Enter.

If the model structure and equation agree and there are no syntax errors in the equation, the dialog box will disappear. If there are problems with the structure or equations, you will receive an error message indicating what is wrong.

Variable Type: Level

✍ Click on *Population*.

The Equation Editor opens and is slightly different from what we saw with the variable *births*.



The dropdown list box on the left shows the type of variable: **Level**. Left of the equation editing box is the INTEG function that defines a Level (integrating the variable over time). An equation is already present in the equation editing box. Because we connected rates with the names *births* and *deaths* to the Level, Vensim automatically enters the rates to the Level equation. Rates constructed by clicking first outside, and then on the Level are considered positive (inwards) flows; rates constructed by clicking on the Level, and then outside the Level are considered negative (draining) flows. If the rates are drawn in a different direction the sign can be changed in the Equation Editor. The equation for this Level is correct, *births* add to *Population*, *deaths* subtract from *Population*, so we need not change them.

The Equation Editor for a Level has an extra editing box to set the starting or initial value; the cursor is placed there.

✍ In the **Initial Value** editing box, type in 1000.

This value is the initial number of rabbits at the start of the simulation (time zero).

✍ Type in the units of *rabbit* in the Units box. Click on **OK** or press Enter.

✍ Click on *birth rate*. Type in the numbers 0.125 in the editing box.

✍ Type in the units *fraction/Year* (if you prefer, enter instead 1/Year), this means that the fractional birth rate is measured in fraction (of rabbits) per year. Another way of saying this is (rabbits born / rabbit population) / Year *(rabbit/rabbit)/Year = fraction/Year* (rabbit cancels out). Click **OK** or press Enter.

✍ Complete the remaining two equations as they are shown in the Equations listing below.

All model variables should appear clear in the sketch after all the equations are entered. Units that have been previously entered can be selected again by clicking on the arrow in the Units box and selecting from the dropdown list.

pop.mdl Equations

```
average lifetime = 8
Units: Year

birth rate = 0.125
Units: fraction/Year

births = Population * birth rate
Units: rabbit/Year

deaths = Population / average lifetime
Units: rabbit/Year

Population = INTEG(
   births - deaths,
         1000)
Units: rabbit
```

Checking for Model Syntax and Units Errors

Before we simulate the model, we should check it for errors in equations and units.

✍ Select **Model>Check Model** from the menu (or press Ctrl + T); you should get an information box saying "Model is OK."

If the model has errors, the Equation Editor will open with the variable containing the error. Check that the equation uses all the inputs and looks the same as in the listing above. Check that the structure of your model is the same as in the diagram above.

✍ Select **Model>Units Check** from the menu (or press Ctrl + U); you should get an information box saying "Units are AOK."

If a units error is generated, read the Output window to see which variables are failing the check. Open the **Equation Editor** on each variable and check the units against the list above. Units that do not check out often indicate poor or incorrect equation formulation.

NOTE The Units Check feature can also be accessed from the Analysis tools except in PLE and PLE Plus. You need to modify the toolset (menu **Tools>Analysis Toolset>Modify…**) and add the **Units** tool, or open the Analysis toolset *default2.vts* which contains the **Units** tool.

Units Equivalents (Synonyms)

When entering units, you may want to enter the plural form as well as the singular. For example, rabbits as well as rabbit. This will fail a Units check because Vensim does not see the plural form as the same word. The easiest way to accommodate this is to set rabbit and rabbits as equivalent units, or synonyms.

✍ Select menu **Model>Settings...** and click the **Units Equiv** tab. In the editing box, type `rabbit,rabbits` then click the button **Add Editing**. Then click **OK** to close the dialog.

Simulating the Model

✍ Double click on the **Runname** editing box on the Toolbar and type *equilib* for the first run name.

✍ Click on the **Simulate** button (or just press Enter when the cursor is in the **Runname** box).

The model will simulate. This model is so small that is may seem like nothing happened, if you are not sure click on the **Simulate** button again. If you are asked if you want to overwrite the dataset the model did simulate.

Model Analysis

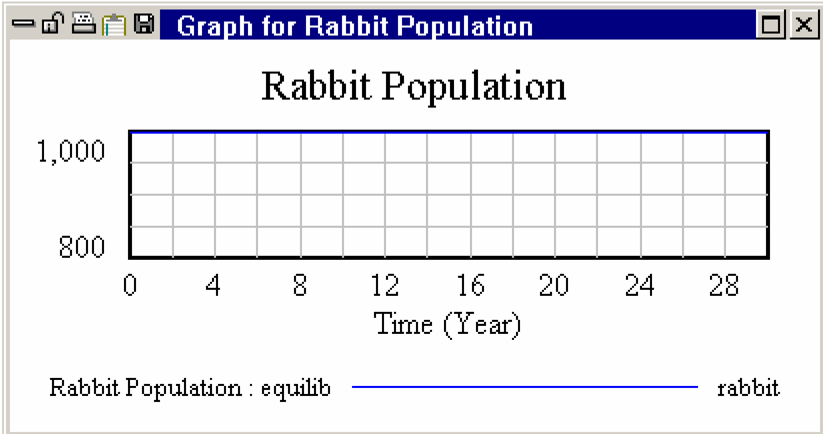
This model has been designed to show equilibrium conditions in the rabbit population. The constants *`birth rate`* and *`average lifetime`* are both set to generate a rate of flow of 12.5 % of *`Population`*, therefore each feedback loop is balanced numerically, resulting in no change in the value of *`Population`*.

Graph and Table tools

✍ Click on the Level *`Population`* in the sketch.

This selects it as the Workbench Variable; another way to do this is to choose *`Population`* from the Variable Selection control. Check the title bar at the top of the Vensim window to see that *`Population`* is selected.

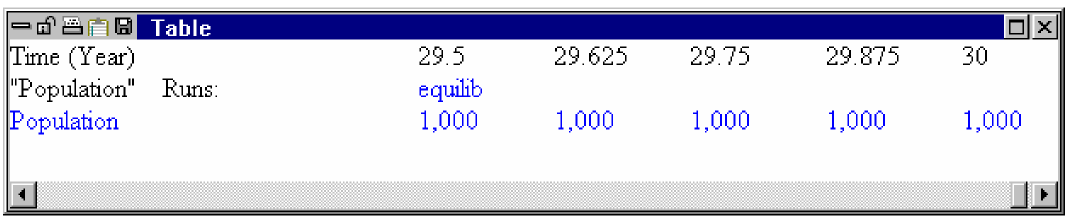
✍ Click on the **Graph** tool. A graph of *`Population`* is generated:



`Population` appears as a flat line at the top of the graph at 1000 rabbits. To check that no change is occurring:

✍ Click on the **Table** tool.

An Output window shows that `Population` is unchanging. Scroll the window to confirm that late in the simulation, `Population` is still 1000.



Time (Year)		29.5	29.625	29.75	29.875	30
"Population"	Runs:	equilib				
Population		1,000	1,000	1,000	1,000	1,000

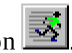
Comparing Simulations

A key feature of Vensim is the ability to do multiple simulations on a model under different conditions to test the impact that changes in constants (or lookups) have on model behavior. Vensim also stores all the data for all variables for each simulation run, so that you can easily access information about the behavior of any variable in any run. Experiments are performed by *temporarily* changing Constant or Lookup values and then simulating the model. This way, your underlying model stays the same, an unchanging reference point.

Exponential Growth

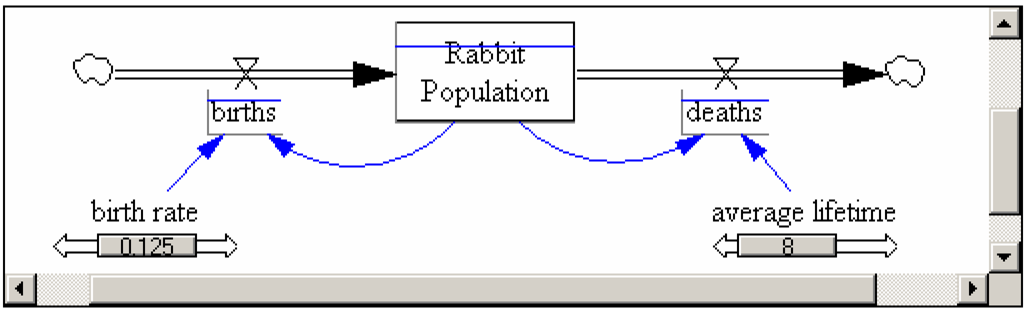
Now that we are satisfied that we have equilibrium conditions, let us make changes to the model constants to generate unconstrained growth. This is one of the simplest possible dynamic behaviors, known as exponential growth.

Simulation Experiments

✍ Click on the **SyntheSim** button . Answer yes to whether you want to overwrite the existing run.

The toolbar will change to

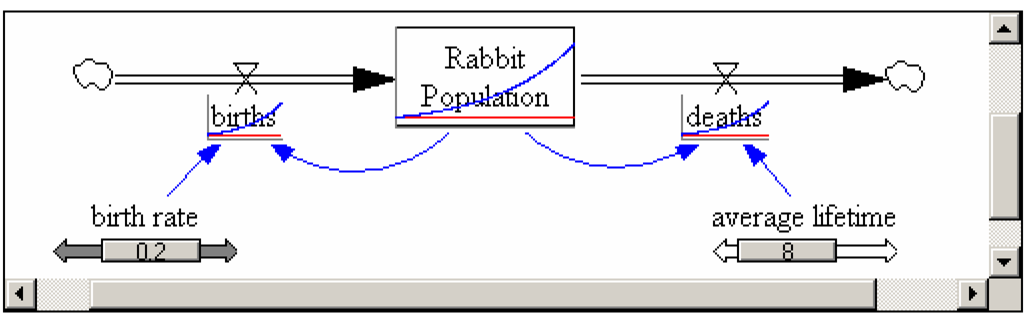
and you will see variable behavior for all elements on the diagram.



✍ Click on the **Runname** editing box and replace *equilib* with the name *growth*.

✍ Drag the slider beneath `birth rate` up till it shows 0.2 as its value. If you have trouble getting the exact value let go of the mouse button and use the left and right arrow keys on the keyboard to move the slider in small increments.

Your sketch should look like:



The blue lines show the current run and the red line the *equilib* run results.

✍ Click on the **Stop** button to stop SyntheSim.

The behavior graphs and sliders will disappear from the sketch.

✍ Press the B key or select the menu item **View>Show Behavior** to see behavior graphs again.

You can toggle between seeing and not seeing behavior graphs by pressing the B key.

Alternative Simulation Setup

As an alternative to entering SyntheSim mode you can set up and perform simulation one at a time. For large models where the simulation takes more than a few seconds this is more practical than trying to enter SyntheSim mode.

✍ Click on the **Set up a Simulation** button 

The Toolbar changes to the simulation toolbar.

This toolbar has features specific to model simulation, allowing changes to the integration technique, and buttons to change model constants and lookups. You will also notice that constants in the sketch of this Population model turn into yellow words with blue background . Also, the sketch tools are grayed out, preventing work in the sketch window.

✍ If you have not already done so click on the **Runname** editing box and replace *equilib* with the name *growth*.

✍ Click on the variable ***`birth rate`*** (appearing blue/yellow in the sketch) and in the editing box type the value 0.2. Press the Enter key. This is a temporary change for this run only and does not permanently alter the value in your model.

✍ Click the **Simulate** button and the model will simulate. If you already created the run *growth* in SyntheSim mode you will be asked if you want to overwrite the existing dataset - answer yes.

✍ If you have not already done so press the B key or select the menu item **View>Show Behavior** to see behavior graphs.

Causes Strip Graph

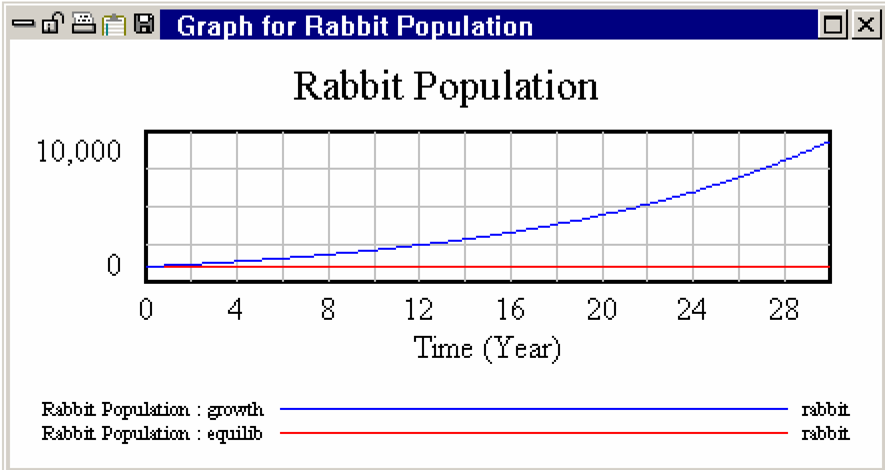
✍ Click on ***`Rabbit Population`*** to select it into the workbench.

✍ Click on the **Control Panel** button  to bring the Control Panel to the front. Click the **Datasets** tab to open the Datasets Control and check that both runs are loaded in the right hand column.

The last run you made (*growth*) is loaded first (at the top of the dialog). Most Analysis tools act on both datasets, allowing comparison of behavior from both runs.

✍ Click on the **Graph** tool.

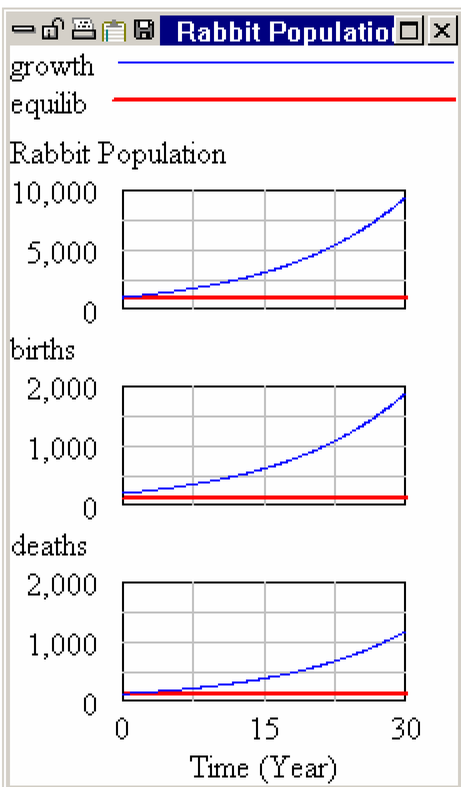
A graph will be generated showing both runs (see below).



✍ Press the Del key or click the **Close** button to remove the graph.

✍ Now click on the **Causes Strip** tool.

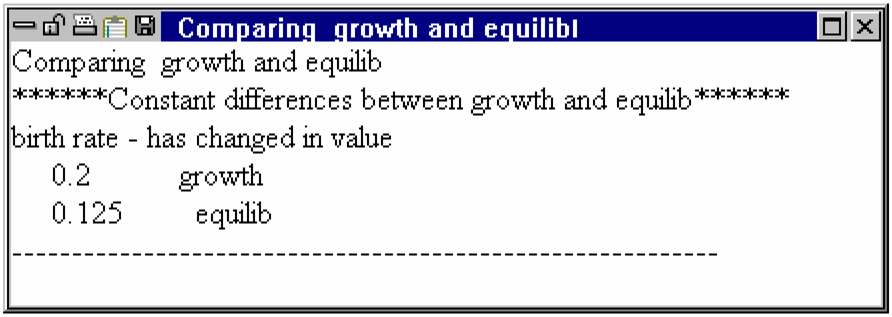
A strip graph is generated showing ***`Population`*** and its causes ***`births`*** and ***`deaths`***.



Runs Compare

To discover the differences between the first and second runs, we will use a tool that compares all Constant (and Lookup) differences. This tool acts on the first two loaded runs (check in the Datasets Control).

✍ Click on the **Runs Compare** tool. The text report below shows the differences in the Constant `birth rate` for runs *equilib* and *growth*.


Comparing growth and equilibl

Comparing growth and equilib

******Constant differences between growth and equilib******

birth rate - has changed in value

0.2 growth

0.125 equilib

`Population` grew in the *growth* run because the `birth rate` was set to a higher value than the equilibrium value. This made the positive feedback loop through `births` stronger than the negative feedback loop through `deaths`, resulting in `Population` growth over time.

✍ Select the Menu Item Windows>Close All Output.

All of the windows you have created using the Analysis tools will be closed.

Exponential Decay

Next, we will make changes to a model Constant to generate exponential decay or decline in the population. Like exponential growth, this is one of the simplest possible dynamic behaviors.

✍ Double click on the run name and replace *growth* with *decay*.

Do one of:

✍ Click on the SyntheSim button.

✍ Drag the Slider under `average lifetime` till it displays a value of 4.

✍ Click on the Stop Button

Or

✍ Click on the **Set Up a Simulation** button.

✍ Click on the variable `average lifetime` (appearing blue/yellow in the sketch) and in the editing box type the value `4`. Press the Enter key.

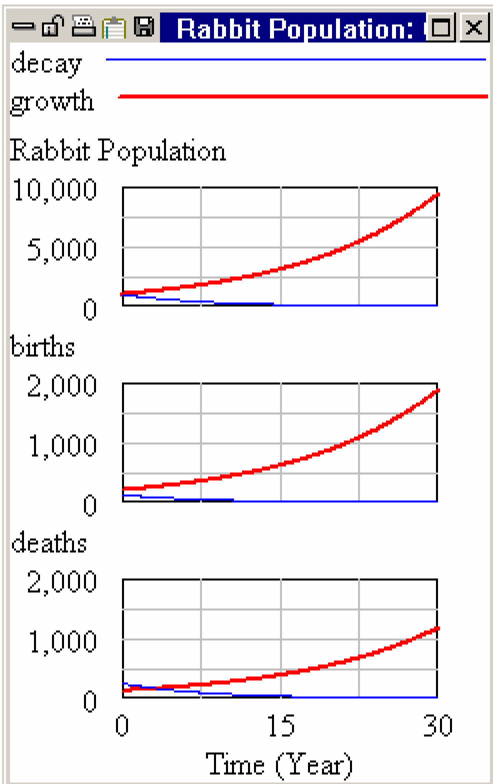
✍ Click the **Simulate** button and the model will simulate.

Both the above sets of steps result in the creation of the new run *decay*.

✍ Click on the **Graph** tool and compare the three runs.

✍ Click on the **Control Panel** button on the Main Toolbar. In the Datasets Control box, double click on the run *equilib* in the right box; this will unload the run so the Analysis tools will not examine it. Note that it can be reloaded just as easily. (You can also single click to select the run then use the **Move** button (<<) or (>>) to unload or load the dataset.)

✍ Click on the **Causes Strip** tool; a strip graph is generated showing `Population` and its causes — `births` and `deaths`, for the two last runs.



`Population` declines in the *decay* run because the `average lifetime` was set to a lower value than the equilibrium value. This made the negative feedback loop through `deaths` stronger than the positive feedback loop through `births`, resulting in `Population` decline over time.

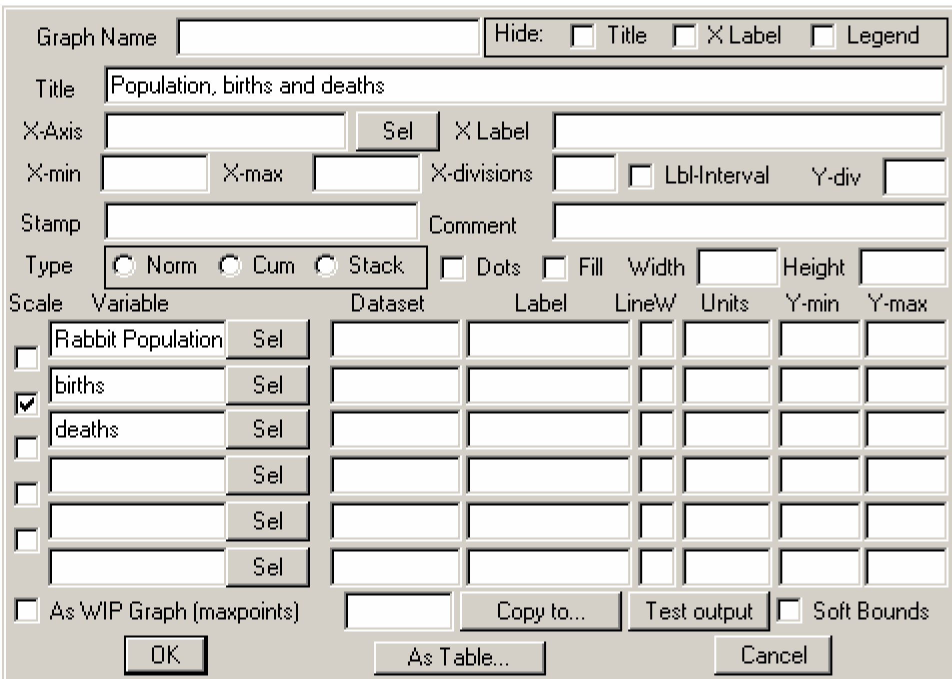
Input and Output Objects

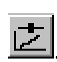
You can use Input Output Objects to imbed sliders, graphs and tables in a sketch. We will demonstrate this here by embedding a custom graph.

✍ Click on the **Control Panel** button and then select the **Graphs** tab.

- Click on the **New** button. The Custom Graph editor will open.
- Enter the title "Population, Births and Deaths."
- Click on the first **Sel** button and select ***`Rabbit Population`*** from the list.
- Click on the second **Sel** button and select ***`births`*** from the list.
- Click on the third **Sel** button and select ***`deaths`*** from the list.
- Click on the **Scale** checkbox to the left between ***`births`*** and ***`deaths`***.

The custom graph editor should look like:

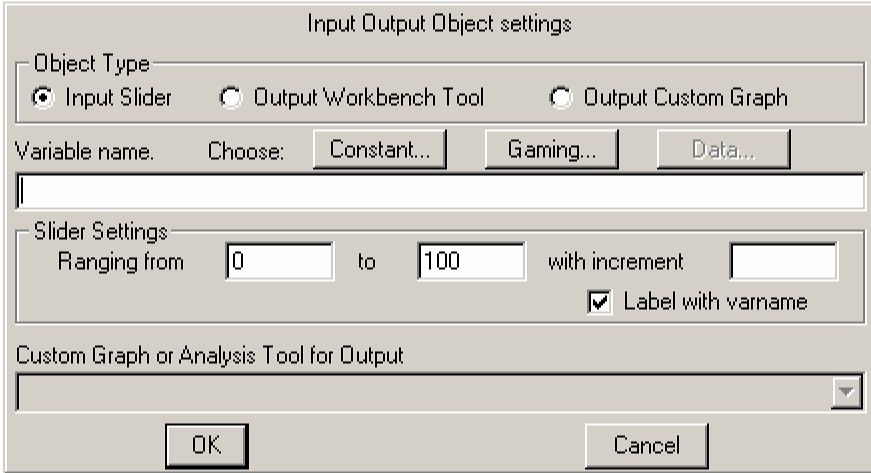


- Click on **OK**.
- Select the **Input Output Object** sketch tool .

NOTE If you forgot to stop SyntheSim mode the Sketch toolbar will be grayed. Click on the **Stop** button then try the above step again.

- Click on an empty area of your sketch, below or to the right of the diagram.

The Input Output Object dialog will open.



- Click on the **Output Custom Graph** radio button.
- From the dropdown below **Custom Graph or Analysis Tool for Output** select POPULATION_,BIRTH_AND_D - it should be the only entry available.
- Click on **OK**.
- Position and size the graph to fit your taste.
- Double click on the run name and replace decay with experiment.
- Click on the **SyntheSim** button.
- Experiment with moving the sliders around and seeing the results on the graph.

Chapter 12 Input and Output Controls has more details on customizing the information displayed on the sketch.

7 Functions and Simulation Errors

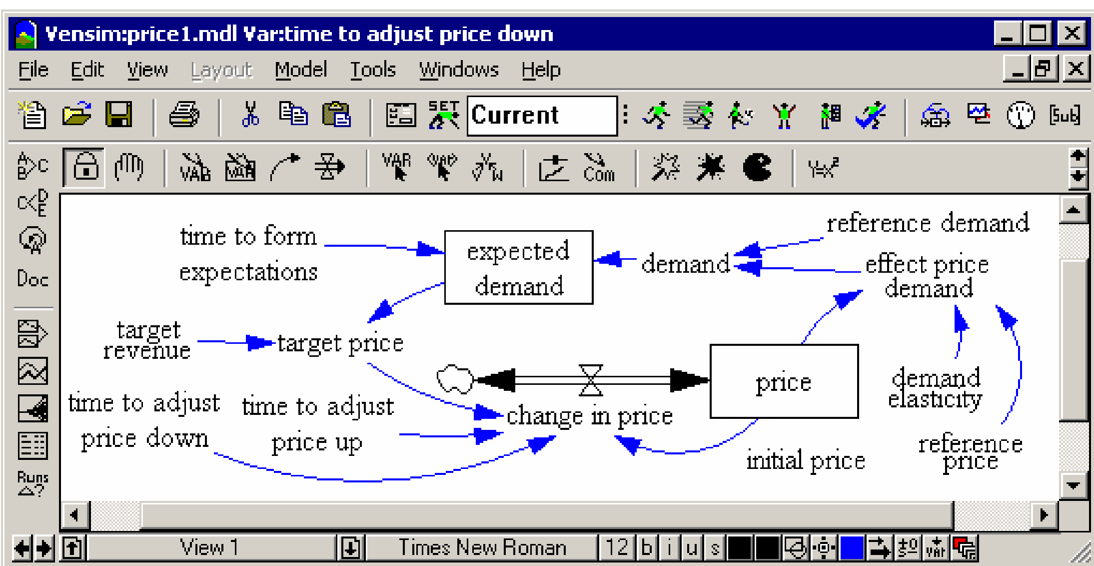
The population model presented in the previous chapter is a simple model that uses only multiplication and division in its equations. While addition, subtraction, multiplication and division are the most common components of equations, sometimes it is necessary to use different types of relationships. Vensim has a number of functions that can be used in equations and in this Chapter we will work through the process of adding these functions to equations. In the next chapter we will cover Lookup functions which allow you to specify functional relationships by drawing a curve.

In addition to exploring the use of functions in this chapter we will look at how to deal with what are called "Floating Point Errors." These are errors that occur when the numbers in a model get too big, there is a division by zero, or a function is given an argument that is out of range.

Price and Demand

As an example we will use a very simple model fragment with some structure around price and demand. The example below is not very complete. The pricing decision is made to achieve a certain revenue goal with the expectation that demand will not be changed by price, while demand does in fact depend on price. This model embodies a single dynamic hypothesis, and is interesting to experiment in isolation even though its behavior, as you will see, is quite unrealistic.

✍ Create the diagram pictured below or open the model *guide\chap03\complete\price1.mdl*.



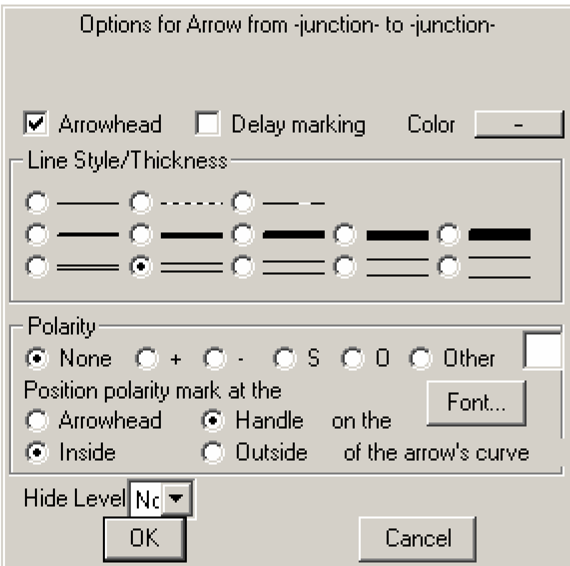
Use the steps outlined in the previous chapter to create this model. There are two elements that are different in this diagram from what was done there. The first is the two-way flow for ***change in price***. The second is the absence of flows into the Level ***expected demand***. To draw the two way flow:

✍ Use the Rate tool to draw a flow from the left of ***price*** into ***price*** and label the rage ***change in price***.

✍ Click on the Move/Size tool.

✍ With the Right Click or Control Click on the small handle appearing in the middle of the pipe from the cloud to ***change in price***.

✍ In the dialog that appears check the box **Arrowhead**.



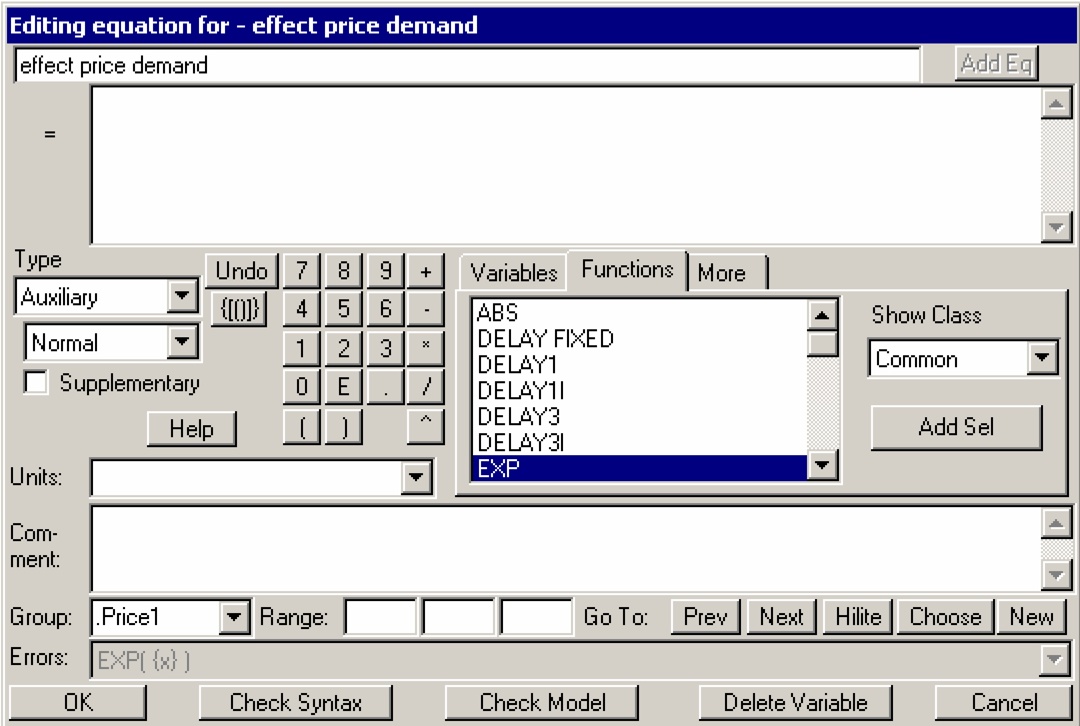
The creation of the arrows into the level ***expected demand*** is done in the same way that all other arrows are created. Some more comments on the way this Level is depicted will be made when the equation for it is created.

Entering a Function

✍ Select the Equation edit tool.

✍ Click on effect price demand.

✍ Click on the **Functions** tab to the right



You will see a list of available functions displayed. If you click on any item in the list it will display the functions arguments at the bottom of the Equation editor. You can also type the first letter in a functions name to scroll the list to functions beginning with that letter. The up and down arrow keys will move through the list displaying argument information at the bottom of the window.

- Click on EXP in the list of functions, scrolling if necessary.
- Press the Enter key.

In the editing window you should see EXP({x}) with the {x} highlighted.

- Press the - key or click on **-** on the buttons.
- Click on the **Variables** tab and click on the variable ***`demand elasticity`***.
- Press the * key or click on * on the button.
- Click on the **Functions** tab.
- Click on LN in the list of function (pressing the L key is a good shortcut to get there) and press Enter.
- Click on the **Variables** tab.
- Click on ***`price`***.

✍ Press the / key or click on the / button.

✍ Click on *reference price* in the variable list.

Your equation should read:

```
effect price demand=
   EXP(-demand elasticity*LN(price/reference price))
```

You can always type this in instead of working through the list of functions and sometimes this is easier to do.

A couple of notes on this equation are important. In this equation LN is the function for a natural logarithm and EXP(x) is the function that takes the special number *e* (about 2.72) to the power x. This equation could also be written as either one of:

```
effect price demand=
   POWER(price/reference price,-demand elasticity*)
```

```
effect price demand=
   (price/reference price)^(-demand elasticity)
```

(In fact, for this equation there are three additional formulas that involve switching *price* and *reference price* then changing the sign on *demand elasticity*.)

This equation is a standard constant elasticity demand curve. The use of *price/reference price* is a normalization that also prevents any units errors from occurring. We will discuss normalizations further in the next chapter.

Independent of which functions are used, the method for entering them is the same. You can either type them in, or select them from the list.

IF THEN ELSE

It is pretty common to need to be able to switch between alternative formulations based on some condition. The IF THEN ELSE function allows you to do this. It is implemented as a function in Vensim for two reasons. First, like other functions it returns a value. Second the statement

```
d = IF a THEN b ELSE c
```

actually assigns the variable "*IF a THEN b ELSE c*" to the variable *d*. In order for Vensim to differentiate variables from statements something such as :IF: a :THEN: b :ELSE: c would need to be used and this seems to create more distraction than it resolves. In Vensim this assignment takes the form

```
d = IF THEN ELSE(a,b,c)
```

✍ With the Equation edit tool selected click on *Change in Price*.

✍ Click on the **Functions** tab then select IF THEN ELSE from the list and press Enter.

✍ Click on the **Variables** tab.

- ✍ Click on ***target price*** press the > key (or click on the **More** tab and click on the **>** button) then click on ***price***.
- ✍ Double click on {ontrue} in the equation to highlight it.
- ✍ Hold down the Ctrl key and press Enter.
- ✍ Press the left parenthesis (key, click on ***target price*** press the minus - key, click on ***price***, press the right parenthesis) key, press the divide / key then click on ***time to adjust price up***.
- ✍ Double click on {onfalse} in the equation to highlight it.
- ✍ Hold down the Ctrl key and press Enter.
- ✍ Press the left parenthesis (key, click on ***target price*** press the minus - key, click on ***price***, press the right parenthesis) key, press the divided / key then click on ***time to adjust price down***.

Your equation should read:

```
change in price=
   IF THEN ELSE(price > target price,
   (target price-price)/time to adjust price up,
   (target price-price)/time to adjust price down)
```

The function arguments appear in such a way that double clicking on them will allow you to replace them. Because the equation arguments are enclosed in Curly braces {} they are actually treated as comments and can even be left in, though this is rarely desirable. The use of Ctrl+Enter puts line breaks into the equations to keep them more readable.

SMOOTH

The SMOOTH function is commonly used to take time averages and represent expectations. It is different from LN, EXP and IF THEN ELSE in that it has time behavior built into it. That is, if you know what value x takes on then you can compute EXP(x), but just knowing x does not tell you the value of SMOOTH(x,4), you also need to know what value the SMOOTH previously had. This is because the SMOOTH function has a level implicitly built into it.

We will write the equation

```
expected demand=SMOOTH(demand, time to form expectations)
```

This equation is exactly the same as:

```
expected demand=INTEG((demand-expected demand)/
         time to form expectations,
         demand)
```

When you use a SMOOTH function Vensim actually creates two variables. One, ***expected demand***, is treated as an Auxiliary that is equal to a Level mad up by Vensim called ***#SMOOTH(demand,timetoformexpectations)#***. Normally Vensim will hide such variables internally, but you can display them by checking **Macro Variables: Show** in the **Settings** tab of the Global Options dialog (using the **Tools>Options** menu item). This is not available in Vensim PLE or PLE Plus.

To enter the smooth equation:

✍ Open the Equation Editor on *`expected demand`*.

Because *`expected demand`* was entered in a box Vensim expects it to be a Level and the Equation Editor will open with this type selected. To enter the SMOOTH equation, however, we want to change this type Auxiliary. We still want to keep the box around the variable, however, because of the hidden level it contains.

✍ Change the equation **Type** from Level to Auxiliary using the dropdown list on the left.

✍ Click on the **Functions** tab, select the function SMOOTH and press Enter.

✍ Click on the **Variables** tab and click on *`demand`*.

✍ Double click on {*`stime`*} then click on *`time to form expectations`* in the variable list.

✍ Add the units *`Box/Month`*, the same as the units for *`demand`*.

✍ Click on OK to close the Equation editor.

There are a number of functions which, like smooth, introduce hidden variables and dynamics. These include DELAY1, DELAY1I, DELAY3, DELAY3I, FORECAST, SMOOTH3, SMOOTH3I, SMOOTHI and TREND. It is often helpful to put variables using these functions in a box to emphasize the hidden Levels. For the SMOOTH function itself, since it is so simple, it is often clearer to use the alternative INTEG formula instead of SMOOTH, but this is a matter of taste.

Whether it is written as SMOOTH(x,t) or INTEG((x-sx)/t,x), diagrammatically we have a Level with information arrows instead of rates coming in. There are different schools of thought on whether or not Levels should be allowed to appear without rates. We will often use the convention shown here for informational concepts, such as *`expected demand`*. In this case, there is no physical process that increases or decreases an information concept, it simply adjusts in response to pressures and imbalances. Vensim itself will let you draw both rates and informational arrows going into levels, so you need to choose a convention that will be clear to those you want to show your work to.

STEP

✍ Open the Equation editor on *`target revenue`*.

✍ Type "10000+" to begin the equation.

✍ Click on the **Functions** Tab.

✍ Select STEP from the list and press the Enter key.

✍ Type "5000."

✍ Double click on {stime} in the equation.

✍ Type "10.".

✍ Enter the units *`$/Month`* and click on **OK**.

The final equation should read

```
target revenue=10000+STEP(5000,10)
```

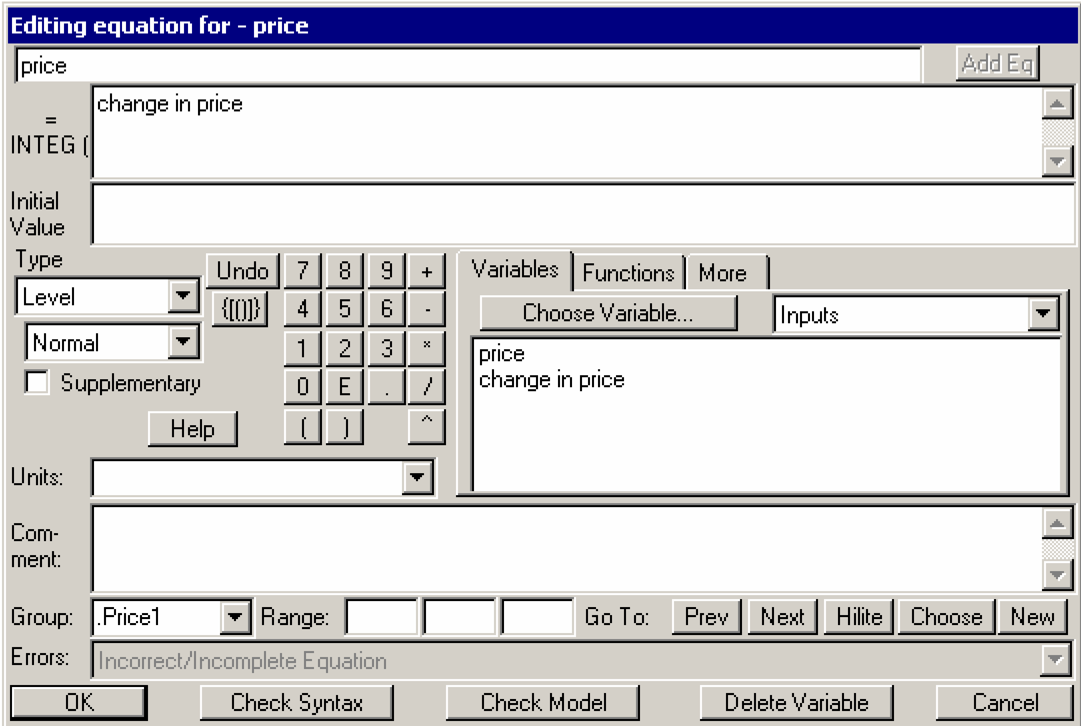
The STEP function is one of the most commonly used input functions. This function returns 0 until *`Time`* reaches {stime} and then it returns {sheight}. In our example it will return 0 till time 10 then it will return 5000 so that *`target revenue`* will start at 10000, remain constant till time 10 and then jump to 15000. The STEP function is important because a step change is a very good way to get a model to show the behavior modes it can generate. For example in a supply chain model, you can use a step change in demand to take the supply

There are a number of functions related to the STEP function. PULSE, PULSE TRAIN, RAMP and the RANDOM... functions. All of these functions return a value that is different at different times. The RANDOM... functions return values that change randomly from time to time, though they will be the same from run to run so that you can replicate results.

Another common source of time varying behavior is Data - this is discussed in Chapter 16 "Using Data in Models." You can also use the variable *`Time`* as an input to other functions such as SIN to get different types of behavior.

Initial Conditions

The initial value for *`Price`* is given by *`initial price`* but there is no arrow from *`initial price`* into *`Price`*. When you open the Equation editor on *`Price`* you will not see *`initial price`* listed as in input.



- ✍ Click on the **Choose Variable...** button (the **Choose Initial Variable** button in Vensim PLE and PLE Plus).
- ✍ Select ***initial price*** from the list presented and click on **OK**.
- ✍ Enter the Units as ***$/Box***.
- ✍ Click on **OK** to close the Equation editor.

Note that if you had drawn an arrow from ***initial price*** into ***Price*** then ***initial price*** would have appeared on the input list. However, when you close the Equation editor the arrow will disappear because initial causes are not, by default, shown on model diagrams. If you want to keep the initial arrow select the menu item **Model>Settings**, click on the check the **Sketch Appearance** tab and check **Show initial causes on model diagrams.**

Model Equations

The following are the complete equations for this model. Most of these are quite straightforward. The model *guide\chap07\complete\price2.mdl* contains these equations.

```
change in price=
   IF THEN ELSE(target price > price,
   (target price - price)/time to adjust price up,
```

```
   (target price-price)/time to adjust price down)
   Units: $/Box/Month

demand = reference demand * effect price demand
Units: Box/Month

demand elasticity= 1
Units: Dmnl

effect price demand=
   EXP(-demand elasticity * LN(price/reference price))
Units: Dmnl

expected demand=SMOOTH(demand,time to form expectations)
Units: Box/Month

FINAL TIME  = 100
Units: Month

initial price= 100
Units: $/Box

INITIAL TIME  = 0
Units: Month

price= INTEG (change in price,
         initial price)
Units: $/Box

reference demand=100
Units: Box/Month

reference price=100
Units: $/Box

SAVEPER  = TIME STEP
Units: Month

target price=target revenue/expected demand
Units: $/Box

target revenue=10000+STEP(5000,10)
Units: $/Month

TIME STEP  = 1
Units: Month

time to adjust price down=10
Units: Month

time to adjust price up=10
Units: Month
time to form expectations=2
Units: Month
```

Simulation Errors

The model you have created should be ready to simulate.

✍ Select the menu item **Model>Check Model** or use the shortcut key combination Ctrl+T.

You should get a message saying "Model is OK." If you do not get this message the Equation Editor will open on an equation which has a problem. You will need to correct the errors reported before proceeding.

✍ Select the menu item **Model>Units Check** or use the shortcut key combination Ctrl+U.

You should get a message saying "Units are A.O.K." If you do not get this message you will need to correct the errors.

Next we will intentionally cause this model to generate some simulation errors and look at how to resolve these problems.

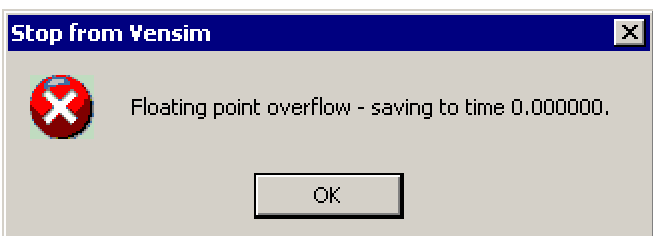
Errors at Time Zero

✍ Click on the **Setup a Simulation** button.

✍ Click on *`Reference Demand`*, type in the value 0 (zero) and press Enter.

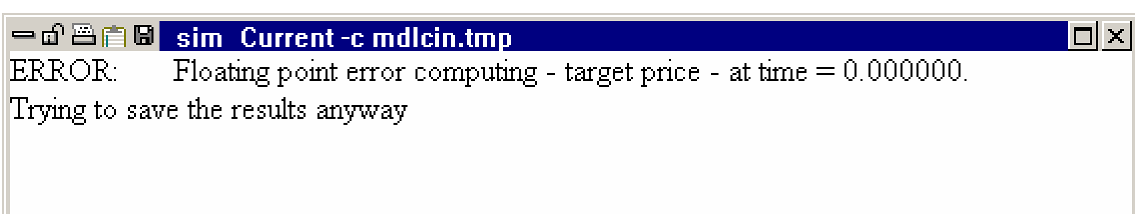
✍ Click on the **Simulate** button.

You should get the error message



✍ Click on **OK**.

There will be an error window that appears.



This window tells you what variable was being computed at what time when the problem was detected. Because the error occurred at time 0, it is only practical to trace it using the Table tool.

✍ Click on ***target price*** in the error window to select it into the workbench.

If you are using Vensim PLE or PLE Plus:

✍ Click on the Table tool.

✍ Click on the Causes Tree tool.

✍ Click on ***expected demand*** in the Causes tree output.

✍ Click on the Table tool.

✍ Click on ***target revenue*** in the Causes tree output.

✍ Click on the Table tool.

or if you are Using Vensim Standard, Runtime, Professional or DSS

✍ Right click on the Table tool.

✍ Under **Show Link** click on **Cause** and click on **OK**.

✍ Click on the Table tool.

then

You will see output that looks something like:

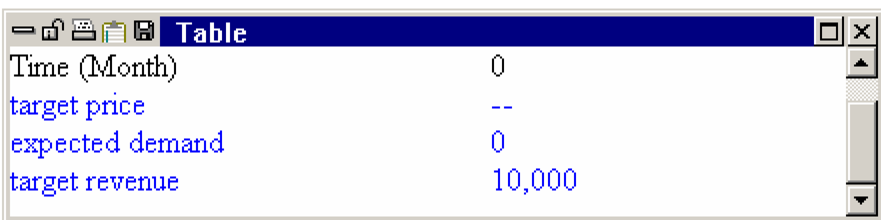


Table	
Time (Month)	0
target price	--
expected demand	0
target revenue	10,000

Next to ***target price*** you will see the value "--." This indicates that ***target price*** was never successfully computed. Both ***expected demand*** and ***target revenue*** were computed and their values are shown. Since the equation for ***target price*** is:

```
target price=target revenue/expected demand
```

It is clear why there is an error - division by 0. To determine why ***expected demand*** is 0 we can continue to trace causes. You should be able to demonstrate quite quickly that ***expected demand*** is 0 because ***demand*** is 0 because ***reference demand*** is 0.

We can fix the equation for ***target price*** so that it will not cause and error even when ***expected demand*** is 0. This is done in "Model Revisions" below.

Errors During the Simulation

✍ Click on the **Setup a Simulation** button.

- Click on ***demand elasticity***, type in the value 2 (two) and press Enter.
- Click on ***time to form expectations***, type in the value 1 (one) and press Enter.
- Click on the **Simulate** button and answer **Yes** to the query about overwriting.

You should see about the same thing that we saw last time except this time the message should be:

```
ERROR: Floating point error computing - target price - at time = 30.000000.
Trying to save the results anyway
```

You can apply same analysis technique we used previously to understand the problem. Here is a table of ***target price*** and its causes:

Table

Time (Month)	29	30
target price	2.26674e+027	2.26674e+027
expected demand	6.61744e-024	0
target revenue	15,000	15,000

NOTE In order to see values for all times you will need to delete the Table tool output and then click on the Table tool again.

There are two things to notice about this output. First the value for ***target price*** at time 29 and 30 is the same even though the inputs are different. When an error occurs during simulation the values reported for variables that were not successfully computed will simply be those of the previous time. Second, the reason for the error is the same as it was in the previous case.

Since the error occurred during the simulation we can use the Causes strip to understand what lead to the problem.

- Click on ***target price*** to select it into the workbench.
- Click on the Causes strip.
- Trace the causes of the behavior using the Causes strip through the sequence ***expected demand***, ***demand***, ***effect price demand***, ***price***, ***change in price*** and back to ***target price***.

We have traced around a positive feedback loop that is loosely higher price, lower demand, lower expected sales higher price needed to meet revenue targets. This is exponential growth and it causes an error because it is occurring so quickly.

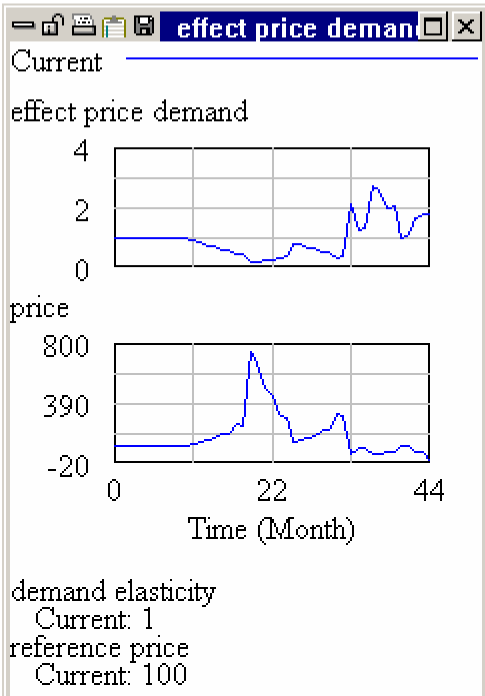
Mysterious Oscillations

- Click on the **Setup a Simulation** button.
- Click on ***time to form expectations***, type in the value 0.5 and press Enter.

✍ Click on the **Simulate** button and answer **Yes** to the query about overwriting.

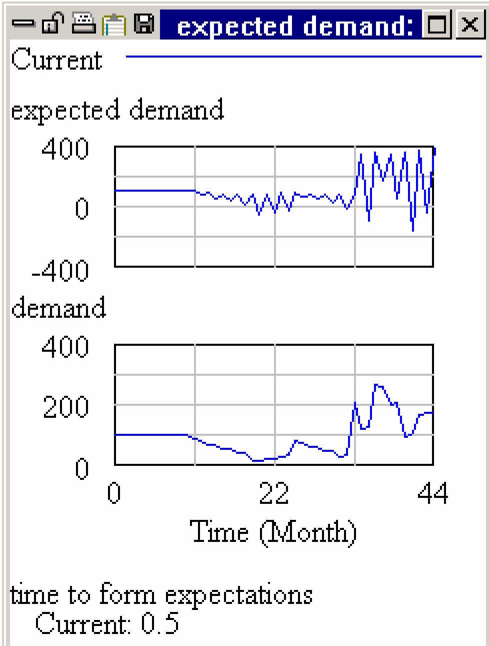
You should be informed of an error occurring while computing ***effect price demand*** at time 44.

✍ Click on ***effect price demand*** and click on the Causes Strip.



The values jump around a lot, but the source of the error is the negative value that ***price*** takes on at the end.

✍ Trace the causes of the behavior using the Causes strip through the sequence ***effect price demand***, ***price***, ***change in price***, ***target price***, and then ***expected demand***.



`expected demand` goes negative, even through `demand` is always positive. This happens because we set the time constant for changing `expected demand` to a value smaller than `TIME STEP`.

Odd behavior such as that displayed here often results because of a problem like this. If you see this type of oscillation and can not find the problem try changing TIME STEP to a smaller number. If this makes a big difference in behavior you probably have a time constant somewhere that is too short. Note that such time constants might not be explicitly defined in the way `expected demand` is.

Errors Integrating

There is one other error message that is displayed. This takes the form:

```
ERROR:
Floating point error integrating - Big Level - at time = 340.0
Trying to save the results anyway
```

This error indicates that even though the all the equations could be computed when the rates were added into the named level it got too big. This type of error can usually be traced to a positive feedback loop much like our example on Errors During the Simulation.

Model Revisions

There is one correction we might want to make for this model to be more robust. We can replace the equation for ***target price*** with something that will behave correctly even when ***expected demand*** is 0. The model with this correction is saved in *guide\chap07\complete\price3.mdl*.

XIDZ

✍ Add the variable ***maximum price*** to the sketch near ***target price***.

✍ Draw and arrow from ***maximum price*** to ***target price***.

✍ Select the Equation edit tool and click on ***target price*** and enter the new equation:

```
target price=MIN(
   XIDZ(target revenue, expected demand, maximum price),
   maximum price)
Units: $/Box
```

✍ Add the equation for ***maximum price***.

```
maximum price=25000
Units: $/Box
```

The XIDZ function performs division except when that division would be by 0 (actually a number smaller than 1.0E-6 in magnitude) in which case it returns the third argument. In other words

```
XIDZ(target revenue, expected demand, maximum price)
```

is the same as

```
target revenue/expected demand
```

except when the ***expected demand*** is 0. In this case the first returns ***maximum price*** and the second causes an error.

The XIDZ function is very useful when you suspect that variables could legitimately take on 0 value and you still want simulations to continue. A closely related function is ZIDZ which returns 0 when dividing by 0.

8 Building a Function with Lookups

In the previous chapter we developed a model that used a number of functions. While the built in functions can be used to represent many relationship they are not always enough. It is often easiest to create special functions with the properties or shape that you want.

Lookups allow you to define customized relationships between a variable and its causes. An equation can be defined with a specially-constructed function:

```
y = my function(x)
```

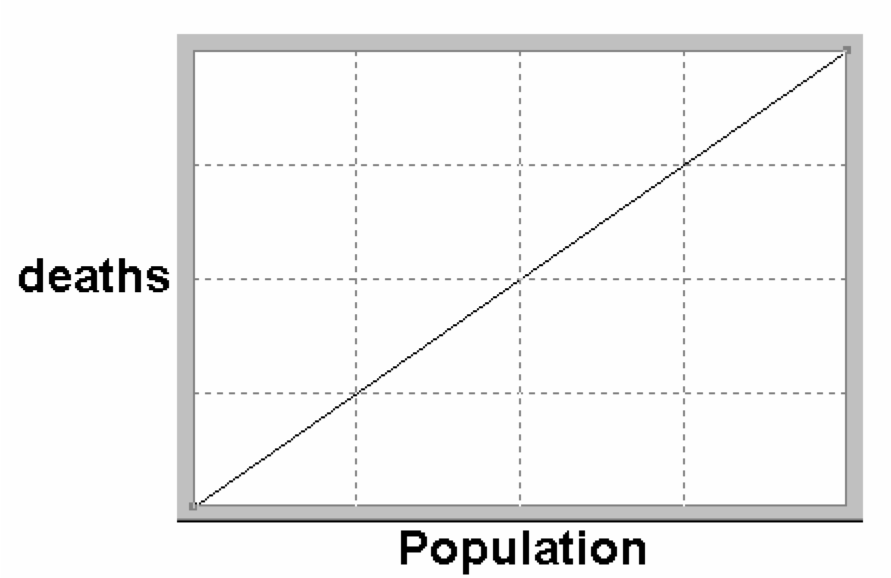
The output variable *`y`* is changed by input variable *`x`* through the Lookup function *`my function`*, which has a (typically nonlinear) shape that you specify.

Lookups are also known as Lookup Functions, Graphical Functions, Lookup Tables, or simply Tables. They can be constructed as a table of numbers (in the Equation Editor), or as a graph (in the Graph Lookup Editor).

Limits to Rabbit Growth

The population model of Chapter 6 is extended here to include the consequences of population growth in an environment with a limited carrying capacity. The unchecked growth of the previous model (*pop.mdl*) is replaced by growth which is limited by the size of the population relative to the carrying capacity of the local environment (*rabbit.mdl*).

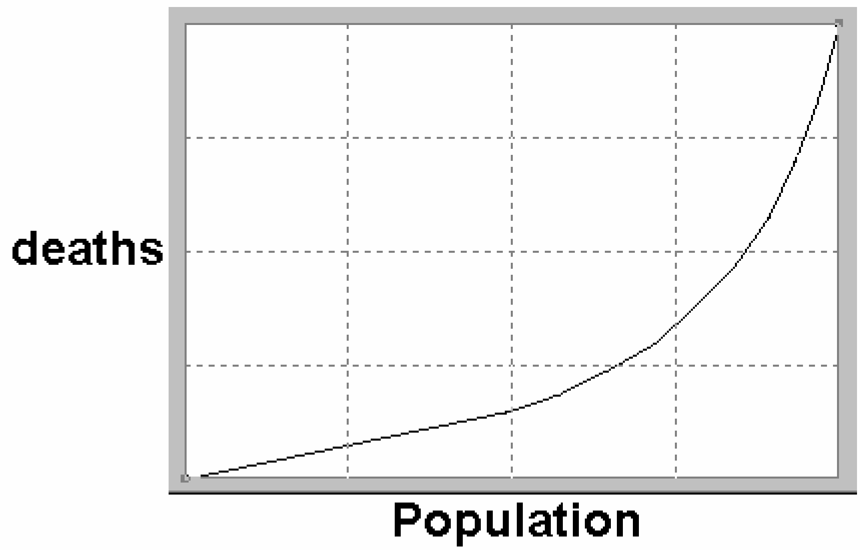
The value of the variable *`deaths`* in *pop.mdl* is directly proportional to the size of the rabbit population. In effect, we see a relationship between *`Population`* and *`deaths`* that is linear:



This *does not* mean that `deaths` will increase linearly over time. The linearity means `deaths` will grow in the same manner as `Population` (if `Population` grows exponentially, so will `deaths`).

We could make a Lookup table (as above) that expresses this linear relationship, but it is easier just to use a constant (that has the same value as the slope of the Lookup) multiplied by `Population`. In fact, the model developed in Chapter 6 uses `Population/average lifetime` so that the slope of the straight line would be `1/average lifetime`. This formulation was chosen because it is much easier to understand what `average lifetime` is (and what a change in it means) than to understand the shape and slope of a curve.

What we want to do in this model is have `deaths` increase more quickly than `Population` as population grows to a large size. This happens because higher populations are nearer resource limits (such as food) and therefore rabbits will, on average, die more quickly. We are looking for a function such as:



You can develop such a function using Lookups, and that is what you will do next. Before that happens, however, it is important normalize the input and output of the Lookup.

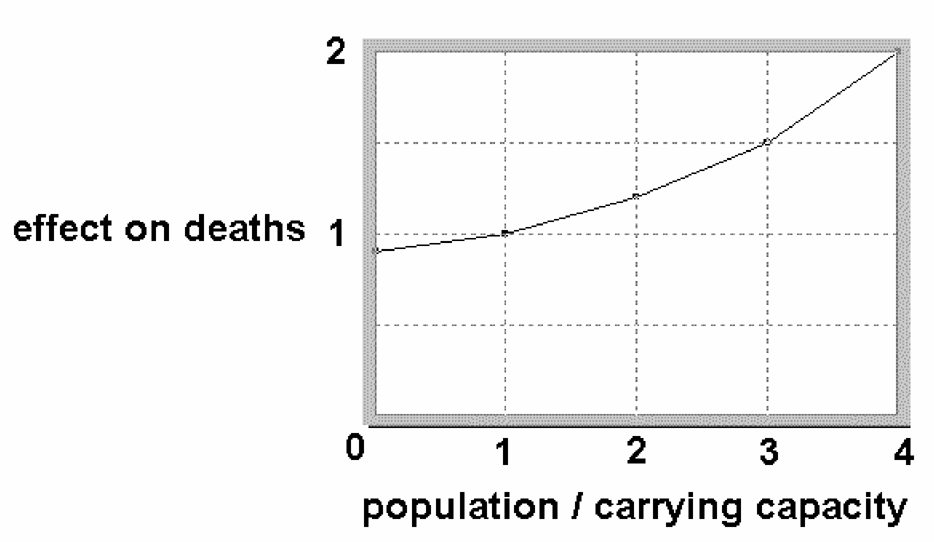
Normalized Lookups

The graphical function drawn above has the number of rabbits as its input and the number of rabbits that die per year as its output. This is a difficult graph to create, and worse, it is very tricky to change. Suppose, for example, you want to understand what happens when a longer lived breed of rabbits is introduced — the whole function needs to be redone. Or suppose you want to understand the effects of increasing the carrying capacity of the rabbit's environment — the whole function again needs to be redone.

A normalized input is built around reference points such as 0,0 and 1,1. The input is adjusted to be dimensionless and independent of the units of measure or scale of other variables in the model. The output is very often dimensionless and is also independent of the units of measure and scale of other variables. For example, suppose we were to measure ***`Population`*** in thousands of rabbits and ***`deaths`*** in thousands of rabbits per month. A Lookup function taking ***`Population`*** as its input and having ***`deaths`*** as its output would no longer be valid.

On the other hand, a Lookup function, normalized by using ***`Population`*** relative to ***`carrying capacity`*** *as* its input and having the output ***`effect on deaths`*** act on a baseline or normal number of deaths (***`Population / average lifetime`***), does not need to be changed when units of measure or scale change.

Normalization allows us to achieve the desired behavioral relationship with a generalized set of numbers in the Lookup function. If information about the size or characteristics of the population changes, you can simply change the value of `carrying capacity` or ***`average lifetime`***, rather than the whole Lookup function.



To normalize, divide the input variable by a normal or average value (e.g., *`Population / carrying capacity`*). When the actual *`Population`* is equal to this normal value, the input (to the Lookup) is 1. Other values of *`Population`* will vary the input either higher or lower than 1. The output values of a Lookup are also usually made to vary about the value of 1. The output of the Lookup can then be used to drive another model variable above or below its normal value. Another way of saying this is: when the input variable is equal to its normal or average value, the output from the Lookup is 1 and therefore has no effect on the connected output variable's value.

NOTE This formula is similar to the Lookup equation example in the beginning of this chapter `y = fn (x)`, but takes two variables as the (normalized) input and computes the output relative to a normalized value : `y = normal y * fn (x/normal x)`.

Sketching the Model (rabbit.mdl)

This model (*rabbit.mdl*) demonstrates the simple and direct approach to building models with Lookup functions. The normalization is done *within* the Lookup output variable ***`effect of rabbit crowding on deaths`*** and the output of this variable acts directly on the rate ***`deaths`***. To see an example where the input and outputs are separated, see the section Separate Normalized Variables later in this chapter, or open the model *rabbit2.mdl*. This is functionally the same model as *rabbit.mdl* but contains more variables. You decide which is clearer.

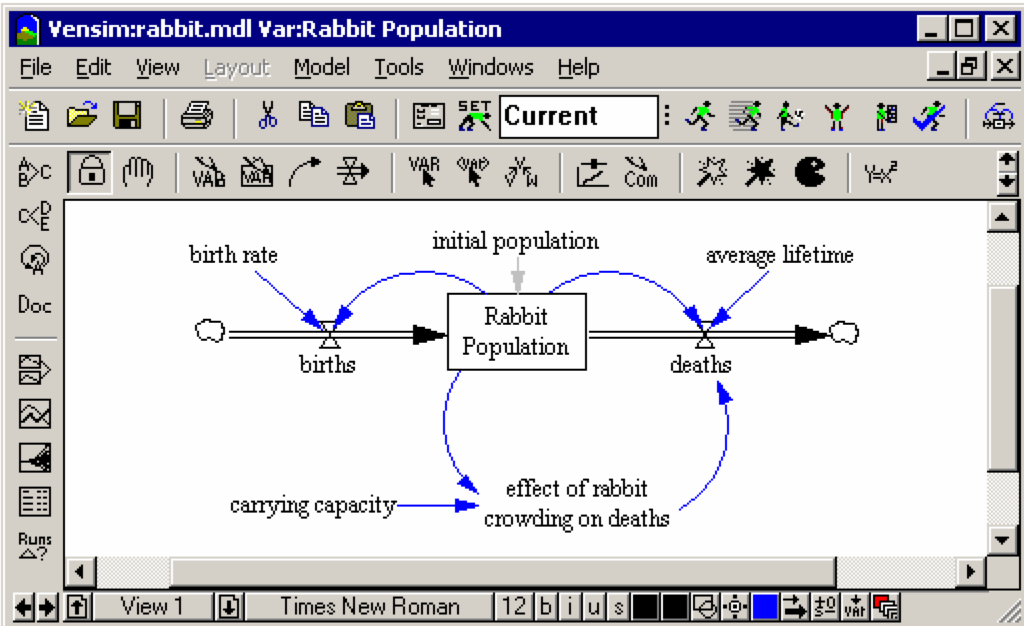
✍ Click the **New Model** button.

✍ In the Model Settings dialog, Time Bounds tab, type 30 for FINAL TIME, type (or select from the drop down box) 0.125 for TIME STEP. Click on the drop-down box for **Units for Time**, and select **Year** (or type in Year).

✍ Click the **Units Equiv** tab. In the editing box, type `rabbit,rabbits` then click the button **Add Editing**. Click on **OK** (or press Enter).

✍ Sketch the model shown below in the diagram below.

✍ Save your model (e.g., *rabbit.mdl*) in the *guide\chap08* directory.



Entering Equations

✍ Click the **Equations** tool, click on the following variables and enter the equations and units of measurements as follows:

```
Rabbit Population = INTEG (
   births-deaths,
         initial population)
Units: rabbit

initial population = 1000
Units: rabbit

birth rate=    0.23
Units: fraction/Year

average lifetime = 8
Units: Year

births = Rabbit Population * birth rate
Units: rabbit/Year
```

```
deaths =       (Rabbit Population / average lifetime) *
         effect of rabbit crowding on deaths
Units: rabbit/Year
```

```
carrying capacity = 1000
Units: rabbit
```

The variable ***carrying capacity*** is *not* the maximum number of rabbits that the environment can hold. Instead, ***carrying capacity*** represents the *normal* number of rabbits for that environment. We would use a different formulation if we wanted to represent maximum carrying capacity.

Note that we are initializing the Level with the Constant ***initial population*** rather than typing in a number. This allows us to make changes to the value during simulation experiments. Note also that the ***deaths*** equation has a multiplier ***effect of rabbit crowding on deaths*** which will change the value of deaths.

Remember: To add a variable as an initial value use the **Choose Variable** (**Choose Initial Variable** in PLE and PLE Plus) button on the **Variables** tab.

Creating and Normalizing Lookups

✍ With the **Equations** tool selected, click on ***effect of rabbit crowding on deaths***.

Under the **Type** label there are two drop down boxes, one showing Auxiliary, the other showing Normal.

✍ Click on the drop down box showing Normal and select **with Lookup**.

This creates a Lookup table embedded within the Auxiliary variable.

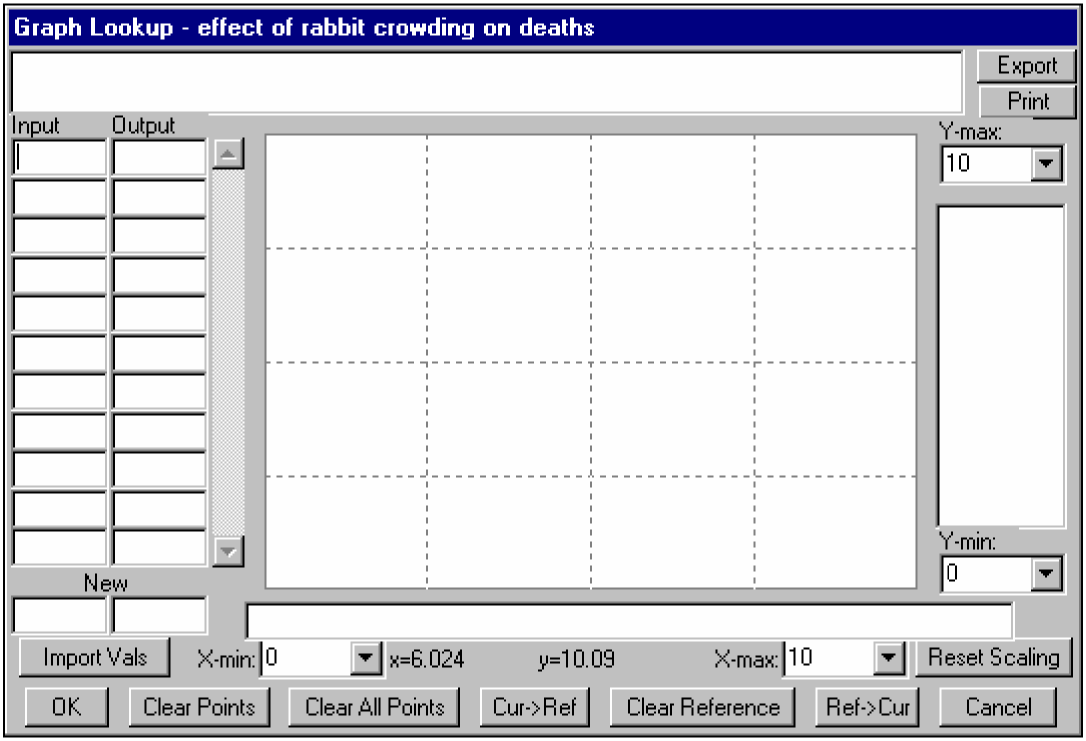
✍ Click on the **Variables** tab then click on the variable ***Rabbit Population*** in the list.

✍ Type a divide by symbol (/) then click on ***carrying capacity***.

```
effect of rabbit crowding on deaths = WITH LOOKUP(
               Rabbit Population/carrying capacity
```

As ***Rabbit Population*** changes, the ***effect of rabbit crowding on deaths*** will change according to the shape of the Lookup function. Now we need to create the actual table of figures or graph that describes the Lookup.

✍ Click on the button **As Graph** in the Equation Editor (below the **Type** boxes). The Graph Lookup Editor opens:



- ✍ Click on the left hand **New** (values) box near the bottom left corner and type in 0 then press the Enter key. The cursor moves to the right hand box; type 0.9 and press Enter again. The cursor moves back to the left hand box and the old numbers are moved to the input/output columns making way for new numbers to be entered.

- ✍ Continue typing in the rest of the pairs of values below, pressing the Enter key each time a value is typed. The graph will draw itself.

```
(0,0.9), (1,1), (2,1.2), (3,1.5), (4,2)
```

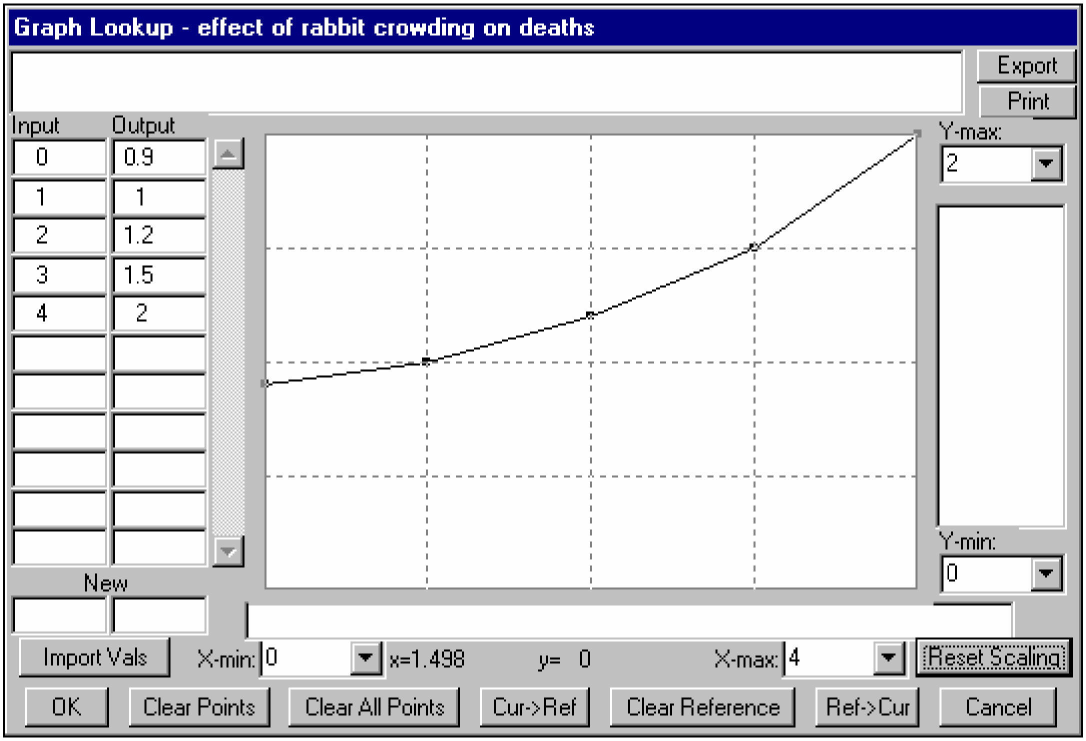
- ✍ Alternatively, you can sketch the graph using the pointer by clicking on the graph to add points, and dragging the points to reposition them on the graph. You will first need to set the **X-max** and **Y-max** values to 4 and 2 respectively.

- ✍ Click on the button **Reset Scaling** to set the X and Y scales to your points.

Do not worry about getting the exact numerical values for points shown in the figure if you are adding points with the mouse. The shape of the curve is more important that the exact values. But do set (1,1) exactly because this is a reference point: when *`Rabbit Population`* is equal to *`carrying capacity`*, there is no change on the normal rate of *`deaths`*.

Editing Values

- You can modify values in the **Input/Output** list, or by dragging a point around on the graph.
- To remove a point, click on the button **Clear Points** and then with the **Delete** icon, click on the point on the graph. Your graph should look like:



- Click on **OK** and the Graph Lookup Editor closes.

Now you will see the graph equation expressed as a table of values enclosed in parentheses. These values could have been typed in directly, but instead we generated them in the Graph Lookup Editor.

- Add the units `Dmnl` (for Dimensionless, which you could also have typed in) to the **Units** editing box, then click **OK** to close the Equation Editor.

The units Dimensionless are pretty important to understand. When we normalized the input to the Lookup, we divided the `Rabbit Population` (measured in rabbits) by `carrying capacity` (also measured in rabbits) leaving a dimensionless variable.

- Click on the **Save** button on the Toolbar to save your model.

Checking for Model Syntax and Units Errors

Before you simulate the model, you should check it for errors in equations and units.

✍ Select **Model>Check Model** (or press Ctrl+T), you should get an information box saying "Model is OK."

If the model has errors, check that the structure is the same as in diagram. If the structure looks the same, open the Equation Editor on each variable and check the equation against the list above.

✍ Select **Model>Units Check** from the menu (or press Ctrl + U); you should get an information box saying "Units are AOK."

If a units error is generated, read the Output window to see which variables are failing the check. Open the Equation Editor on each variable and check the units against the equations listed above. Units that do not check correctly are often an indication of poor or incorrect equation formulations.

Simulating the Model

✍ Click on the **Runname** editing box on the Toolbar and type a name for the first run (e.g., *base*).

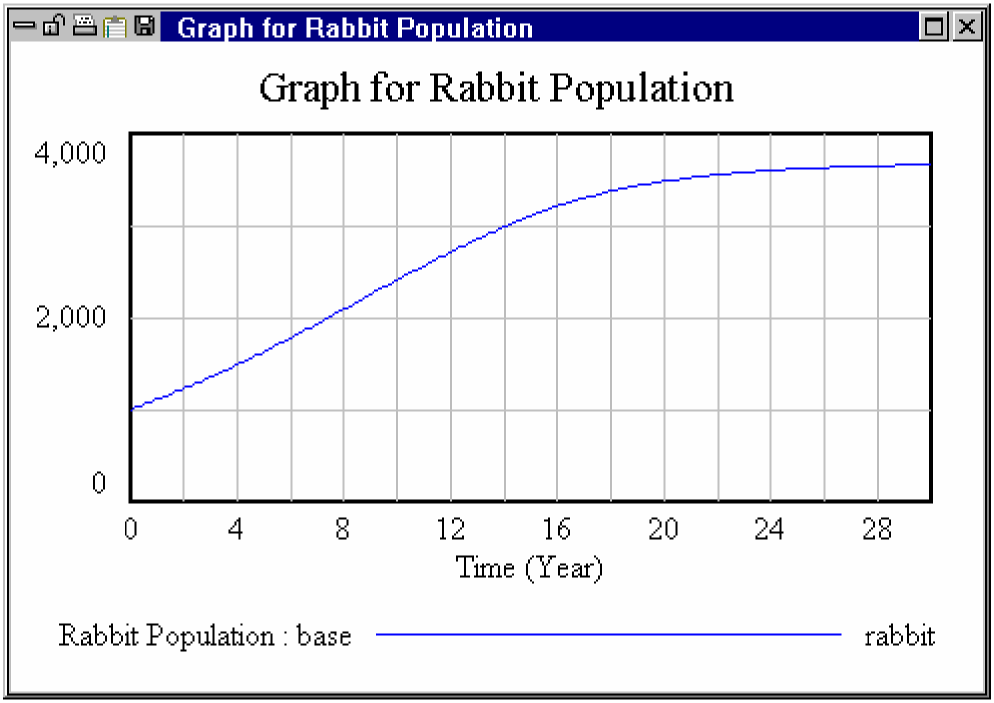
✍ Click on the **Simulate** button.

The model will simulate, showing a work-in-progress window until completion (on a fast computer, you might not see this window).

Model Analysis

✍ Click on the Level `Rabbit Population` in the sketch. This selects it as the Workbench Variable. Check the title bar at the top of the Vensim window to see that `Rabbit Population` is selected.

✍ Click on the **Graph** tool. A graph of `Rabbit Population` is generated:

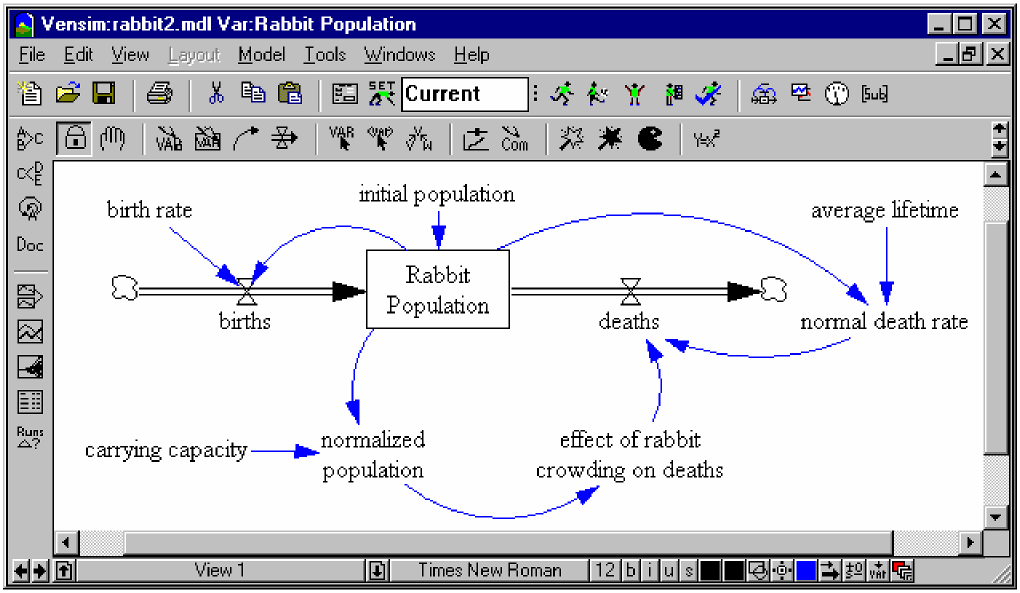


We can see that the **`Rabbit Population`** first grows exponentially then grows at a slower and slower rate until it eventually approaches a maximum value of approximately 3500 (because of the effect of the Lookup table).

Separate Normalized Variables

This section is optional and produces a model with the same behavior as the *rabbit.mdl* model already constructed.

Lookup tables are best used to drive model variables above or below their "normal" values. In the previous model (rabbit.mdl), the normal value was implicit and hidden within the Rate equation for deaths. In other models, you might want to have an explicit normal value. This might be a Constant, or an Auxiliary as ***`normal death rate`*** in the *rabbit2.mdl* model below:



This model also includes an explict (separated) normalized input, rather than normalizing the input variable inside the variable ***effect of rabbit crowding on deaths***.

- ✍ Save the previous rabbit model as another name (*rabbit2.mdl*) and then create the structure above.
- ✍ To move the arrows from the Rate ***deaths*** to the Auxiliary ***normal death rate***, use the **Move/Size** tool to grab the arrowhead from deaths and drop it on the Auxiliary. Alternatively, you can delete the arrows using the **Delete** tool by clicking on the arrowhead, then draw new arrows with the **Arrow** tool.
- ✍ Click on ***normal death rate*** and enter the following equation (the same equation for deaths before we introduced the effect from the Lookup) and units then press Enter:

```
normal death rate=
   Rabbit Population / average lifetime
Units: rabbit/Year
```

- ✍ Click on ***deaths*** and replace the equation with the one below then press Enter.

```
deaths=
   normal death rate * effect of rabbit crowding on deaths
```

- ✍ Click on ***normalized population*** and enter the equation below:

```
normalized population =
   Rabbit Population / carrying capacity
Units: Dmnl
```

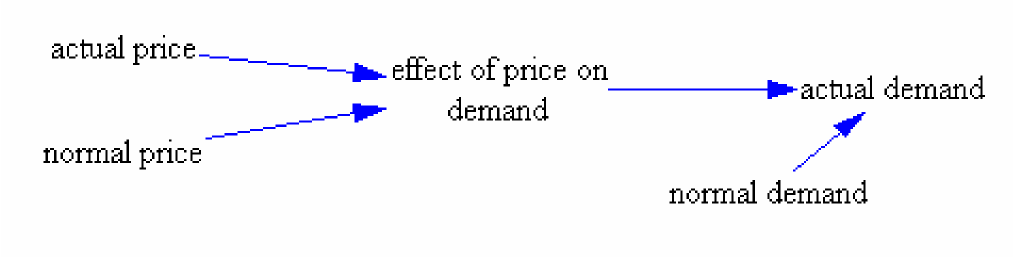
- ✍ Click on ***effect of rabbit crowding on deaths*** and replace the equation in the WITH LOOKUP editing box with (leaving the Lookup table and units as they are):

```
effect of rabbit crowding on deaths= WITH LOOKUP (
   normalized population
```

The Equation Editor should close and no variables on the diagram should have highlights.

✍ Click on the **Save** button on the Toolbar to save your model.

Note that the normal output variable `normal death rate` could (in a different model) be a Constant, and not the Auxiliary that it is here. See for example, the structure below:



Simulation

✍ Before simulating, perform a Units Check (Ctrl + U) and a Model Check (Ctrl + T).

✍ Choose a simulation run name and simulate the mo del. Check output with the graph tools. You should get exactly the same behavior as the previous model (see graph earlier).

One thing to note is the extra variable `normal death rate`. This calculates what would be the amount of deaths if there was no population crowding affecting the death rate (via the Lookup table). We could have embedded this calculation in the Rate `deaths` as in the previous model (rabbit), but we chose to separate it for clarity and to teach the use of Lookup output driving a "normalized" variable.

Changing Model Lookups

Now let us make a temporary change to the Lookup and simulate the model again.

✍ Click on the **Set Up a Simulation** button.

✍ Click on the **Runname** editing box and type *run2* or some other run name.

✍ Click on the variable `effect of rabbit crowding on deaths` appearing yellow/blue in the sketch.

The Graph Lookup Editor will appear.

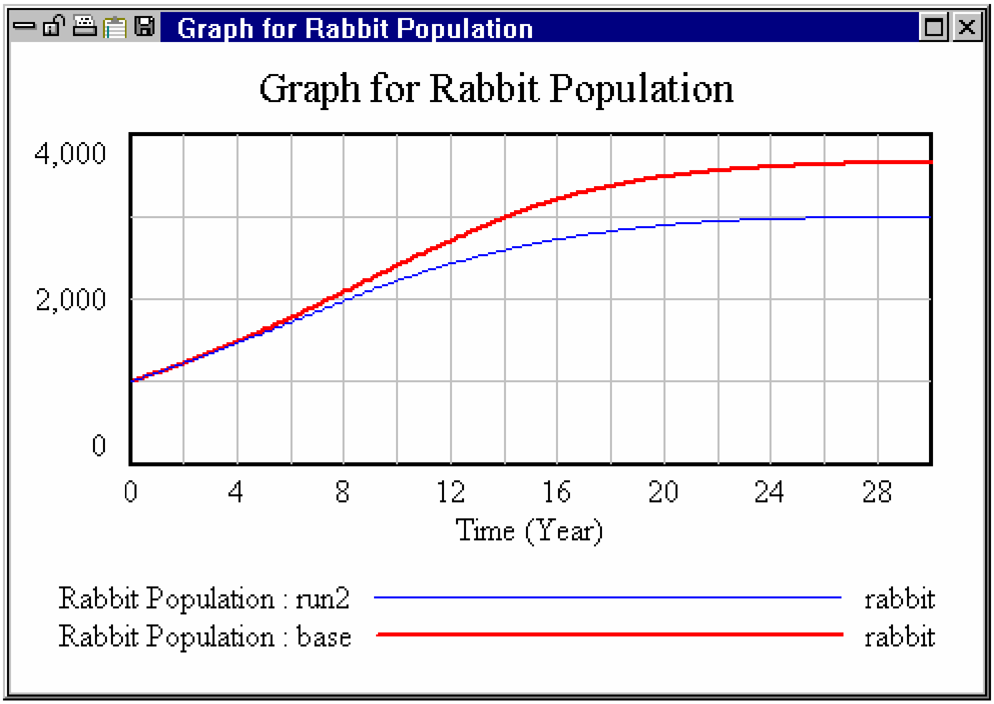
✍ With the mouse, move some of the points on the graph to change the steepness of the curve. For example, drag the points a little higher. If you need to, increase or decrease the scale by clicking on the drop down boxes for **Y-max: Y-min: X-max: X-min:** or typing in new values. You can also add extra points or delete points. (You could also change the values from the keyboard in the **Input** and **Output** fields on the left.) Click the **OK** button.

This is a *temporary* change for this run only, and does not permanently alter the values in your model.

✍ Click the **Simulate** button and the model will simulate.

✍ Click on ***`Rabbit Population`*** to select it, then click on the **Graph** tool.

You might get a graph such as the one below, showing reduced size in final ***`Rabbit Population`***, or something quite different. The graph below was a result of an increased effect of the Lookup table (increased values).



Named Lookups

In the previous models we have used an Auxiliary variable with subtype with Lookup to enter the nonlinear effect of population density on the rate at which rabbits die. In some cases it is desirable to place a name on the functional form. This is especially true if you want to use the Lookup in more than one place.

✍ Save the model *rabbit2.mdl* as a new name (*rabbit3.mdl*).

✍ Click on the Variable tool and add the new variable ***`effect of rabbit crowding on deaths function`***.

✍ Draw an arrow from the effect of rabbit crowding on deaths function to effect of rabbit crowding on deaths.

- ✍ Open the equation editor on effect of rabbit crowding on deaths function.
- ✍ Click on the dropdown for type and select Lookup. Click on the **As Graph** button to open the Graph Editor and enter the Lookup values as you did before.
- ✍ Close the Graph Editor, add in units Dmnl and click on **OK** to close the Equation Editor. Your equation should look like:

```
effect of rabbit crowding on deaths function( [(0,0)-
   (4,2)],(0,0.9),(1,1),(2,1.2),(3,1.5),(4,2))
Units: Dmnl
```

- ✍ Open the equation editor on effect of rabbit crowding on deaths.
- ✍ Select the Subtype **Normal** from the dropdown. The lower window will disappear. Click on the beginning of the equation and select *`effect of rabbit crowding on deaths function`* from the **Variable** list. Add parentheses () around *`normalized population`*.

```
effect of rabbit crowding on deaths=
         effect of rabbit crowding on deaths function(
               normalized population)
```

- ✍ Click on **OK** to close the dialog.

This model is exactly the same as *rabbit2.mdl* except that the Lookup relationship has been explicitly named. While naming Lookups in this manner is cumbersome for a simple relationship such as the one defined here it can be very helpful in more complex situations, especially if you want to use the same nonlinear relationship in more than one place. When you click on the Simulation Setup button the named Lookup will highlight allowing you to change it.

9 Multiple Views

The larger a model is the more difficult it is to understand the model. When developing a model diagram, if everything appears in one place, things can quickly become overwhelming. The solution to this is to break the model up into a number of more manageable pieces called Views. Views are supported in all Vensim configurations though older versions of Vensim PLE did not have this capability.

How Views Work

Large models can be presented more clearly in multiple views than in a single view. Views can be thought of as similar to pages of a book, each page telling a portion of the story. Each view displays a sketch and is connected to one (or more) of the other views through variables or shadow variables. Multiple views allow models to be broken into sectors, such as production, financial, customer and so on.

Graphic Models and Variables

Vensim models are definitively expressed in equation or text form. A sketch or graphical view of a model might not show all of the variables present or all of the relationships among those variables. When sketching a new model, you add variables to a view. These variables can be removed (cut) from the view but still be present in the model. Variables and structure also can be hidden with the **Magic Wand** tool. Variables can be added directly to the model by using the Equation Editor (or Text Editor in Professional and DSS); these variables will not show on the sketch unless you add them (using the **Variable** tool) or Vensim adds them (as a cause to another variable).

Models represented graphically can show the same variable in many different views; conversely a variable present in the model might not be displayed in any view. The variable can be shown as a regular variable (with causes attached), or a shadow variable (without causes attached).

NOTE It is possible to build separate working models in different views, but this is not recommended unless you intend to link them causally together at some later time. It is better to build complete new models (this gives them their own name and Time Bounds).

Customer Diffusion Model

This model describes a simple diffusion process, where Potential Customers of a product are influenced into buying the product by word of mouth from Customers (who already own the product). The first view shows the diffusion process. The second view will add a production capacity variable which will potentially limit the quantity of product sold at any one time. The third view will describe the sales revenue generated from sales of the product.

Some of the theory behind models of this type is discussed in Chapter 4 of the Modeling Guide.

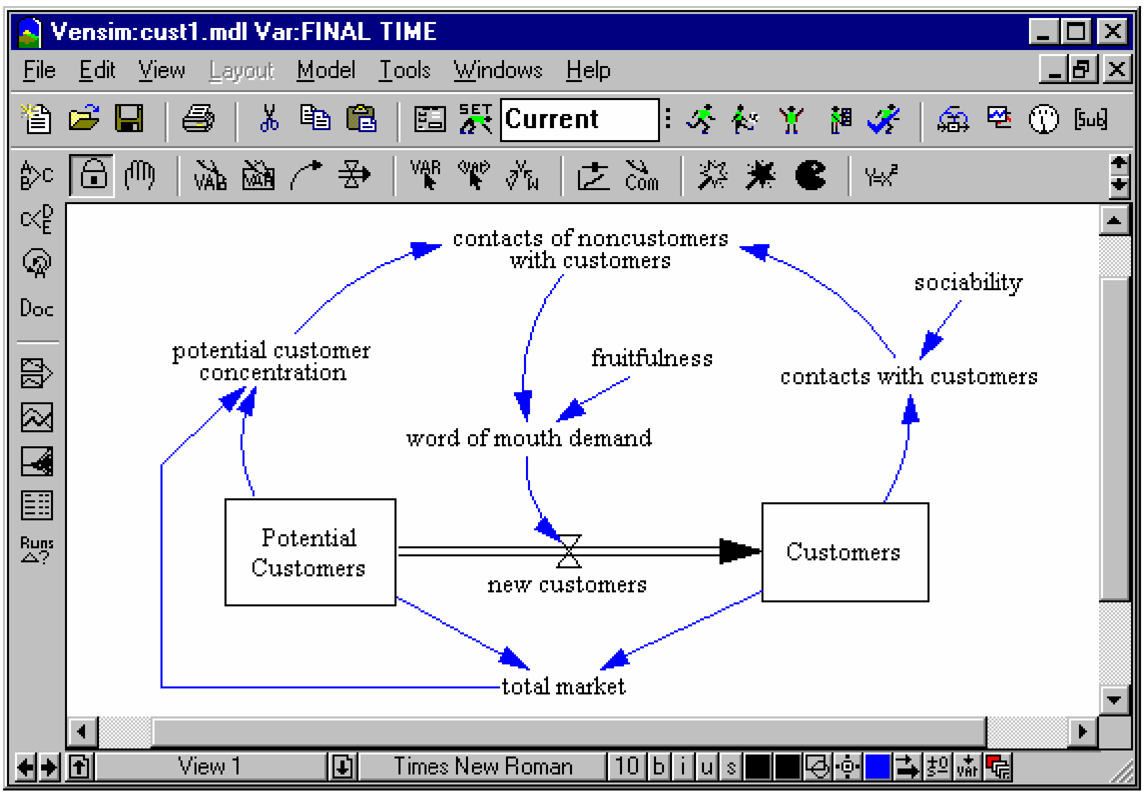
Building the Diffusion Model (cust1.mdl)

EITHER

- ✍ Open the existing model: Click the **Open Model** button and select the model *cust1.mdl* from the *guid\chap09\complete* directory and save it as *cust1.mdl* in the *guide\chap09* directory, then skip forward to section "Simulating the Model."

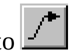
OR

- ✍ Build a new model: Click the **New Model** button or select the menu item **File>New Model**
- ✍ Click **OK** to accept the default Time Bounds.
- ✍ Sketch the model shown in the figure below.
- ✍ Save your model as *cust1.mdl* in the *guide\chap09* directory.



The sketch shown here contains what is called a Polyline Arrow. A Polyline arrow is like an ordinary arrow except that instead of an arc it appears as a series of line segments. Vensim PLE Plus user's will need to draw a regular arrow here (there is no way to draw a Polyline Arrow in PLE or PLE Plus). To draw a Polyline Arrow using Standard, Processional or DSS:

✍ Right-Click or Ctrl-Click on Arrow tool in the Sketch Toolset.

✍ Click on the **Polyline** option and Click **OK**. The icon will change to  and the mouse pointer will also change.

✍ Click on *`Total Market`*, move to the left and click again, move straight up and click again then click on *`potential customer concentration`*.

✍ Right-Click on the Arrow tool and reset the type to **Normal**.

Entering Equations

✍ Click on the **Equations** tool and enter the following equations and units of measurement for each variable in the model:

```
new customers = word of mouth demand
Units: person/Month

contacts of noncustomers with customers = contacts with customers *
   potential customer concentration
Units: contacts/Month

contacts with customers = Customers * sociability
Units: contacts/Month

Customers = INTEG(
   new customers,
         1000)
Units: person

fruitfulness = 0.01
Units: person/contacts

potential customer concentration = Potential Customers / total
   market
Units: dmnl

Potential Customers = INTEG(
    - new customers,
         1e+006)
Units: person

sociability = 20
Units: contacts/person/Month

total market = Customers + Potential Customers
Units: person

word of mouth demand = contacts of noncustomers with customers *
   fruitfulness
Units: person/Month
```

Checking for Model Syntax and Units Errors

Before you simulate the model, you should check it for errors in formulas and units.

- ✍ Select **Model>Check Model** (or Press Ctrl + T), you should get an information box saying "Model is OK."

If the model has errors, check that the structure is the same as in diagram. If the structure looks the same, open the Equation Editor on each variable and check the formula against the list above.

- ✍ Select **Model>Units Check** from the menu (or press Ctrl + U); you should get an information box saying "Units are AOK."

If a units error is generated, read the Output window to see which variables are failing the check. Open the Equation Editor on each variable and check the units against the list above.

- ✍ Save the model by clicking the **Save** button, or selecting menu **File>Save**, or pressing Ctrl + S.

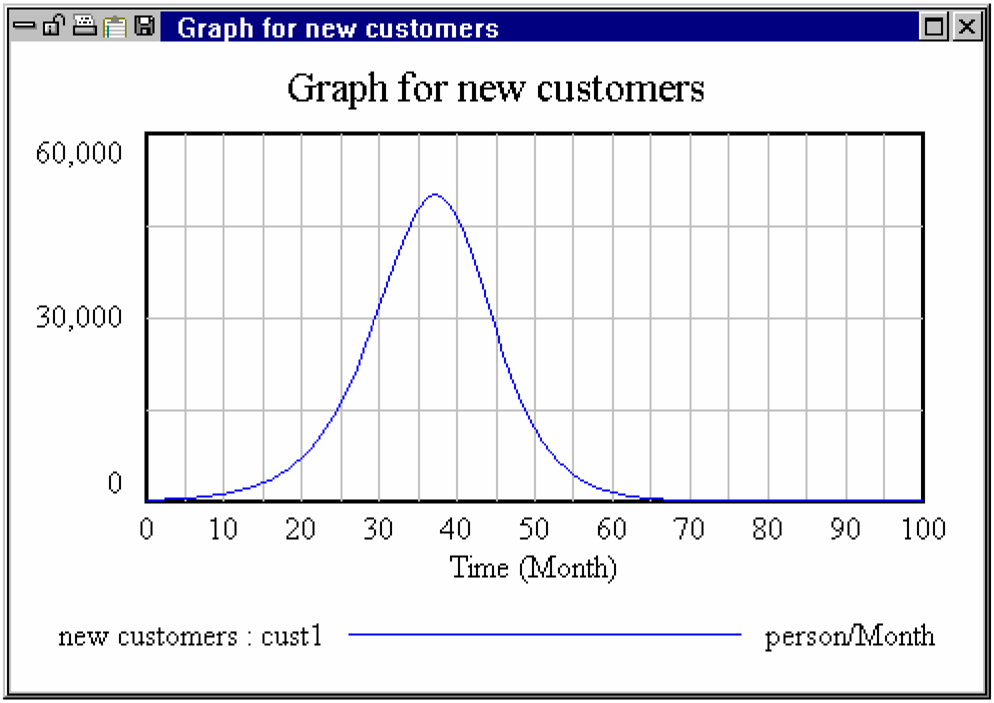
Simulating the Model

- ✍ Double click on the **Runname** editing box and type *cust1* for the first dataset.
- ✍ Click on the **Simulate** button.

Model Analysis

- ✍ Click on the **Graph** tool or the **Causes Strip** graph to investigate the behavior of key variables, such as *`Potential Customers`*, *`Customers`*, and *`new customers`* (as well as any other variables that you want to see).

The variable *`new customers`* should give you the following graph:



Naming and Saving Your Model

Before starting the next section, name the view, and save the model under a new name so that we will have two distinct working models.

✍ Select menu **View>Rename** and type in the name Customer, then click **OK**.

✍ Select **File>Save As…** and enter the model name *cust2.mdl*, then click **Save** .

Adding the Capacity View (cust2.mdl)

Now we would like to expand our model by considering production capacity issues. Customer demand might outstrip our capacity to supply the product. Therefore, we will build a production capacity view that is linked to the customer view and limits the customer demand if capacity is reached.

✍ Select the menu item **View>New**.

Let us give this view a name so we can easily navigate between different views.

✍ Select the menu item **View>Rename**. Type in the name Capacity and click **OK**.

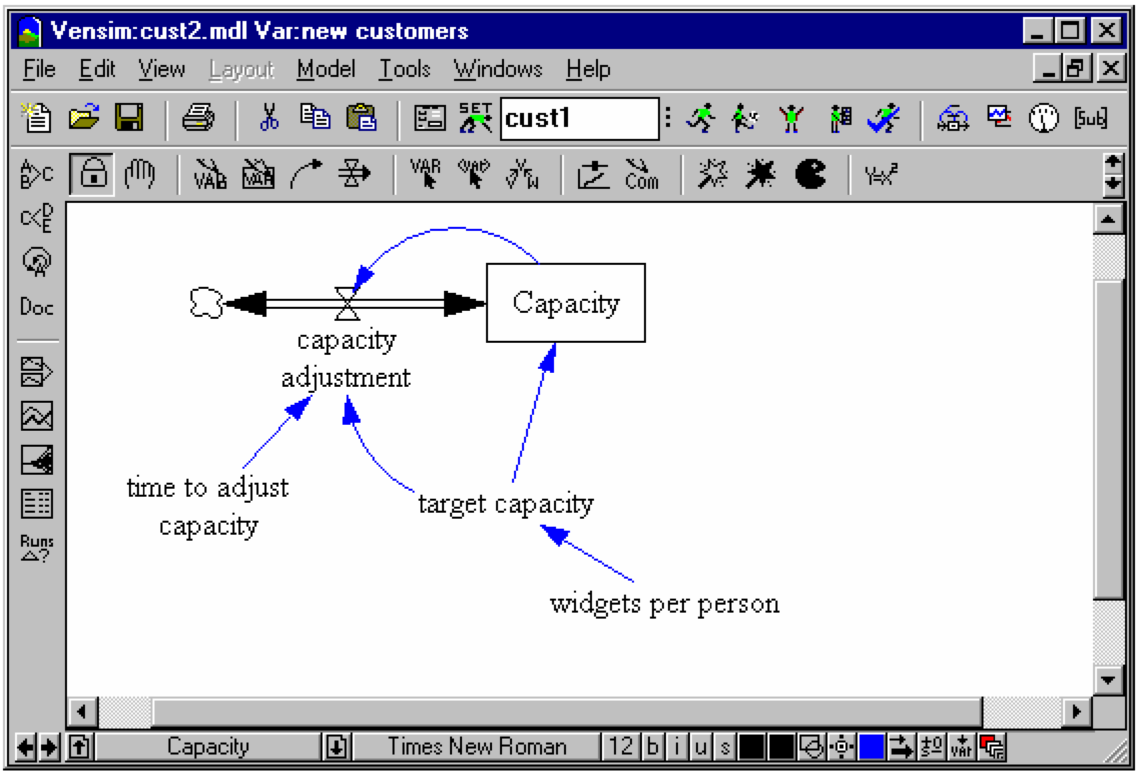
✍ Select the **Box Variable** tool and click on the sketch. Type the name `Capacity` and press Enter.

✍ Select the **Rate** tool, then click on the sketch to the left of `Capacity` then click on `Capacity`. Type the name `capacity adjustment` and press Enter.

✍ Using the **Move/Size** tool, click with the right mouse button on the handle (the little circle) in the middle of the left hand Rate pipe. The Arrow Options dialog will open. Check the box for **Arrowhead** at the top of the dialog then click **OK**.

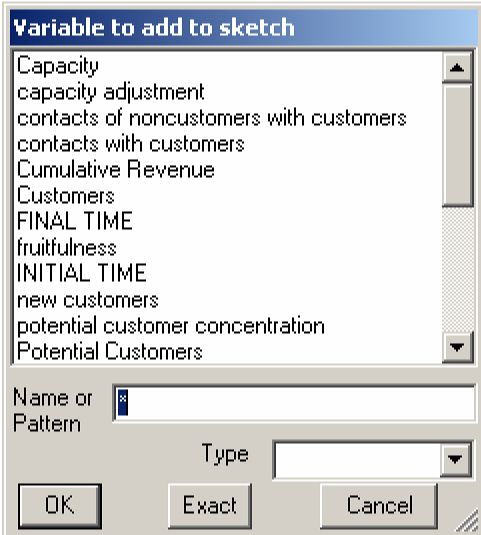
This extra arrowhead indicates that this Rate pipe can flow in either direction. Or in other words, the Rate can increase or decrease the Level `Capacity`. Note that actual behavior of the Rate is governed by the equation, not by the arrowhead that we just added.

✍ Create the other variables and connect them with arrows as shown below:



Shadow Variable Tool

Select the **Shadow Variable** tool and click on the sketch below `target capacity`. A dialog box opens:



✍ Either select the variable ***word of mouth demand*** from the list, or type the first few letters of ***word of mouth demand*** until it becomes highlighted in the list, then press Enter or click on **OK**.

✍ Select the **Arrow** tool and click on the Shadow Variable ***word of mouth demand*** then click on ***target capacity***.

The structure of this view is now complete. The structure is causally linked to the first view through the variable ***word of mouth demand***.

NOTE The shadow variable is only inserted for use in causing *other* things to change. If you tried to connect an arrow from another variable to the shadow variable, the arrow will not connect. If you want a variable from another view to be caused by something in this view, you need to use the **Model Variable** tool, which adds a variable *and all its causes* to a view.

Adding Equations

✍ Select the **Equations** tool.

All variables will appear black, except for the Shadow Variable ***word of mouth demand*** which has an equation defined in the first view.

✍ Enter the following equations and units of measurement for each variable in this view:

```
Capacity = INTEG(
   capacity adjustment,
         target capacity)
Units: Widget/Month

capacity adjustment=
   (target capacity - Capacity) / time to adjust capacity
```

```
Units: Widget/Month/Month

target capacity=
   word of mouth demand * widgets per person
Units: Widget/Month
time to adjust capacity=12
Units: Month
widgets per person= 1
Units: Widget/person
```

Now we will return to the first view and complete the feedback loop by linking its structure to the variables in the view Capacity.

✍ Click on the **View** button (on the Status Bar — currently reads "Capacity") and choose Customer.

NOTE You can also use the Page Up / Page Down keys on your keyboard to step through views.

✍ Select the **Shadow Variable** tool and click below and right of the variable *`new customers`*. Choose *`Capacity`* from the list (or type the first few letters of *`Capacity`*) and press Enter (or click **OK**).

✍ Repeat this process for the variable *`widgets per person`*.

✍ If necessary, move the variable *`total market`* lower and move the arrows to make room.

✍ Select the **Arrow** tool and connect Capacity to new customers, then connect widgets per person to new customers.

Altering An Equation

✍ Select the **Equations** tool.

Only the equation for *`new customers`* should be highlighted black, because this is the only variable in this view to which we added causes. Sales of the product are limited by the factory's ability to produce, so we will write an equation which returns the smaller value of *`word of mouth demand`* or factory capacity divided by units of product per customer (*`Capacity/widgets per person`*).

✍ Click on *`new customers`* and change the equation to the one below:

```
new customers=
   MIN(word of mouth demand, Capacity/widgets per person)
```

You can select the MIN function from the list under the **Functions** tab, or just type it in as above.

Adding a Sales Revenue View

Let us add a view which will track the sales revenue from selling the product, and also the cumulative revenue from all the sales.

✍ Select the menu item **View**>**New**.

✍ Select the menu item **View**>**Rename** and type in Sales Revenue, then press Enter.

Model Variable Tool

If you are using Vensim PLE or PLE Plus there is no Model Variable Tool. Just enter ***`widgets per person`*** and ***`new customers`*** as shadow variables then add the remaining variables.

✍ Select the **Model Variable** tool. Click on the view and select the variable ***`new customers`*** then press Enter.

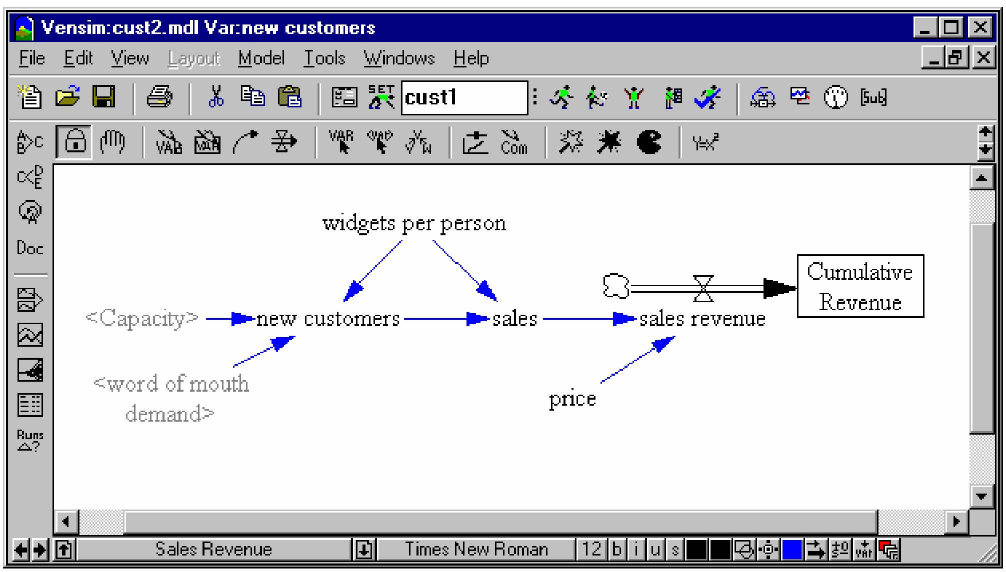
This adds ***`new customers`*** and its causes to the view. The causes will be on top of one another, so we need to rearrange them for clarity.

✍ Using the **Move/Size** tool (or the currently selected **Model Variable** tool), rearrange the variables so they sit on the left of ***`new customers`***.

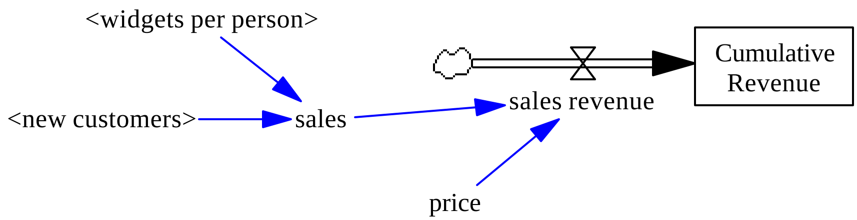
✍ Select the **Model Variable** tool. Click on the shadow variable ***`widgets per person`***.

This will turn the shadow variable into a normal variable. Because it is a Constant, there are no causes connected to ***`widgets per person`***.

✍ Add the variables and arrows as shown in the diagram below:



If you are using PLE or PLE Plus your diagram will appear as:



More Equations

✍ Click on the **Equations** tool and enter the following equations and units of measurement for each variable that appears black in the view (the others already have equations entered).

```
sales=new customers * widgets per person
Units: widget/Month
price=50
Units: dollar/widget
sales revenue=sales * price
Units: dollar/Month
```

✍ When you click on *`Cumulative Revenue`* to write its equation, click on the check box marked **Supplementary**. This tells Vensim that the variable is not used anywhere else.

```
Cumulative Revenue = INTEG(
   sales revenue,
         0)
Units: dollar
```

✍ Select **File>Save** (or press Ctrl + S).

Simulating the Model

✍ Check the model for errors with **Model>Check Model** (Ctrl + T).

✍ Check the units for consistency by selecting **Model>Units Check** (Ctrl + U).

✍ Simulate the model using the dataset name *cust2*.

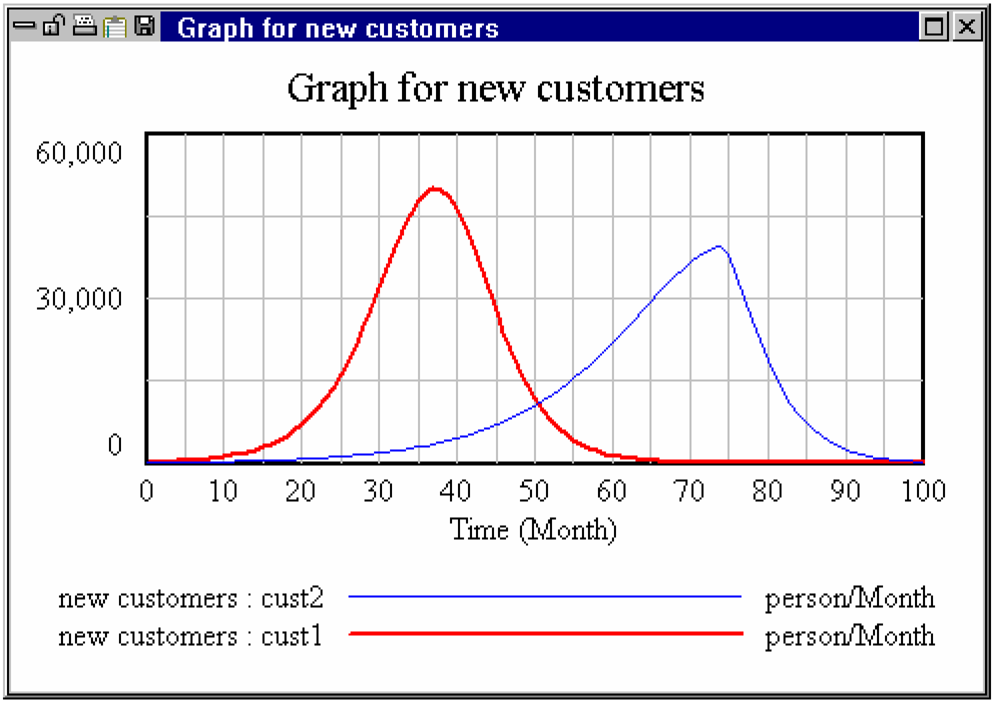
Analyzing the Model

✍ Use the **Graph** tool or the **Causes Strip** graph to investigate behavior of key variables, such as *`Potential Customers`*, *`Customers`*, and *`new customers`* (as well as any other variables that you want to see).

If you opened the existing *cust2.mdl* rather than creating your own, you will notice that the only results displayed are those for the current dataset (*cust2*). Use the Control Panel **Datasets** tab to load *cust1*.

✍ Select the variable *`new customers`* as the Workbench Variable and click on the **Graph** tool.

NOTE *cust1* has stored results for all the variables in *cust1.mdl* and does not have any results for the variables unique to our new model. If you ask to see a graph for a variable not in the original model you will only see results from the run *cust2*.



We notice a difference between the two runs. Because of the constraint in production capacity, the second run shows a much slower rise in sales and a later peak, and the total time over which sales are made is longer.

Saving Your Model

Before stating the next section, save your model under a new name so that you have three distinct working models.

✍ Select **File>Save As…** and enter the model name *cust3.mdl* then click **OK**.

✍ Select **File>Close** to close the model.

Detailed Capacity Model (cust3.mdl)

Suppose we want to model more accurately the production capacity sector of a firm. If, somewhere, we have an existing model that describes the capacity sector better than our current model, we can use this model to enhance and refine our current model. We have a model called *cap1.mdl* that builds capacity based on investment with a construction delay, and that depreciates capacity only after the capacity life is used up. This implies that we cannot reduce capacity any faster than it depreciates (unlike our other model where capacity could be reduced at the same Rate as it was built). This section introduces the concept of merging two different models (structure and equations) to form one complete working model.

Copying and Pasting

We are going to make use of model structure from another existing model (*cap1.mdl*). This model has an alternative, and somewhat improved, set of equations for determining capacity. It does not, however, have any equations around customer demand. Instead it uses a built in function called STEP to determine *`desired production`*. (See Chapter 7 for some discussion of the STEP function). We want to paste this structure into our model and then make the connections from variables we have already defined to the variables from *cap1.mdl*.

✍ Click the **Open Model** button and open the model *cap1.mdl* in the directory *guide\chap09.*

✍ Select the menu item **Edit>Select all** (or Ctrl + A, or using the **Move/Size** tool, drag a box to cover all the structure).

✍ Click the **Copy** button, or select menu item **Edit>Copy** (or Ctrl + C).

✍ Click the **Open Model** button, or select menu item **File>Open Model…** and open the model *cust3.mdl* (or select *cust3.mdl* from the **File** menu recent files list).

✍ If you are not in the Capacity view, click on the **View** button and choose the view Capacity.

✍ Click the **Paste** button, or select menu item **Edit>Paste** (or Ctrl + V).

A Choose Paste Mode dialog box opens. The choices for pasting are

Replicate — this pastes the structure and equations into the model view, renaming any variables that already appear with the same name.

Picture — this pastes a picture of the structure, but not the equations. If what is being pasted contains variables that do not exist in the current model, a warning dialog opens.

✍ Choose the **Replicate** option (the default) because we want to add the variables and equations to our model. Click the **OK** button.

The new structure is dropped onto the sketch view in the same location as it was copied from.

✍ Position the **Move/Size** tool in the middle of the highlighted box then press and hold down the mouse button and drag the box below the existing structure.

✍ Click outside the box to drop the variables permanently on the sketch.

NOTE If you click outside of the box before you have moved it, the new structure will drop on top of the existing structure; not what we want at this stage. If you have clicked outside the box before moving it: Select **Edit>Undo** and the pasted structure will disappear. Then paste again choosing Replicate.

✍ Select the **Merge** tool. Move the cursor on top of the new capacity variable (*`Capacity 0) and`* press and hold down the mouse button. Move *`Capacity 0`* directly on top of *`Capacity`* and release the mouse button.

You will see the message "This will delete the variable *Capacity* and replace it with *Capacity 0*. Do you want to continue?"

✍ Click the **Yes** button.

✍ Move the cursor over *Capacity 0* and press and hold the mouse button, then move *Capacity 0* back to where it used to be on the view (between *investment* and *reductions*) and release the mouse button.

Merging the variable we want to keep (*Capacity 0*) *onto* the variable we want to replace (*Capacity*) keeps all the causal connections and equations for *Capacity 0* as originally entered in the *cap1.mdl* model. The equations and causes of *Capacity* are destroyed. However, all variables that used *Capacity* as an input (*capacity adjustment* and *new customers*) now use the variable *Capacity 0* as their input.

The new capacity structure needs inputs to change *desired production*, which depends on the amount of word of mouth sales and the number of units sold per sale. In the old capacity structure, the calculation for this occurred in the variable *target capacity*. Therefore, to keep this equation intact, we need to merge the old variable *target capacity* onto the new variable *target capacity 0*.

✍ Using the **Merge** tool, drag *target capacity* onto *target capacity 0*. Click on **Yes** to answer the warning prompt.

We also want to keep the Constant *time to adjust capacity* with its original value.

✍ Using the **Merge** tool, drag *time to adjust capacity* onto *time to adjust capacity 0*. Click on **Yes** to answer the warning prompt.

✍ Drag the two variables word of mouth demand and widgets per person to a location below target capacity.

✍ Select the **Delete** tool. Click on the variable *capacity adjustment* to delete the variable from the model. Click on the cloud that is left hanging where the Rate used to be.

✍ Using the **Move/Size** tool, rearrange the variables and arrows until the diagram looks neat.

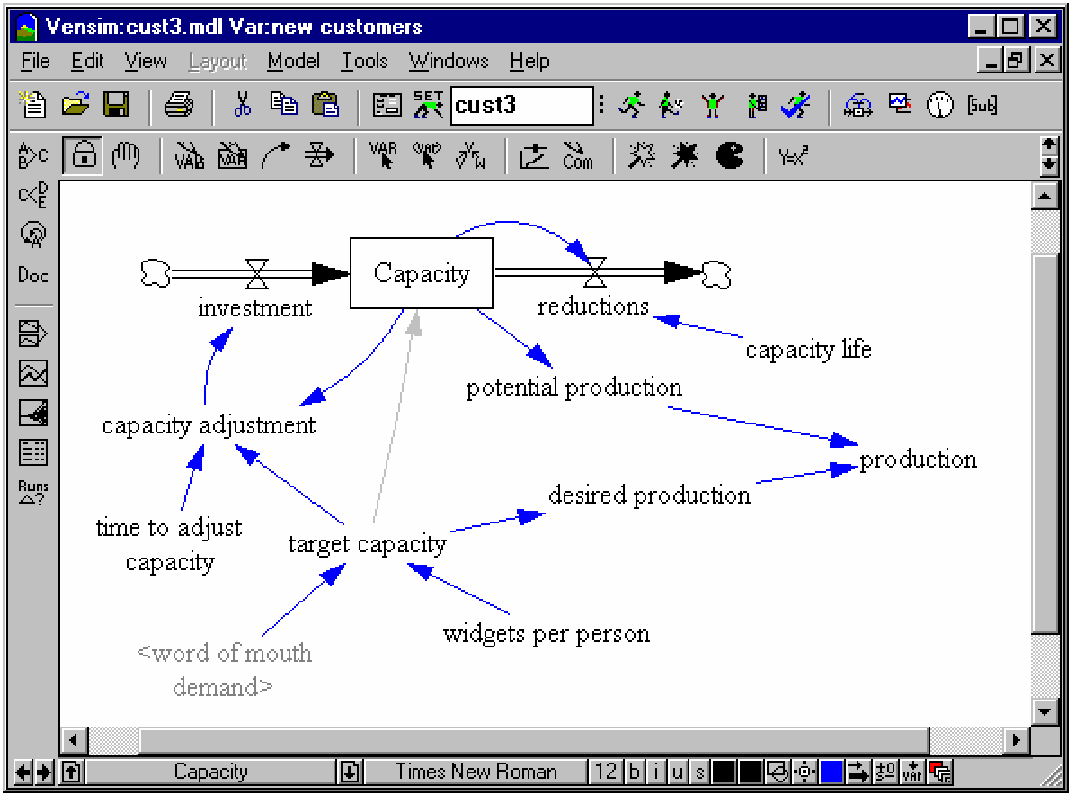
✍ If you find it hard to move arrows because you cannot find the handle, simply choose the **Arrow** tool and make a new arrow between the variables. You should get the warning box "This connection exists, remove the old arrow?" Click **Yes**.

✍ Select the **Variable** tool. Click on *Capacity 0* and an editing box with the name opens. Place the cursor at the end and delete the *0* and the space, then press Enter.

✍ Click on *capacity adjustment 0*, place the cursor at the end and delete the *0* and the space, then press Enter.

✍ Select the **Arrow** tool. Join *target capacity* to *desired production* (see below).

All of this sketch manipulation has allowed us to keep the desired equation formulations and the desired names for variables. Your view should appear similar to the diagram below:



✍ Select the **Equations** tool. Click on `desired production` and replace the numbers and step functions with the input variable `target capacity`, as below:

```
desired production=
   target capacity
Units: Gadget/Month
```

Now you should see no variables highlighted in black. If you do, click on them and check the equation against the equation for that variable appearing in the equation listing below, changing them if necessary.

✍ Click on `production` then click on the check box marked **Supplementary**

Capacity View Equations

```
Capacity= INTEG (
         investment-reductions,
               target capacity )
Units: Gadget/Month
```

```
capacity adjustment=
   (target capacity - Capacity)/time to adjust capacity
Units: Gadget/Month/Month

capacity life=
   20
Units: Month

desired production=
   target capacity
Units: Gadget/Month

investment=
    capacity adjustment
Units: Gadget/Month/Month

potential production=
   Capacity
Units: Gadget/Month

production=
   MIN(desired production, potential production)
Units: Gadget/Month

reductions=
   Capacity/capacity life
Units: Gadget/(Month*Month)

target capacity=
   word of mouth demand * widgets per person
Units: Widget/Month

time to adjust capacity=
   12
Units: Month
```

Units Synonyms

Units Synonyms in Vensim are different names that refer to the same unit of measurement. When typing in units, you might find yourself making a unit of measurement singular in one place and plural in another. The Units Check feature will consider them different units unless told that they are synonyms.

Some synonyms are already defined; Month and Months, Year and Years, etc.

✍ Select **Model>Units Check** from the menu (or press Ctrl + U).

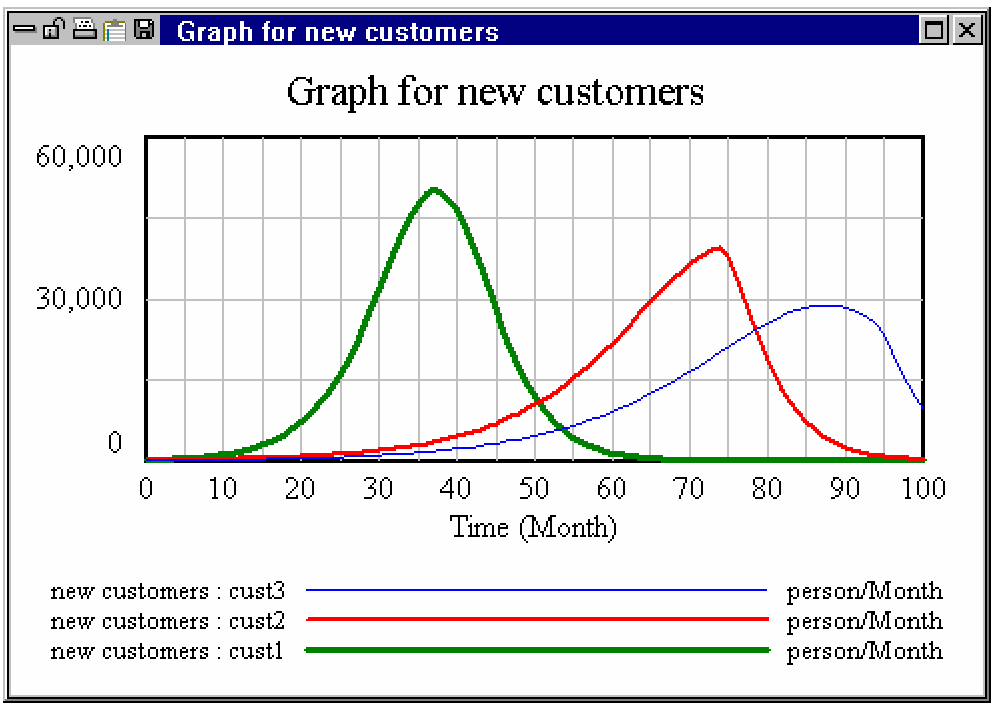
You should get 3 units errors. These are all due to the presence of different names for the things that get produced and sold — Widgets and Gadgets. We could go through each equation looking for Gadget and replacing it with Widget (or vice versa). However, Gadget and Widget are referring to the same unit and we can therefore define them as synonyms. We will also add the plural forms (Gadgets, Widgets) in case we typed a plural by mistake.

- ✍ Select the menu item **Model>Settings…** and click the tab **Units Equiv**.
- ✍ In the editing box, type Gadget, Gadgets, Widget, Widgets and then click on the button **Add Editing**, then on **OK**.
- ✍ Select **Model>Units Check** from the menu (or press Ctrl + U).

Now units should check out AOK. If they do not, read the units errors Output window and try to figure out why. Refer to the units for each variable in the equation set above.

Simulating and Analyzing the Model

- ✍ Check the model for errors with **Model>Check Model** (or Ctrl + T).
- ✍ Simulate the model using the dataset name *cust3*.
- ✍ Click on the **Control Panel** button then on the **Datasets** tab. Load the datasets *cust1* and *cust2* if not already loaded.
- ✍ Select the variable ***`new customers`*** as the Workbench Variable and click on the **Graph** tool.

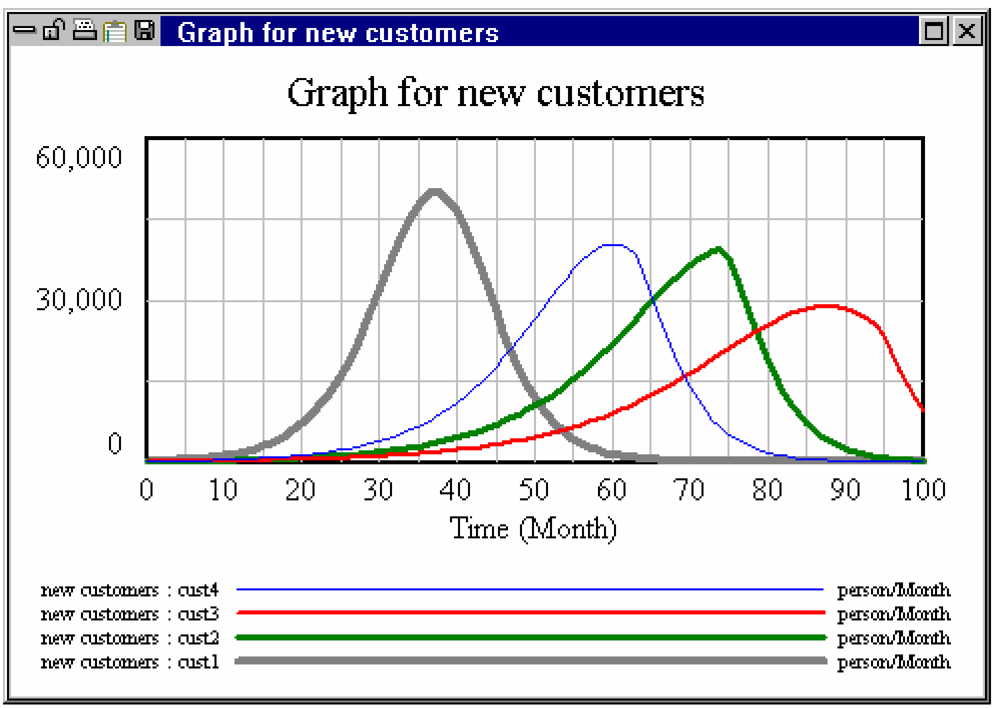


Notice how the curve for *new customers* in *cust3* builds more slowly, peaks later and lasts longer than for the other runs. This behavior comes from the constraint in *Capacity*, which is lower in *cust3.mdl* than *cust2.mdl* because of capacity reductions (retirement due to old age).

Capacity Investment Policy

In supplying a new market we stand to loose market share (*new customers* and *Customers*) because we are supplying the product later than some competitor. What can we do to supply the product (and hence bring on *new customers*) sooner? Let us try building *Capacity* faster by lowering the *time to adjust capacity*.

- ✍ Type the name *cust4* into the **Runname** editing box.
- ✍ Click on the button **Set Up a Simulation**.
- ✍ In the Capacity view, click on the highlighted Constant *time to adjust capacity* and type in the number 4 then press Enter.
- ✍ Click on the **Simulate** button.
- ✍ Select the variable *new customers* as the Workbench Variable and click on the **Graph** tool.



Now we see that *new customers* has moved closer to the ideal of the first model *cust1.mdl*. The *Capacity* constraint has been pushed back, though not eliminated.

10 Customizing Output

You can customize output by configuring analysis tools (except PLE and PLE Plus), and by creating custom graphs and tables.

Output from Analysis Tools

Except in Vensim PLE and PLE Plus you can customize all of the analysis tools to change the format and content of their output.

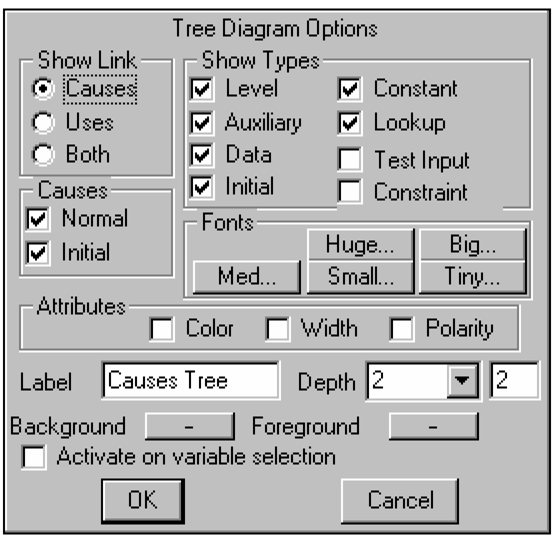
Tree Diagrams

Tree diagrams can be customized to show the graphics associated with the model: arrow thickness, color, and polarity.

✍ Open the model *project.mdl* in the directory *torial\chap04\complete*.

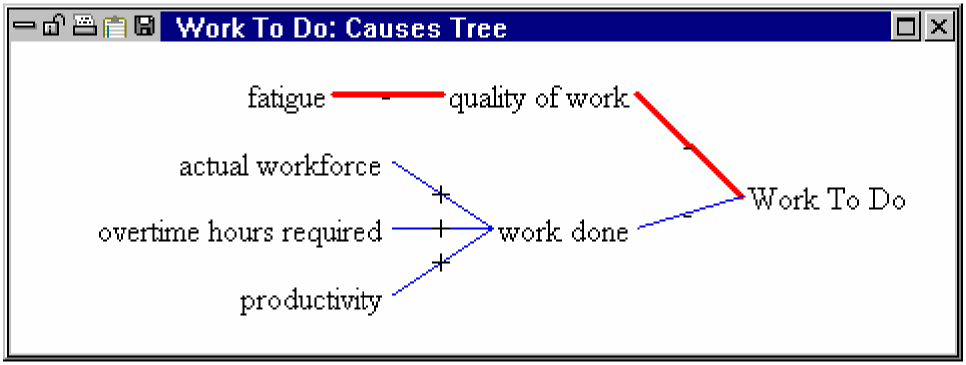
Click on `Work To Do` to select it as the Workbench Variable.

✍ Click with the right mouse button on the **Causes Tree** tool.



✍ Click the checkboxes under **Attributes** for **Color**, **Width**, and **Polarity** then click **OK**.

✍ Click with the mouse button on the **Causes Tree** tool.



Graphs and Strip Graphs

Graphsand Strip Graphs can display lines in different colors and thicknesses as well as with attached numbers and symbols. We will manipulate the options for the **Graph** tool; the same methods can be applied to the **Strip Graph** tool.

EITHER

✍ Open the *cust3.mdl* model you created in Chapter 8

OR

✍ Open the model *cust3.mdl* in the directory *guide\chap08\complete*.

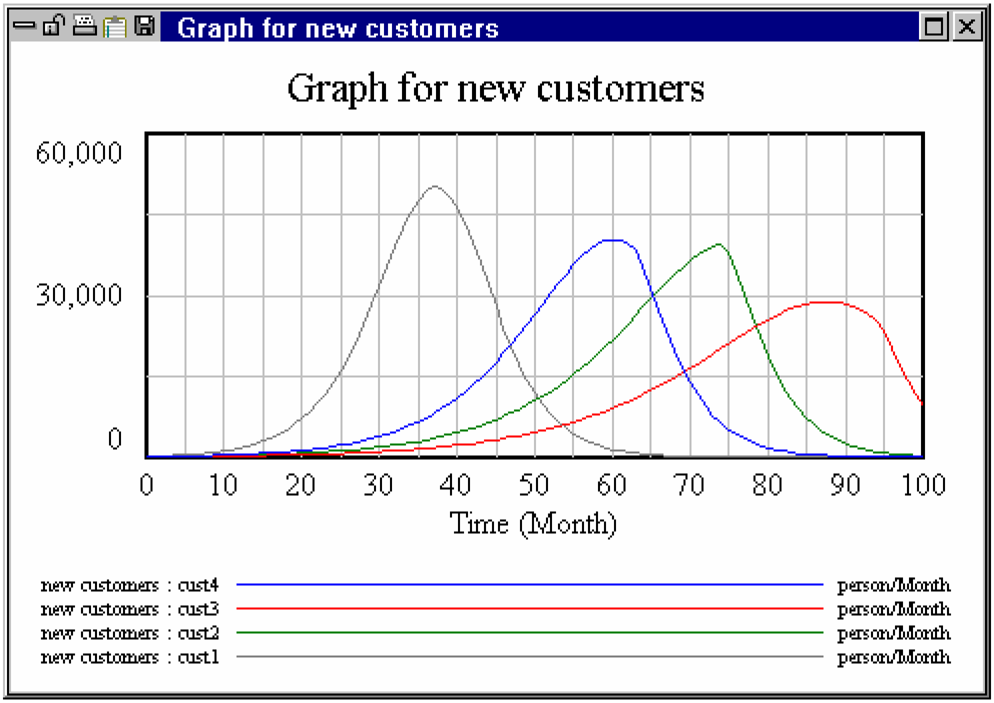
✍ Click on the **Control Panel** button then on the **Datasets** tab. Check to see that the runs *cust1* through *cust4* are all loaded. If they are not loaded, double click on each run in the **Available** list box to load them.

✍ Click on the variable `new customers` appearing in the sketch to select it as the Workbench Variable.

Thin Lines (Color)

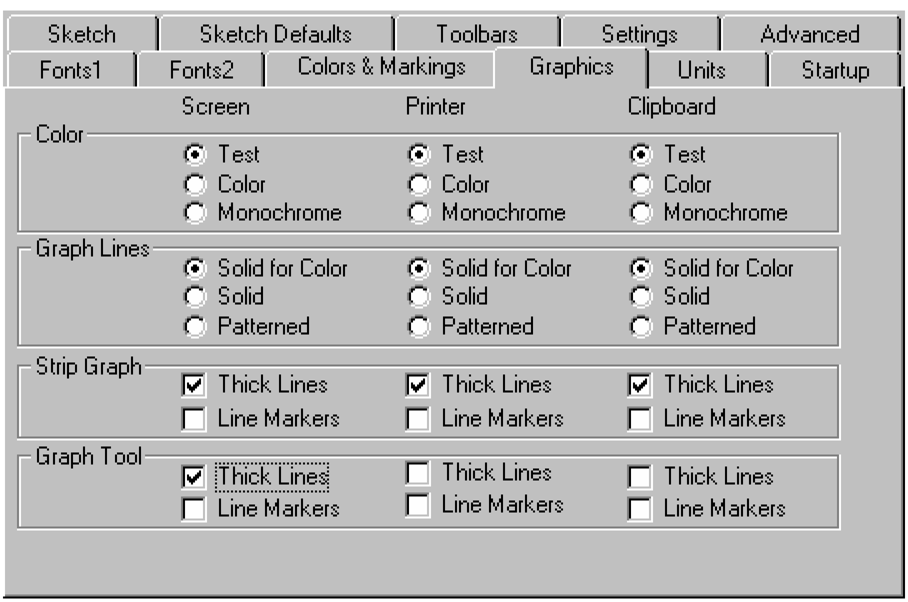
✍ Click on the **Graph** tool.

The Vensim default for the **Graph** tool is for graphing thin lines with different colors (the colors do not show in the printed User's Guide):



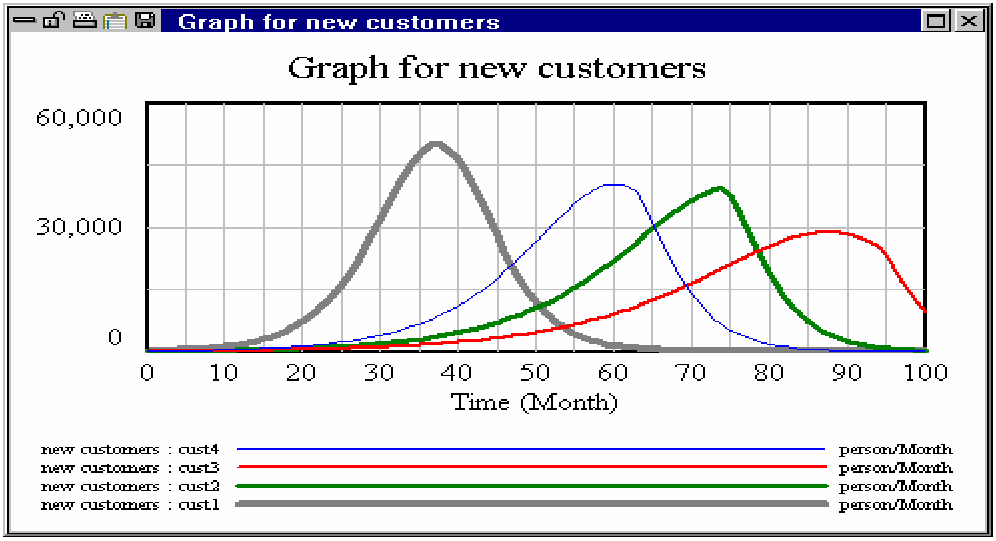
Thick Lines (Color)

✍ Select the menu **Tools>Options…** and click on the **Graphics** tab. Under the **Graph Tool** field, click the first checkbox for **Thick Lines** (underneath **Screen**). Click on **OK**.



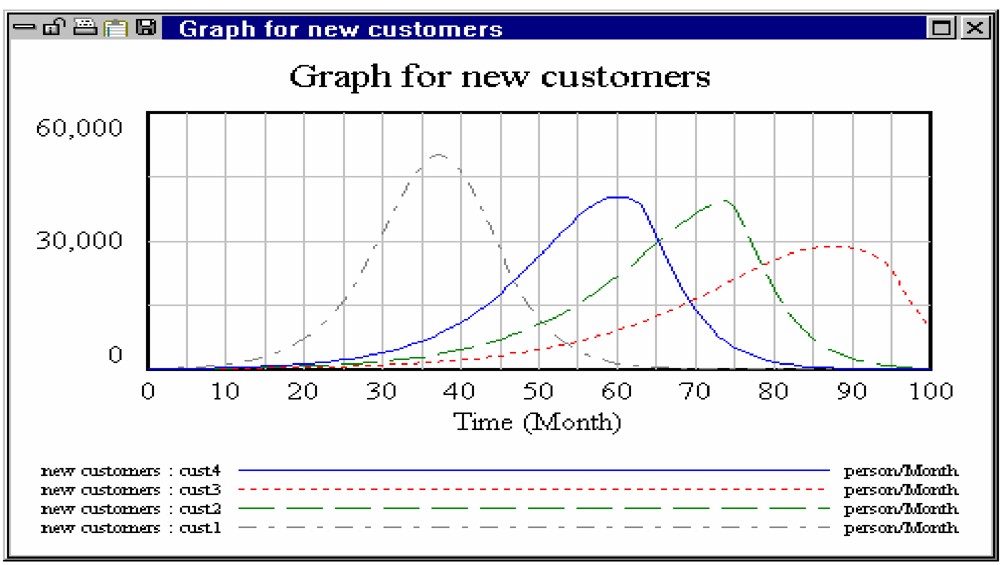
The output from the **Graph** tool will be updated to have thick lines. Note that the graphics options works on all graphs that are open (and any new graphs created).

The appearance of lines can be adjusted separately for the screen (left), the printer (center) and the clipboard (right). We are changing only the screen here.



Patterned Graph Lines (Color)

✍ Select the menu **Tools>Options…**. Under the **Graph Lines** field, click the first option button for **Patterned** (underneath **Screen**). Click on **OK**.

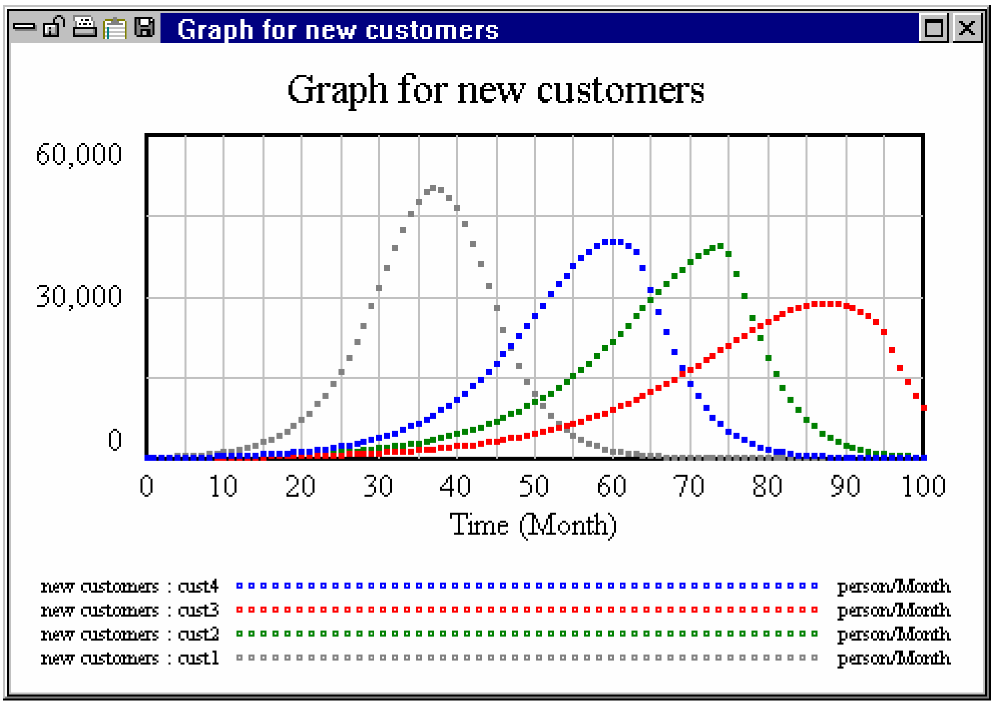


Reset the Global Options dialog to defaults by:

- Select the menu **Tools>Options…**. Under the **Graph Lines** field, click the first option button for **Solid for Color**. If you want, check **Thick lines** for **Graph tool** (if you prefer them). Click **OK**.
- Close the graph by clicking the close button or pressing the Del key.

Dots-Only Graphs

- Click with the right mouse button on the **Graph** tool. In the **Line Type** field, click the option button for **Dots Only**. Click **OK**.
- Click on the **Graph** tool.



Reset the **Graph** tool options by:

- Click with the right mouse button on the **Graph** tool. In the **Line Type** field, click the option button for **Linear Interp**. Click **OK**.

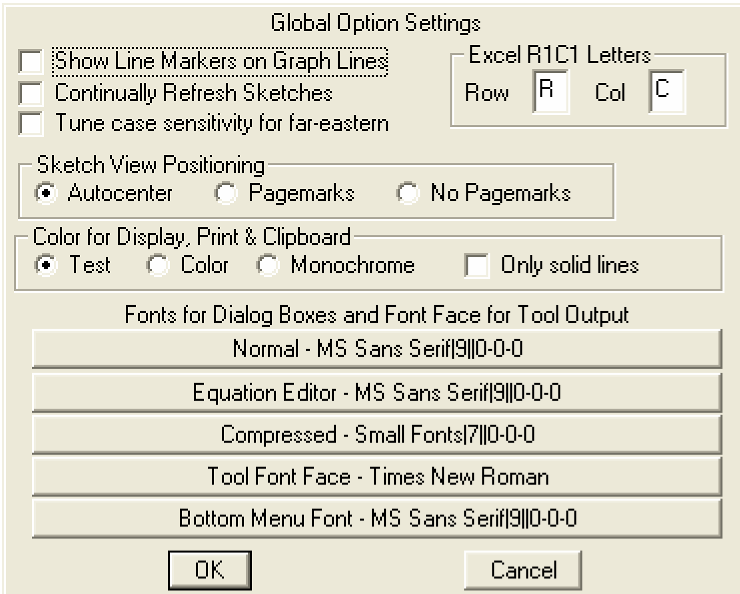
Graph Lines with Markers

In addition to different line styles, it is possible to add marked to graph lines. In Vensim PLE and PLE Plus these are simply toggled on and off, while in the other configuration they can be customized.

Marker are characters that are displayed along a graph line. The defaults are numbers (and letters), but symbols can also be used.

In Vensim PLE or PLE Plus:

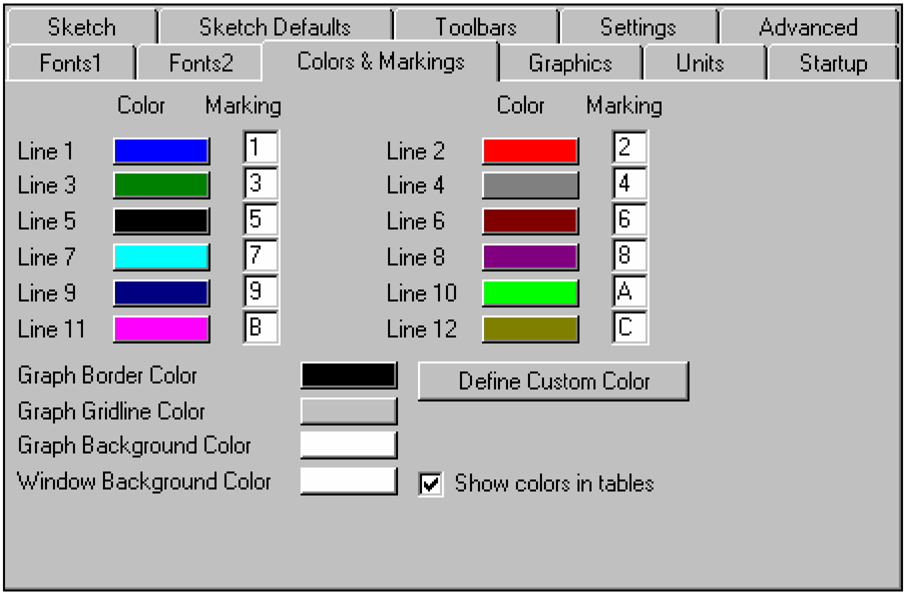
✍ Select the menu **Options>Options…**



✍ Click on Show Line Markers on Graph Lines.

In other configurations:

✍ Select the menu **Tools>Options…** and click on the **Colors & Markings** tab.

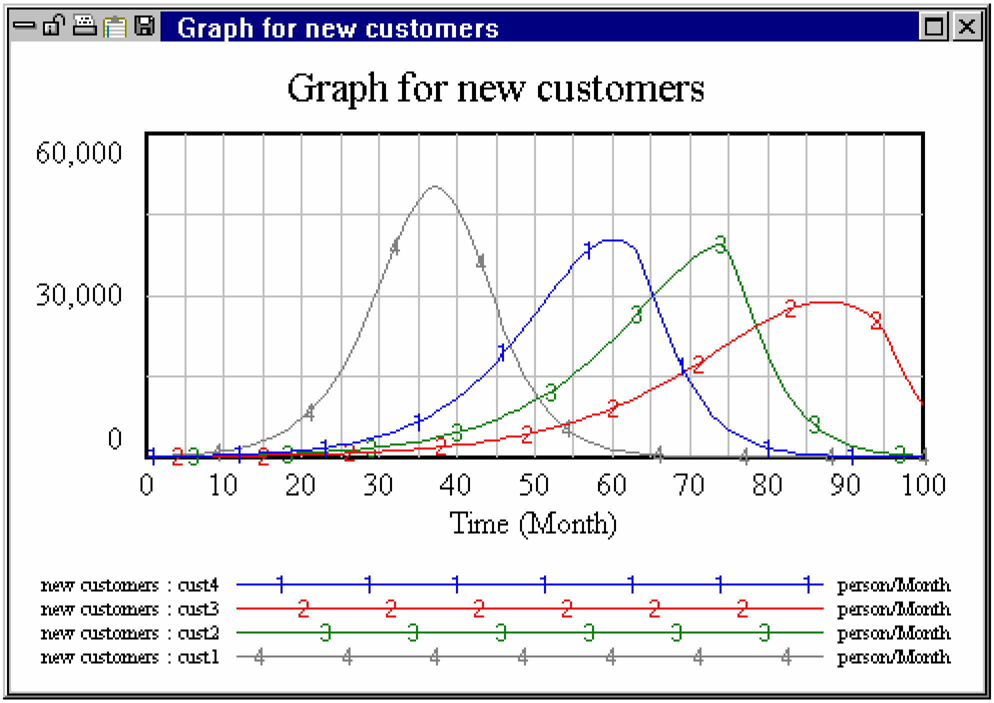


Note the **Color** and **Marking** associated with each line. You can change these colors to any other color, and change the marking to any single character.

✍ Click on the **Graphics** tab. Under the **Graph Tool** field, click the left most checkbox (under **Screen**) for **Lines Markers** so that it is checked. Click on **OK**.

Then for all configurations:

✍ Click on the **Graph** tool.



Reset the Global Options dialog to defaults by unchecking the **Line Markers** checkbox.

NOTE Changing settings on the **Colors & Markings** and **Graphics** tabs of the Global Options dialog will change the appearance of existing graphs both on the screen and when they are printed.

Custom Graphs

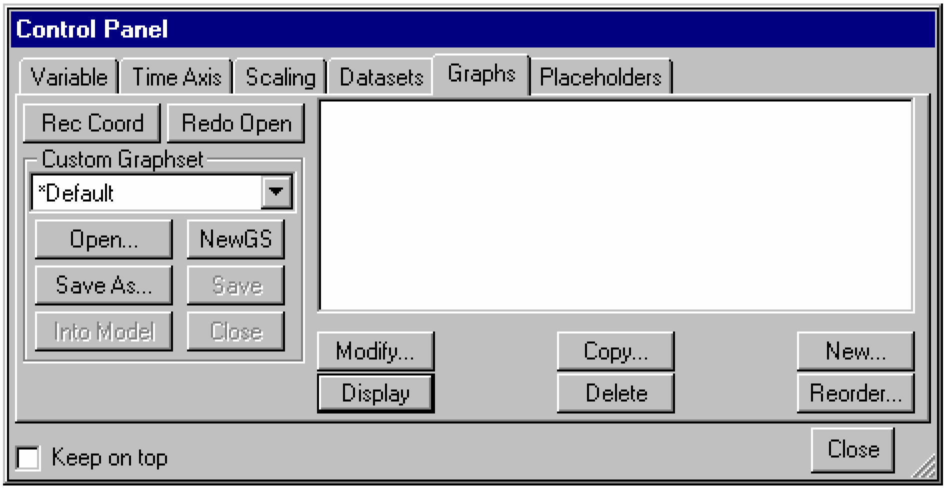
Custom Graphs are used to customize the content of a graph so as to show selected variables, runs, and style, in one graph. Custom Graphs are created from the Graph Control in the Control Panel.

If *cust3.mdl* is not already open:

✍ Open the model *cust3.mdl* in the directory *guide\chap08\complete*.

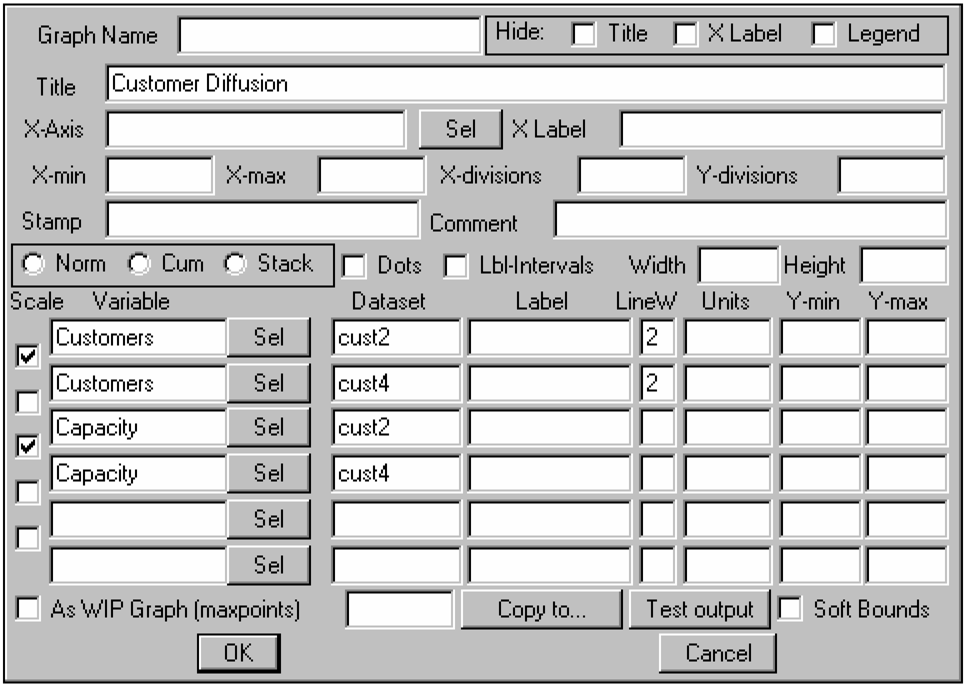
To make a Custom Graph:

✍ Click on the **Control Panel** button on the Toolbar. Click on the tab **Graphs** in the Control Panel.



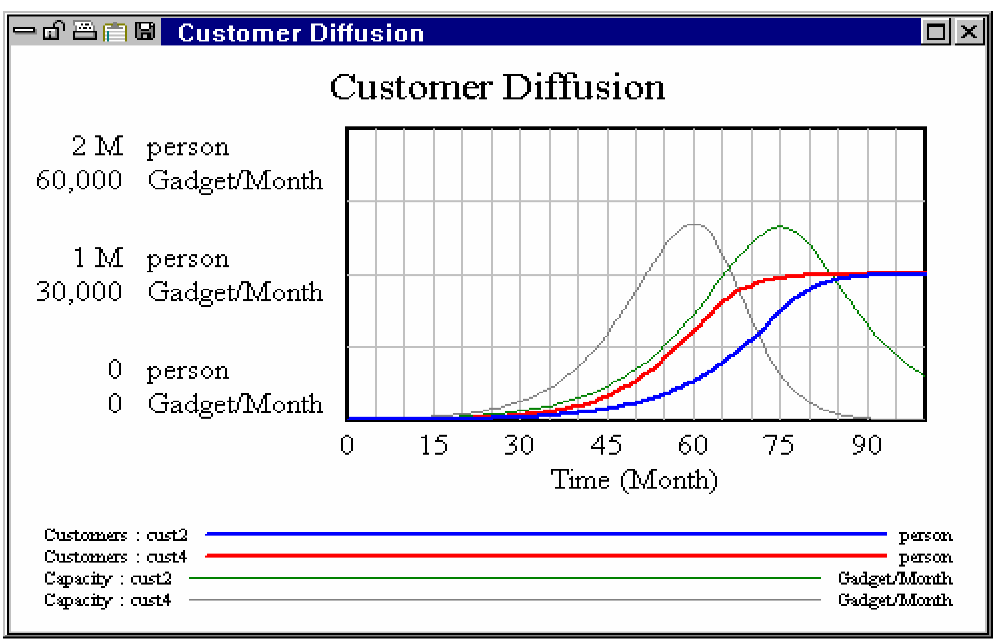
- Click the button **New…**. The Custom Graph Editor opens with the cursor positioned at the graph **Title** editing box.
- Type in a name for the graph (e.g., Customer Diffusion)
- Using the mouse, move to the **Variable** boxes on the left side of the graph editor and click on the top button labeled **Sel**. A Variable Selection dialog box appears. Move the scrollbar down the list and double click on ***`Customers`*** (or type the first letters in Customers until it is highlighted, then press Enter).
- Click on the second button down labeled **Sel**. Double click on the variable ***`Customers`***.
- Click on the third button down labeled **Sel**. Double click on the variable ***`Capacity`***.
- Click on the fourth button down labeled **Sel**. Double click on the variable ***`Capacity`***.
- Click the **Scale** checkbox that lies left and between the first two variables.
- Click the **Scale** checkbox that lies left and between the third and fourth variable.
- Click on the **Dataset** editing box just right of the first variable. Type in the run name *cust2*.
- Click on the **Dataset** editing box just right of the second variable. Type in the run name *cust4*.
- Click on the **Dataset** editing box just right of the third variable. Type in the run name *cust2*.
- Click on the **Dataset** editing box just right of the fourth variable. Type in the run name *cust4*.
- Click on the **LineW** (Line Width) editing box just right of the first variable, type 2.
- Click on the **LineW** (Line Width) editing box just right of the second variable, type 2.

The Custom Graph Editor should look like this:



✍ Click on **OK**. The Custom Graph Editor closes and the Graph Control is left open.

✍ Click on the name of your graph in the Graph Control, then click on the button **Display**.



The graph is displayed. You can easily see how the behavior of one variable relates to another. Experiment with different options in the Custom Graph Editor by clicking on the button **Modify** in the Graph Control, and changing things. If you do not include a dataset next to a variable name in the graph dialog, the **Display** button will show the first loaded run (the run at the top of the **Loaded** list in the Datasets Control).

Custom Tables

Custom Tables are used to customize the content of a table so as to show selected variables and runs in one table. Like Custom Graphs they are created from the Graph Control in the Control Panel.

If *cust3.mdl* is not already open:

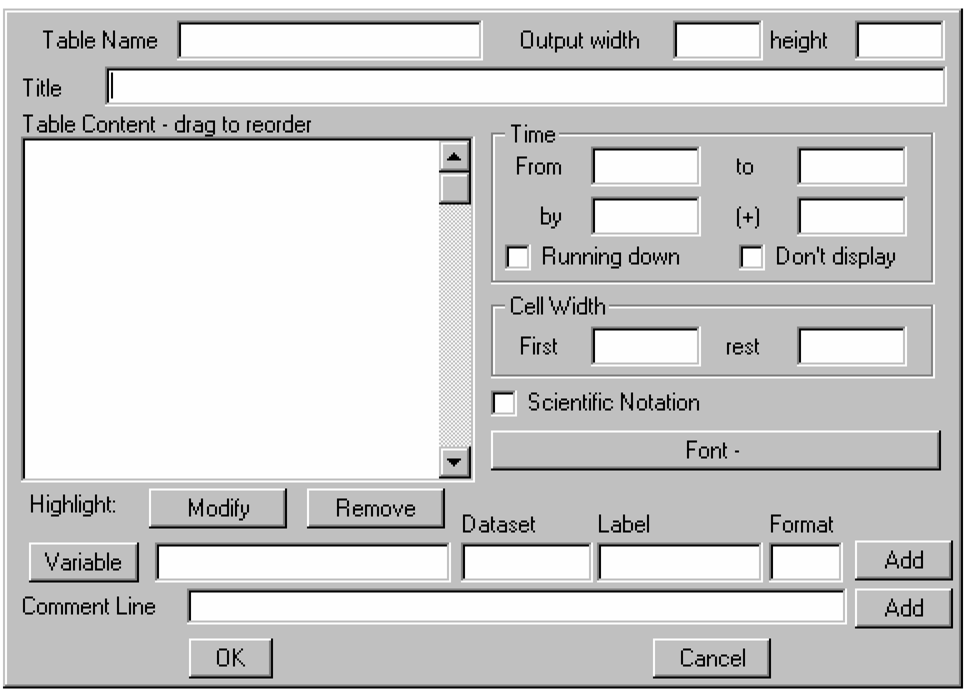
✍ Open the model *cust3.mdl* in the directory *guide\chap08\complete*.

To make a Custom Table:

✍ Click on the **Control Panel** button on the Toolbar. Click on the tab **Graphs** in the Control Panel.

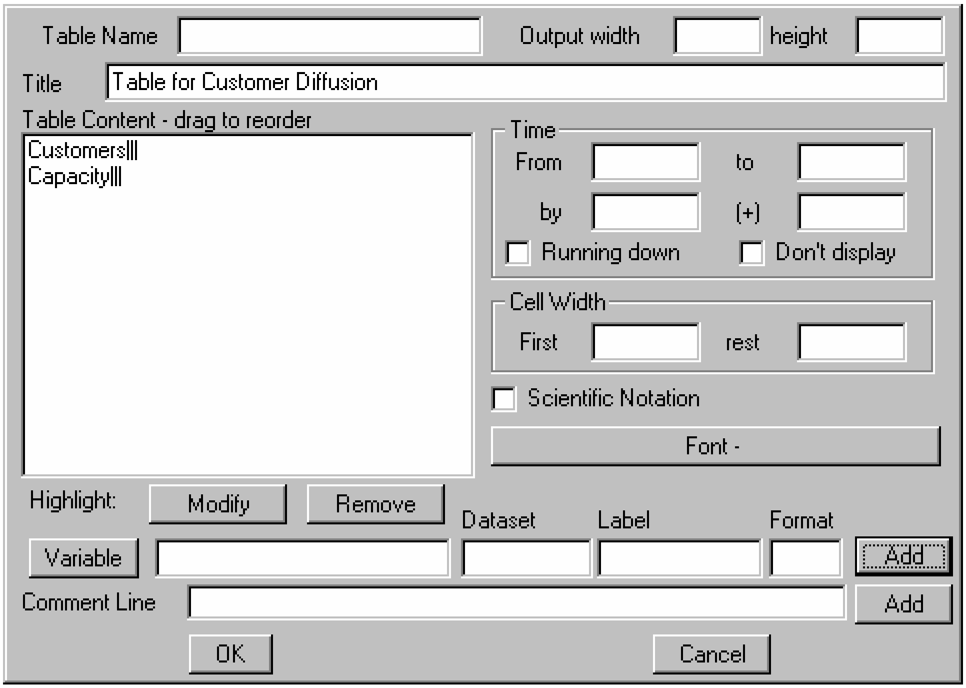
✍ Click the button **New…**. The Custom Graph Editor opens with the cursor positioned at the graph **Title** editing box.

✍ Click on **As Table...** on the bottom of the dialog. The Custom Graph Editor opens with the cursor positioned at the graph **Title** editing box.

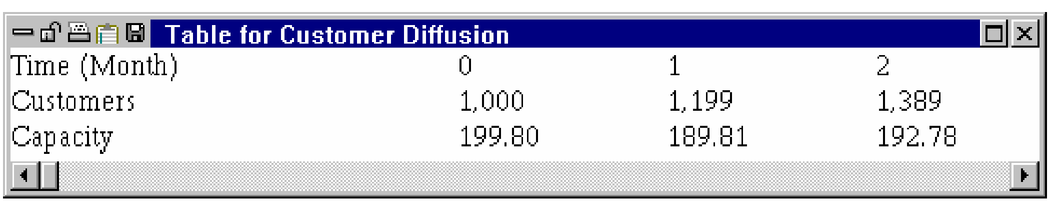


✍ Type in a name for the graph (e.g., Table for Customer Diffusion)

- ✍ Using the mouse, click on the button labeled **Variable** on the lower left of the dialog. A Variable Selection dialog box appears. Move the scrollbar down the list and double click on ***Customers*** (or type the first letters in Customers until it is highlighted, then press Enter).
- ✍ Click on the **Add** button to the right of the variable name you just entered.
- ✍ Click again on the **Variable** button and double click on the variable ***Capacity***.
- ✍ Click again on the **Add** button.
- ✍ The Custom Table Editor should look like this:



- ✍ Click on **OK**. The Custom Table Editor closes and the Graph Control is left open.
- ✍ Click on the name of your table in the Graph Control, then click on the button **Display**.



The table is displayed, with the variables you chose. You can easily add additional variables and change the labels that appear in the left column.

Experiment with different options in the Custom Table Editor by clicking on the button **Modify** in the Graph Control, and changing things. If you want to change the order that variables are displayed simply drag them to a different position in the list. You can also add comments and other information. You can specify a format following C format conventions (for examp le %.0f for numbers rounded to integers and %.6g for 6 decimal places) for each variable.

11 Games in Vensim

This material does not apply to Vensim PLE.

What Are Games?

Games are a way of actively engaging in the progress of a simulation. Games are examples of the "flight simulator" approach, where the user participates in decisions that affect the simulation outcome for each step in time. A Vensim simulation model can be run as a game by stepping through time and making changes to gaming variables along the way. In contrast, a normal simulation model runs through the complete time span based on the initial setup of the model.

The Real Estate Game (houses.mdl)

EITHER

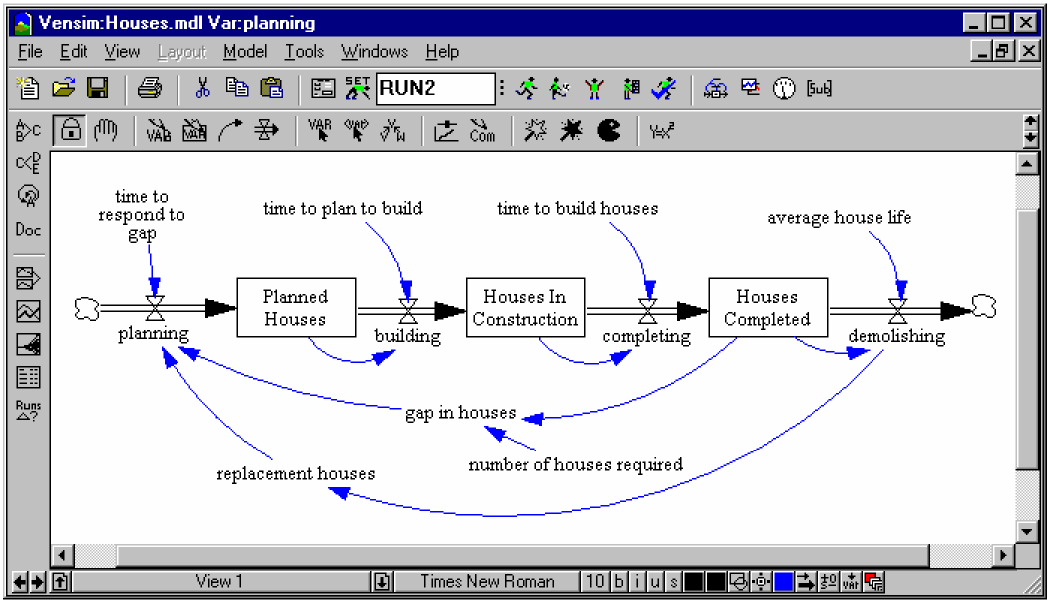
✍ Click the **Open Model** button and open the model *houses.mdl* in the directory *guide\chap11*.

OR

✍ Build the model as shown in the diagram and equation listing below and save it in the directory *guide\chap11* with a different name (e.g., *myhouses.mdl*). Time Bounds are INITIAL TIME = 0, FINAL TIME = 100, TIME STEP = 0.5, Units for Time: Month.

Model Structure

Below is a model of construction in the real estate industry. A long delay exists between the need to close the gap in houses and the completing of construction of those houses. This model features a negative feedback loop with several delays.



houses.mdl Equations

```
average house life = 1200
Units: Month

building = Planned Houses / time to plan to build
Units: house/Month

completing = Houses In Construction / time to build houses
Units: house/Month

demolishing = Houses Completed / average house life
Units: house/Month

gap in houses = number of houses required - Houses Completed
Units: house

Houses Completed = INTEG( completing - demolishing , 5000)
Units: house

Houses In Construction = INTEG( building - completing ,
         building * time to build houses )
Units: house

Planned Houses = INTEG( planning - building ,
         planning * time to plan to build )
Units: house

planning = MAX ( 0, replacement houses + ( gap in houses /
         time to respond to gap ) )
Units: house/Month
```

```
replacement houses = demolishing
Units: house/Month

time to build houses = 6
Units: Month

time to plan to build = 3
Units: Month

time to respond to gap = 8
Units: Month
```

Built-in Functions

We are going to start this model in equilibrium. We will set `number of houses required` to 5000, which is the initial value for `houses completed`. Because of this, `gap in houses`, will be 0 so that `planning` is equal to `replacement houses` which, in turn is equal to `demolishing`. Because of the way we initialized the other levels (for example `Planned Houses` is initialized equal to `planning * time to plan to build`) each of these is already in equilibrium. We should therefore have a mo del that will simulate without any values changing.

While it is important to check to be sure that our understanding is correct and the model will simulation without anything changing, we also want to see more model behavior. Thus, instead of using just 5000 for `number of houses required` we want to hold it at 5000 for a time (say ten months) and then increase it (say to 5050). To do this we use the equation:

```
number of houses required = 5000 + STEP ( 50, 10)
Units: house
```

The STEP function takes two arguments, **height** and **start time**, which are enclosed in parentheses. It takes on a value of 0 until **start time** is reached and from then on takes on value **height**. This function is a particularly good input for a model because it is a simple input change that generates a broad behavior response. Other functions that are useful for "exciting" or "disturbing" a model in this way include PULSE and RAMP.

To add the above equation, open the Equation Editor on `number of houses`.

✍ Type in the number 5000, then the plus sign.

✍ Click on the **Functions** tab then scroll down until you see the STEP function appear in the list. Click once on the STEP function and click on the **Add Sel** button.

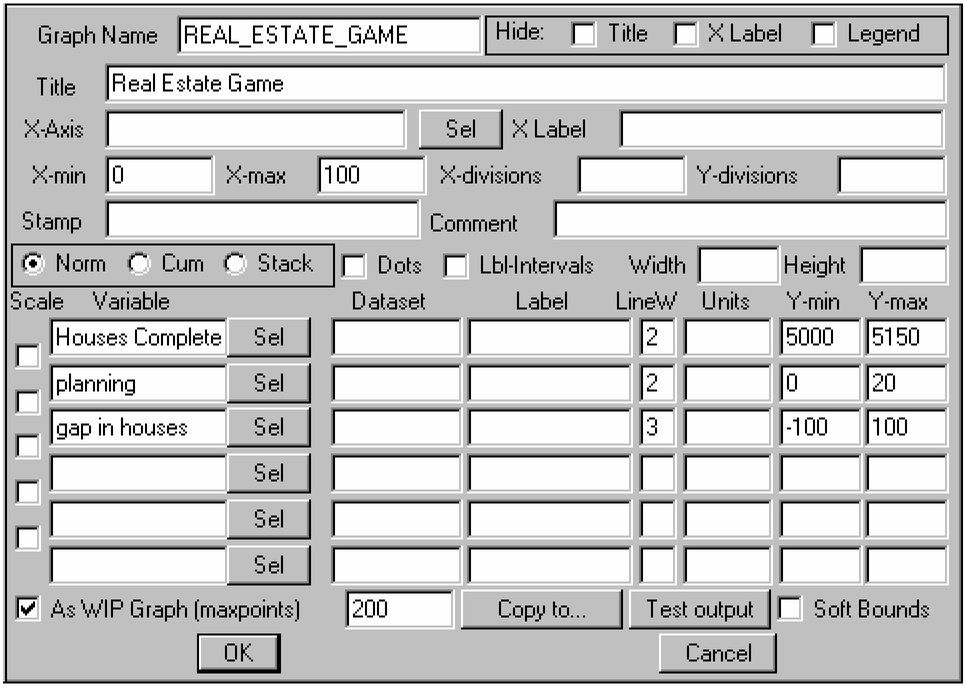
✍ The argument {height} should be highlighted, just type over it with the value 50.

✍ Double click on the {stime} argument and type in 10.

✍ Enter the units `new` and click **OK**.

WIP Graph

✍ Create the Custom Graph definition shown below. Be sure to check the **As WIP Graph** checkbox. Chapter 10 describes how to create Custom Graphs.



Adding Game Variables

The goal in playing this game is to meet the demand for houses (a zero ***`gap in houses`***); you do this by setting and changing the variable ***`planning`*** which introduces new houses into the planning and construction process. Right now ***`planning`*** is determined by a formula. This formula allows you to simulate the model, but does not provide a mechanism for you to intervene and change the value of ***`planning`*** during a simulation. You need to change ***`planning`*** to a Game variable. To do this:

✍ Select the **Equations** tool.

✍ Click on the variable ***`planning`***.

We see the equation:

```
planning = MAX( 0, replacement houses + (gap in houses /
         time to respond to gap))
Units: house/Month
```

The equation is formulated so planning can never go negative; you can plan to build some houses (positive) or plan to build no houses (zero). To make this a Game variable, we change the variable type in the lower drop-down variable type box:

✍ Click on the drop-down arrow on the lower variable **Type** box (currently reads **Normal**) and choose **Gaming** from the list. Click the **OK** button.

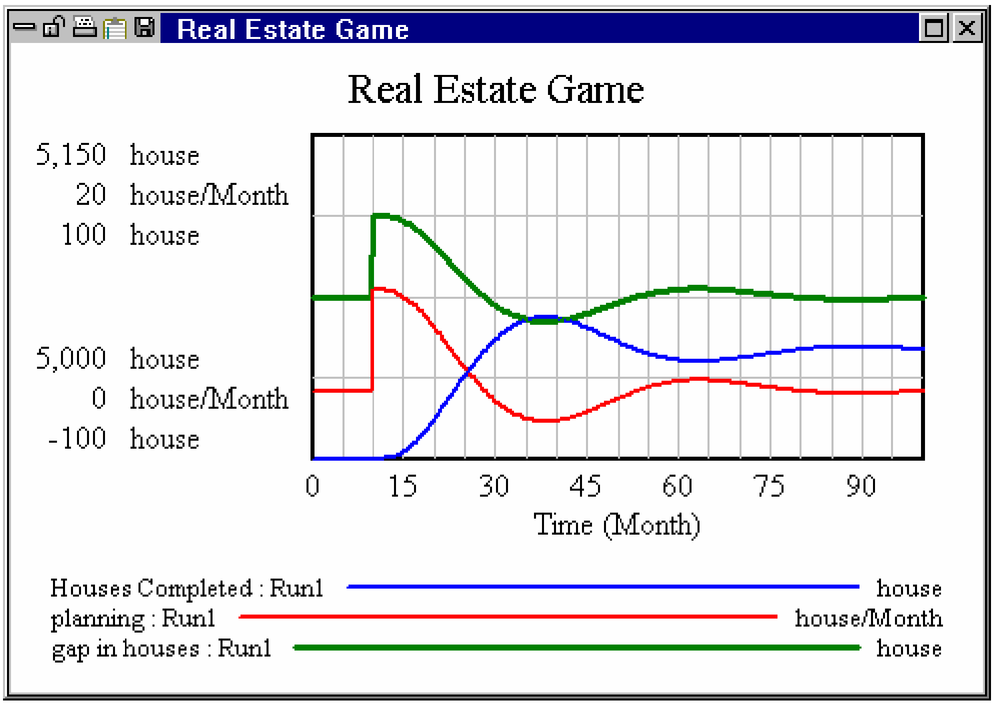
✍ Save the model.

You can convert any Auxiliary, Rate or Constant into a Game variable in this manner. During a simulation a Game variable does just what it would do if it were an Auxi liary, Rate or Constant. However, during a game, you can set the value of a Game variable at each time as the game progresses.

Simulating the Model

Before playing the game, let us look at how the model behaves when simulated.

✍ Double click on the **Runname** editing box, type in *run1* (or any name), then click the **Simulate** button.

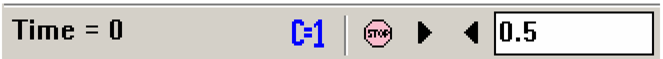


A Work-In-Progress (WIP) custom graph is generated showing behavior for three key variables in the model: ***Houses Completed***, ***planning***, and ***gap in houses*** (the thickest line). Note the oscillation: overshoot then undershoot of the goal. The model is trying to drive the ***gap in houses*** to zero. The step in the model is coming from ***number of houses required***. Let us see if we can do any better by planning houses ourselves to try and keep a zero ***gap in houses***.

Playing the Game

✍ Double click on the **Runname** editing box, type in *game1* (or any name), then click the **Game** button.

A WIP (Work-In-Progress) Custom Graph is generated and the Toolbar changes to the Game Toolbar:



The Game **Time** is displayed on the left. The **Change Gaming Variables** button (C=1) provides one way to change the value of Game variables during the game. The **Stop** button stops the game. **Move Forward** and **Move Backward** buttons move the game by the amount shown in the **Amount to Move** editing box (currently displaying 0.5). The three standard window class buttons are on the right.

Moving Forward in a Game

✍ Move the WIP graph to the right side of your screen so that it is not covering the sketch.

You should see that the variable `planning` is highlighted yellow with blue text. This provides the second way to change the value of Game variables during the game.

✍ Click on the variable `planning`, you will see its initial value (4.166), press Enter to leave without changing the value.

Note that the WIP graph disappeared behind the Build window. If we change the Game variable from the toolbar using the button **Change Gaming Variables**, the WIP graph will stay on top. Another way to keep the two windows visible is to reduce the size so both fit on your screen.

✍ Click on the reduce button  for the Build window (upper right corner, but below the reduce button for the Vensim application).

The Build window containing the sketch will shrink to a smaller size.

✍ Resize and position the Build window and the graph window so you can see both the model (or at least the variable `planning`) and the WIP graph.

✍ Double click on the **Amount to Move** editing box on the Game Toolbar and type in **5**.

✍ Click on the **Move Forward** button 

The WIP graph will start to draw. Behavior is in equilibrium; we do not need to change any parameters because the `gap in houses` is currently zero (exactly on the middle gridline).

✍ Click on **Move Forward** once more, you will see the step upwards in the `gap in houses`.

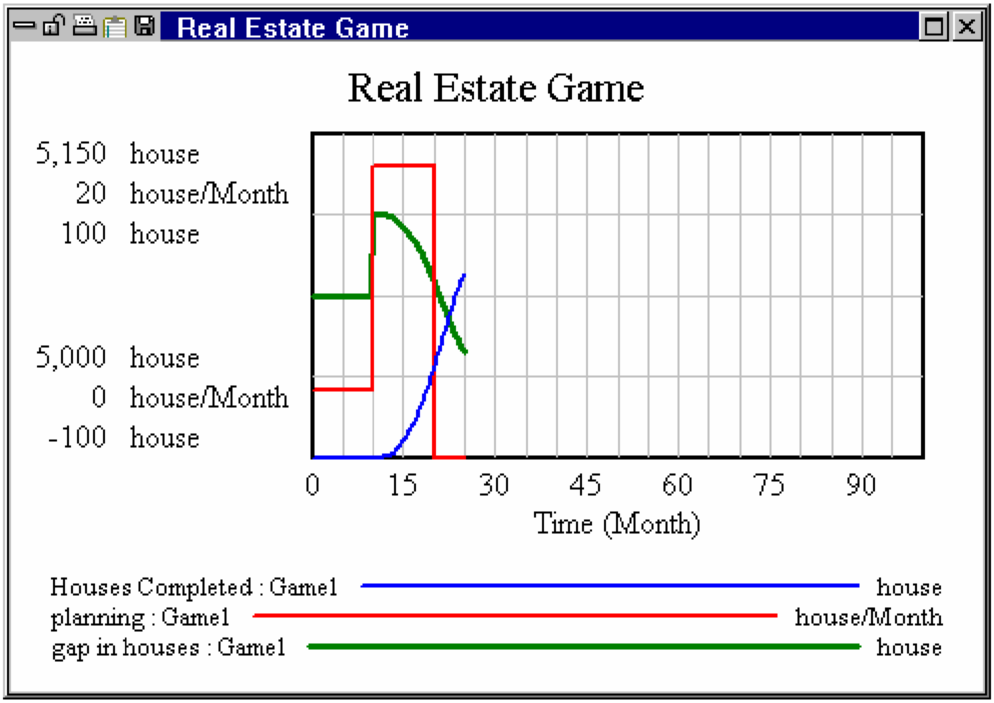
✍ Click on `planning` on the sketch, type a value of 18, then press Enter.

✍ Click on **Move Forward** two more times.

See how `gap in houses` reduces, while `Houses Completed` takes an upturn. We have almost closed the gap to zero. We had better stop building so many houses.

✍ Click on `planning` on the sketch, type in 0, then press Enter.

✍ Click on Move Forward.



Wow! We have overshot the mark; our goal (`gap in houses`) is now negative (below the middle gridline). Since we cannot plan negative houses, we had better plan zero houses for a while.

✍ Click on **Move Forward** until the `gap in houses` is positive (just above zero at about Time = 50).

Now we should start building again so that we don't get a positive `gap in houses` (where more houses are required). We can anticipate this somewhat by building a little before `gap in houses` is positive.

Backing Up in a Game

✍ Click on the **Move Backward** button  twice (until `gap in houses` is negative).

Of course, we can't back up the real world. But in order to try out different options when the game has progressed too far in some direction, we can back up in the game.

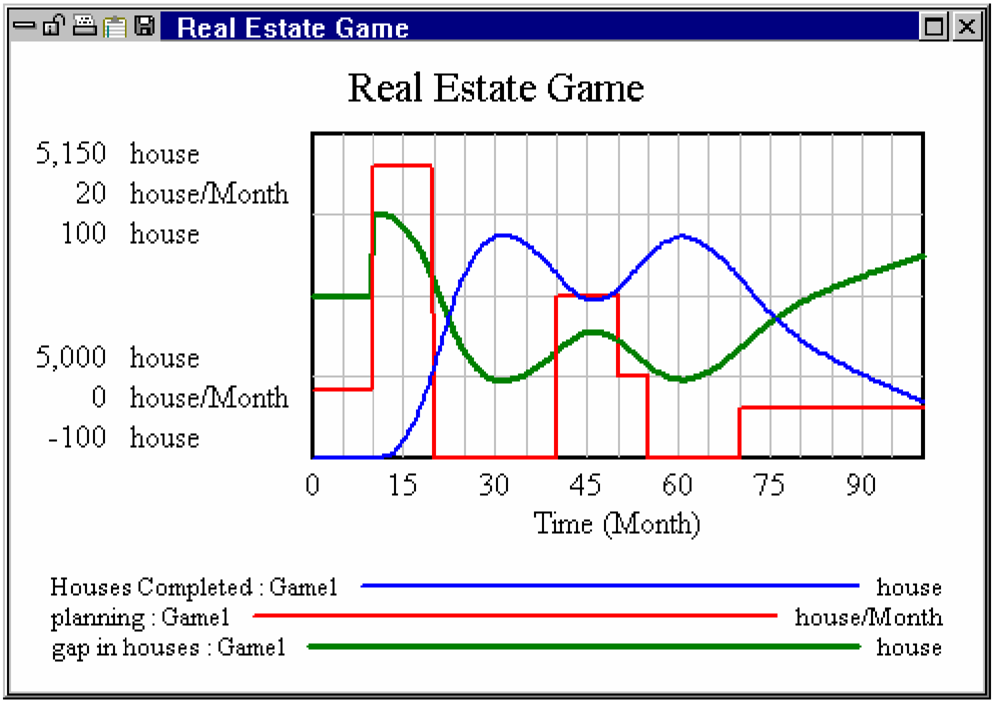
✍ Click on `planning` on the sketch, type in a larger number (for example, 10), then press Enter.

✍ Click on Move Forward.

✍ Continue playing the game, trying to keep the `gap in houses` at or near zero until the final time of 100 is reached.

✍ Click the **Stop** button 

Your WIP graph will end up looking something like this:



Your game results are probably not much better (and maybe even worse) than the original simulation. In the graph above, `gap in houses` (which we tried to keep at zero) fluctuates wildly in response to our `planning` decisions.

12 Input Output Controls

You can customize the way your sketches work with simulation models by adding Input and Output Controls to model views. These can be added alongside model structure or in separate views from model structure. Using Input Output Controls you can build your own "control room" for managing model inputs and viewing simulation results.

In this chapter you will build a control room for an existing model. Input Output Controls are not part of model structure and, therefore, do not influence model behavior. They can easily be added to a finished model to make it easier for another person to use the model. The Controls are supported in the Vensim Model Reader and provide a simple mechanism to make your models easier to consume. Input Output Controls also adapt to changes in model structure quite well. If you rename a variable the corresponding control will automatically be updated. If you delete a variable or change its type any associated controls will no longer work, but this will not stop you from working on the model. Rather they will simply appear blank or inactive.

Word of Mouth Sales

For this example we will use a word of mouth sales model. Some of the conceptual foundations for developing models of this kind are covered in Chapter 4 of the Modeling Guide.

EITHER

✍ Open the model *wom1.mdl* contained in the directory *guide\chap12*.

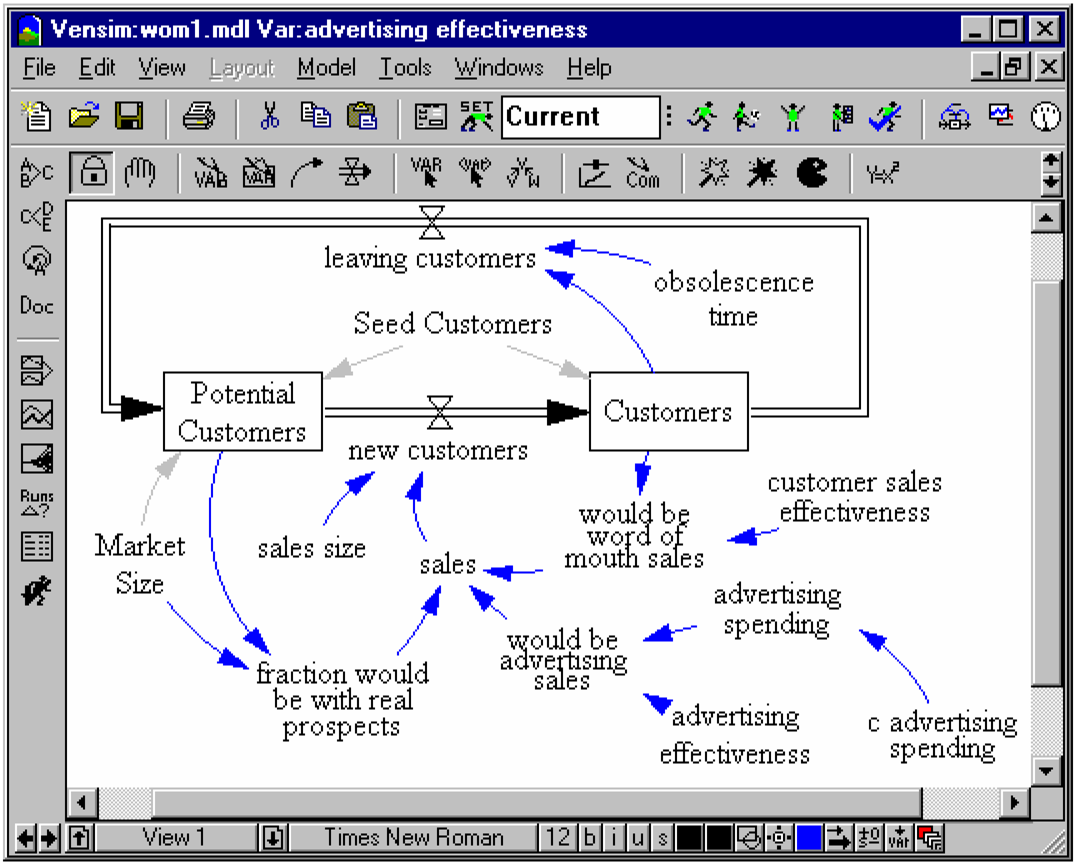
OR

✍ Build the model as shown in the diagram and equation listing below and save it in the directory *chap12* as *wom2.mdl*. Time Bounds are INITIAL TIME = 0, FINAL TIME = 5, TIME STEP = 0.0625, Units for Time: Year.

wom1.mdl Equations

```
advertising effectiveness = 0.1
Units: Widget/$

advertising spending = GAME( c advertising spending )
Units: $/Year

c advertising spending = 2e+007
Units: $/Year
customer sales effectiveness = 3
Units: Widget/(Year*Person)

Customers = INTEG(new customers - leaving customers, Seed Customers)
```

```
Units: Person
```




```
fraction would be with real prospects =
               Potential Customers / Market Size
Units: Dmnl

leaving customers = Customers / obsolescence time
   Units: Person/Year

Market Size = 1e+008
Units: Person

new customers = sales / sales size
Units: Person/Year

obsolescence time = 2
Units: Year

Potential Customers = INTEG( - new customers + leaving customers ,
          Market Size - Seed Customers )
Units: Person
```

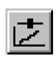
```
sales = (would be word of mouth sales + would be advertising sales)
   * fraction would be with real prospects
Units: Widget/Year

sales size = 1
Units: Widget/Person

Seed Customers = 10000
Units: Person

would be advertising sales = advertising spending *
   advertising effectiveness
Units: Widget/Year

would be word of mouth sales =Customers*customer sales effectiveness
Units: Widget/Year
```

Output Controls

Begin by creating a new view to which we can add Output Controls.

✍ Click on the Status Bar View button (this is at the bottom of the screen and labeled **View 1** for this model) and select ****New****.

A new and empty View will open.

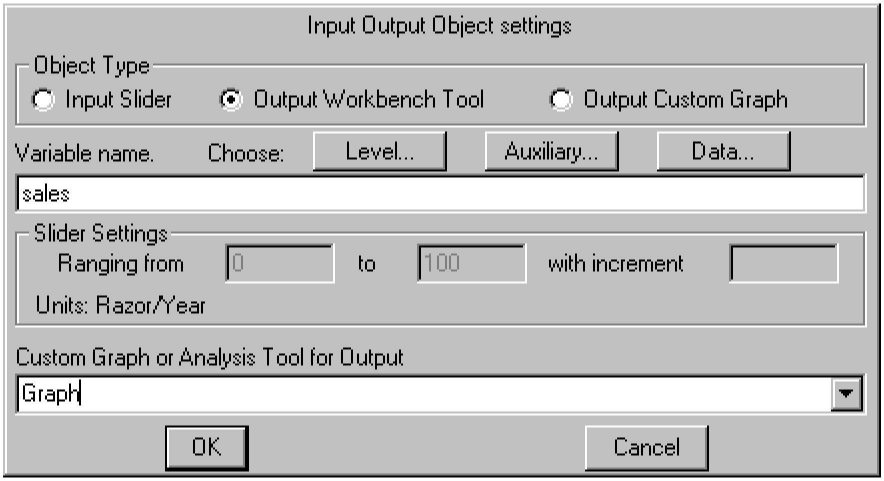
✍ Select the **Input Output** tool  by clicking on it or by pressing the 0 (zero) key.

✍ Move the mouse to the right hand side of the sketch and click. This will open the Input Output Object Settings dialog.

✍ Select type **Output Workbench Tool** then click on the **Auxiliary** button and Select `sales` from the list (clicking OK to close the Variable Selection dialog).

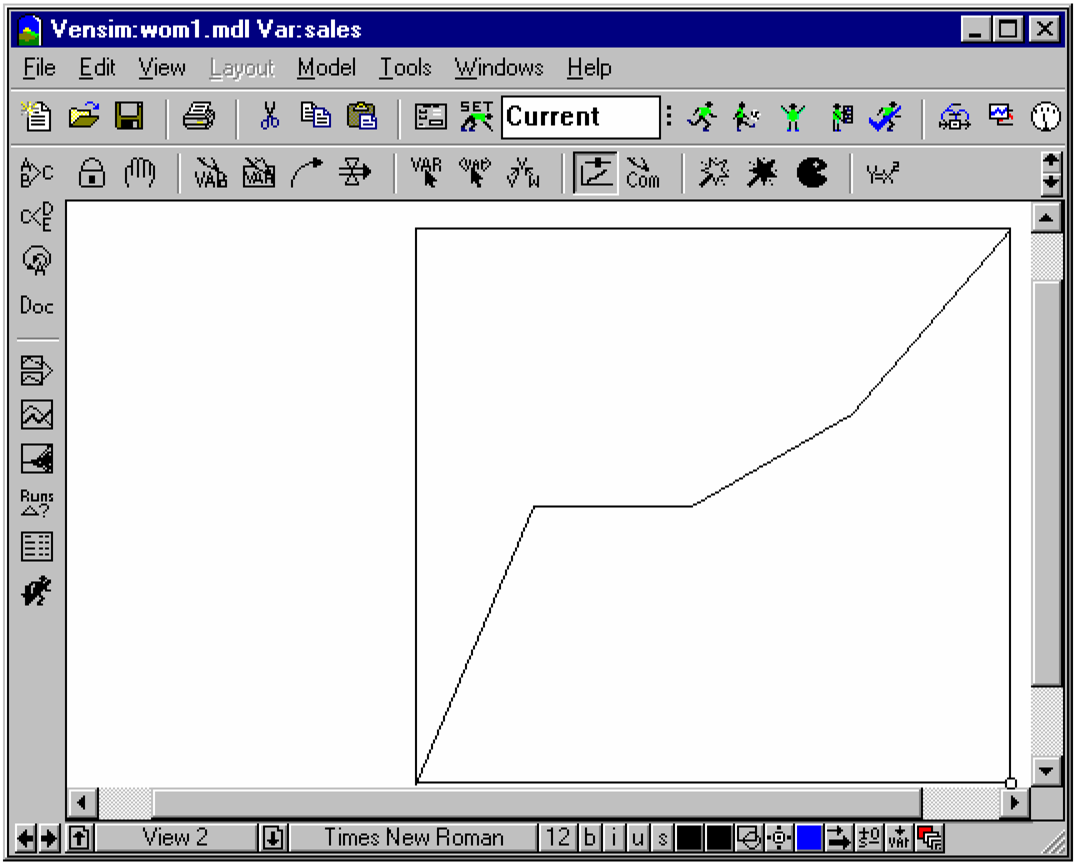
✍ Click on the dropdown and select the tool **Graph** from the dropdown list.

You dialog should look like:



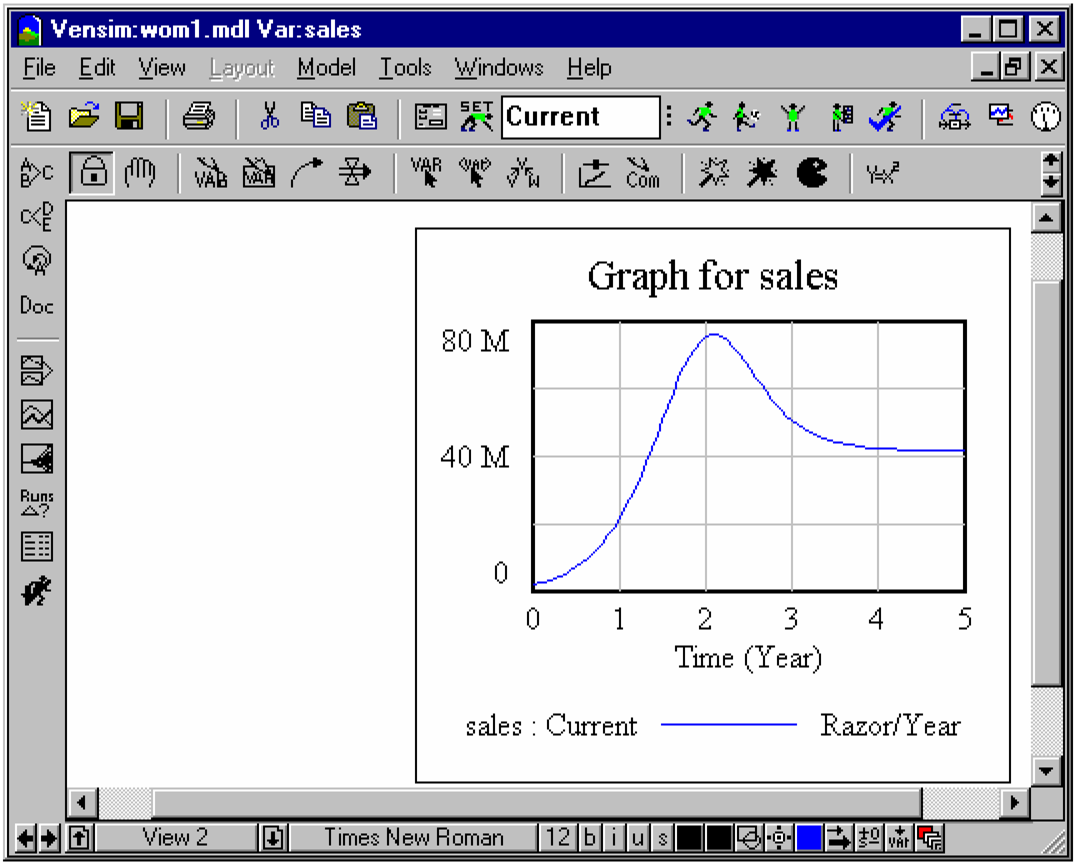
✍ Click on **OK**.

You will see a fairly large Rectangle with the sizing handle visible. Resize this to fill roughly the right hand side of the sketch. Your model should look like this:



✍ Click on the **Simulate** button  on the Toolbar (or type Ctrl+R).

The model will simulate and the graph will fill in.



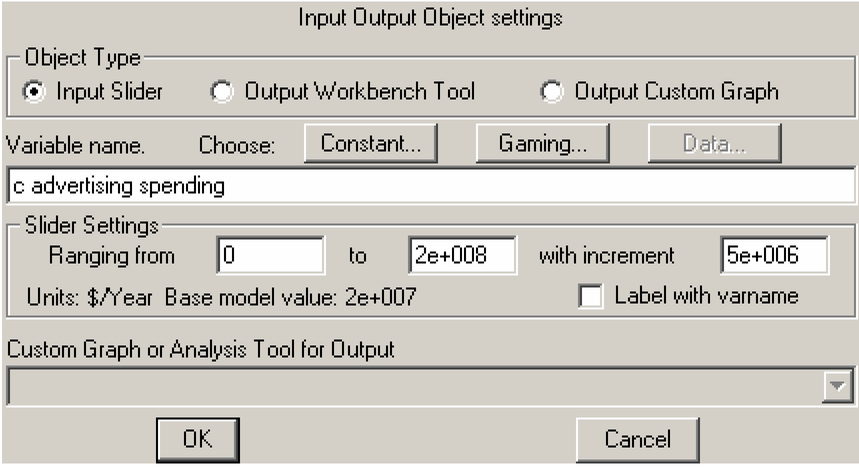
Input Controls

We will continue working in the same view.

Slider for Advertising

- Select the **Input Output** tool and click on the upper left portion of the sketch. The Input Output Object Settings dialog will open.
- Select type **Input Slider** (probably already selected by default).
- Click on the **Constant...** button and select `c advertising spending` from the dialog.
- Enter the numbers 0, 200 E6, and 0.5E6 in the **Ranging from**, **to**, and **with increment** fields.
- Click on **Label with varname** to uncheck it.

Your dialog should look like:



✍ Click on **OK**.

A Slider will be drawn on the screen.

✍ Adjust its size so that it more or less takes up the space to the graph.

✍ Select the **Comment** tool. Click above the Slider.

✍ Type in the comment "Advertising Spending ($/Year)," select Shape **None** and click on OK. This is a label for the Slider you just created.

Slider for Product Life

Now repeat this entire process creating a Slider for *`obsolescence time`* with a range from 0.5 to 10 and increment of 0.1. Place the label "Product Lifetime (years)" above this slider.

Other Changeable Constants

✍ Select the **Shadow Variable** tool  and add in the model Constants *`customer sales effectiveness`* and *`Market Size`*.

✍ Highlight the new Shadow Variables and set the color on them to black from the Status Bar color button or Right-Click on them in turn and set their text color to black.

Lining Things Up

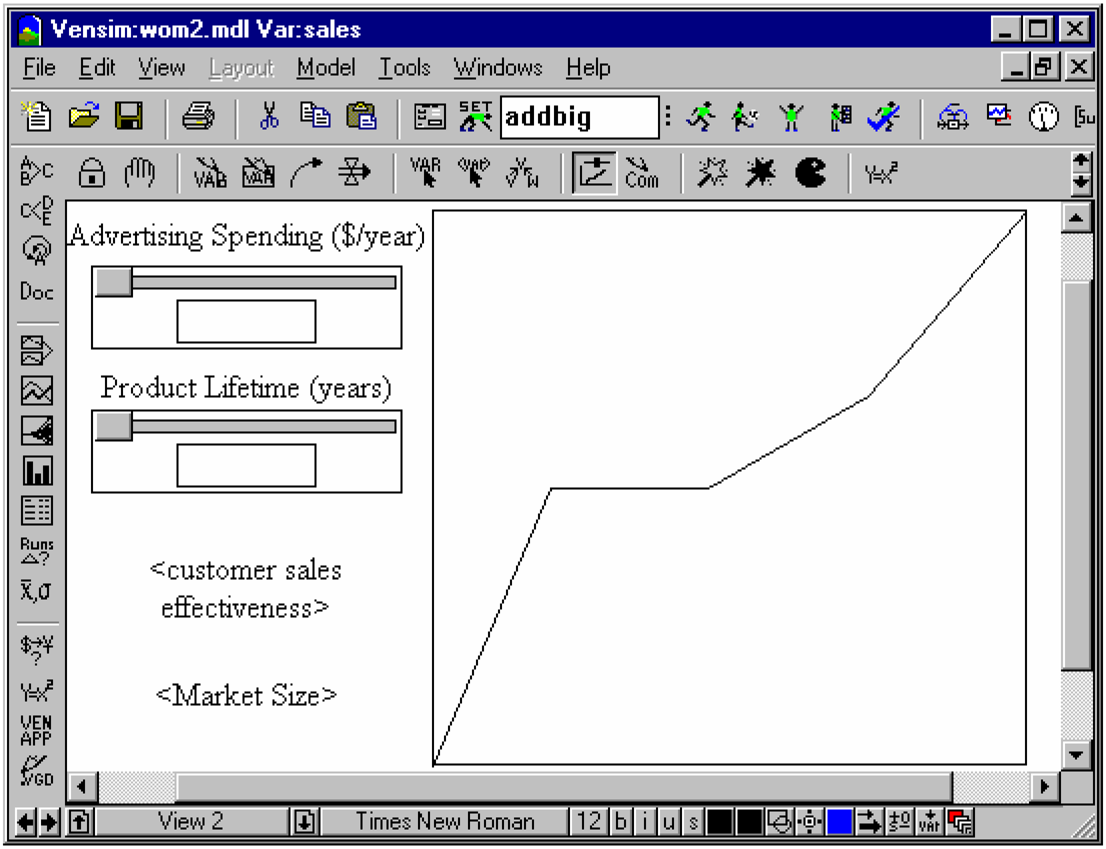
✍ To make the things you have added line up nicely select the **Size/Move** tool  then drag over all the elements on the left hand side to highlight them.

✍ Shift-Click on the topmost comment (Advertising Spending ($/year)) once to unhighlight it, and again to highlight it.

✍ From the **Layout** Menu select the item **Center on LastSel**.

✍ Position the whole selection by dragging the middle of the selection box until it looks good.

You should end up with.

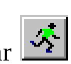


Now is a good time to save your work under a new name. The changes that have been made to *wom1.mdl* are also saved in the model *wom2.mdl* contained in the directory *guide\chap11\Complete* if you want to compare them with your own changes.

Simulating the Model

✍ Click on the **Set Up a Simulation** button in the Toolbar  (or type Ctrl+E). Type in a new run name.

The Graph will fill in, the Sliders will activate, and the two shadow variables will highlight.

✍ Drag a slider and then click on the **Simulate** button in the Toolbar  or type (Ctrl+R).

The model will simulate and the results for the new run will be displayed in the graph. Repeat this making other changes. You can change the run name or keep it the same. Each time you simulate the results will be displayed in the graph on the right.

You can also click on the **SyntheSim** button and see the graph update automatically as you move the sliders. In this case sliders will also appear for *`customer sales effectiveness`* and *`Market Size`*.

Gaming Control

The word of mouth model has one gaming variable *`Advertising Effectiveness`*. We can use the same basic screen layout that we did for the simulation control setup.

✍ Select the **Move/Resize** tool .

✍ Highlight the Output Object on the right, the Comment "Advertising Spending ($/Year)" and the Slider below this.

You can do this by Clicking on one, then Shift-Clicking on the other two.

✍ Select **Edit>Copy** (or Ctrl+C).

NOTE It is easiest to copy Input Output Objects with the Move/Resize tool active. When you activate the Lock tool these objects no longer get selected on Click or Shift-Click, though they can be selected by dragging around them.

Pasting to a New View

✍ Start a new view by clicking on **New** in the Status Bar View button at the bottom.

✍ Use the Edit command Paste (or Ctrl+V).

✍ Click OK on the paste dialog. There isn't really any structure to paste so **Replicate** and **Paste Picture** have the same effect.

✍ Select the **Input Output** tool and then click on the Slider.

The Input Output Control dialog will open.

✍ Click on the **Gaming...** button and click select *`advertising spending`* (the only thing in the list). Click on **OK**.

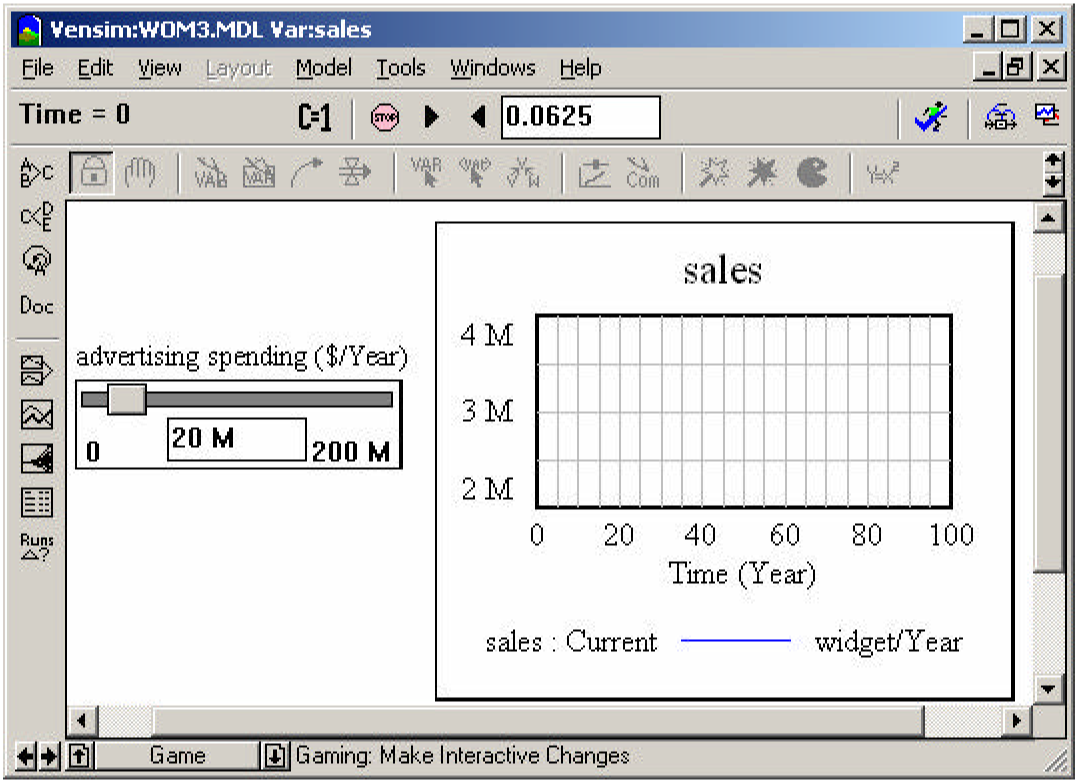
Note that you do not need the change the label, since it is still advertising spending that is being controlled.

Running a Game

✍ Select the Control Panel **Datasets** tab and unload all datasets.

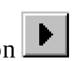
✍ Click on the **Game** button to start a game.

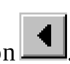
The graph will fill (perhaps with a full time scale but possibly with only a 0-100 scale if you are overwriting the only loaded run). The scale should adjust as you move forward in time. The Slider will become active and the Toolbar will change to reflect the gaming state:



The Gaming Interval appears highlighted in an editing box in the Toolbar. By default the Gaming Interval is set to the TIME STEP which is 0.0625.

- Type in 0.25.

- Make changes using the Slider and click on the **Advance** button  to move forward in time.

The graph will update. You can also move back in time using the **Backup** button .

Now is a good time to save your model. The changes that have been made here are also saved in the model *wom3.mdl* contained in the directory *guide\chap12\Complete* if you want to compare them with your own changes.

Publishing the Model

The model as it now stands has a simple interface for changing constants and running simulations and it also has a simple interface for running games. You can help people who have no knowledge of the model by adding navigation buttons for the Views, and also build in a Gaming Interval that is more appropriate than the default TIME STEP..

Game Interval

To set the default Game Interval to be a more reasonable number simply add the Constant ***`Game Interval`*** to the model, mark it as a Supplementary variable and set its value equal to 0.5 (Years). When a game is started, the Constant 0.5 will be read in as the interval for Game steps.

Commentary and Navigation Links

To help a newcomer to the model, it is appropriate to set up some instructions and guides to help them navigate around.

✍ Create a new View by selecting **New** on the Status Bar **View** button.

Renaming Views

✍ Select menu **View>Rename**. In the dialog that opens type in "Overview" and click on **OK**.

Your view will now be called "Overview." You should rename the other views in the same manner to "Structure," "Simulate," and "Game."

Reordering Views

✍ Select menu **View>Reorder**.

✍ Press the mouse button on "Overview" in the list of Views.

✍ Holding down the mouse button move the mouse up to near the top.

"Overview" should disappear and the shape of the pointer should change to a crosshair.

✍ Move the crosshair up so that it is centered near the top of the first name (Structure) and let go of the mouse button.

"Overview" should appear in the first position. If it does not just repeat the operation letting go of the mouse a little lower. If you move the center of the crosshair outside of the list "Overview" will just drop back to its old position.

✍ Click **OK**.

Adding Commentary

If you are not currently on the "Overview" View, move there by selecting "Overview" from the bottom View button or by using the Page Up and Page Down keys.

✍ Select the **Comment** tool  and click on the top part of the screen.

✍ Type in some commentary you think will be helpful. See below to see what comments we added.

Navigation Links

A Navigation Link is just a comment with the **Navigate** field filled in. When you click on a Navigation link (with the **Lock** tool selected) the View is automatically changed to the View named in the **Navigate** field.

✍ With the **Comment** tool, click on the diagram below your existing comments.

✍ Type in "Model Structure" and select **Box** in the **Shape** field.

✍ Click on the **...** button to the right of **Navigate View**.

A list of views will appear in a new dialog.

✍ Select "Structure" from the list and click on **OK.**

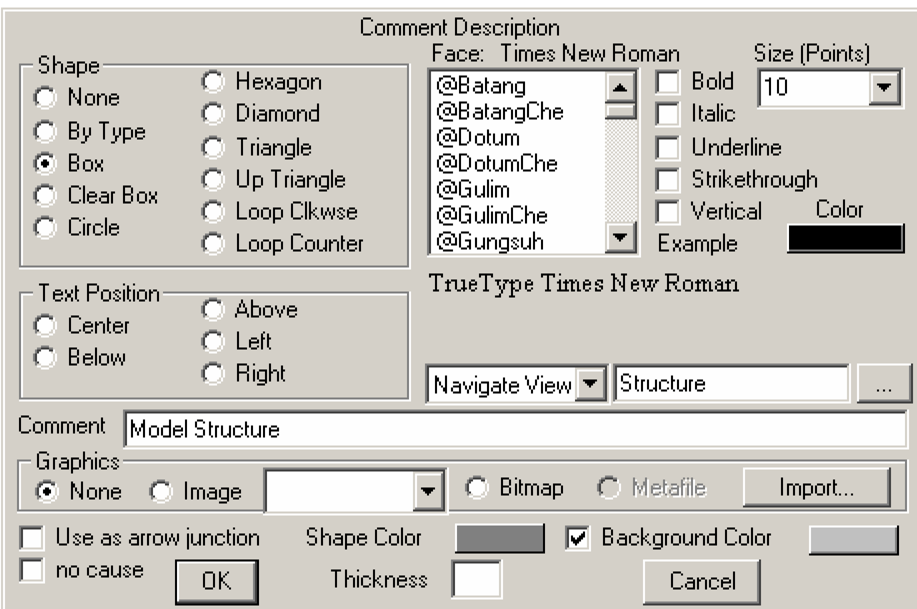
✍ Click on the button to the right of **Shape Color** and click on a dark gray button.

✍ Click on the button to the right of **Background Color** and click on a light gray color.

✍ Fill in the field **Thickness** below the colors with two.

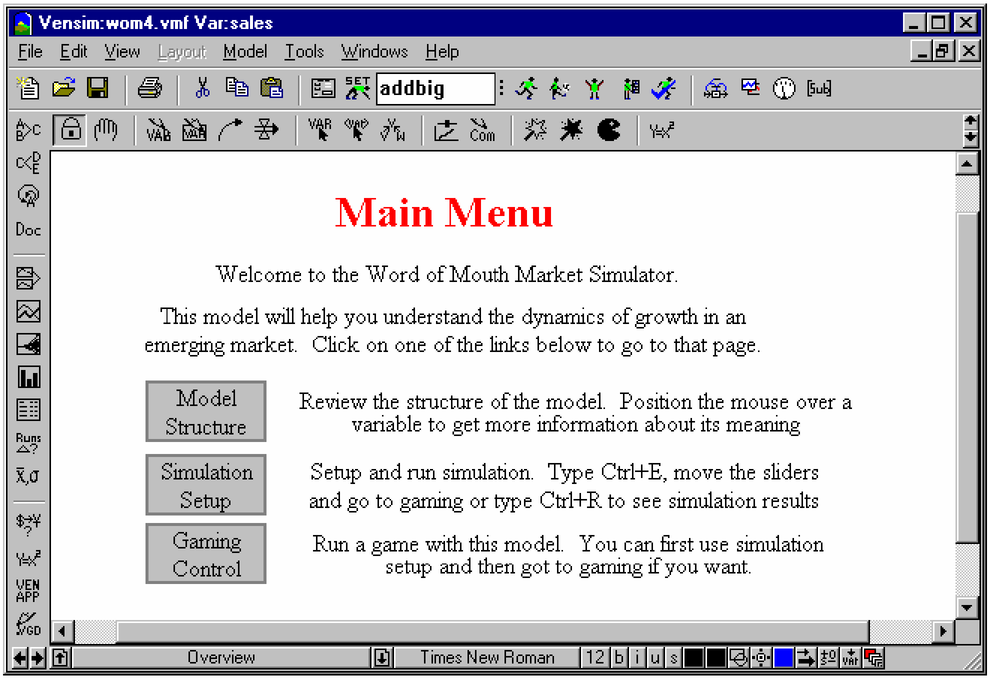
The use of a gray color and dark gray box with **Thickness** set to 2 makes the Comment look a little more like a button.

The comment description dialog should appear as:



✍ Set up Navigation Links to each of the other views giving them the labels "Simulation Setup" and "Gaming Control."

You should have a diagram that looks like this:



✍ Add in Navigation links back to the Overview View from each of the other Views. In the Simulation View you also need to add a Navigation link to the Game View.

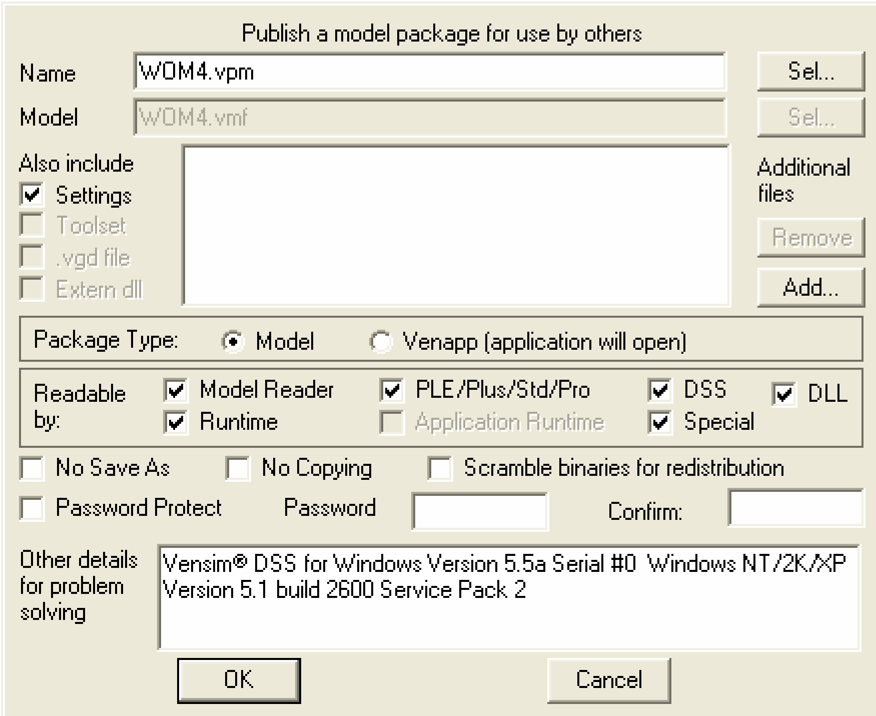
Now is a good time to save your model. The changes that have been made here are also saved in the model *wom4.vmf* contained in the directory *guide\chap12\Complete* if you want to compare them with your own changes. The *.vmf* file extension used instead of the normal *.mdl* extension and this is discussed below.

Test It Out

Select the Lock tool and try the Navigation links. Also repeat the process of setting up and running a simulation. Review the appearance of the "Game" view in simulation setup mode. Review the appearance of the "Simulate" view in Gaming mode. The behavior of your model with the Lock tool selected is essentially the same as its behavior will be in the Vensim Model Reader.

Publish a Package

The Vensim model reader is a read-only program that can't read text format (*.mdl)* models. Because of this it is necessary to save the models you develop in a special binary form before they can be used in the Vensim model reader. To do this just choose **File>Publish** from the File Menu:



If you have runs active the above dialog will also list these. For this model we do not need them – click on the name and click on remove for each run listed.

You can also save the model as a binary format (*.vmf)* model using **File>Save As...** from the File menu. In the save as dialog choose **Binary Format Models** as the file type and enter in a name. You can also just type in the name with extension .vmf (as in *wom4.vmf*) and Vensim will determine the type of file you want to save as from the extension you have typed.

Publishing to a package gives you control over how the model can be opened and used, see Chapter 19 for more details. Saving as a binary format model allows the model to be opened and modified just as it could be for a text format model. The binary format files are bigger and less suited to archiving since they can only be opened with Vensim. Binary files do, however, have the advantage of making it easier to work with **Based On** in the Simulation Control dialog as discussed in Chapter 8 of the Reference Manual. Binary format models also make it easier to work with compiled simulations as discussed in the *Vensim DSS Reference Supplement*.

You have now created a model that you can send to anyone and they can use by downloading the free Vensim Model Reader.

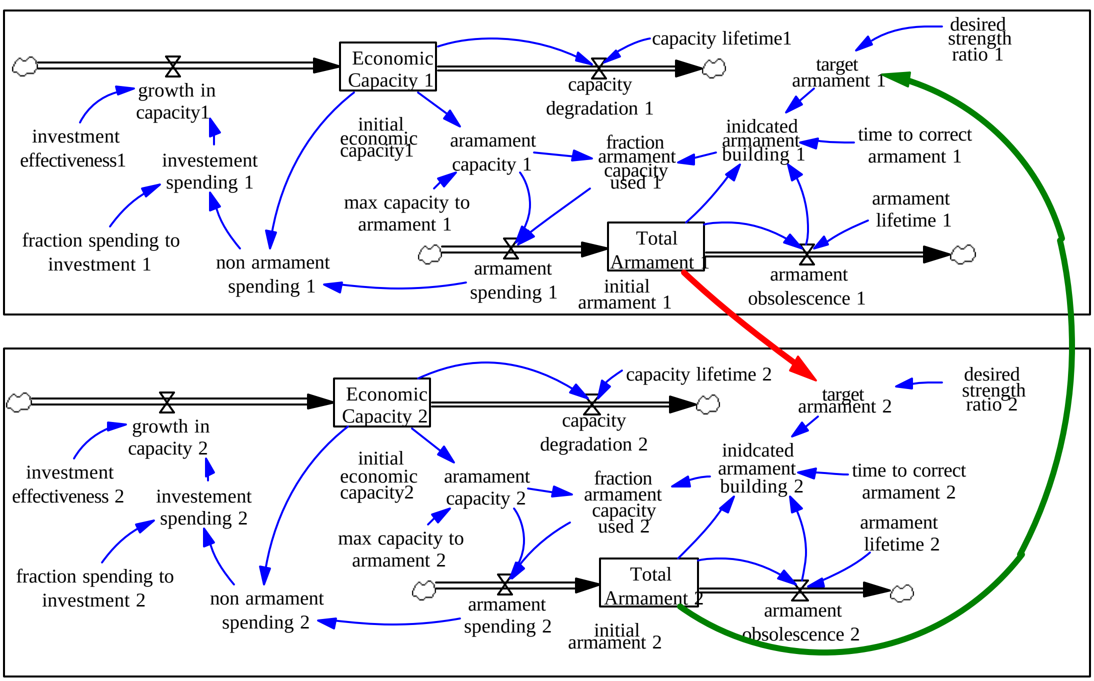
13 SyntheSim

Simulating models is traditionally done as a batch process. Set up some assumptions, make some a simulation, and then look at some results. Twenty five years ago, this was a necessity. Computers were large, expensive, and very slow by today's standards. Access was through a time shared terminal or perhaps even punch cards and jobs had to be submitted for execution and the output picked up from the main computer office or, if you were lucky, a close at hand printer or even your own teletype terminal.

Technology has changed a great deal, but the batch processing of simulation has remained a legacy till now. While it has been possible to view the results of simulations as they progress, this has not really altered the change, compute, review approach to simulation. With Vensim 5 a better approach to interacting with models has finally arrived, and it is practical for a surprisingly large fraction of the models in use. We call this approach SyntheSim.

Simple Arms Race Model

Construct the following model or open *guide\chap13\complete\arms2.mdl*. This is a simple arms race model that includes economic growth.



New Diagramming Flourishes

The above diagram is straightforward except for two things. One is the large rectangles used to enclose each side in the arms race. The second is the long noncircular arrow connecting the two sides. The following works through the process of constructing these. If you want you can start from the model *guide\chap13\complete\arms1.mdl*.

Long Connections

The arrow from `Total Armament 2` to `target armament 1` is not a regular circular arrow. It was created by using a number of smaller arrows linked through comments that allow arrows (called junctions). To create this you need to first create the junctions, then add the arrows, then get rid of the extra arrowheads and size the junctions to 0.

- ✍ Select the **Comment** tool.
- ✍ Click on an empty part of the sketch to the right or `armament obsolescence 1`.
- ✍ Select blue as the background color.
- ✍ Click on the **Use as arrow junction** checkbox.
- ✍ Click on OK. We will call this a junction box.
- ✍ Resize the junction box to be quite small (around 10x10).
- ✍ Repeat the above operations to the right of `armament obsolescence 2`, or copy and paste the junction box.
- ✍ Select the **Arrow** tool.
- ✍ Draw an arrow from `Total Armament 2` to the lower junction box.
- ✍ Draw an arrow from the lower junction box to the upper.
- ✍ Draw an arrow from the upper junction box to `Total Armament 1.`
- ✍ Right click on the two lower arrows and uncheck the **Arrowhead** box. Set the arrow width to the second thickest and the color to dark green.
- ✍ Change the arrow width and color on the upper arrow.
- ✍ Adjust the positions of the junction boxes and arrows to look good.
- ✍ Resize the junction boxes to have 0 size. Just push the handle up and to the left till nothing is showing.

Hint: If you want to move the junction boxes after you have sized them to 0 you can select the Move/Shape tool and select them by rubber banding around them then move the selection rectangle.

Containing Boxes

- Select the **Comment** tool and click in a blank area of the upper left hand corner of the sketch.
- In the dialog that appears select **Shape Box** and **Thickness** 2 (this is right at the bottom) then click on **OK**.
- Resize the box so that it contains the upper portion of the model.

NOTE After you have created the box the sizing handle will be visible for a few seconds. If the sizing handle disappears just click on the **Move/Size** tool to make it visible again.

- Click on the **Lock** tool.
- Click on the new box. It will highlight.
- Click on the **Push the highlighted words to the background** button on the status bar. The highlighting will disappear.
- Repeat the above process for the lower half of the model.

You can also give containing boxes a background color if you wish (pick something light). They can be helpful to separate things visually, though they don't have any impact on causal relationships.

Min, Max and Increment

In the Equation editor you can enter a minimum value, a maximum value and an increment for each equation. In Vensim PLE and PLE Plus these fields are labeled explicitly:

in the other configurations they are labeled **Range**

For dynamic model variables the range is used to generate warning messages when variables get too small or too big. For example having a range of [10,500] would give a warning whenever the variable went below 10 or above 500.

For Constants the range is used to specify how big or small a constant can get. In this case the increment indicates how much to change a Constant by as a slider is moved. For example the range [10,500,5] would allow the Constant to range from 10 to 500 changing 5 at a time. The range [0,1,1] could be used for an On/Off switch.

If you do not specify any range for a constant Vensim will make one up based on the model value. If you do not specify any increment Vensim will just divide the range up into about 80 even intervals.

Model Equations

The model equations are quite straightforward. The ranges are shown in square brackets following the units. With the exception of `target armament`, the equations for ...2 are exactly the same as those for ...1 with 2s instead of 1s at the end and so these equations are not repeated here.

```
armament capacity 1= Economic Capacity 1 * max capacity to armament
   1
   Units: M$/Year

armament lifetime 1=20
Units: Year [2,60,1]

armament obsolescence 1=Total Armament 1/armament lifetime 1
Units: M$/Year

armament spending 1=armament capacity 1 * fraction armament capacity
   used 1
Units: M$/Year

capacity degradation 1=    Economic Capacity 1/capacity lifetime1
Units: M$/Year/Year

capacity lifetime 2=30
Units: Year [3,80,1]

desired strength ratio 1=1
Units: Dmnl [0,5,0.1]

Economic Capacity 1= INTEG (growth in capacity1 - capacity
   degradation 1,
                     initial economic capacity1)
Units: M$/Year

FINAL TIME  = 100
Units: Year

fraction armament capacity used 1= WITH LOOKUP (
   ZIDZ(indicated armament building 1,aramament capacity 1),
([(0,0)-
   (10,1),(0,0),(1,1),(10,1)],(0,0),(0.4,0.4),(2,0.8),(3,0.9),(5,1),(10,1)
   ))
Units: Dmnl

fraction spending to investment 1=0.3
Units: Fraction [0,0.6,0.05]

growth in capacity 2=investment spending 2 * investment
   effectiveness 2
Units: M$/Year/Year

indicated armament building 1=MAX(0,armament obsolescence 1 +
               (target armament 1 - Total Armament 1)/time to
   correct armament 1)
```

```
Units: M$/Year

initial armament 1=50
Units: M$ [0,200,5]

initial economic capacity1=100
Units: M$/Year [10,500,5]

INITIAL TIME  = 0
Units: Year

investment spending 1=     non armament spending 1 *
         fraction spending to investment 1
Units: M$/Year

investment effectiveness 2=0.15
Units: 1/Year [0.01,0.4,0.01]

max capacity to armament 1=0.4
Units: Fraction [0,1,0.05]

non armament spending 1=Economic Capacity 1 - armament spending 1
Units: M$/Year

SAVEPER  = TIME STEP
Units: Year [0,?]

target armament 1=Total Armament 2 * desired strength ratio 1
Units: M$

target armament 2=Total Armament 1 * desired strength ratio 2
Units: M$

TIME STEP  = 0.125
Units: Year [0,?]

time to correct armament 1=5
Units: Year [1,20]

Total Armament 1= INTEG (armament spending 1-armament obsolescence
   1,
         initial armament 1)
Units: M$
```

Structure, Behavior and Speed

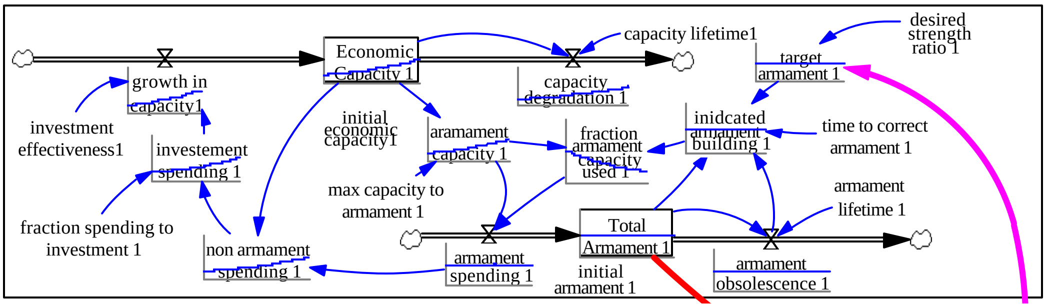
SyntheSim is so called because it allows you to use Simulation to Synthesize the structure and behavior of a model. Events, behavior and structure are often presented as a hierarchy of ways of looking at the world with events arising from underlying behavior patterns and behavior arising because of structure. Simulation models represent the structure which, through simulation, generates the behavior. The connection between structure and behavior is strong, but it can be difficult to gain an understanding of how structure causes behavior. By superimposing behavior on structure and instantly updating behavior as you make changes, SyntheSim provides you with a new and very powerful tool for understanding your model.

Showing Behavior

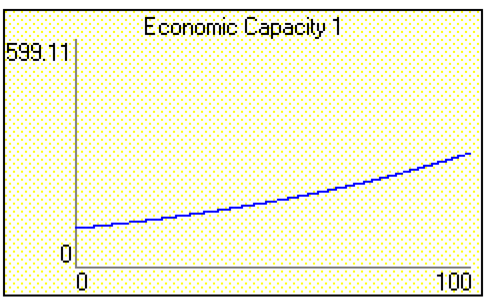
✍ Click on the **Simulate** button to simulate the model.

✍ Select the menu item **View>Show Behavior** or press the shortcut B key.

Instead of just a diagram you will see:



The behavior over time for each of the dynamic variables is displayed directly on the sketch. You can work with the sketch, move words and arrows and change equations with behavior showing. If you position the mouse over a variable and wait, a larger version of the small graph will appear just as a tool tip would.



The graph is the same as the thumbnail graph that appears on the diagram with labels added. You can also click on a variable to select it into the workbench.

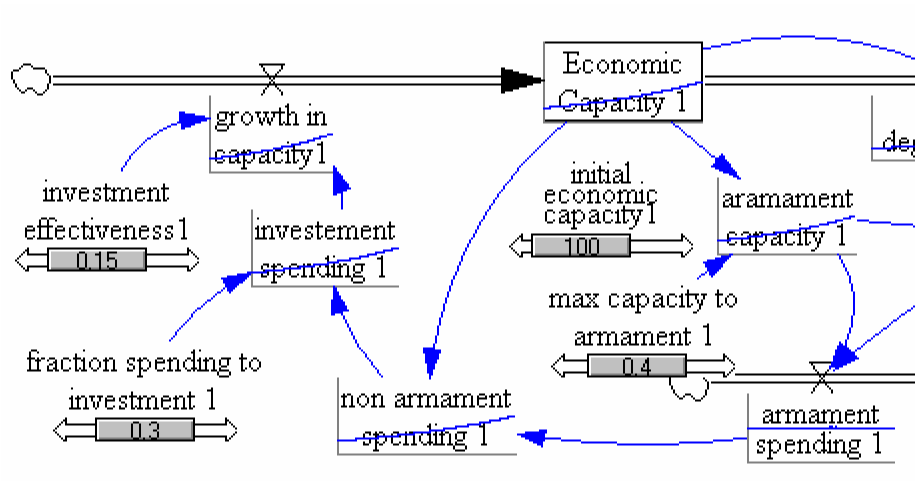
If you make a number of simulations each one will show with a different color on all of the thumbnail graphs. By loading and unloading toolsets from the control panel you can control what is displayed. There is an example of doing this later in the chapter.

Making Simulations

✍ Click on the **SyntheSim** button .

✍ Answer yes to the question of whether or not you want to overwrite the existing run.

In addition to the graphs on each variable you will see that each constant has a slider:



The sliders are created automatically for all model constants just below the variable name (except in the case of a variable with a shape and the name below that shape in which case the slider will appear just above the name).

In SyntheSim mode every time you make a change to a model Constant or Lookup the model will be simulated and the results presented. There are several ways to change things.

Moving Sliders

You can change constants by dragging the button on the sliders.

✍ Position the mouse over the slider just below `desired strength ratio 1`.

✍ Press the mouse button.

When you do this the button will depress and the pointer may move slightly to better reflect the current slider position.

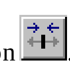
✍ Slowly move the slider to the right till it reaches about 2.

As you are doing this watch what happens to the levels in the model. See how armaments move from something that is flat, to something that is growing more and more quickly to a more S shaped behavior. The graphs will briefly flash red to indicate that their scales are increasing and blue to indicate that they are decreasing.

If you think you missed something just backup and try again.

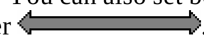
✍ Let go of the mouse button.

The slider you have been moving will have grey rails indicating that it is the current slider.

✍ Press the home key or click on the **Reset Current Slider** button .

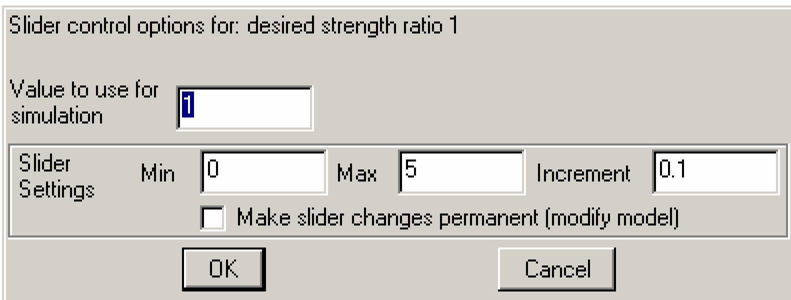
The value for this slider will return to the model value and the model will be simulated again with the original conditions. You should see the armaments graphs flatten

Setting Slider Values

The sliders move back and forth along the range set in the equations for the constant or, if no range was set, along a range computed by Vensim given the Constant value. You can also set both the value of the Constant and the slider range by clicking on the rails of the slider .

✍ Click on the rails of the slider (not on the slider itself) below ***`desired strength ratio 1`***.

A dialog box will open. If you miss and click on the button it will depress - release the mouse button and try again.



✍ Change the **Value to use for simulation** to 2, the **Min** to 1, the **Max** to 2 and the **Increment** to 0.01. Then click on **OK**.

The model will simulate with ***`desired strength ratio 1`*** set to 2 and display the results. The slider will be at the rightmost side of its range.

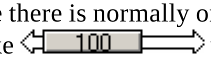
✍ Drag the slider back and forth and review the results.

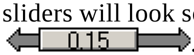
You now have much more control over the slider within a narrower range of values.

✍ Press the Home key or click on the **Reset Current** slider button.

The value of ***`desired strength ratio 1`*** will be reset to 1, but the slider range will remain at what you changed it to. The range will not be changed until you either switch Views or stop SyntheSim. If you check the **Make Slider Changes Permanent (modify model)** checkbox when setting the range the changes will be made permanently to the model.

Using the Arrow Keys

In addition to using the mouse to move the slider you can also use the arrow keys to change the values of constants. When you are in SyntheSim mode there is normally one slider that looks different from the others. Most sliders will look something like 

while one of them will have the rails filled in like 

. The one with the rails filled in is called the active slider. Whenever you click on a slider it becomes active. You can also use the Tab (and Shift+Tab) keys to move between the sliders, though the order is that in which you created the Constants and is not always very clear visually.

✍ The active slider should be the one underneath ***desired strength ratio 1***. If it is not press the Tab key until it is.

✍ Press the right arrow key.

The value of the slider should increase from 1 to 1.01, and you will see a small change in the graphs. Click again and the value will change to 1.02 and so on.

✍ Press the left arrow key.

The value of the slider will decrease by .01. The arrow keys are very useful if you want to move very slowly through a Constant's range. They can also be helpful if you are working with bigger models and you want to do only one simulation each time a value is changed (dragging a slider will often end up making two or three simulations).

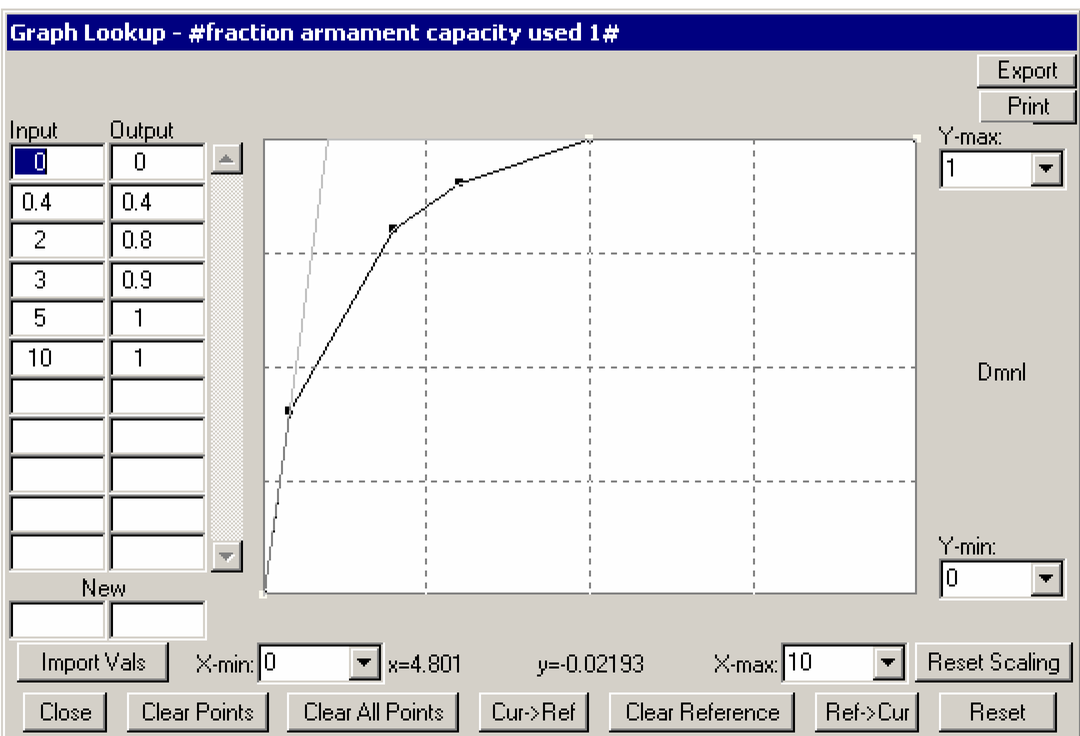
Changing Lookups

Changing Lookups is done by opening the Lookup editor and modifying the Lookup there. As you make modifications the model will be repeatedly simulated and the results displayed.

✍ Drag the slider underneath ***desired strength ratio 1*** till it has about 2 as its value.

✍ Click on fraction armament capacity used 1.

The Lookup editor will open.



The Lookup editor in SyntheSim mode has a **Reset** button instead of close button. Each time you make a change to the values in the lookup the model is simulated so there is no way to cancel the change. The **Reset** button will put the values back to the original model values.

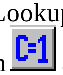
This Lookup has a reference line (appearing in grey) along which the fraction of resources indicated are used. The graph itself should always lie at or below this reference line.

✍ Position the mouse over the third point on the graph and then drag this point up toward the reference line.

Watch the behavior of the variables as the point moves. You can move the Lookup editor by dragging its title bar if you want to expose other variables.

✍ Click on the **Close** button to close the Equation Editor.

Changing from the Toolbar (Not PLE or PLE Plus)

In addition to making changes to Constants and Lookups on the Screen you can also make changes by clicking on the **Change Model Constants** button  and the **Change Lookups** button . These bring up dialogs with lists of Constants and Lookups respectively.

✍ Click on the **Change Model Constants** button.

✍ Click on *`desired strength ratio 1`* in the list and then click on **Modify** or press the Enter key. The current value will appear highlighted in an editing box.

✍ Type in 1.8 and press the Enter key. The model will simulate. You will also see the slider below *`desired strength ratio 1`* change in value.

✍ Click on **Close**.

✍ Click on the **Change Lookups** button.

✍ Click on *`#attraction armament capacity used 1#`* (the # signs are added by Vensim) and click on **Modify**.

✍ Make changes to the Lookup as before then click on **Close**.

Doing Simulation Experiments

So far we have been looking at only a single graph for each variable. . You can keep simulations that are interesting and also display results from multiple simulations on the thumbnail graphs.

✍ Click on the **Reset all Constants** button or press Ctrl+Home.

This resets not only constant changes but also any lookup changes you might have made.

✍ Click on the **Save this Run to...** button .

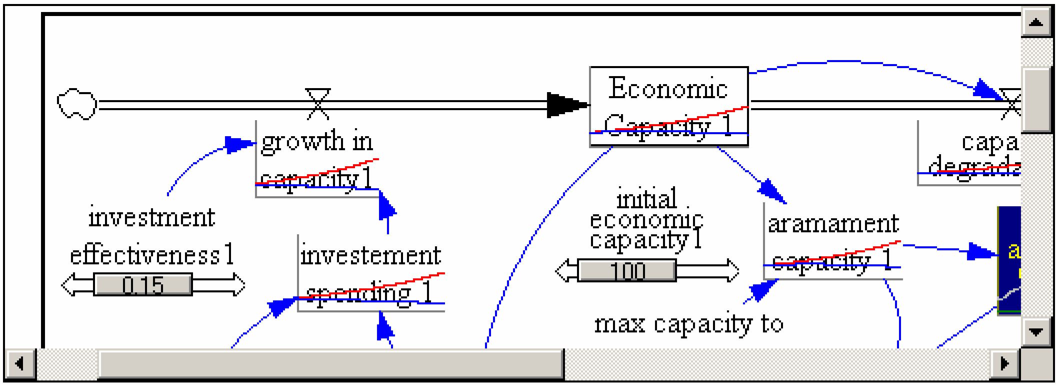
✍ Type in the name **baserun**.

✍ Click on **Save**.

You will notice that the screen blinks, but nothing seems to change. This is because the base run is the same as the current experiment.

✍ Move the slider on `desired strength ratio 1` up to 2.

There should be two graphs on each variable.

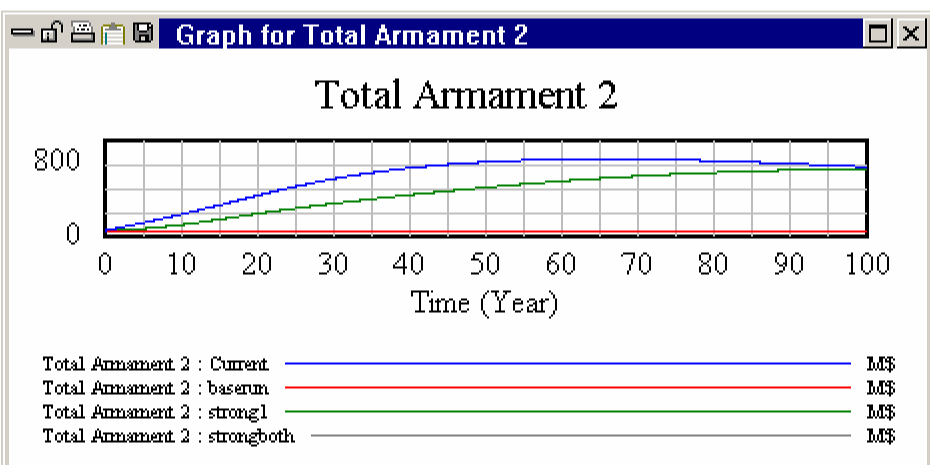


✍ Click on the **Save this Run to...** button. Type the name *strong1* and click on **Save**.

✍ Move the slider on `desired strength ratio 2` up to 2. (This is in the lower part of the model. It is the first time we have changed this value.)

✍ Click on the **Save this Run to...** button. Type the name *strongboth* and click on **Save**.

✍ Click on `Total Armament 2` and click on the **Graph** tool. You will see the graph:



Notice how the *strong1* and *strong2* run end up in almost the same place.

✍ Click on `Economic Capacity 2` and click on the **Graph** tool.

When country two is more aggressive about building up arms the consequence is not more arms so much as less economic activity. If you play with the desired strength numbers you will see that after a point making them larger yields a lower arms level at the end of the run.

Loading and Unloading Runs

You can load and unload runs and change other settings from the control panel. If you change the Time Axis settings this will change the appearance of graphs, though it will not have an effect on the thumbnail graphs (these display the full simulation range).

✍ Click on the **Control Panel** button and click on the **Datasets** tab.

✍ Click on *strong1* and click on the << button.

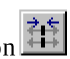
✍ Click on *baserun* and click on the << button.

As you make these changes you will see the thumbnail graphs redraw to reflect the list of loaded runs. You can have up to 16 runs loaded, thought it is rarely useful to have more than a few.

Breaking Feedback Loops

One very effective way of understanding model behavior is to break feedback loops and see what changes. This is very easy to do with SyntheSim.

Stopping the Race

✍ Click on the **Reset All Constants** button  or press the Ctrl+Home keys.

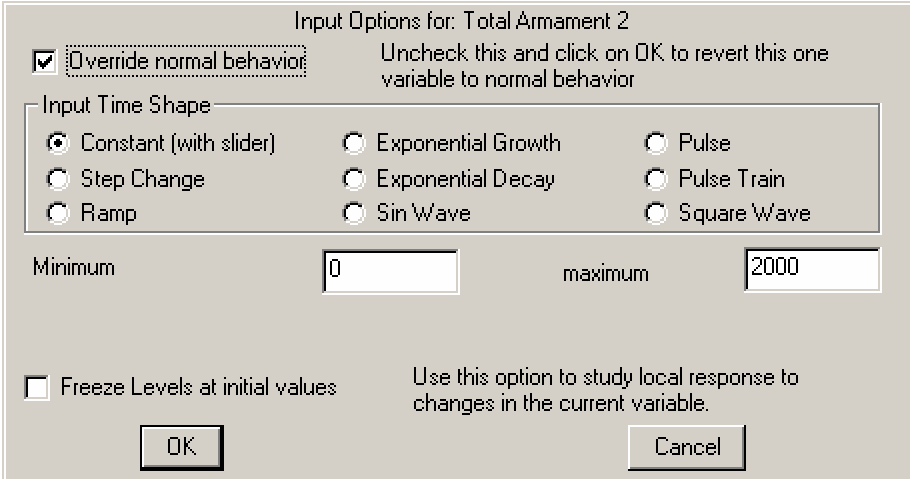
This will reset all sliders to the original positions (and reset all Lookups). Note that if you had only moved one slider and that slider is still active this is the same as resetting the current slider.

✍ Drag the slider under `desired strength ratio 1` to 2.

The two graph lines should now be on top of one another. If this is not the case make sure that the two runs you have loaded are *strong1* and *current*.

✍ Click on `Total Armament 2` with the right mouse button (or hold down the Ctrl key and click).

The variable override dialog will appear.



When a variable is overridden, rather than using the value computed for it an externally input value will be used. This value can be a constant, or one of a number of alternative input patterns.

✍ In the **maximum** field enter 1000.

✍ Click on **OK**.

The model will be simulated with *`Total Armament 2`* constant at 500. The variable itself will be displayed in a different color to indicate that it is not being computed in the normal manner and there will be an input slider just below it.

✍ Drag the slider under *`Total Armament 2`* down to have a value of 50 - this is the original initial value for this variable.

✍ Click on *`Total Armament 1`* then click on the **Graph** tool.

`Total Armament 1` starts at 50, then rises fairly quickly to 100 and stays there. There is no ongoing increase.

✍ Click on **Stop Override** in the toolbar.

The graphs should return to showing only a single line.

Overriding Behavior

The economy in this model provides an important controlling influence on the ability to produce armaments. If armament spending had been formulated as the indicated spending times an effect from capacity, we could simply override that effect and make it always one. Unfortunately, such a formulation is difficult to create because it is important that actual armament spending never exceed economic capacity. For this model, the appropriate way to remove the controlling influence of the economy is not to make something constant, but rather to impart behavior to economic activity that is not influenced by the level of spending on arms.

✍ Click on `Economic Capacity 1` with the right mouse button (or hold down the Ctrl key and click).

✍ In the dialog click on the **Exponential Growth** radio button.

✍ For **Starting From** fill in 100.

✍ For Growth Rate (%/unit time) fill in 10.

Look at the behavior of the different variables. *`Total Armament 1`* grows more, but is still S shaped. In fact, `Economic Capacity 2` is falling off quite strongly relative to the *strong1* run and this is hampering the ability of 2 to product armaments. Let make the same change to `Economic Capacity 2` that we made to `Economic Capacity 1`.

✍ Click on `Economic Capacity 2` with the right mouse button (or hold down the Ctrl key and click).

✍ In the dialog click on the **Exponential Growth** radio button.

✍ For **Starting From** fill in 100.

✍ For Growth Rate (%/unit time) fill in 10.

Now look at the behavior. We get very rapid growth in the level of armaments for both sides. We can be confident that economic capacity is not a constraining factor because in both countries the fraction of capacity devoted to armament production is small and diminishing.

✍ Raise desired strength ratio 2 to also be 2.

Economic growth is constraining once again - even at 10%/year.

Building a Control Panel

In addition to working with thumbnail sketches SyntheSim will work with graphs and tables imbedded as Input Output Objects.

✍ Click on the **Stop Simulation** button.

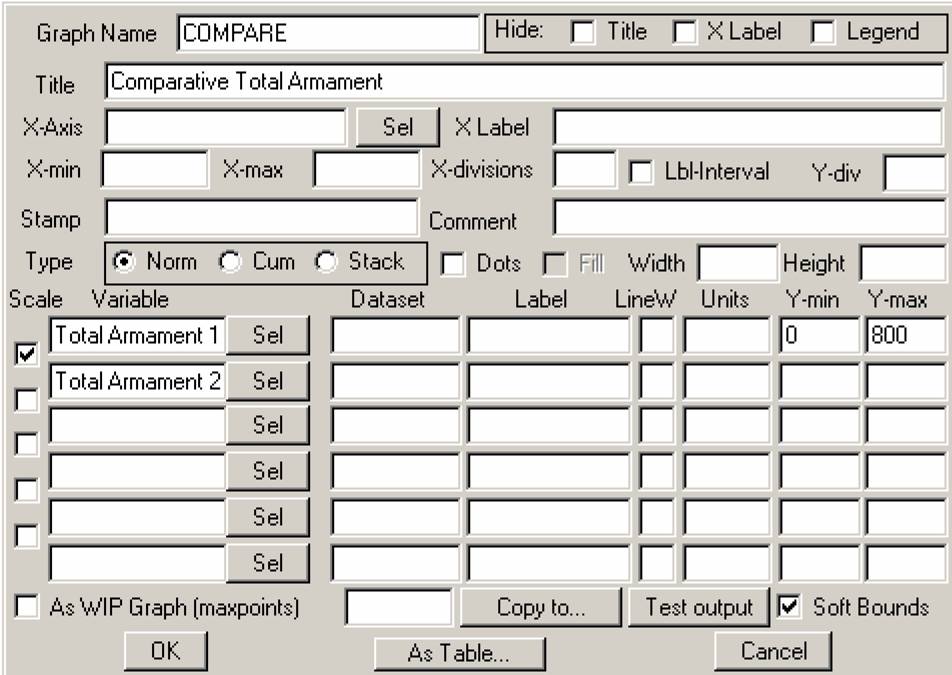
✍ Click on the **Control Panel** button and click on the **Graphs** tab.

✍ Click on the **New** button.

✍ In the Custom Graph editor add the title "Comparative Total Armament" and add the variables `Total Armament 1` and `Total Armament 2`. For **Graph Name** enter COMPARE.

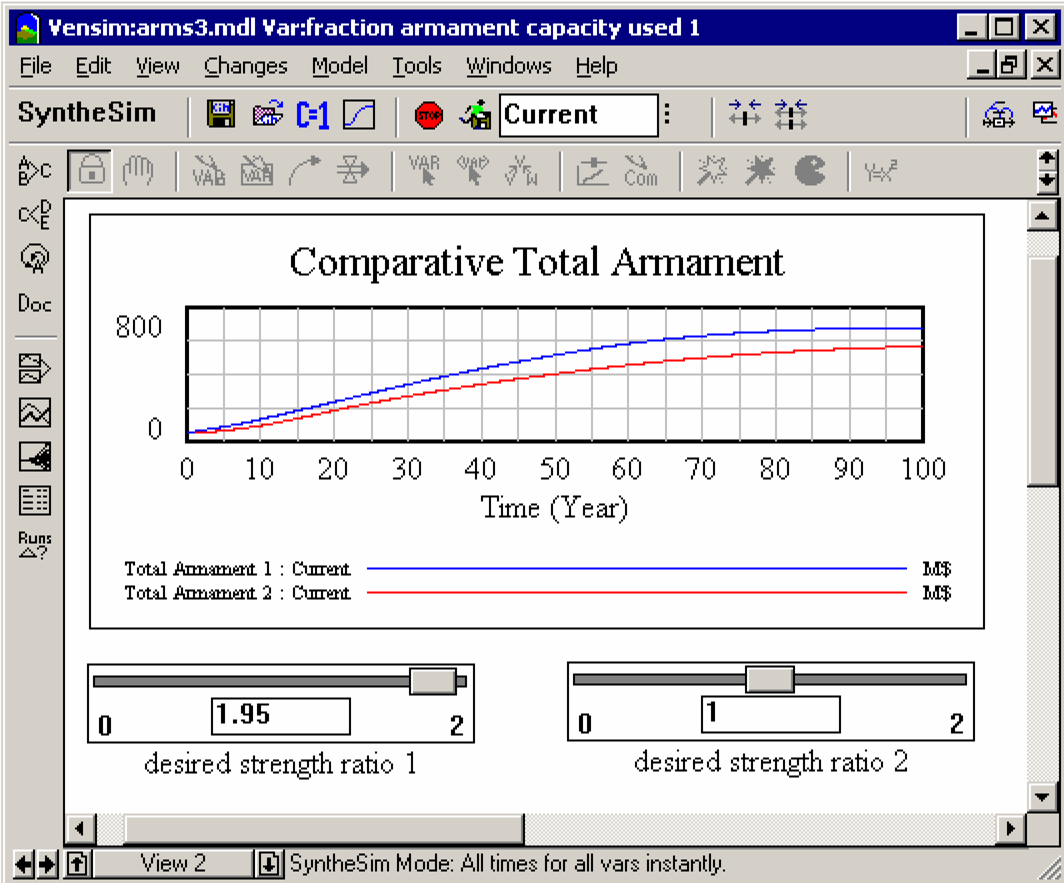
✍ Check the **Scale** checkbox to the left of the variable names. Add 0 under **Y-Min** and 800 under **Y-Max** at the right in the top row.

✍ Check the **Soft Bounds** checkbox at the bottom. The dialog should appear as:



- Click on **OK**.
- Select the Menu Item **View>New**.
- Click on the **Input Output** tool.
- In the dialog click on the **Output Custom Graph** radio button. Select COMPARE from the dropdown list at the bottom and click on **OK.**
- Size the graph to be quite large.
- Click below the graph at the left.
- In the Input/Output Object dialog click on the **Constant...** button and select ***`desired strength ratio 1`***.
- Set the range from 0 to 2 with increment .01 and click on **OK.**
- Click below the graph at the right.
- In the Input/Output Object dialog click on the **Constant...** button and select ***`desired strength ratio 2`***.
- Set the range from 0 to 2 with increment .01 and click on **OK.**

You can make the sliders slightly bigger and position them to look good. Go to the control panel and clear any runs, then start SyntheSim mode and move the sliders. You should see something like:

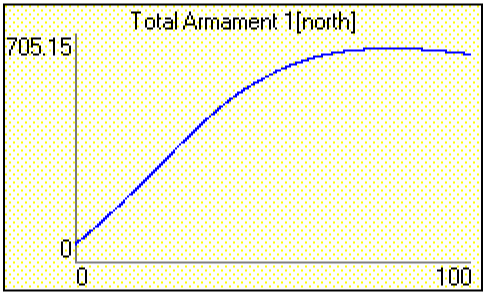


You can go back to the previous view and continue to make changes to model constants. When you return to this view you will see the results of those changes shown in the graph.

Subscripts and SyntheSim (DSS and Professional only)

A few words are in order on the interaction of subscripts and SyntheSim. While you are in SyntheSim mode only one graph appears for each variable and only one constant appears for each slider. Vensim determines which subscript to display information for by referring to the Subscript control dialog. The first selected subscript combination is displayed (if nothing is selected the first subscript combination is displayed).

You can determine exactly which variable is being displayed by hovering over it till the ToolTip shows:

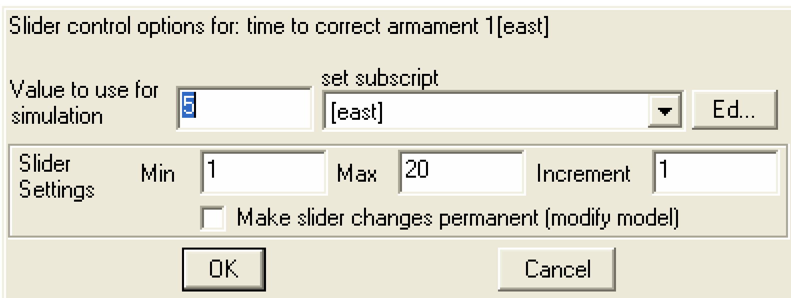


Similarly you can determine which Constant you are changing by hovering over it till the ToolTip shows:

[north] = 20 Year

NOTE You need to hover over the name, not the slider to get the ToolTip to show.

You can also reset one slider to refer to a different subscript by clicking on the slider handles:



Just set a different subscript. The new subscript will stick to the slider till you change subscript selection from the Subscript control or change views. The Ed… button allows you to change values on the full set of subscripts for the constant – but does not change the subscript the slider is active for.

14 Reality Check®

You build and use models in order to address problems. As these models are built, there are various checks that must be done against reality. These checks may be explicit and take the form of tests of model behavior or subsector behavior under different assumptions, or they may be implicit mental simulations and analysis based on your own understanding of models and the modeling process. In either case these checks are very important in ensuring that the models you develop can adequately address the problems they are being applied to.

Reality Check gives you a straightforward way to make statements you think must be true about a model for it to be useful, and the machinery to automatically test a model for conformance with those statements. Reality Check is a new technology that adds significantly to your ability to validate and defend the models you build. It can also focus discussion away from specific assumptions made in models onto more solidly held beliefs about the nature of reality.

This chapter:

? Introduces the concept of a Reality Check.

? Shows you how to define Constraints and Test Inputs.

? Shows you how to test a model for conformance to Reality Check equations.

? Provides a simple example of model building using Reality Check.

Models and Reality

Models are representations of reality, or our perceptions of reality. In order to validate the usefulness of a model, it is important to determine whether things that are observed in reality also hold true in the model. This validation can be done using formal or informal methods to compare measurements and model behavior. Comparison can be done by looking at time series data, seeing if conditions correspond to qualitative descriptions, testing sensitivity of assumptions in a model, and deriving reasonable explanations for model generated behavior and behavior patterns.

Another important component in model validation is the detailed consideration of assumptions about structure. Agents should not require information that is not available to them to make decisions. There needs to be strict enforcement of causal connectedness. Material needs to be conserved.

Between the details of structure and the overwhelming richness of behavior, there is a great deal that can be said about a model that is rarely acted upon. If you were to complete the sentence "For a model or submodel to be reasonable when I ______________ it should __________" you would see that there are many things that you could do to a model to find problems and build your confidence in it.

In most cases, some of the things required for a model to be reasonable are so important that they get tested. In many cases, the things are said not about a complete model but about a small component of the model, or even an equation. In such cases you, as a model builder, can draw on your experiences and the work of others relating to behavior of generic structures and specific formulations.

Ultimately however, most of the things that need to be true for a model to be reasonable are never tested. Using traditional modeling techniques, the testing process requires cutting out sectors, driving them with different inputs, changing basic structure in selected locations, making lots of simulations, and reviewing the output. Even when this gets done, it is often done on a version of the model that is later revised, and the effect of the revisions not tested.

Reality Check equations provide you with a language for specifying what is required for a model to be reasonable, and the machinery to go in and automatically test for conformance to those requirements. The specifications you make are not tied to a version of a model. They are separate from the normal model equations and do not interfere with the normal function of the model. Pressing a button lets you see whether or not the model is in violation of the constraints that reality imposes.

Domains of Expertise

Although Reality Check equations in Vensim have been implemented as an extension of the Vensim modeling language, the skills and experience required to write Reality Check equations is different from that required to write models. Reality Check equations are assertions about the nature of behavior in reality. They do not require the creation of structure capable of generating particular behavior. Reality Check equations create behavioral conditions and then check to see if the structure of the model causes the appropriate behavioral response.

Because Reality Check equations are formulated in a behavior-behavior world, the people best at formulating them are the people with the most knowledge of behavior—for the most part experts in the domain of study. Because of this, Reality Check equations can do much more than find bugs in models. Reality Check equations allow the consumers of models to have confidence in the quality of the results they are getting.

Defining Reality Check Equations

You enter Reality Check equations in the same way you enter other equations in Vensim. You can use the Sketch Tool to define inputs to Reality Check equations, or simply write the equations directly in the Equation Editor or Text Editor. Structurally, Reality Check equations are not inputs to any model equations, but use model variables in their definitions. When Reality Check equations are exercised, they can change the value of model variables as well as the equations used to compute those variables. Again, we emphasize that Reality Check equations are not statements about causal structure but simply statements about behavior—"If this happens, then that must happen".

The appropriate naming conventions for Reality Check equations are different from those for model variables. Model variables should be named as nouns or noun phrases which more and less have obvious meaning—*Workforce*, *productivity*, *capacity*, *determination*, *propensity to save* and so on. Reality Check equations, on the other hand, should be brief phrases that describe the nature of the check—*no workers no production*, *rain means flooding* and so on. The best guide in naming Reality Check equations is to think of them as true or false, and name the Reality Check the statement that would be made when it took on a true value.

There are two types of equations that can be defined in Vensim to make use of the Reality Check functionality—Constraints and Test Inputs. Constraints make statements about the consequences that should result from a given set of conditions. They are called Constraints because they specify the way in which the Test Inputs should constrain behavior. The violation of a Constraint indicates a problem with the model. Test Inputs are a way of specifying the conditions or circumstances under which a Constraint is binding. Since Test Inputs can be used in Constraints, they are described first.

Test Inputs

Test Inputs allow you to define alternative conditions by changing equations for a variable in the model. The basic format for a Test Input is:

name :TEST INPUT: *variable* = *expr*

Where *name* is the name of a Test Input. What appears to the right of `:TEST INPUT:` is exactly the same format equation you would normally use to define an auxiliary variable, and can involve only model variables. The equation you write is also restricted in that you cannot use dynamic functions (such as SMOOTH), defining functions (such as ACTIVE INITIAL) or user defined Macros. If you need this functionality, you can create extra model variables for use in your Test Inputs.

You can only use Test Inputs in the conditional portion of a Constraint equation. The major reason for defining Test Inputs is to give a name to the experiment being conducted. This can make reading the Constraint much easier. If you do not define Test Inputs, you can define Constraints directly using the *variable = expr* portion of the Test Input equation. The same restricted equation format applies if you do this.

Dynamic Test Inputs

In addition to an alternate assignment expression for a Test Input, it is often desirable to force a change in a variable after a period of time in a simulation. For example, you might want to force production to ramp down to 0 between time 10 and 12, but before time 10 just let production be what it would have been. This type of a change is useful both for writing complete reality Checks and for studying the response of a model to interesting Test Inputs..

To create Test Inputs with a time profile there are a series of functions that begin with RC — RC COMPARE, RC DECAY, RC GROW, RC RAMP and RC STEP (details in Chapter 4 of the Reference Manual). These all behave in a similar manner. For example:

```
TI Production to zero :TEST INPUT:
         production = RC RAMP(production,0,2,10)
```

This Test Input will cause production to go to ramp from whatever it is at time 10 to zero at time 12.

The RC... functions all take two optional arguments—a start time and a duration. If the duration is omitted, as it is above, the Test Input continues in the changed state to the end of the simulation. If you specify a duration the Test Input will continue in the changed state for the time specified, and then the variable will revert to its normally computed value. When it does revert the values the variable depend upon will likely be different so it most likely will not take on the value it has in a normal simulation.

If the start time (10 in the above example) is omitted, and the model contains the constant RC START TIME the changed specified will begin at this time. If RC START TIME is not in the model the change will start at INITIAL TIME + TIME STEP. Using RC START TIME in this manner is convenient because it allows you to globally change the time at which changes take effect and allows you to leave off additional arguments to the RC functions. Having Test Inputs start during a simulation is helpful because it prevents startup behavior from interfering with the testing of the model and also permits you to run tests with different relative values for variables.

Constraints

Constraints take the form

`name :THE CONDITION: condition :IMPLIES: consequence`

`:THE CONDITION:` and `:IMPLIES:` are special keywords in Vensim. *`condition`* and *`consequence`* are logical expressions described below. The name of a Constraint must use letters and numbers just as other variables in Vensim. Constraints do not need units of measure, though if you are using the Text Editor you must put in the tilde symbol ~ as a placeholder. You can attach comments to Constraints just as you do with other variables in Vensim.

Logical Expressions

Constraints use a condition and a consequence that are both defined as logical expressions. An example of this would be:

```
no capital no production :THE CONDITION: Capital = 0 :IMPLIES:
         production= 0
```

When testing Reality Check equations, Vensim will force a condition to be true whether the model generated values suggest it should be true or not, and test the consequence for truth. When the condition is true, but the consequence is not, Vensim reports the problem as a Reality Check failure. Vensim also does passive testing as described below.

Logical expressions can be more complicated then this. They are built up by using the comparisons `=`, `>`, `<`, and `<>` along with `:OR:`, `:AND:` and `:NOT:`. A valid logical expression could be:

```
Population > 8E9 :AND: (food ratio < .75 :OR:
         Pollution > critical pollution)
```

Here a number of things are being compared and this expression will be true if `Population > 8E9` and either *`food ratio < .75`* or *`Pollution > critical pollution`* or both. Logical expressions can quickly become difficult to understand and you are encouraged not to combine too many things in a condition. In the consequence portion, it is often useful to have many items combined with `:AND:`s (to test several consequences of a single condition), but more complicated structures are rarely helpful.

The condition portion of a Constraint definition is restricted to the comparison of variables to other variables and variables to numbers. The only exception to this is that you can use *`var=expr`*, or a named Test Input as one of the condition's logical components. Thus:

```
pop lt cc :THE CONDITION : Population < Carrying Capacity * 1.1
```

```
    :IMPLIES: deaths from crowding < 1000
```

is wrong because it takes the form *var<expr*, whereas

```
pop lt cc :THE CONDITION : Population = Carrying Capacity * 1.1
    :IMPLIES: deaths from crowding < 1000
```

uses *var=expr* and is a legitimate expression. Expressions included directly in Constraint conditions in this manner can use `TIME TRANSITION` and must conform to the rules for forming Test Inputs.

Test Inputs may be used for the condition of a Constraint, as in:

```
pop lt cc :THE CONDITION : pop at cc plus 10
    :IMPLIES: deaths from crowding < 1000
```

where *pop at cc plus 10* is a Test Input. Note that Test Inputs are treated as logical variables taking on a value of true if they are active and false if they are not.

All components of a Constraint are restricted to using simple functions (`MIN`, `MAX`, `SUM` etc.). Other than this, there is no restriction on logical expressions in the consequence portion of a Constraint definition.

It is not generally useful to test equality in a consequence because equality tests are very likely to fail even when there is nothing wrong with the model. This is because even for concepts that are definitionally equivalent, but computed in different manners, there are likely to be slight numerical differences which will be flagged during an equality test.

Dynamic Tests in the Consequences

Reality Check equations that have Test Inputs using RC... functions typically use a corresponding RC...CHECK function in the consequence. The RC ... CHECK functions work in an analogous way to RC... functions in Test Inputs. While Test Inputs change the value of a variable, the consequence portion of a Constraint makes a comparison of the value of a variable. The RC...CHECK function takes one more argument than the corresponding RC... function. This argument is the grace period and it allows a delay after a Test Input occurs before the consequences are tested. For example:

```
TI Production to zero :TEST INPUT:
         production = RC STEP(production,0)
RC No Production no Shipments :THE CONDITION:
         TI Production to zero :IMPLIES:
         sales <= RC RAMP CHECK(.5,sales,.0001)
```

The Test Input causes *production* to step to zero at RC START TIME. After a grace period of .5 sales is checked to see if it is less than or equal to .0001 times the value it had at RC START TIME. The use of .0001 instead of 0 prevents nuisance violations that might occur with continuous formulations and is good practice.

The grace period is the first argument to all the RC...CHECK functions except for RC COMPARE CHECK which takes uses the literal name of a file first.

:CROSSING:

In the implication portion of a Reality Check you can use :CROSSING: and :AT LEAST ONCE: with > and < to look for above/below type relations, as in:

```
... :IMPLIES:
         Inventory > :CROSSING: RC STEP CHECK(0,Inventory,1)
```

This will test to be sure that at RC START TIME Inventory is first bigger than its baseline value and then becomes smaller and stays smaller. If there were two `:CROSSING:` keywords in a row as in:

```
   Inventory > :CROSSING: :CROSSING: RC STEP CHECK(0,Inventory,1)
```

Inventory would need to start bigger, then become smaller, then become bigger and stay bigger.

If you use one or more `:CROSSING:` keywords, you can end the sequence with `:IGNORE:` to indicate that after the required number of crossings have been made, it does not matter if any more crossings occur. For example

```
   Inventory > :CROSSING: :IGNORE: RC STEP CHECK(0,Inventory,1)
```

This says that *`Inventory`* needs to start bigger, then become smaller, then it does not matter what it does.

:AT LEAST ONCE:

Analogous to the `:CROSSING:` keyword is a `:AT LEAST ONCE:` keyword that simply requires that the condition be true once after RC START TIME. For example

```
   Inventory > :AT LEAST ONCE: RC STEP CHECK(0,Inventory,1)
```

says that *`Inventory`* needs to exceed its value at RC START TIME at least once during the rest of the simulation. It might always be above, or cross from below to above. It the value is above once it can also cross below and the condition will remain true.

Empty Conditions

The condition part of a Constraint may be empty as in:

```
debt bounded :THE CONDITION: :IMPLIES: debt < 4E6
```

This equation states that no matter what happens there will never be more than four million in debt. Note that Vensim does not try to test all possible model conditions when it sees an empty condition. Constraints with empty conditions are checked passively anytime you are using the Reality Check function, and will detect a high debt.

For the simple example given here, using 4E6 as the maximum value that debt could take on in the equation for debt would also result in a message when debt exceeds 4E6 as long as warnings are not suppressed. The Empty condition can, however, be used to evaluate much more general expressions.

Wildcard Tests in the Consequence

In addition to testing a variable, you can test all variables to see if they satisfy a condition. To do this use a * instead of a variable name. For example you might write the Constraint

```
all peaceful :THE CONDITION: FINAL TIME=101 :IMPLIES:
```

```
* < 1E9 :AND: * >= -1E3
```

which will test to be sure that all variables stay in the range -1,000 to 1 billion. The condition *`FINAL TIME = 101`*, which simply runs the simulation an extra year, is used instead of the empty condition because this is a time consuming check to make and Constraints with empty conditions are checked passively every time any Reality Check is being made.

Simulation and Reality Check

Before we discuss the mechanics of actually running Reality Check equations, it is useful to describe very briefly what happens inside the model. Reality Check equations involve systematic intervention in the basic structure of the model. They are qualitatively different from sensitivity analysis in that there are not any well defined pathways of influence. Test Inputs and Constraints can cause changes to be made at almost any point in a model.

In order to accomplish the changes involved in running Reality Check equations, Vensim restructures your model, adding equations and modifying the sequence in which equations are computed to match. After completing Reality Check equations, Vensim returns the model to its original structure. This means that doing causal tracing on a run made with one or more Test Inputs active can give surprising and seemingly incorrect results.

In some cases, restructuring and reordering may leave the model ill formed. The most common problem is that the model may contain simultaneous equations and therefore not be computable. If this is the case, Vensim will report the problem and not complete the simulation. Because of the behavior-behavior nature of Reality Check equations, the existence of simultaneous equations does not necessarily represent a conceptual problem, but will prevent simulation. If possible, reformulate the Test Inputs causing the problems to avoid simultaneous equations.

NOTE Reality Check equations are always run using interpreted (not compiled) simulations. This is because they require continual restructuring of equations.

Active Constraint Checking

During active Constraint testing, Vensim forces the condition part of Constraint equations to be true, changing variable values or model structure where necessary.

? If the condition is an equality condition or Test Input, Vensim appends the equation to the existing model equation (remember that the equation may reference the original computed value for a variable). The new value for the variable is then used wherever the variable is used.

? If the condition uses inequality, Vensim first tests to see if the inequality is true.

    ? If it is true, the value of the variable is not changed.

    ? If it is not true, Vensim makes it true by assigning the value just as if it were an equality condition.

Having forced conditions to be true Vensim tests to see if the consequences are also true.

Passive Constraint Checking

In passive Constraint checking, Vensim simply evaluates both halves of the Constraint equations as logical expressions. If a consequence is false while its condition is true, an error is reported.

Whenever you run a Reality Check, Vensim tests all Constraints that are not explicitly activated for passive conformance. This checking does not get done during normal simulation.

Error Reporting

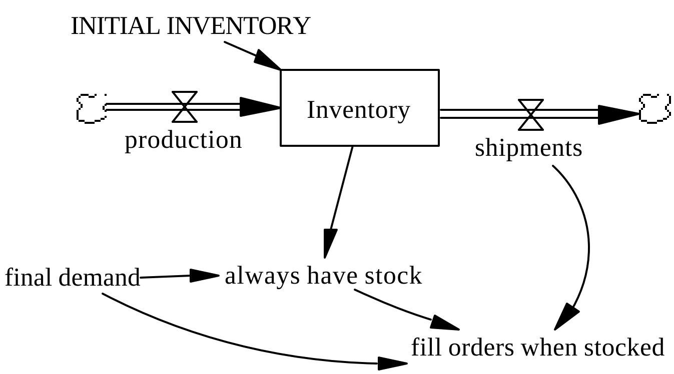
Violations of Constraints are reported the same way Lookup table overflows are. The first time a Constraint is violated an error is reported. Then a message is sent when the Constraint is no longer violated indicating a return to conformance. The next violation is reported, and so on.

Entering Reality Check Equations

You enter Reality Check equations the same way you enter normal model equations. In the Text Editor, you simply type them in. In the Sketch Editor, you can enter them in diagrams by showing all of the elements to be checked as causes of the Test Inputs and Constraints. The Test Inputs and Constraints are not part of the model's causal structure. In addition, the right hand side of alternative defining equations in Test Inputs are not shown as causes of the variable being given the Test Input. Thus the equations:

```
Inventory = INTEG(production-shipments,INITIAL_INVENTORY) ~~|
always have stock :TEST INPUT: Inventory = 3*final demand ~~|
fill orders when stocked :THE CONDITION: always have stock
         :IMPLIES: shipments >= final demand ~~|
```

would appear on the diagram as:



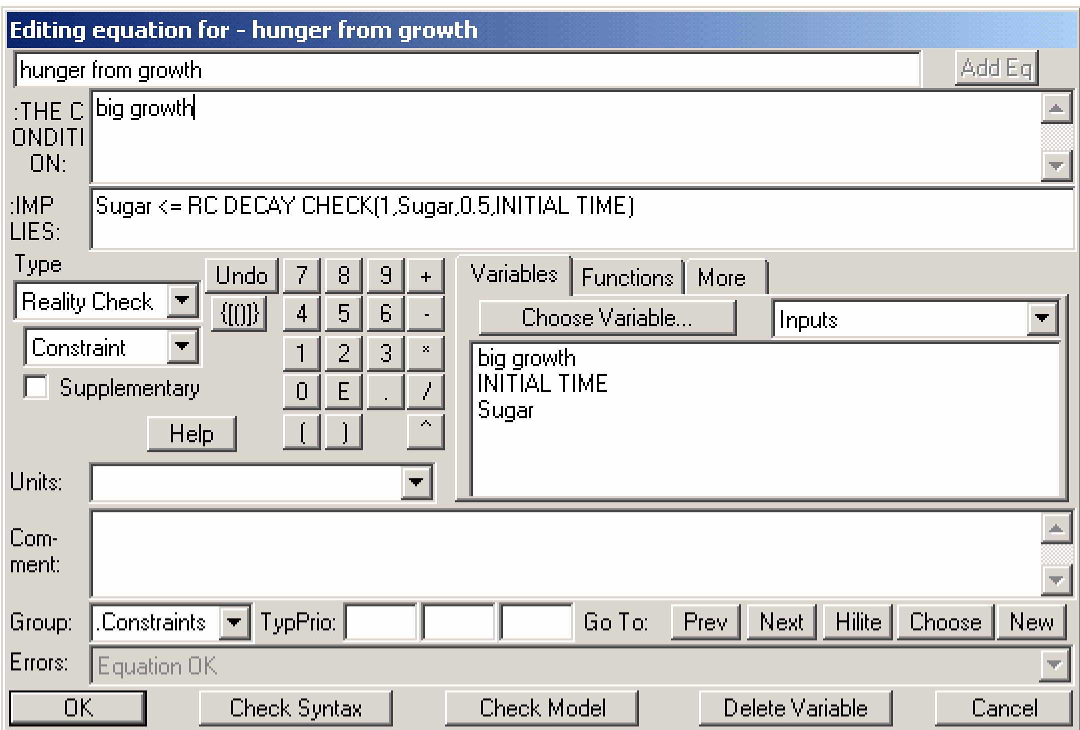
In it is likely that you will want to keep the equations and causal structure used in defining Reality Check equations separate from the normal model equations. One convenient way to do this is to put them in a separate group and include them on separate views.

In the diagram, the mixing of the different types of names can be confusing. When looking for arrows going in to `fill orders when stocked`, it is not what causes `fill orders when stocked` that is wanted, but what we need to know to determine if this is true. Given the difficulties of understanding diagrams just showing causal connections and flows, you may want to omit this added complexity from the working diagram. It is probably easiest to place Reality Check equations in a separate view so they will not be confused with model structure.

It can also be very convenient to adopt a naming conversion for Reality Check statements. For example you might start all Test Inputs with TI and all Constraints with RC.

Equation Editor

You enter Reality Check equations just like you would enter other equations. Create a variable, open with the Equation Editor, select the **Type** Reality Check and the **Suptype** Constraint or Test Input and fill in the equation.



When you select **Type** Reality Check, you will notice that the **Range** label changes to **TypPrio** (this appears near the bottom just to the right of **Group**). This is a mechanism for filtering Reality Check statements in the Reality Check Control dialog and will be discussed below. You can enter a number for both Type and Priority. The Values that Priority can take on are arbitrary, but 1-10 would be a common prioritization. Type should take on an integer value between 0 and 64. The settings are not available in PLE or PLE Plus.

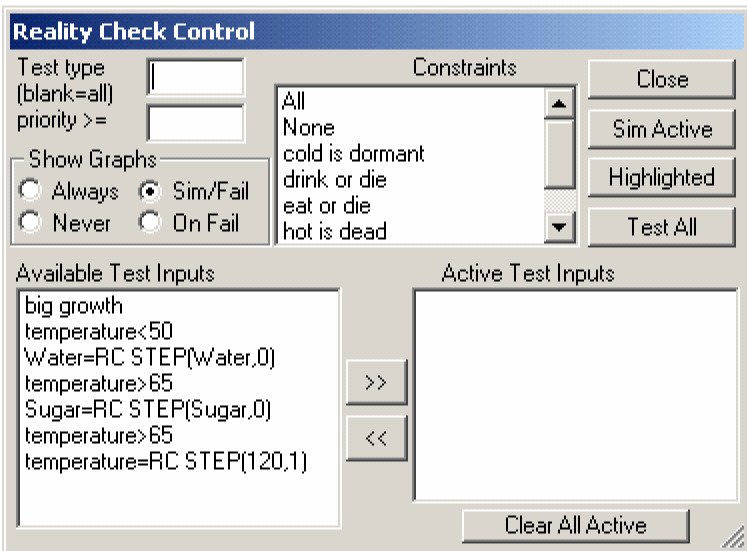
Running Reality Check

Reality Check is run by first setting up a simulation and then testing one or more of the Reality Check equations you have entered.

You can start Reality Check from the Toolbar or from the Simu lation Control dialog (see Chapter 8 of the Reference Manual for details on these). You can make changes and adjust other options just as you would for a normal simulation. After you have done this click on **Reality Check** in the Simulation Control dialog or click on the **Reality Check** button  on the Toolbar.

NOTE Any changes you have made to constant values, data to be used, and so on prior to launching Reality Check will continue to hold throughout the Reality Check session.

When you launch Reality Check, the Reality Check Control dialog appears:



Test type (not PLE or PLE Plus) allows you to specify which type of Reality Check you want to test. If this is blank all types will be tested. The type of a Reality Check is indicated in the **TypPrio** field of the Equation Editor. In the Text Editor Type, and Priority are enclosed in square brackets [ ] in the units field (~[type,priority]). This field is only applicable when the **Test All** button is used to start testing.

Priority >= (not PLE or PLE Plus) will restrict testing to only Constraints having a priority bigger to or equal to that specified. This field is only applicable when the **Test All** button is used to start testing.

NOTE If Constraints do not have a priority specified, they are treated as highest priority. If Constraints do not have a type specified they are assumed to match all types.

Show Graphs is used to display a graph of the variable being tested in the Consequence portion of a Constraint against the behavior that variable needs to conform to. This is done using a special invocation of the Graph Tool and can be very helpful for understanding the manner in which the failure occurred.

? **Always**, if checked, causes a graph to come up for each Constraint Checked.

? **Never**, if checked, suppresses graph output.

? **Sim/Fail**, if checked, causes graphs to appear whenever a Reality Check fails, and also when a single simulation is made using the **Sim Active** or **Highlighted** buttons.

? **On Fail**, if checked, causes a graph to come up only if a Reality Check fails. When you use the **Test All** button this is the same as **Sim/Fail**. For the **Sim Active** and **Highlighted** buttons this suppresses the graph unless there is actually a failure.

Constraints is a list of Constraints you have entered. Clicking on one of these will highlight the corresponding Test Inputs. You can then activate one or more of these Test Inputs.

Test Inputs is a list of the Test Inputs in the model. This list includes everything you defined explicitly as a Test Input, and all comparisons in the logical expressions making up the condition components of Constraint equations. Comparisons are shown directly and not given a name.

>> moves the highlighted items in the Test Inputs list to the Active Test Inputs list.

<< removes the highlighted items in the Active Test Inputs list.

Clicking on an element in the list highlights it. Control-clicking toggles the highlight status adding or removing it from the selection of highlighted elements. Double clicking moves the item to the Active Test Inputs list.

Active Test Inputs shows a list of the Test Inputs, explicit and implicit, that are active. These will be used when the Simulate Button is clicked. Clicking on an element highlights it, and unhighlights anything else that may be highlighted. Control-clicking toggles an element's highlight. Double clicking removes an element from the list.

Clear All Active removes all elements of the Active Test Inputs list.

Sim Active simulates the model using the active Test Inputs in the list. All other Constraints are tested passively. If the simulation succeeds the results will be stored just as for a normal simulation, and you can review them with the tools on the Workbench. Be careful to note that causal tracing will not always give the expected results since model structure may have been changed during the simulation.

Highlighted simulates the model using the item highlighted in the Constraint list. Because of the logical structure of the condition portion of a Constraint this may actually require more than one simulation. The final simulation will be stored just as a normal simulation.

Test All tests all the Constraints in the model one at a time by forming the collection of Test Inputs necessary to activate each Constraint. Because of the logical structure of the condition portion of a Constraint equation, it may take more than one simulation to test a Constraint. This testing can be time consuming, activity and errors are reported in a separate window. Running Test All does not store any simulation results but simply reports the results of Reality Check.

Close closes the Reality Check Control dialog.

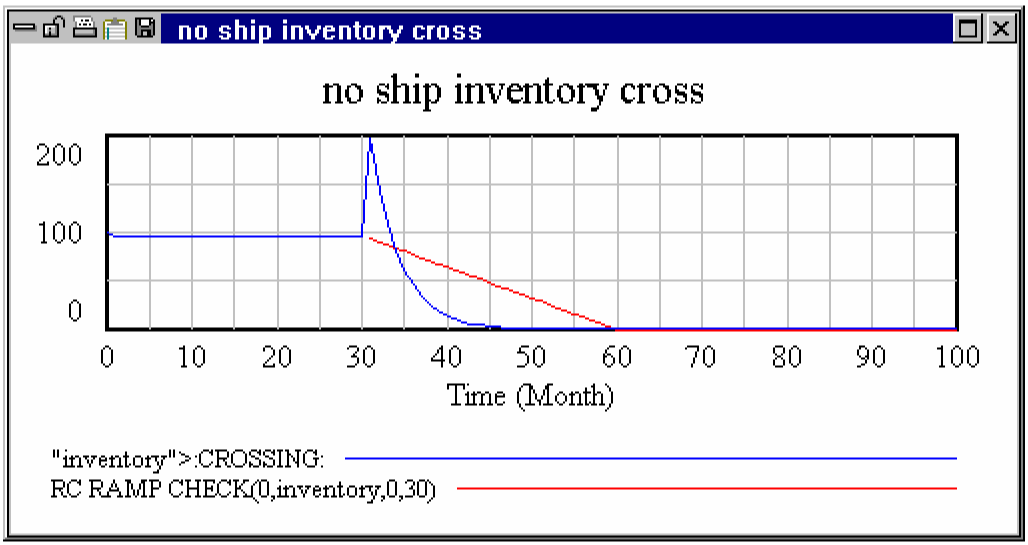
The Reality Check Tool

You can add a Reality Check tool to the Analysis toolset (does not apply to PLE or PLE Plus). This tool is a shortcut for starting Reality Check, highlighting a Constraint and clicking on **Highlighted**. The tool works on the Constraint that is currently selected into the Workbench. If the Workbench variable is not a Constraint an error message is reported.

Reality Check Results

Reality Check results are reported in a text window. The window shows which Constraints were checked and whether any Constraint was violated. A new window is brought up each time you click on **Sim Active**, **Highlighted** or **Test All**. Examples are given below.

One or more graphs may also appear. These graphs are intended to display the implication portion of a Constraint:



Here the spiked line shows what inventory does while the sloped line shows what it is being compared to.

Reviewing Simulation Results

Each time you click on **Sim Active** or **Highlighted** a new simulation is made. That simulation is given the name currently specified in the Runname box (on the Toolbar or in the Simulation Control dialog). You can, while the Reality Check Control is open, look at the results in the simulation just as you would any other simulation. You can then make another simulation and review those results. Each simulation you make will overwrite the last one. If you want to compare two Reality Check simulations, you should close the Reality Check Control and then start again placing a different name in the simulation Runname box.

NOTE If you change constants in the Simulation Control dialog or on screen using Set Up a Simulation from the Toolbar, those constant changes will be held throughout your Reality Check session.

Reality Check and Yeast Growth

To demonstrate the mechanics of writing and using Reality Check equations, it is useful to work through a very simple example. Suppose that you want to model the growth of yeast in a bowl of water. The water is held at a constant temperature that can be set, and has a fixed amount of sugar in it to start.

We first list some of the Reality Check equations that must be satisfied:

? If temperature is below 50 degrees yeast growth should stop, and the yeast go dormant.

? If temperature is above 100 degrees the yeast should die out.

? If there is no sugar and the yeast are not dormant the yeast should die out.

? If there is no water and the yeast are not dormant the yeast should die out.

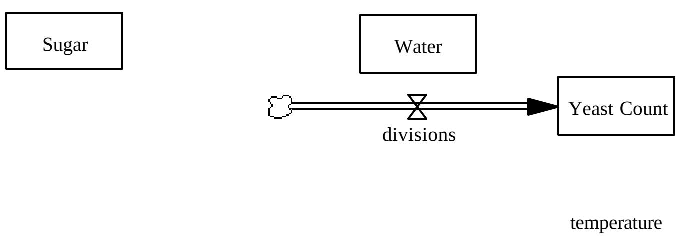
? If the yeast continue to grow they will consume all the sugar.

? If the yeast continue to grow they will consume all the water.

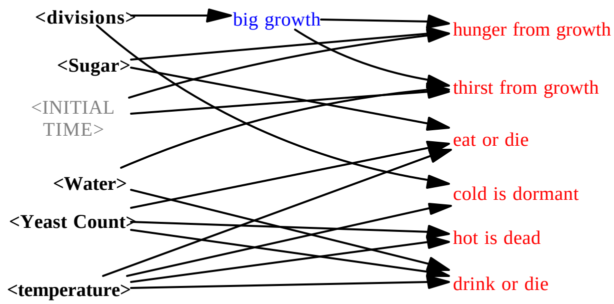
The other thing we know is that yeast reproduce by dividing and that when conditions are right they can reproduce with a doubling time of about 10 minutes.

Test Input and Constraint Equations

We will start the modeling process by defining the variables that we need to test our Reality Check equations and then entering our Reality Check equations. The Reality Check equations we have outlined talk about the number of yeast, the amount of water, the amount of sugar, the number of yeast divisions and temperature. Identifying levels, we put these into a skeletal model as:

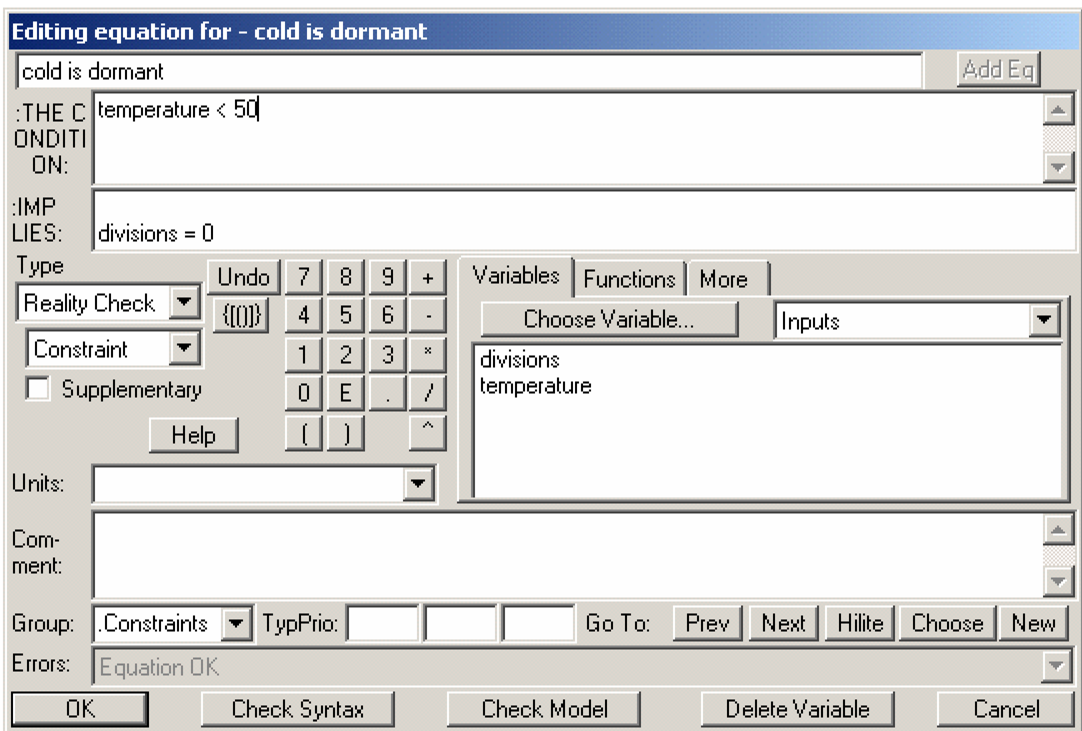


For clarity, we want to build up our Reality Check equations in a second view. If you are using Vensim PLE you will need to put them in the same view with the model structure. The first step is to place all of the model elements as shadow variables. Using shadow variables allows the model structure to change without requiring modification to the Reality Check diagram.



There is no fixed set of rules for structuring Reality Check diagrams. In our experience, making a column for normal model variables, Test Inputs and Constraints is the easiest format to follow. It can also be helpful to color code (e.g., blue for Test Inputs and red for Constraints). The arrows can get pretty messy this way, but if you organize Reality Check information by the variables that are affected, this is usually not a big problem.

You can enter the Constraints and Test Inputs in the Equation Editor:



You will need to select the equation **Type** Reality Check and the subtype Constraint. The conditional and consequence are split into two separate windows.

The Reality Check equations are:

```
cold is dormant :THE CONDITION: temperature < 50 :IMPLIES:
         divisions = 0
```

```
hot is dead :THE CONDITION: temperature = RC STEP(120,1)
         :IMPLIES: Yeast Count <= RC DECAY CHECK(1,Yeast Count,2)
```

For ***`hot is dead`*** we have used a step in the ***`temperature`*** and then used a RC DECAY CHECK function to check ***`Yeast Count`***. For this model a decay is appropriate for looking at the behavior of ***`Yeast Count`*** since we are focusing in on situations of dying out.

```
eat or die :THE CONDITION: Sugar = RC STEP(Sugar,0)
:AND: temperature > 65 :IMPLIES:
         Yeast Count <= RC DECAY CHECK(1,Yeast Count,15)
```

Here, the Constraint requires the ***`Yeast Count`*** decline toward zero with an average death time of 15.

```
drink or die :THE CONDITION: Water = RC STEP(Water,0)
         :AND: temperature > 65
         :IMPLIES: Yeast Count <= RC DECAY CHECK(1,Yeast Count,1)
```

```
big growth :TEST INPUT: divisions = 1e+022
```

```
hunger from growth :THE CONDITION: big growth :IMPLIES:
         Sugar <= RC DECAY CHECK(1,Sugar,0.5,INITIAL TIME)
```

```
thirst from growth :THE CONDITION: big growth :IMPLIES:
         Water <= RC DECAY CHECK(1,Water,0.5,INITIAL TIME)
```

Notice that for the last two Constraints the RC DECAY CHECK function starts checking at INITIAL TIME since the Test Input starts from the beginning of the simulation. If the Test Input had used an RC STEP function to start during the simulation the INITIAL TIME argument would have been left off of the last two RC DECAY CHECK uses.

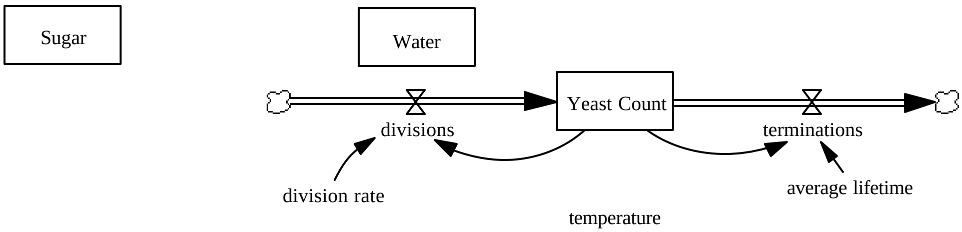
Finally we have:

```
RC START TIME = 10
```

The time is arbitrary. It might be that starting early there will be few yeast and lots of sugar, while starting later there would be more yeast and less sugar. When a model passes Reality Check equations to your satisfaction it is a good idea to change RC START TIME and retest.

An Initial Model

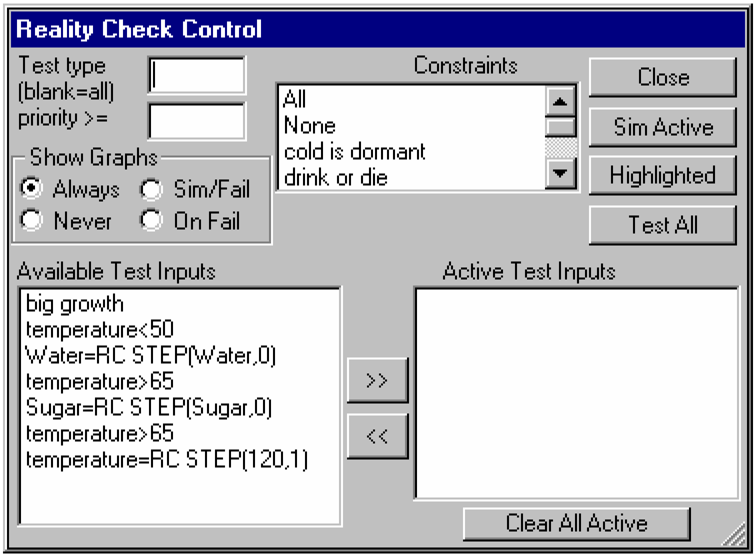
We have a short list of things we know will need to be in our model just to make it possible to use Reality Check equations. Armed with a very basic understanding of exponential growth, we can fill in our skeleton framework to build a model. Suppose we start with the simplest possible model (*yeast1.mdl*):




```
INITIAL TIME  = 0, FINAL TIME  = 300, Units: Minute

average lifetime  = 250
Units: Minute

division rate  = 0.08
Units: 1/Minute

divisions  = Yeast Count*division rate
Units: Cell/Minute

Sugar  = 100
Units: g

temperature  = 85
Units: Farenheit

terminations  = Yeast Count/average lifetime
Units: Cell/Minute

TIME STEP  = 1
Units: Minute

Water  = 100
Units: ml

Yeast Count  = INTEG( divisions-terminations,100)
Units: Cell
```

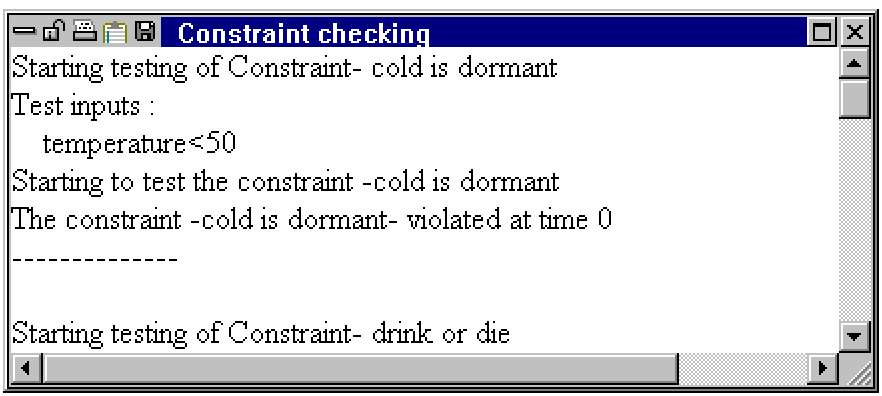
In this model we are not making use of ***Sugar***, ***Water*** or ***Temperature***, and they have been set to constants with the **Type** drop down box in the Equation Editor.

Next, we run our Reality Check equations on this model. Click on the **Reality Check** button on the Toolbar, or open the Simulation Control Dialog and click on the **Reality Check** button.



In the Reality Check Control dialog you will see a list of all the Constraints for the model. Below that is a list of Test Inputs. One, *big growth*, was named explicitly while the others are simply derived from the conditional parts of the different Constraints.

Clicking on Test All begins a series of simulations. The following violations are reported:

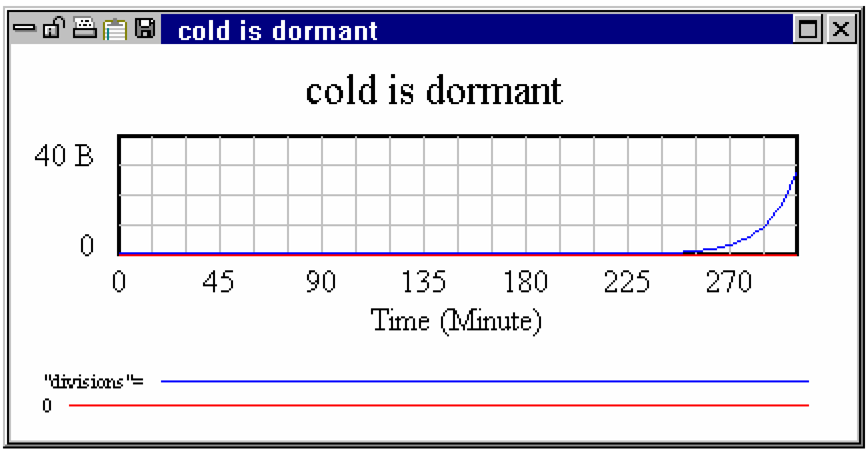


The condensed report is:

```
Starting testing of Constraint- cold is dormant
Test inputs :
    temperature<50
Starting to test the constraint -cold is dormant
The constraint -cold is dormant- violated at time 0
--------------
The constraint -drink or die- violated at time 11
```

```
--------------
The constraint -eat or die- violated at time 11
--------------
The constraint -hot is dead- violated at time 11
--------------
The constraint -hunger from growth- violated at time 1
--------------
The constraint -thirst from growth- violated at time 1
--------------
```

Every Constraint has been violated. This is not a great surprise since the elements involved in the Constraints were not connected in the model. The model as presented represents a basic growth mechanism. No attention has been given to either control or containment which is what all the Constraints relate to. In addition to the error window there will be 6 graphs such as:



The top line shows what ***divisions*** does, while the bottom line shows what it ought to do.

The report window ends with a summary of what has happened.

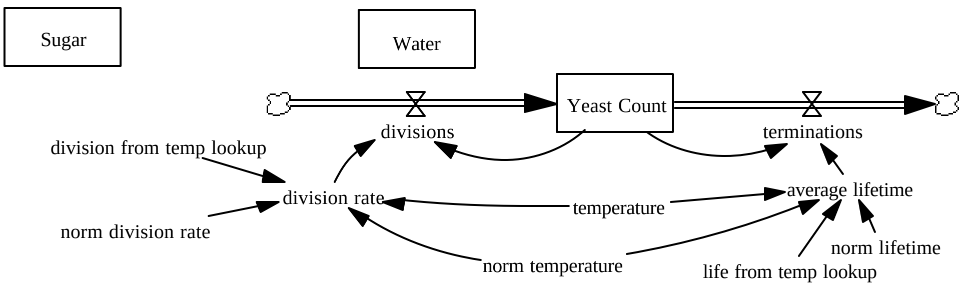
```
*****************
0 successes and 6 failures testing 6 Reality Check equations
The Reality Check Index as run is 0
Closeness score is  0.0% on 6 measurements
```

The first line is a summary count. The second line reports the Reality Check Index. This is defined as the number of successes divided by the product of the number of dynamic variables in the model with the total number of variables in the model. Since for every variable pair there is the potential for one or more Reality Check equations this index is something that should be near one for a model with a complete Reality Check set.

Finally a closeness score is reported. The closeness score is a the average closeness of a Reality Check. If a Reality Check Passes, its closeness if 1. It it fails its closeness is 1 minus the average absolute error divided by the amount of variation in the variable being checked. Thus a Reality Check that only just fails has a Closeness score close to 1, so that the closeness score is a continuous measure of how badly, on average, Constraints have been violated.

Temperature, Divisions and Terminations

Two conditions relate temperature to growth. If temperature is low everything goes dormant. If temperature is high the yeast die. We replace the simple equations for `division` and `termination` with this in mind we create *yeast2.mdl as:*




```
divisions = Yeast Count * division rate
division rate = norm division rate * division from temp lookup (
   temperature / norm temperature )
norm division rate = 0.08
norm temperature = 80
division from temp lookup ( (0,0),(0.8,0),(1,1),(2,1) )
terminations = Yeast Count / average lifetime
average lifetime = norm lifetime * life from temp lookup (
   temperature / norm temperature )
norm lifetime = 250
life from temp lookup ((0,1),(1,1),(1.25,0.02),(2,0.001) )
```

Now testing the Constraints shows that *`cold is dormant`* is respected.

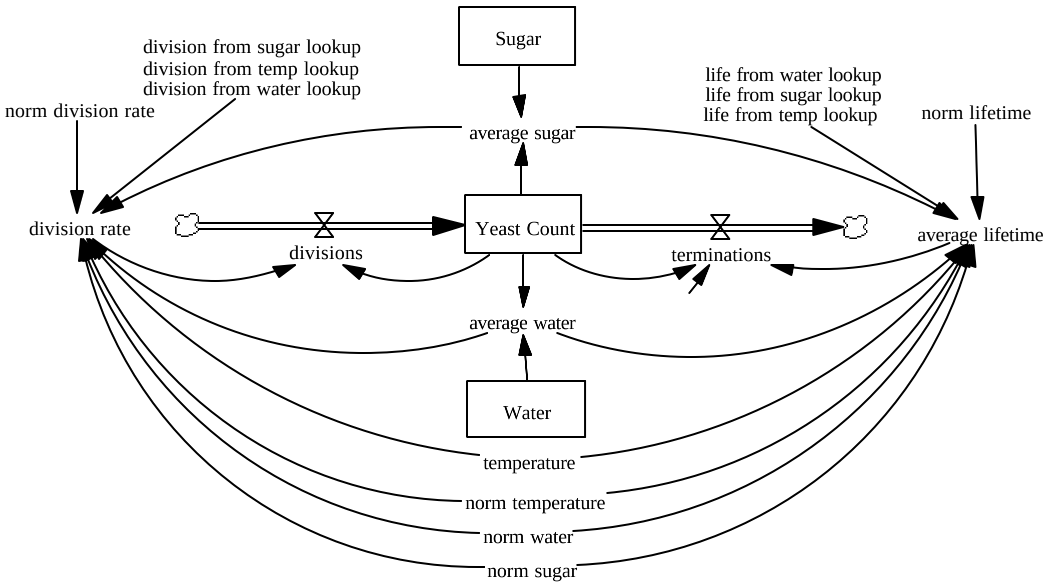
Constraint is violated because the yeast do not die out quickly enough when the temperature is increased. In this case, we can restructure the model to meet the Constraint, or relax the Constraint slightly, depending on which is more realistic. One way to relax the Constraint is to change the time transition to allow more time for the yeast to die out (we change the *`RC DECAY CHECK`* argument from 2 to 5):

```
hot is dead :THE CONDITION:  temperature = RC STEP(120,1)
         :IMPLIES:  Yeast Count <= RC DECAY CHECK(1,Yeast Count,5)
```

Now testing the Constraints shows that *`cold is dormant`* and *`hot is dead`* are both respected. This kind of interplay, trading off strictness of Constraints and making adjustments to equations is valid. It provides a means of focusing more attention on the Constraints that may be violated to a small extent.

Divisions as influenced by Water and Sugar

The next two Constraint equations state that water and sugar are necessary for survival. We model the impact of insufficient water and sugar on both the division rate of the yeast and the average lifetime of the Yeast. The new model (*yeast3.mdl) looks like:*



The equations are available with the sample model. Note that the Test Inputs chosen cause extreme conditions to exist for Water and Sugar, and the way the model is formulated makes ***terminations*** skyrocket. Using Euler integration, the system becomes a 1 period oscillator with Yeast Count going negative. To prevent this, another integration technique can be used, or the equation for ***terminations*** changed to:

```
terminations  = MIN(Yeast Count/TIME STEP,
         Yeast Count/average lifetime)
```

This formulation prevents uninteresting dynamics, but also can set ***Yeast Count*** to precisely zero. This means that ***average sugar*** and ***average water*** need to be computed as

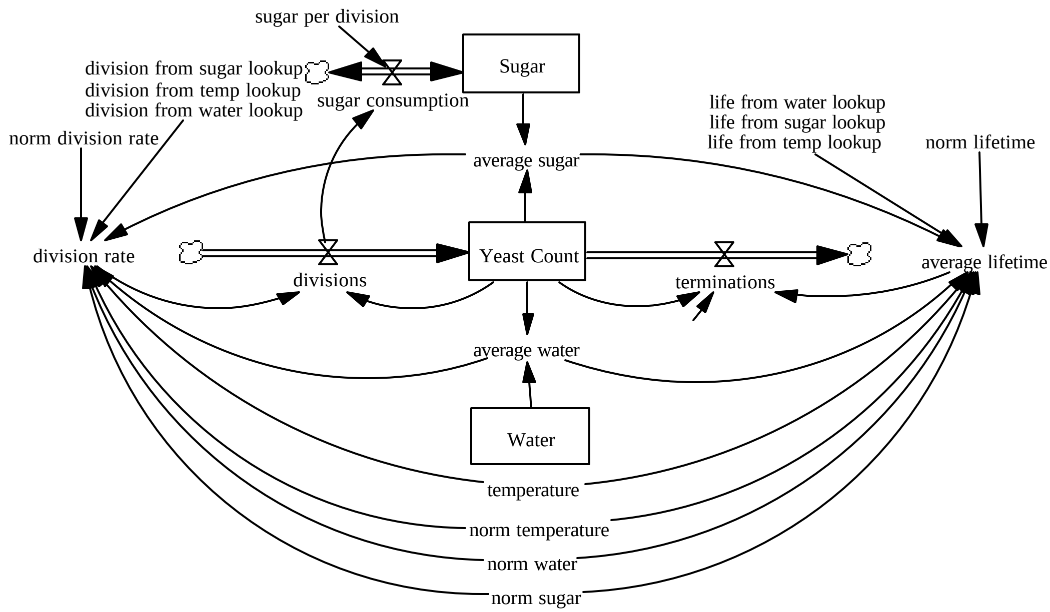
```
average sugar = ZIDZ(Sugar,Yeast Count)
average water = ZIDZ(Water,Yeast Count)
```

to prevent a numerical error (overflow) when ***Yeast Count*** is 0.

With the changed structure, the model now passes four out of the six Constraints.

Water and Sugar as Influenced by Divisions

Nearing the end, we still have not made the model conform to the last two Constraints. The problem is that we have not yet made a connection from what the yeast are doing to how much water and sugar there is. The simplest way to model this is to consider the division process as a consumer of sugar and water resources. Thus we have the model *yeast4.mdl*:



And this model conforms to all the Constraints written. But is it a good model? In this case the answer is probably "not yet". For a detailed an accurate representation of yeast growth we would need to get some experimental data, and calibrate the model for temperature and resource deprivation effects. The point about this model is that is does not violate the most obvious common sense Reality Check equations.

We have shown a number of different models. What about the Reality Check equations? Except for the small change made to `hot is dead`, they are unchanged and have exactly the same diagram.

Conclusion

The use of Reality Check serves two purposes. First, it is written record of things that were assumed must be true for a model to make sense. These are things that typically go undocumented, even though they may be the most important product of the modeling exercise. Very simple insights such as the notion that if the yeast keep growing they will eventually run out of water are really important things to understand about a system.

The second important thing about this example is that it illustrates a way of building models that is both effective and relatively efficient. You start from a basic core and then add the complementary structure necessary to get conformance to Constraints. This provides a rigor and direction that can increase greatly both the speed and quality of modeling work.

15 Sensitivity Testing

Sensitivity Testing is not available in Vensim PLE.

Monte Carlo Simulations

Sensitivity testing is the process of changing your assumptions about the value of Constants in the model and examining the resulting output. Manual sensitivity testing involves changing the value of a Constant (or several Constants at once) and simulating, then changing the value of the Constant again and simulating again, and repeating this action many times to get a spread of output values.

Monte Carlo simulation, also known as multivariate sensitivity simulation (MVSS), makes this procedure automatic. Hundreds or even thousands of simulations can be performed, with Constants sampled over a range of values, and output stored for later analysis. Latin Hypercube sampling is a specialized form of sensitivity testing that allows faster sensitivity testing on very large models.

Market Growth Model (sales.mdl)

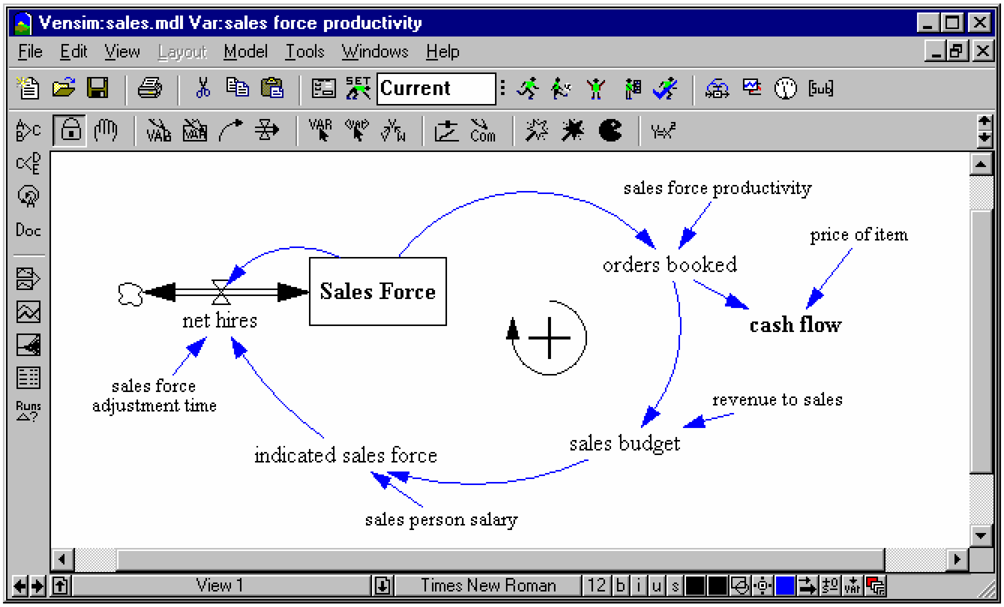
The sales model shown below contains a major positive feedback loop, in which more sales people can generate more sales, thereby increasing revenue and allowing more sales people to be hired. A minor negative feedback loop allows adjustment of the sales force over a period of time through hiring and layoffs.

EITHER

✍ Open the model *sales.mdl* contained in the directory *guide\chap15*.

OR

✍ Build the model as shown in the diagram and equation listing below and save it in the directory *guide\chap15* with a different name (e.g., *sales1.mdl*). Time Bounds are INITIAL TIME = 0, FINAL TIME = 60, TIME STEP = 0.25, Units for Time: Month.



Sales.mdl Equations

```
cash flow=
    orders booked * price of item
Units: dollars/Month

indicated sales force=
    sales budget / sales person salary
Units: person

net hires=
   (indicated sales force - Sales Force)/sales force adjustment time
Units: person/Month

orders booked=
    Sales Force * sales force productivity
Units: unit/Month

price of item=
    100
Units: dollars/unit

revenue to sales=
    10
Units: dollars/unit

sales budget=
    orders booked * revenue to sales
Units: dollars/Month
```

```
Sales Force= INTEG (
         net hires,
               50)
Units: person

sales force adjustment time=
    6
Units: Month

sales force productivity=
    210
Units: unit/(person*Month)

sales person salary=
    2000
Units: dollars/(person*Month)
```

Base Simulation

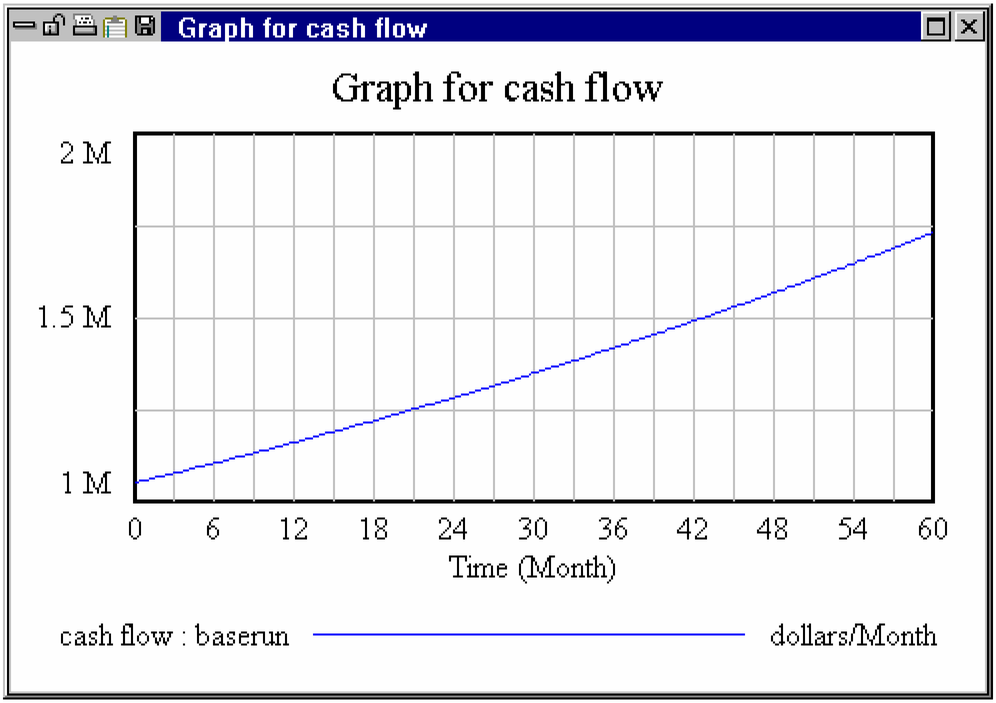
First we will simulate the model to see the behavior with the normal values for the model's Constants.

✍ Double click on the **Runname** editing box, type in *baserun*.

✍ Click the **Simulate** button.

✍ Double click *`Sales Force`* then click on the **Graph** tool. Double click *`cash flow`* then click on the **Graph** tool.

We find that both *`Sales Force`* and *`cash flow`* are slowly growing.



✍ Select the Control Panel **Datasets** tab and double click on *baserun* to unload it.

Uncertainty in Multiple Parameters

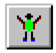
This model contains five constants that we can vary to examine their effect on simulation output. We will assume that we know the exact values for two constants: ***price of item*** and ***revenue to sales*** (because these are policy decisions that managers can readily set). The uncertain parameters are ***sales force productivity***, ***sales force adjustment time***, and ***sales person salary***. We will select these parameters and assign maximum and minimum values along with a random distribution over which to vary them to see their impact on model behavior. Note that we could select only one parameter if we wanted to see how sensitive model behavior is to one parameter.

NOTE Parameter is a synonym for Constant. In the Vensim documentation parameter is frequently used to refer to model Constants that we select for variation during sensitivity analysis and optimization.

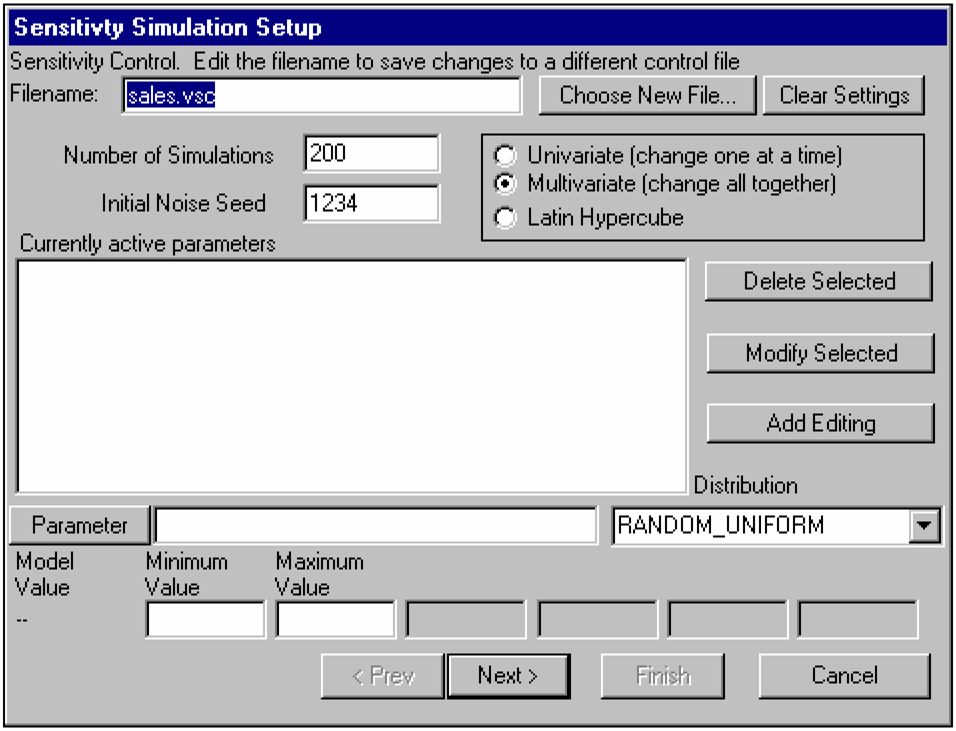
Sensitivity Control Parameters

Vensim has two methods of setting up sensitivity simulations. One is to use the Sensitivity tab of the Simulation Control to set up the control files. The other way is to use the Sensitivity button on the Toolbar to activate the Sensitivity Wizard.

✍ Double click on the **Runname** editing box and type the name *sensitivity* for the dataset (currently *baserun*).

✍ Click the **Sensitivity** button  on the Toolbar.

The Sensitivity Wizard opens at the Sensitivity Control



✍ Make sure that the radio button for **Multivariate** is selected and that the **Number of Simulations** is set to 200.

Monte Carlo multivariate sensitivity works by sampling a set of numbers from within bounded domains. To perform one multivariate test, the distribution for each parameter specified is sampled, and the resulting values used in a simulation. When the **Number of Simulations** is set at 200, this process will be repeated 200 times.

✍ Click on the **Parameter** button, a Control Parameter dialog box will open showing all the parameters (Constants) in the model that can be selected for Monte Carlo sampling. Click on `sales force productivity` and click **OK**.

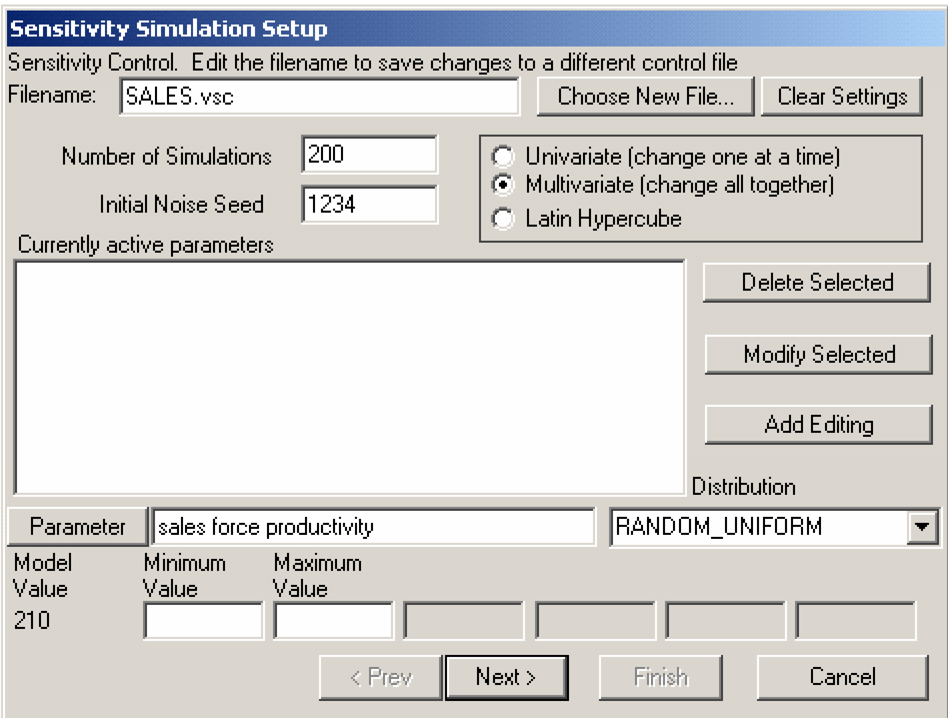
Random Uniform Distribution

In order to do sensitivity simulations you need to define what kind of probability distribution values for each parameter will be drawn from. The simplest distribution is the Random Uniform Distribution, in which any number between the minimum and maximum values is equally likely to occur. The

Random Uniform Distribution is suitable for most sensitivity testing and is selected by default. Another commonly-used distribution is the Normal Distribution (or Bell Curve) in which values near the mean or more likely to occur than values far from the mean.

Vensim provides a variety of different distributions to choose from. The most commonly used distributions are the Uniform, Normal and Triangular distributions. If you do not have any reason to choose a specific distribution sticking with a uniform distribution is usually sensible.

Minimum and maximum values are chosen to bound each parameter. Note the actual model value of 210 showing below the **Parameter** button.



✍ Click on the box labeled **Minimum Value** and type in 200. Click on the box labeled **Maximum Value** and type in 220.

The minimum value of 200 represents the lowest productivity we think the sales force can achieve; the maximum value of 220 represents the highest productivity we think they can achieve.

✍ Click on the **Add Editing** button.

✍ Click on the **Parameter** button and click on *`sales force adjustment time`* and click **OK**.

✍ Click on the box labeled **Minimum Value** and type in 3. Click on the box labeled **Maximum Value** and type in 12. These figures are asymmetrical around the model's value of 6; we think the value might be a little lower or a lot higher. Click on the button **Add Editing**.

✍ Click on the **Parameter** button and click on `sales person salary` and click **OK**.

Random Normal Distribution

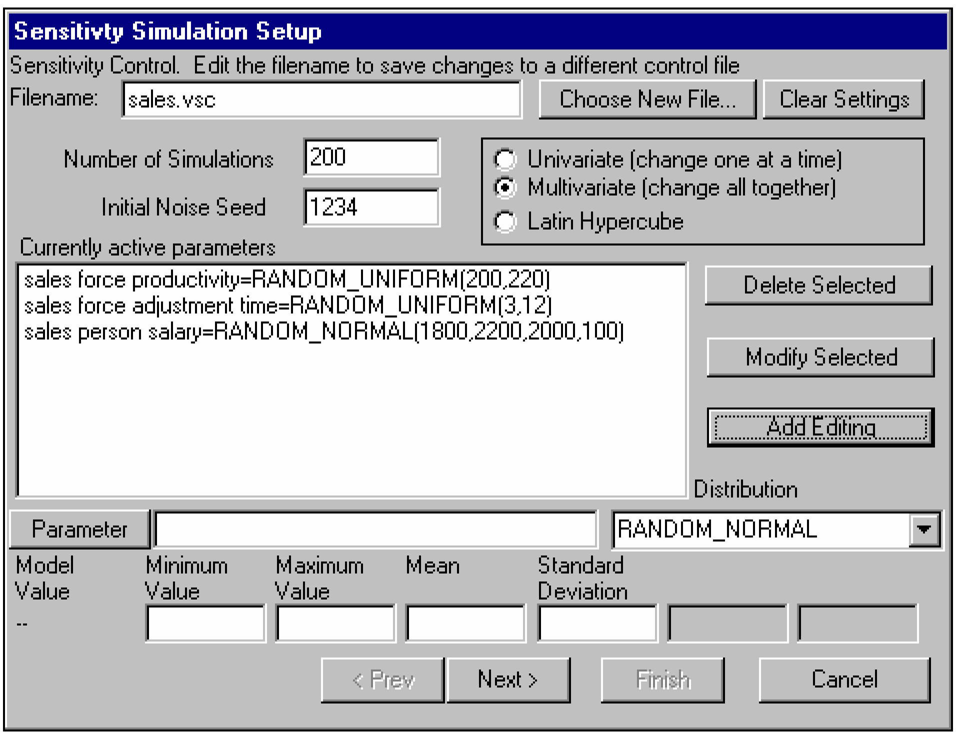
For `sales person salary` we will choose something different from the default distribution. RANDOM NORMAL samples values according to a Normal Distribution, and requires maximum and minimum bounds as well as a mean and standard deviation to be specified. (Technically this is called a Truncated Normal Distribution. The Normal Distribution is unbounded and very large negative and positive values can occur.)

✍ Click on the dropdown arrow of the box **Distribution**. Select RANDOM NORMAL from the list (you will need to use the scrollbar to scroll the list up).

Note that some editing boxes have been added to the bottom of the Sensitivity Control.

✍ Click on the box **Minimum Value** and type in 1800. Click on the box **Maximum Value** and type in 2200. Click on the box **Mean** and type in 2000. Click on the box **Standard Deviation** and type in 100. Click on the button **Add Editing**.

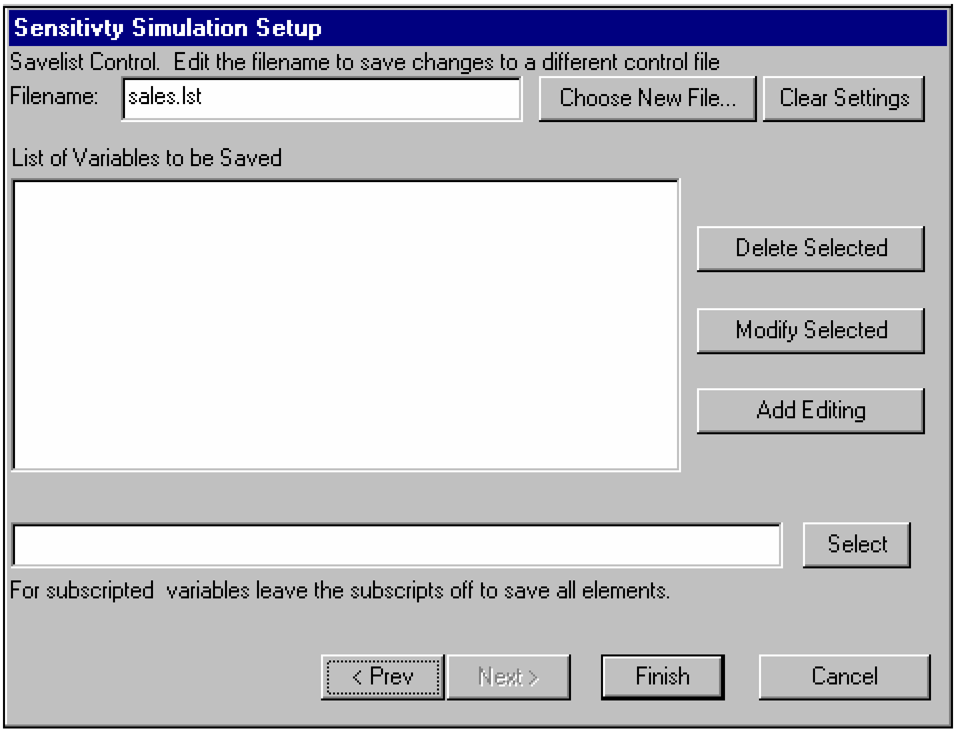
The Sensitivity Control should look the same as below.



✍ Click the **Next** button to move to the Save List control.

Save Lists

Save lists are files that tell Vensim which variables to save values for. Sensitivity simulations generate a huge amount of data, so it is necessary to limit the data saved to only those variables that we are really interested in. You should choose to save values only for variables that you think are of real interest; trying to save sensitivity values for all variables in the model will take a long time and require a large amount of disk space.



✍ Click on the **Select** button, a Variable Save dialog will open showing all model variables. Choose `cash flow` and click **OK**. Click on the button **Add Editing**.

✍ Click on the **Select** button and choose `Sales Force` then click **OK**. Click on the button **Add Editing**.

NOTE We could have simply typed these names individually into the editing box then clicked the button **Add Editing**.

Sensitivity Simulations

✍ Click on the **Finish** button.

The model will simulate once then perform 200 additional simulations while automatically varying the parameters `sales force productivity`, `sales force adjustment time`, and `sales` `person salary`. The dataset now contains the standard behavior for all variables with the model's original Constant values, and a range of values for the behavior of the variables `cash flow` and `Sales Force` generated by the 200 sensitivity simulations.

✍ Open the **Control Panel** and click the **Datasets** tab. Double click on the run *baserun* appearing in the **Loaded** runs list to remove it.

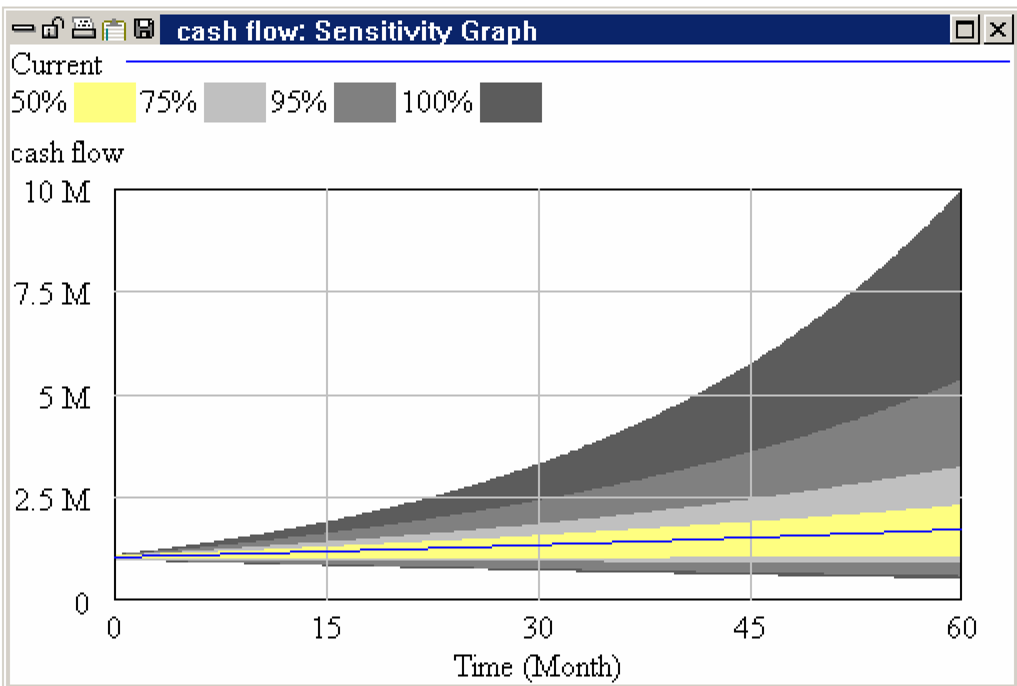
✍ Double click on the variable `cash flow` on the sketch to select it as the Workbench Variable.

Time Graph Sensitivity Output

Results of sensitivity testing can be displayed in different formats. Time graphs display behavior of a variable over a period of time. The variable's spread of values, at any period in time, are displayed either in terms of confidence bounds, or as separate values which combine to form individual simulation traces.

Confidence Bounds

✍ Click on the **Sensitivity Graph** Analysis tool. The default configuration for this tool is to plot confidence bounds.



A graph is generated showing confidence bounds for all the output values of `cash flow` that were generated when the three parameters were randomly varied about their distributions. You can expand the graph to full screen by clicking on the maximize/minimize button just left of the close button in the upper right corner.

The outer bounds of uncertainty (100 %) show maximum values of approximately 10 million dollars and minimum values of approximately 500 thousand dollars at the end of the simulation. Note the possibility of a decline in `cash flow`. The first simulation run (with the values of the Constants contained in the model) is plotted as a line indicated by the run name *sensitivity*.

Mean Values (Not PLE Plus)

A mean value lies in between the confidence bounds, and can be plotted by:

✍ Click with the right mouse button on the **Sensitivity Graph** tool. Click on the check box **Plot Mean Value** in the **Show Sensitivities as:** field.

✍ Click on the check box **Suppress first run plot**. (This will leave only the mean value and confidence bounds.) Click **OK**.

✍ Click on the **Sensitivity Graph** tool.

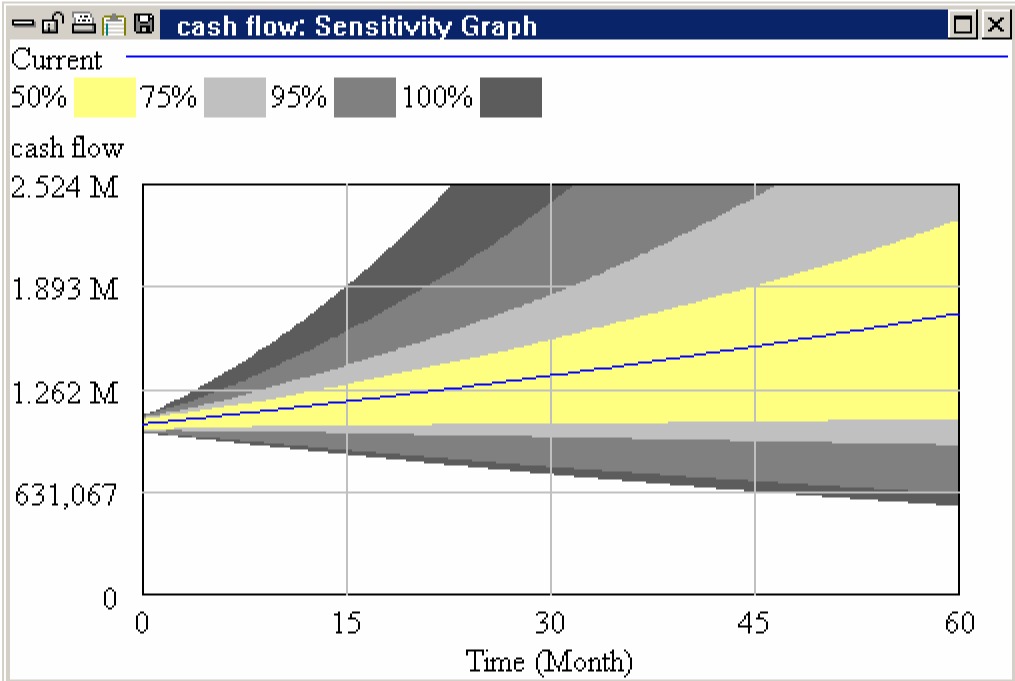
A plot is generated showing the mean value of the confidence bounds as a red line.

Focusing Graph Scales

Let's focus the vertical scale to show greater detail in the lower range of uncertainty values.

✍ Place the pointer on the horizontal graph line that shows 2.5 M in `cash flow`. Hold down the Ctrl key then click and hold down the mouse button. Drag the mouse so the cursor moves down to the bottom of the graph (the graph line that shows 0 in `cash flow`). Release the mouse button.

✍ Click on the **Sensitivity Graph** tool.

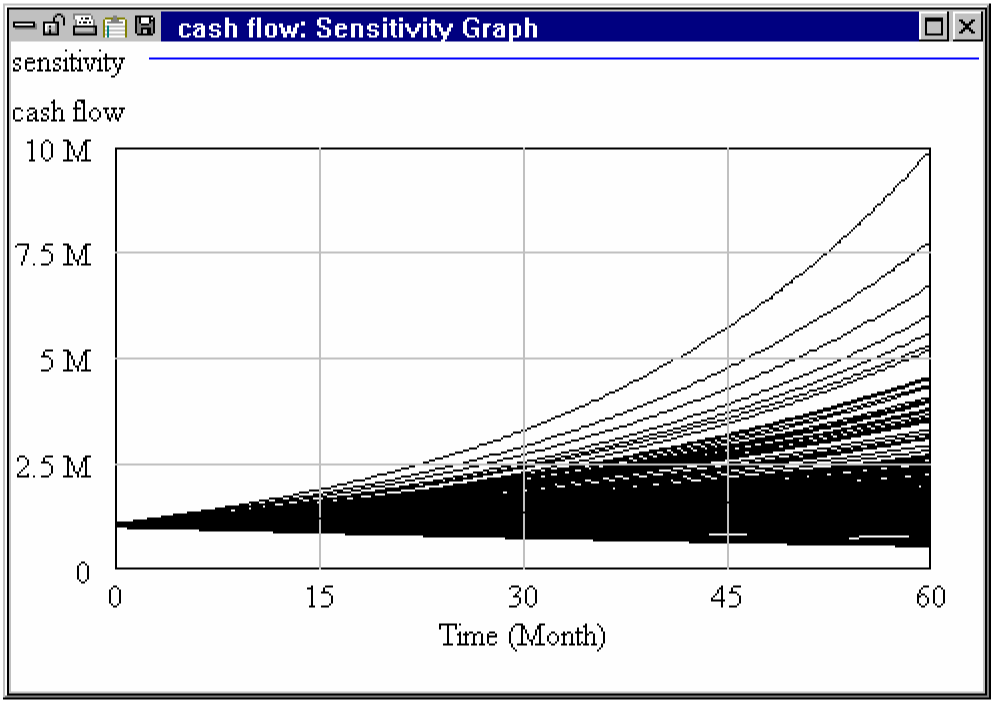


Individual Traces

Another option in viewing sensitivity output is to view all the individual simulation traces. In Vensim PLE Plus click on the lower sensitivity graph . For other configurations first adjust the options:

- Click with the right mouse button on the **Sensitivity Graph** tool. Click on the radio button **Individual Traces** in the **Show Sensitivities as** field then click **OK**.
- Click on the **Sensitivity Graph**.

Individual traces for each simulation are now plotted. You might want to maximize this graph (if not already maximized) to full window size in order to discern some of the individual traces.



✍ Double click on `Sales Force` then click on the **Sensitivity Graph**.

We see very similar behavior, which is not surprising since the behavior of the feedback loop through `Sales Force` directly governs the behavior of `cash flow`.

Histogram Sensitivity Output

Results of sensitivity testing can be displayed in terms of histograms. These provide a cross section of values at a particular period in time. Histograms display the number of simulations for which the variable was in a given range at the specified time. Histograms provide a mechanism for seeing the distribution of values for a variable over all the simulations done and at a specified time. In Vensim PLE Plus the Bar tool is configured to show sensitivity at Special Time. For other configurations you can adjust the settings.

Changing the Default Analysis Toolset

Vensim provides two default Analysis toolsets. *default1.vts* is what we have been working with so far. *default2.vts* contains more tools than *default1.vts*. You can also create and save your own toolset.

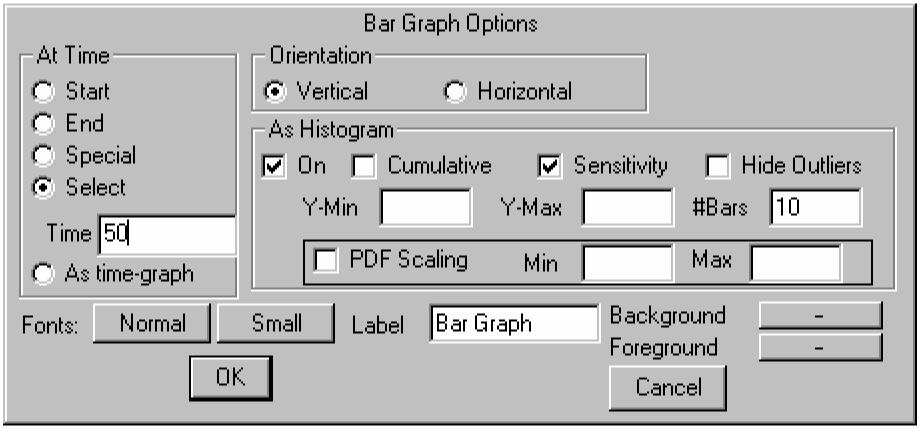
For Vensim PLE Plus simply click on the bar graph tool. Otherwise:

✍ Select the menu item **Tools>Analysis Toolset>Open…**.

✍ Click **No** when asked if you want to save the current toolset.

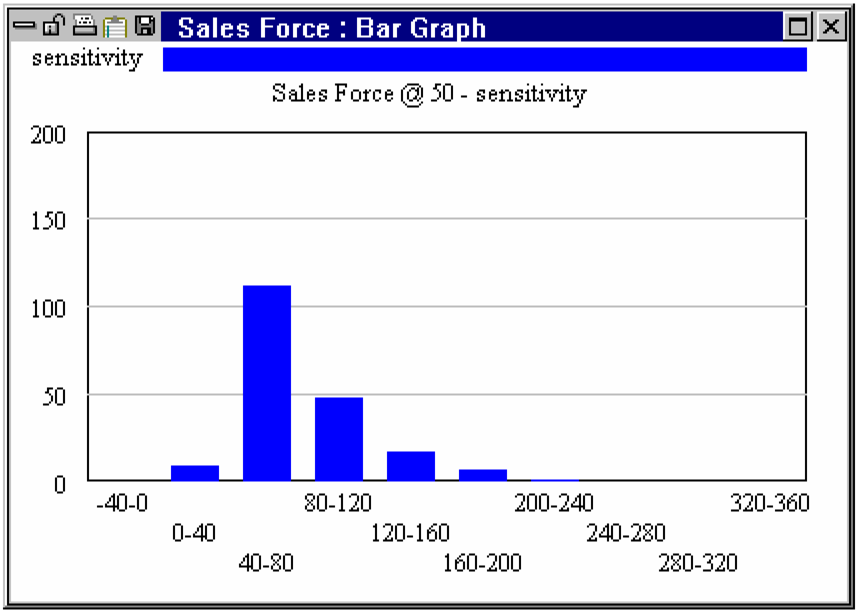
- ✍ Select *default2.vts* and click **Open**.
- ✍ Click with the right mouse button on the **Bar Graph** tool.
- ✍ In the **As Histogram** field, check the box **On**, check the box **Sensitivity**. In the **At Time** field, click the radio button **Select**, then type 50 in the **Time** box. Click **OK**.

The dialog should look like the one shown below.



- ✍ Select the Control Panel **Datasets** tab and check that *sensitivity* is still loaded. Check that `Sales Force` is still the Workbench Variable.
- ✍ Click on the **Bar Graph** tool.

A sensitivity histogram will be generated showing a cross section at the mid time of the simulation, 50 months.



`Sales Force` in people is shown along the X axis. The Y axis represents the number of simulations. Thus the graph says that there were about 115 simulations for which, at time 50, ***`Sales Force`*** was between 40 and 80 and about 50 simulations in which it was between 80 and 120.

Stats Tool

If you are using Vensim Professional or DSS the Stats tool can also be used to get information about a sensitivity simulation. See Chapter 14 of the Reference Manual for details on configuring this tool.

16 Using Data in Models

Vensim PLE does not support the direct connection to spreadsheets, but does support the use of data through importing. Note that this is usually done using Reference Modes as described in Chapter 20.

Types of Data Use

Vensim can use data in two ways. First, as exogenous inputs to drive models, and second, as a basis for comparing the behavior of a model with what actually happened.

Exogenous inputs are known as Data variables and are time series inputs which drive a portion of your model. Data variables are not computed during simulation, but instead refer to an existing data series (time series) for use during simulation. This data series can either be an imported Vensim dataset used by a **Data variable** in the model, or a time series residing in an external file such as an Excel or 123 spreadsheet used by a **Data variable** with a **Data function** in the model.

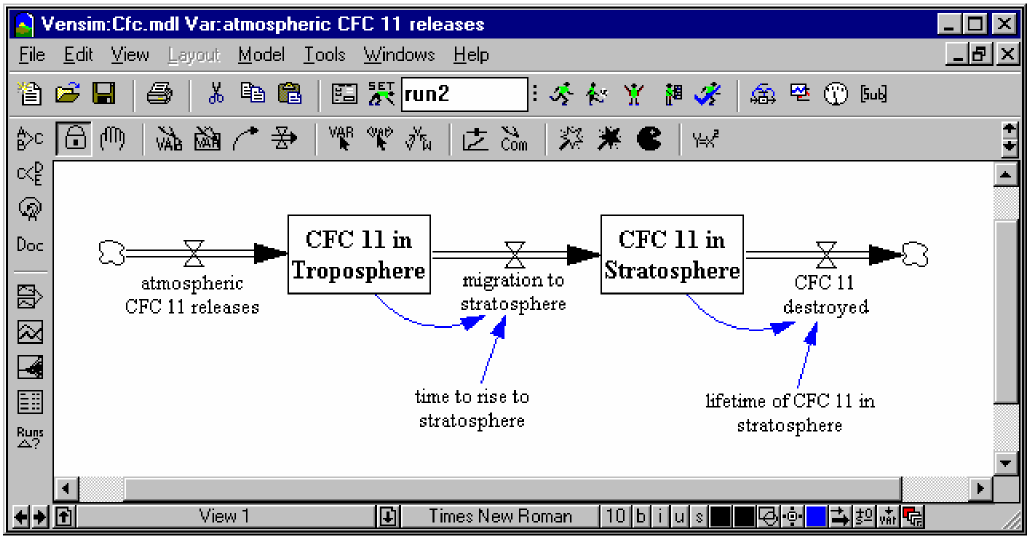
The second way to use data involves loading real world data as a Vensim dataset, then comparing the dataset against model behavior using the Analysis tools. You must have Microsoft Excel or Lotus 123 installed to complete this chapter.

Using Data to Drive a Model (cfc.mdl)

EITHER

✍ Open the model *cfc.mdl* in the directory *guide\chap16*.

✍ ***OR*** Build the model as shown in the diagram and equation listing below. Time Bounds are INITIAL TIME = 1930, FINAL TIME = 2130, TIME STEP = 0.5, Units for Time: Year.



cfc.mdl Equations

```
CFC 11 destroyed = CFC 11 in Stratosphere /
          lifetime of CFC 11 in stratosphere
Units: Mkg/Year

CFC 11 in Stratosphere = INTEG(
   migration to stratosphere - CFC 11 destroyed,
         0)
Units: Mkg

CFC 11 in Troposphere = INTEG(
   atmospheric CFC 11 releases - migration to stratosphere,

         0)
Units: Mkg

lifetime of CFC 11 in stratosphere = 55
Units: Year

migration to stratosphere = CFC 11 in Troposphere /
         time to rise to stratosphere
Units: Mkg/Year

time to rise to stratosphere = 5
Units: Year
```

This model is complete except for the variable ***atmospheric CFC 11 releases***.

✍ Save it in the directory *guide\chap15* with a different name (e.g., *mycfc.mdl*).

Getting Data from a Spreadsheet (Not PLE)

The simplest way to bring data into a model is to use a function to read the data directly from a Microsoft Excel or Lotus 123 spreadsheet. This technique also allows you to access any data source, such as a database, that can be accessed by the spreadsheet.

Select the **Equations** tool, click on the variable *atmospheric CFC 11 releases*.

✍ Click on the variable **Type** drop-down box (on the left side of the Equation Editor) and select the type **Data**. Click on the variable **Type** lower drop-down box (that currently says **Normal**) and select **Equation**.

✍ Click the **Functions** tab and in the **Function Class** drop down box, select **Data Only**. Scroll down the list of Data functions and select **GET XLS DATA** if using Microsoft Excel (or if using Lotus 123, select function **GET 123 DATA**).

The function is entered with four arguments:

'filename'

'tabname'

'time_row_or_col'

'first_data_cell'

IMPORTANT NOTE: you must enter these arguments *with* the single quotes surrounding them. It is also possible to use String variables to enter these, but entering the literals directly is often easier.

CFC emissions are included as time series data of both historical and forecast values in the directory *guide\chap16* in two files:

- ? *cfc11.xls* (Microsoft Excel spreadsheet)
- ? *cfc11.wk4* (Lotus 123 spreadsheet)

✍ Type in the filename 'cfc11.xls' (don't forget the single quotes!) in the first argument if using Microsoft Excel, or 'cfc11.wk4' if using Lotus 123.

✍ Double click the second argument, type in the tab name 'cfc11'.

✍ Double click the third argument, type in the row name '2' (the row for time)

✍ Double click the fourth argument, type in the cell name 'C4' (the starting cell for data)

✍ Choose units of *Mkg/Year* and click **OK** to close the Equation Editor.

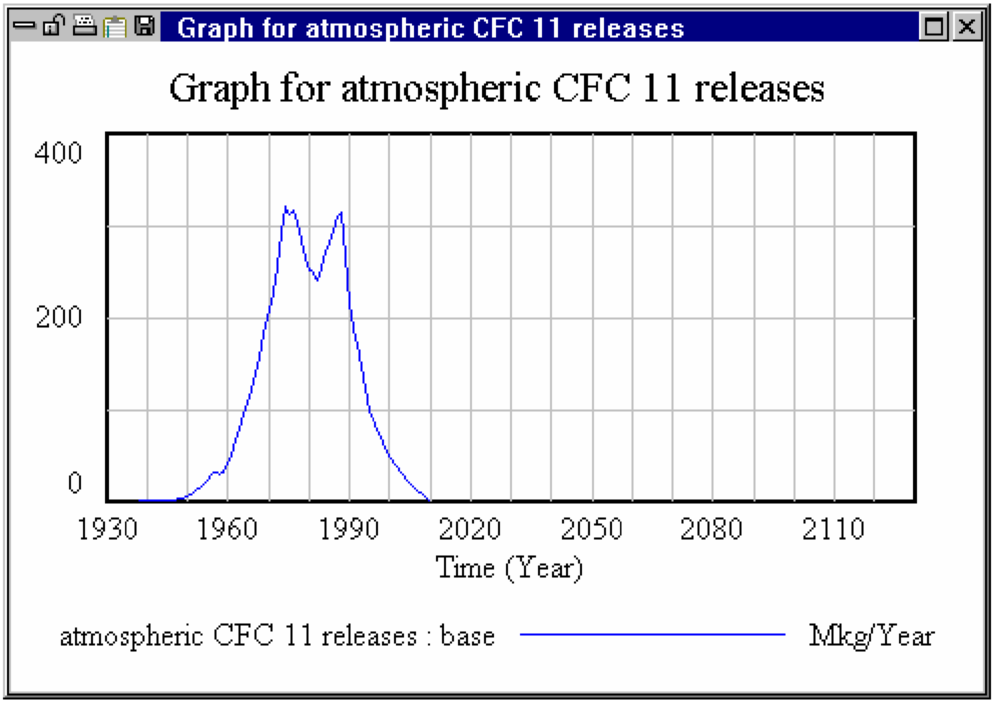
Simulating

Newer versions of Excel and 123 should start automatically when called from the Vensim GET DATA function; however older versions might need to be started before you simulate. Some versions of 123 might need the *cfc11.wk4* file open in the 123 application.

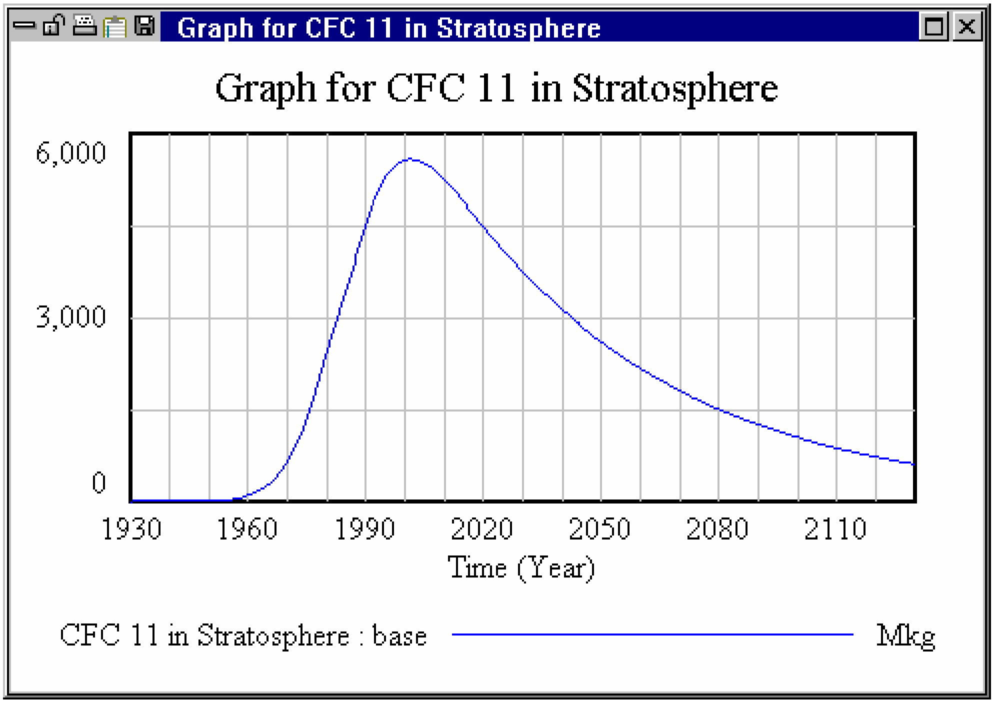
✍ Type in a run name (e.g., *base*) and click the **Simulate** button.

Vensim should automatically open the spreadsheet and simulate while reading values off the file *cfc11*. The model starts simulating at 1930, eight years before CFC 11 releases occurred, and Vensim reports an error because our model starts *before* the driving data. We could have entered zero in the data series for years prior to 1938, but we took the data directly from the data source.

✍ Select ***`atmospheric CFC 11 releases`*** as the Workbench Variable and create a graph:



✍ Create graphs of the two levels. Note the long delay present which results in significant CFC 11 in the stratosphere for the next 100 years.



Note that this simulation has an optimistic assumption about `atmospheric CFC 11 releases`. We will explore the implications of this further below.

Data Variable with Imported Data

Instead of directing Vensim to go to a spreadsheet for data you can tell it to look at datasets to get values. Such Data variables use the subtype "Normal," though that is a rather arbitrary designation. Datasets are created by running simulations, or by importing data from other sources. The dataset must be created or imported before the model will simulate.

✍ Save the model *cfc.mdl* as another name, e.g. *cfc2.mdl*.

✍ Select the **Equations** tool, click on the variable `atmospheric CFC 11 releases`. Click on the variable **Type** lower dropdown box (currently says **Equation**) and select **Normal**. Click **OK**.

When a simulation is run, Vensim looks for a loaded dataset containing the named Data variable. If no dataset is found, the simulation is halted.

CFC emissions are included as time series data of both historical and forecast values in the directory *guide\chap16*. The historical data values are contained in two files:

? *cfc11.dat* format (Vensim data format text file)

? *cfc11.tab* format (tab-delimited text file)

Unlike the spreadsheet data, these files do not contain any projected values. We will use the *cfc11.dat* data file to import a dataset into Vensim.

✍ In Vensim Professional or DSS, select **File>Edit File…** and choose *cfc11.dat* then click **Open**.

OR

✍ Open and examine the ASCII text file *cfc11.dat* using a text editor or word processor.

Opening the file is optional; the only point is to review the data. You will see the data displayed in the format below, with the variable name followed by one column for time and one column for values:

```
atmospheric CFC 11 releases
1938     0.1
1939     0.1
(more...)
1990     216.1
1991     188.3
1992     171.1
```

✍ Close the *cfc11.dat* file.

Importing Text-Formatted Data (.dat)

We need to import the data series by converting the text file *cfc11.dat* to a binary Vensim data file (*cfc11.vdf*).

✍ Select the menu **Model>from .dat format…**. Choose the file *cfc11.dat* and click **Open**.

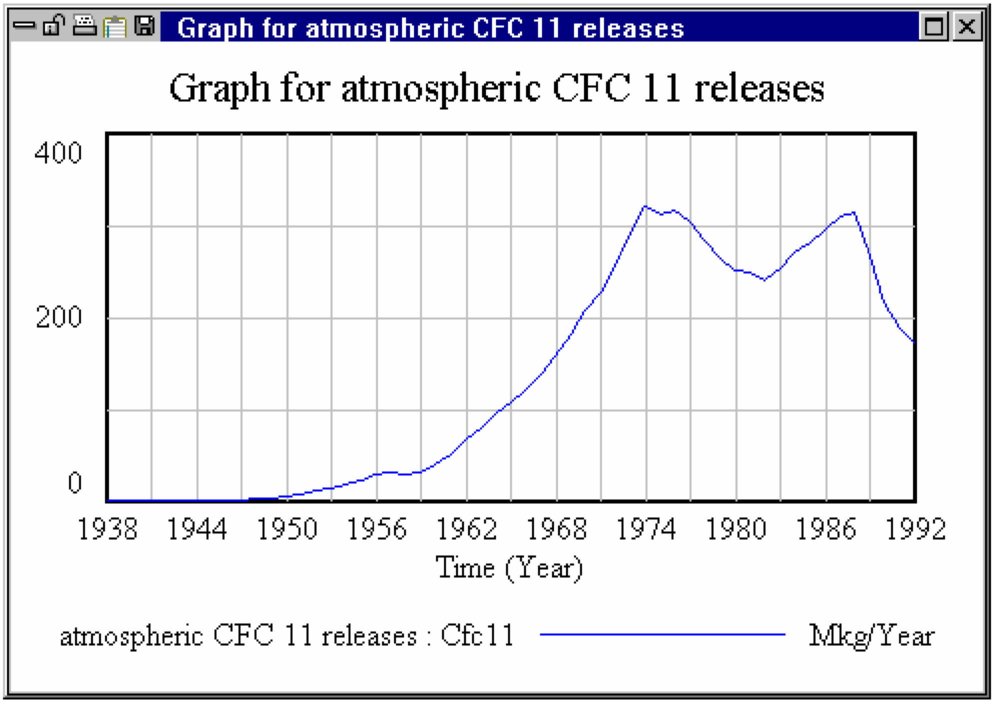
You should get the message "Dat2vdf completed without error" and an Output window will show that a number of values were written for `atmospheric CFC 11 releases` and for time base `Time`. The dataset *cfc11.vdf* is loaded as the first dataset. Analysis tools will now work on this dataset.

✍ Click **OK** to the message box and close the Output window.

✍ Open the Control Panel and click the **Datasets** tab, double click on the previous simulation dataset *base* to remove it, leaving only the dataset *cfc11*.

✍ Double click the variable `atmospheric CFC 11 releases` to select it as the Workbench Variable and then click on the **Graph** tool.

A graph is displayed for `atmospheric CFC 11 releases` over the time base of the dataset.



- Close the graph by clicking the Close button or pressing Del.

Notice that this output runs only through 1992. The projected releases were not included in the input dataset.

Simulation

Data variables in a model need data sources. These data sources are the dataset files that we imported and they need to be specified before simulation can occur. One place to set the data source is in the Simulation Control dialog; the other place is on the Toolbar after the **Set Up a Simulation** button has been pressed.

Toolbar

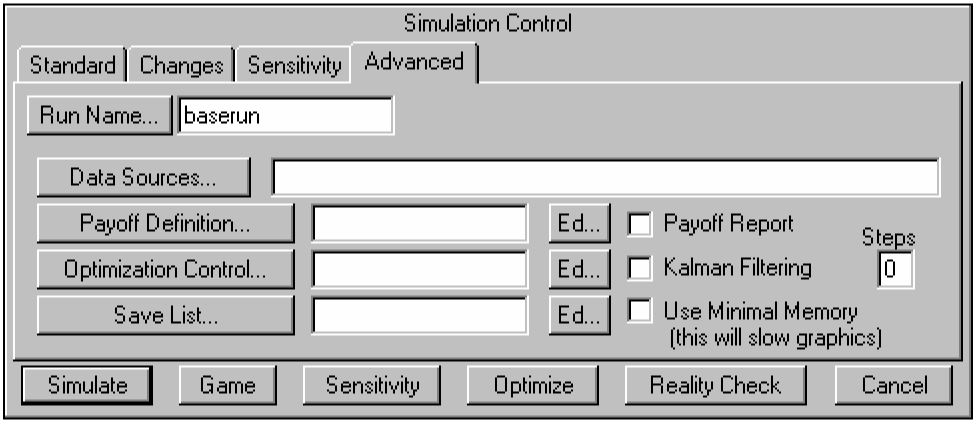
- Press the **Set Up a Simulation** button.
- Type in a run name (e.g., *baserun*.)
- Type in the name of the converted dataset, *cfc11*, in the left hand editing box (the Data editing box just right of the integration method **Euler**) (or press the little button right of the editing box to choose the dataset).

- Press the **Simulate** button.

Another way of specifying a dataset for use in simulation data is to use the Simulation Control (below).

Simulation Control — Optional (Not PLE or PLE Plus)

✍ Click the **Simulation Control** button then click the **Advanced** tab.



The **Advanced** tab contains settings for data files and optimization of models.

✍ Type in a name for the run (e.g., *baserun*).

✍ Click on the **Data Sources…** button then click on the dataset *cfc11.vdf* then click **Open** (or type the name *cfc11.vdf* into the editing box right of **Data Sources...**).

✍ Click the **Simulate** button.

You will receive 3 warnings:

```
WARNING:     Simulation starts before first time for exogenous
   variable - atmospheric CFC 11 releases.
WARNING:     Exogenous variable - atmospheric CFC 11 releases - in
   range at time 1938.
WARNING:     Simulating beyond range for exogenous variable -
   atmospheric CFC 11 releases - at time 1994.
```

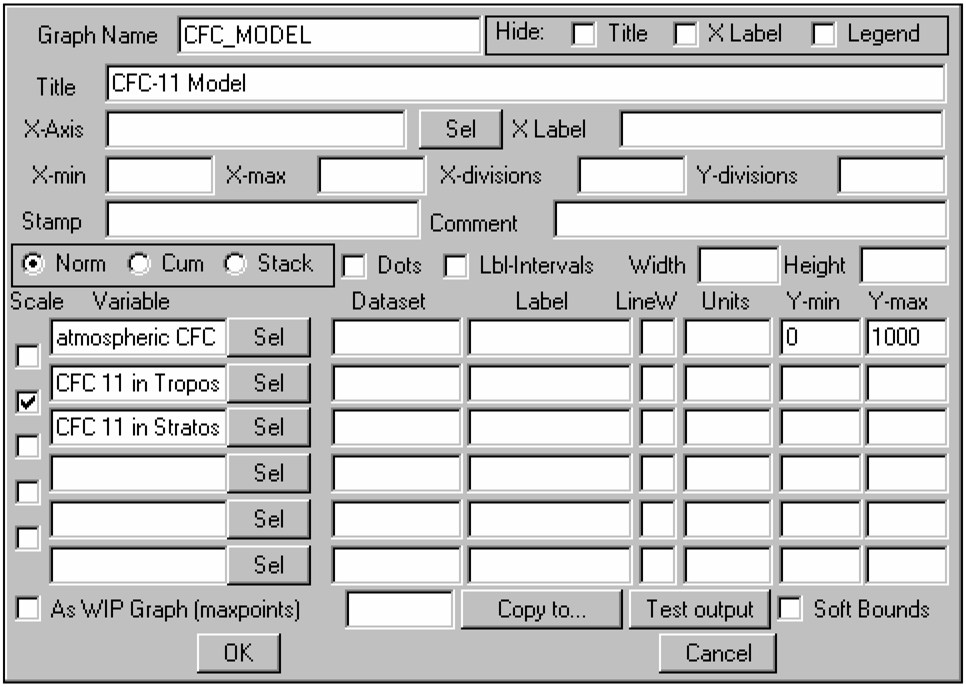
When the simulation is outside of the range of the data the closest value of the data will be used. This means that before 1938 the first value for ***`atmospheric CFC 11 releases`*** will be used and that in 1994 and beyond the last value will be used.

Results

✍ Click on the **Control Panel** button, select the **Datasets** tab and double-click on the dataset *cfc11*.

Because the model used the *cfc11* dataset during simulation to provide data for one variable, the dataset *baserun* also stores the data for this variable.

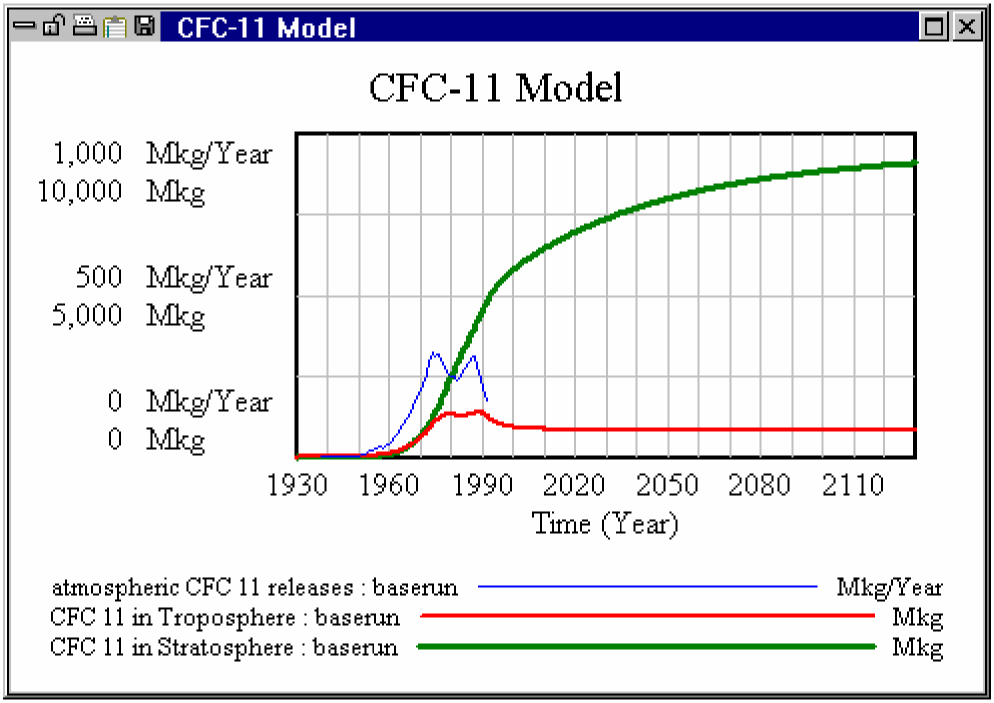
If you have created your own model you should add the following Custom Graph definition:



✍ Click the **Graphs** tab and then click on the button **Display** to show the Custom Graph.

OR

✍ Look at the behavior of the rates and levels in this model with the **Graph** and **Causes Strip graph**.



The input data series ***atmospheric CFC 11 releases*** ends at time 1992 and this means that the last figure in the series (171.1) is used in calculations for any time greater than 1992. Because of this, the ***CFC 11 in Troposphere*** value stabilizes after about the year 2000, while ***CFC 11 in Stratosphere*** keeps climbing.

Tthe consumption and release of CFC 11 is likely to continue to decline over the following years. Releases of CFC 11 are not likely to continue at a fixed rate as high as 171.1 Mkg/Year. Let's open a data series that contains an optimistic forecast for CFC emissions.

Importing Spreadsheet Data (Not PLE)

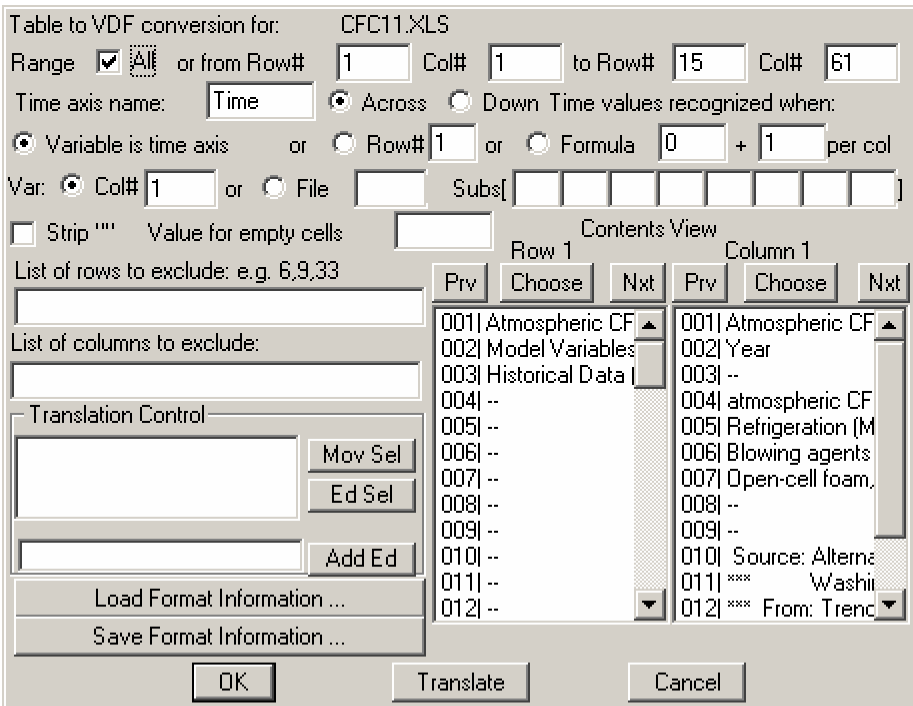
✍ If you have a spreadsheet application (e.g., Lotus 1-2-3 or Microsoft Excel), open one of the files *cfc11.wk1*, or *cfc11.xls* and examine the data.

A portion of the spreadsheet file is shown below:

Model Variables	Historical Data (see Source, below)			
	1938	1939	1940	1941
atmospheric CFC 11 releases	0.1	0.1	0.1	0.1
	0.1	0.1	0.1	0.1
	0	0	0	0
	0	0	0	0

✍ Select the menu **Model>Import Dataset…**. Choose the file *cfc11.xls* or *cfc11.wk1* and click **Open**.

A Table to VDF conversion dialog box opens.



We need to set the import options for Ve nsim to properly read the time data, the variable data, and the variable names. First we need to set the range of cells that encompasses all the data, but nothing else.

✍ Click on the editing box Range: **from Row#** and type in 2.

✍ Click on the editing box Range: **Col#** and type in 3.

✍ Click on the editing box Range: **to Row#** and type in 4.

✍ Check that the Range: **Col#** editing box reads 61.

We do not have a variable named `Time`, so we need to specify from which row to read the time values.

✍ Click on the editing box for **Row#** and type in 2 (same line as **Variable is time axis**)

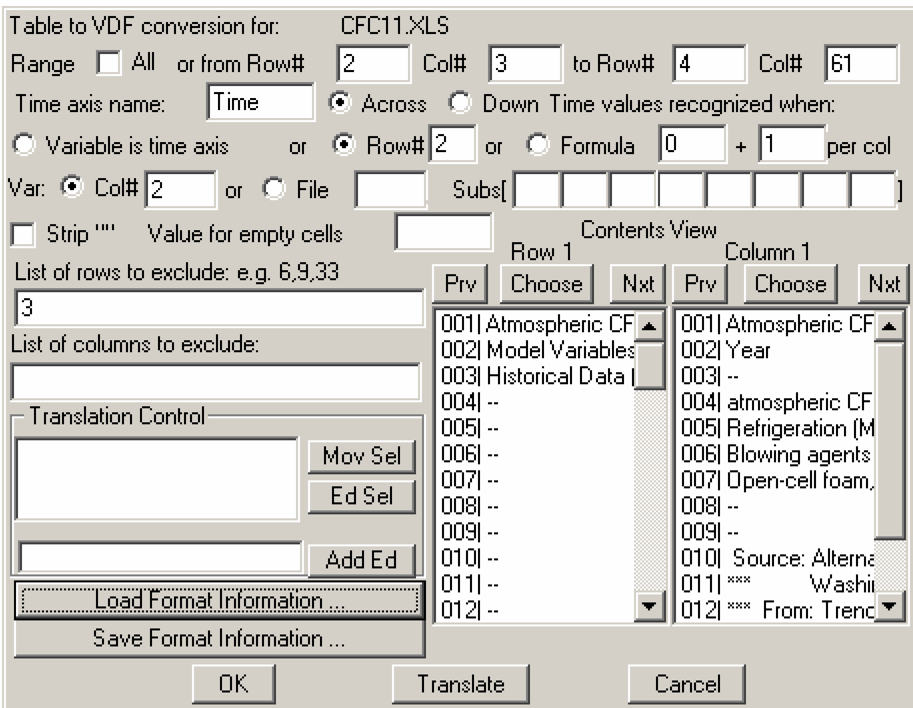
The option button for **Variable is time axis** should turn off and the button for **Row#** should turn on. The default position for the variable name is column #1. The model variable names are actually in column #2, so we need to specify this. We also need to exclude Row 3 of the spreadsheet because it does not contain any data.

✍ Click on the **Var: Col#** editing box and type in 2.

✍ Click on the editing box for **List of rows to exclude** and type in 3.

✍ Click on the button **Save Format Information…** Type in the name *cfc11* and click the **Save** button.

You can now load this format information in the future if you want to convert the same data file again. The Table to vdf conversion dialog box should now look like:



✍ Click the OK button in the Table to VDF conversion dialog box.

You will get the message "Dataset cfc11 already exists. Do you want to overwrite it?"

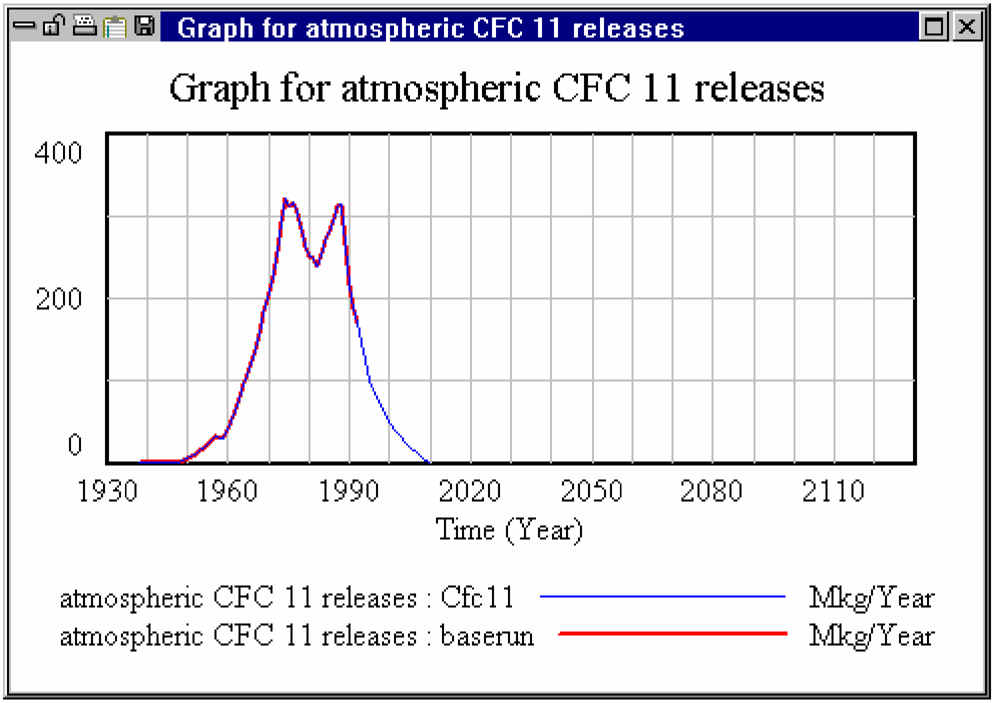
✍ Click **Yes**

You should get the message "Conversion to vdf completed without error" and an Output window showing what was done should open.

✍ Click **OK** to the message box and close the Output window.

✍ Double click on the variable `atmospheric CFC 11 releases` and then click on the **Graph** tool.

Now the graph of the `atmospheric CFC 11 releases` shows a drop to zero in dataset *cfc11* (dataset *baserun* has a line exactly under this, except for the values later than 1992).



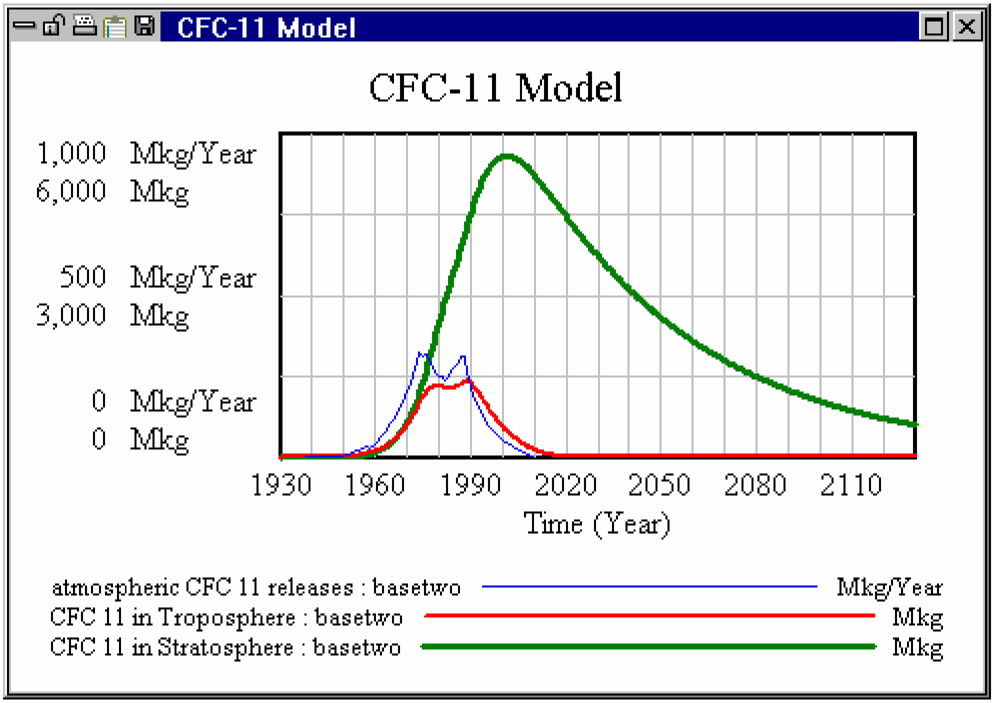
Simulation

✍ Click the **Set Up a Simulation** button. Type in a name for the run (e.g., *basetwo*).

Note that the dataset *cfc11* is still listed in the **Data** editing box.

✍ Click the **Simulate** button.

✍ Click the **Control Panel** button then click the **Datasets** tab and double-click on the dataset *cfc11* and the dataset *baserun* to unload them. Click the **Graphs** tab and click the button **Display**.



Now we see that stopping releases of CFC 11 greatly reduces the amount of ***CFC 11 in Stratosphere***. However, there is a considerable delay. The peak value for ***CFC 11 in Stratosphere*** occurs in the year 2002, even though ***atmospheric CFC 11 release*** began to decline starting in the year 1988. Moreover, the value for ***CFC 11 in Stratosphere*** exceeds its 1988 value until about the year 2060. This model is quite simple and does not represent a complete picture of CFC release and ozone depletion. But it does demonstrate quite clearly how long it can take for the consequences of current activities to be fully realized.

17 Subscripts

Subscripts are only supported in Vensim Professional and DSS only. Models with subscripts can be used with the Vensim Model Reader.

Introducing Subscripts

A Subscript allows one variable and equation to represent a number of different distinct concepts. For example consider the equation:

```
profit = revenue - cost
```

If we are modeling a chain store we might be interested in activities at each store. Thus we could define `store` as a Subscript

```
store : WAYLAND, HARVARD, LONGMEADOW, OLALLA, OAK RIDGE, LAS VEGAS
```

Each name describes a different store (in this case by location). Now we can write the equation

```
profit[store] = revenue[store] - cost[store]
```

This equation says that for *each* store, profit *specific to that store* is computed by taking revenue specific to that store and subtracting cost specific to that store. Thus instead of writing six equations, you need only write one. In addition, if you add another store, you do not need to rewrite the equation.

Variables in a model may have different Subscripts, or no Subscripts. For example, we might have:

```
corporate profit = SUM(profit[store!]) - corporate overhead
```

This equation adds up `profit` across all stores and then subtracts `corporate overhead`. The point here is that you can mix and match subscripts and regular variables, and this usually turns out to be necessary.

A Simple Project Model (proj1.mdl)

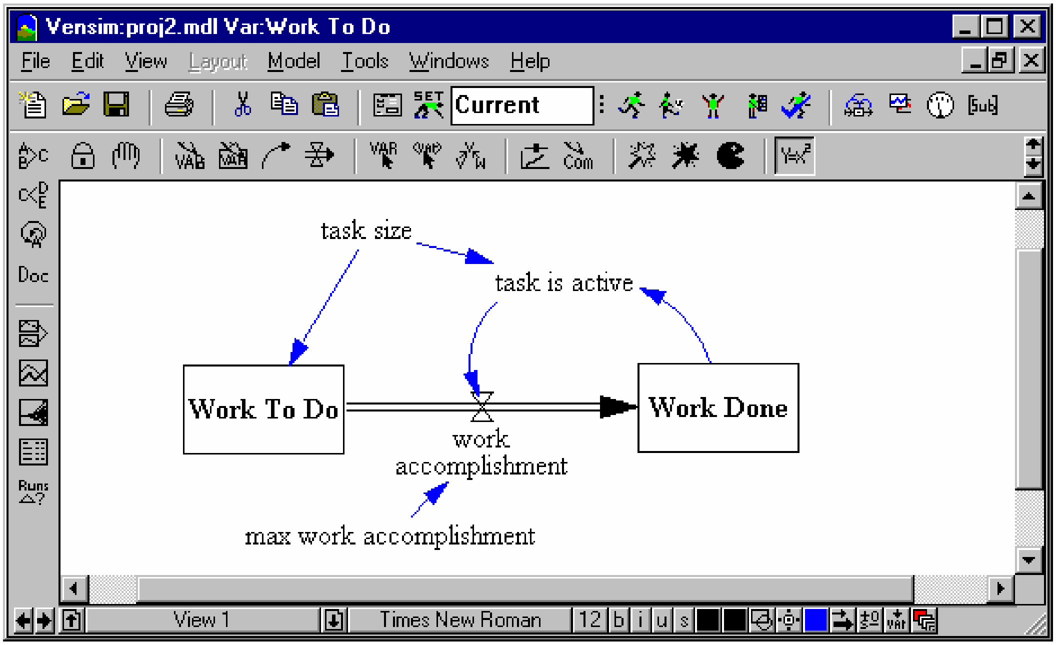
This project model will demonstrate the completion of a project that contains three separate tasks, all modeled with the same structure. *proj1.mdl* models only one task. This model will be saved as *proj2.mdl* and modified to take subscripts representing the three tasks.

EITHER

✍ Open the model *proj1.mdl* contained in the directory *guide\chap17*

OR

✍ Sketch the model structure and enter the equations as shown on the next page (INITIAL TIME = 0, FINAL TIME = 100, TIME STEP = 0.125, Units for Time: Month).



proj1.mdl Equations

```
max work accomplishment=8
Units: Drawing/Month

task is active=
   IF THEN ELSE( Work Done < task size, 1, 0)
Units: Dmnl

task size=180
Units: Drawing

work accomplishment=
   max work accomplishment * task is active
Units: Drawing/Month

Work Done = INTEG(
   work accomplishment,
         0)
Units: Drawing

Work To Do = INTEG(
   - work accomplishment,
         task size)
Units: Drawing
```

✍ Mark `Work To Do` as a Supplementary variable in the Equation Editor.

✍ If you like, make a simulation run to examine the behavior of the model. You will find that `Work Done` (a single task) increases linearly from zero to a final value, then stays constant.

✍ Select menu **File>Save As…** and type in the name *proj2*.

Project With Multiple Tasks (proj2.mdl)

The Subscript Range

First we need to make a new variable that describes the concept of having different tasks. This is called the Subscript Range and a suitable name in this case is `task`.

✍ Click on the **Subscript Control** button on the Toolbar. The Subscript Control opens:

✍ Click on the **New…** button, in the dialog box type the name `task` then press Enter.

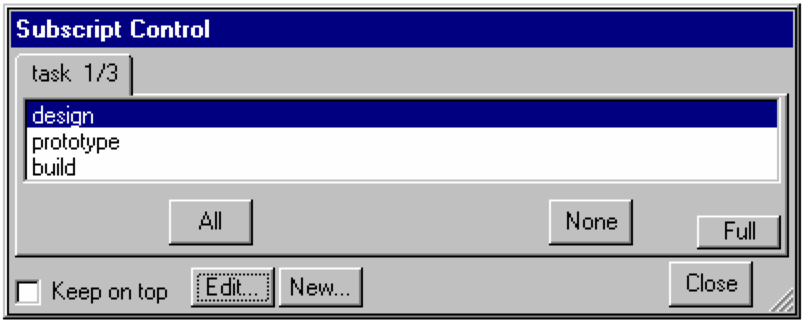
The Equation Editor opens to display the incomplete equation for `task`. This is a Subscript Range and it describes the three tasks: design, prototype, and build. The range `task` does not *equal* these tasks, it *represents* the tasks. Note that the Equation Editor displays the Subscript symbol `:` left of the equation editing box.

✍ Fill in the three tasks, as below (you don't need to type **task :** that is already entered) then click on the **OK** button.

```
task :
   design, prototype, build
```

The three tasks should now be showing in the Subscript Control. Note that the subscript range `task` and the three subscripts do not appear in the sketch. The subscript range and subscripts are part of the definition of the model, not part of the diagrammatic structure.

The Subscript Control



We see the three Subscript Elements appearing in a list box with a tab that displays the Subscript Range name (`task`). The first Subscript Element (`design`) is highlighted. Analysis tools will operate only on the selected (highlighted) Subscript Elements in a Range. To toggle the highlight status of a Subscript Element, click on it. You can also use **All** to highlight everything and **None** to highlight none. You can edit the subscript range in the Equation Editor at any time by clicking the **Edit…** button.

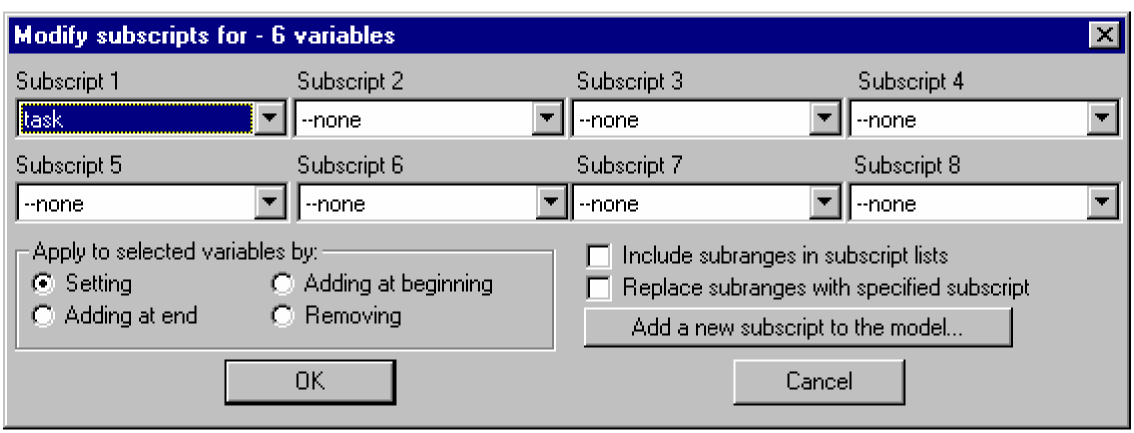
The **Full** button provides a more elaborate control designed for Subscript Ranges with a large number (> 12) elements. Details on the Subscript Control are contained in Chapter 12 of the Reference Manual.

Adding Subscripts to Variables

When you add subscripts to variables, you need to first decide which variables get which subscript. For this model it makes sense to add subscripts to all the variables defined so far.

✍ Click on the sketch, or press the **Build** windows button on the Toolbar. Highlight all of the variables in the model using the menu item **Edit>Select All** or Ctrl+A.

✍ Select menu item **Edit>Set Subscripts** to open the Modify Subscripts dialog.



✍ Click on the first dropdown box (for Subscript 1) and select the **task** (the two choices are **--none** and **task**).

✍ Click on **OK**.

All of the variables in the model will now appear with the subscript *task*. Note that the sketch view does not show these subscripts. The subscripts will appear in the title bar when you select a variable into the workbench and you can view the equations to see them.

✍ Select the **Equations** tool and click on *work accomplishment*. The equation reads:

```
work accomplishment[task]=  max work accomplishment[task]*
      task is active[task]
```

Subscripts appear in Vensim Equations inside of square brackets [ ]. If there is more than one subscript for a variable, the different subscripts are separated by commas (e.g., *inventory[store,item]*).

The most important subscript rule is that a subscript appearing on the right hand side of an equation must also appear on the left hand side. After using the **Edit>Set Subscripts** command, this might not be the case and when you check the model or try to simulate, an error will be reported. For this particular model, this is not a problem.

✍ Click on **Cancel** to close the Equation Editor.

Constant Equations

With the modifications we have made so far, all of the equations in the model are the same. In particular, the equation for *task size* is

```
task size[task]=180
```

This equation says that each *task* has size 180. Part of the purpose of subscripts is to allow different elements to have different values. This can be done by writing multiple equations for a variable as will be discussed in a later section. For constants this can be done simply by separating a series of numbers by commas.

✍ Open the Equation Editor on *task size*. Replace the old equation (180) with the equation below, then click on **OK**.

```
task size[task]=
   180, 250, 500
```

✍ Repeat this for *max work accomplishment* as below:

```
max work accomplishment[task]=
   8, 10, 15
```

Vector Functions

Now we will add a variable to the model that describes the sum total of the work done for all tasks.

✍ Select the **Variable** tool. Click on the sketch below *`Work Done`* and type in the name *`total work done`* then press Enter.

✍ Select the **Arrow** tool and click once on *`Work Done`* and then on *`total work done`*.

✍ Open the Equation Editor on *`total work done`*.

✍ Click on *`work done`* in the **Variables** list.

When you do this it will be added with its subscript so that the `equation will read:`

```
total work done = work done[task]
```

This equation is not complete. If you were to check the model at this point you would get an error message saying *`task`* appears on the right but not the left. *`work done`* must contain the subscript *`task`*, but *`total work done`* does not require the subscript because it will add all the tasks together. For this, we use the SUM function.

```
total work done = SUM(work done[task!])
```

The exclamation point ! indicates which Subscript Range we want to sum over. In this case there is only one subscript, but there can be up to eight subscripts for a variable and it is important to indicate which subscripts the SUM function works on.

✍ Enter the SUM function and subscripts as they appear in the equation above. Enter the units *`drawing`*. Click on the checkbox **Supplementary** on the left hand side of the Equation Editor. Don't forget the exclamation mark ! Click **OK**.

Subscripts in the Equation Editor

In addition to using the **Edit>Set Subscripts** command, you can add subscripts to variables using the Equation Editor. You add subscripts to a name appearing in the equation by positioning the cursor at the end of the name and then clicking on a subscript name in the **Subscripts** tab. Vensim will automatically add square brackets or commas to get an appropriate syntax. The **Subscripts** tab only appears when a model contains subscripts. You can also just type in the subscript names.

The full set of equations for this model should now read:

```
task : design,prototype,build

max work accomplishment[task] = 8, 10, 15
Units: Drawing/Month

task is active[task] = IF THEN ELSE (
               Work Done[task] < task size[task] , 1, 0)
Units: Dmnl

task size[task] = 180, 250, 500
Units: Drawing

work accomplishment[task] = max work accomplishment[task] *
         task is active[task]
Units: Drawing/Month
```

```
Work Done[task] = INTEG( work accomplishment[task] , 0)
Units: Drawing

Work To Do[task] = INTEG(-work accomplishment[task],task size[task])
Units: Drawing

total work done = SUM ( Work Done[task!] )
Units: Drawing
```

Subscript Summary

Note that in all equations except the last, a subscript appears on left of the equals sign. For the Constants, the subscript `task` appears on the left, and three values (one for each task) appear on the right. For most Auxiliaries, the subscript `task` appears on both the right and the left. Things that are differentiated by `task` need to be caused by things that are differentiated by `task`.

An exception to this is the last equation. Here, `task` appears only on the right. The variable `total work done` is a single number representing the sum of all three task's work done. The subscript function SUM tells Vensim to add together all the values for the variable with the subscript that is marked by an exclamation point `(!)`. Sometimes more than one subscript range appears in an equation, so the exclamation point is used to denote the subscript range being acted on (here by the SUM function).

✍ Select the menu **Model>Check Model** (Ctrl + T).

You should get the message "Model is OK". If you don't, you will probably get a message "Incorrect number of subscripts for …(variable name)." The Equation Editor will open and position itself at the incorrect equation. Check that the equation exactly matches the equations above. If that equation matches, check the other equations. Modify, click **OK**, then do **Model>Check Model** again.

Simulation and Analysis

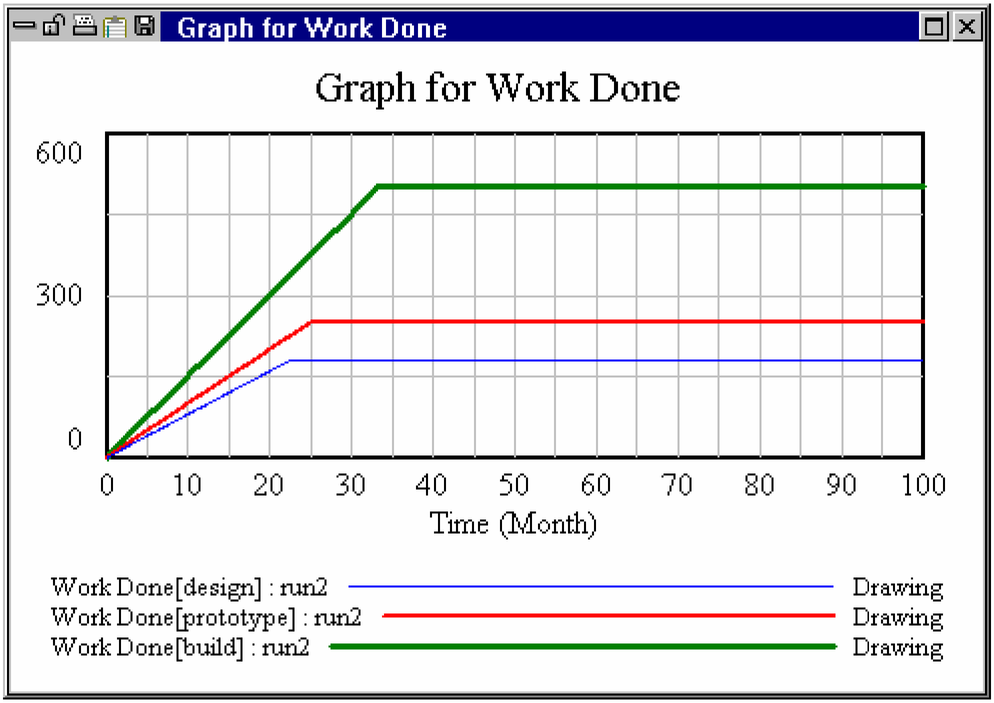
✍ Double click on the **Runname** editing box and type a name for the dataset, e.g., *run2*. Click the **Simulate** button.

After the model simulates use the **Datasets Control** to unload any previous runs then:

✍ Double click on `Work Done`, then click on the Analysis tool **Graph**.

Output is generated showing the behavior of `Work Done` for one subscript: `design`.

✍ Click the **Subscript Control** button on the Toolbar. The Subscript Control opens. Click on the button **All**, then click again on the **Graph** tool.



Because all subscripts are selected, the work done for each task is graphed. Any Analysis tool you invoke will generate output that will reflect which subscripts are selected in the subscript control.

✍ Click on ***total work done*** then click on the **Causes Strip** graph.

Output is generated showing the behavior of the variable ***total work done*** and its causes, ***Work Done*** for each of the three tasks. In this case, the number of subscripts selected is irrelevant: all are shown because all are causes of ***total work done***.

Advanced Subscript Model (proj3.mdl)

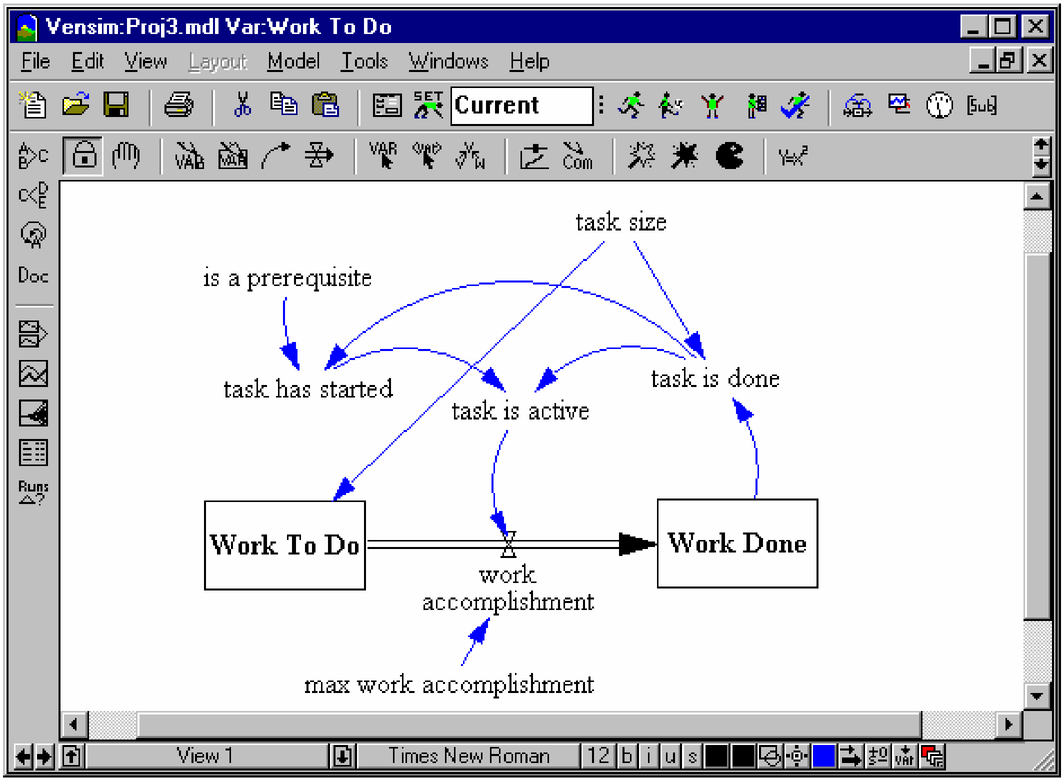
This project model will demonstrate the completion of a project that contains three separate tasks. Each task must be finished before the next begins, that is, each task is a prerequisite for the following task. The structure of completing each task is exactly the same for all tasks.

EITHER

✍ Open the model *proj3.mdl* in the directory *guide\chap17\complete*. Use Save As to save the model in *guide\chap17* or whichever directory you are working in. You can skip to Simulation and Analysis below or work through the model development inspecting but not changing model equations.

OR

✍ Click the **New Model** button. Set the TIME STEP to 0.125 then click **OK** to accept the other default Time Bounds. Sketch the model below and save it to *proj3.mdl*.



✍ Select the **Hide Wand** tool and click on the arrowhead of the arrow from `task size` to `Work To Do` to hide the arrow (because it only initializes the Level).

Subscript Range

First we need to make a subscript range to describe the tasks.

✍ Click on the **Subscript Control** button on the Toolbar.

✍ Click on the **New…** button, in the dialog box type the name `task` then press Enter.

The Equation Editor opens to display the incomplete equation for `task`.

✍ Fill in the three tasks, as below, then click on the **OK** button.

```
task :
   design, prototype, build
```

✍ Select the **Equations** tool and open the Equation Editor on `task size`.

✍ Type in the numbers 180, 250, 500

✍ Click on the top edit box to the right of the name *task size*. Now click on the **Subscripts** tab and click on *task* (the only entry in the list). Vensim will add the square brackets [ ]. Your equation should read:

```
task size[task]=
   180,250,500
Units: Drawing
```

✍ Continue entering the other equations.

You will need to alternate between the **Variables** and **Subscripts** tabs. You will notice that once you have added a subscript to a variable when you click on its name in the **Variable** tab it will be placed in the equation with its subscript. Thus, for each variable, you only need to add the subscript once when you write the equations. Enter the equations as shown below.

```
Work To Do[task] = INTEG(
   - work accomplishment[task],
         task size[task]
Units: Drawing
```

✍ Check the **Supplementary** check box for *Work To Do*.

```
Work Done[task] = INTEG(
   work accomplishment[task],
         0)
Units: Drawing
```

```
work accomplishment[task]=
  IF THEN ELSE(task is active[task],max work accomplishment[task],0)
Units: Drawing/Month
```

The equation for *work accomplishment* uses the logical IF THEN ELSE function, getting work done for each task at the maximum work accomplishment as long as that task is active.

```
max work accomplishment[task]=
   8,10,15
Units: Drawing/Month
```

The four remaining variables describe the relationships of the tasks to each other. If a task has no prerequisite tasks, then it may start immediately. If a task has prerequisites, those prerequisites must finish before the task can start.

Mapping Subscripts

A task can be either (1) a task alone, or (2) a prerequisite task for another task. To be able to describe in one equation the concept of both tasks and prerequisites (which are also tasks), we define another subscript range — *prereqtask* — which is exactly the same as the other subscript range (`task`). This process is called mapping. For this project model, each task is a potential prerequisite for some other task, so we define another subscript range as equivalent to the first, using the mapping symbol `<->`.

✍ Click on the **Subscript Control** button on the Toolbar. Click on the **New…** button, in the dialog box type the name `prereqtask` then press Enter.

✍ Click on the variable **Type** lower drop down box that currently says **Normal** and choose **Equivalence**.

The Equation Editor displays the Equivalence symbol `<->` left of the editing box.

✍ Type in the name `task` in the equation field. It should appear without any brackets.

✍ Click **OK** to close the Equation Editor.

You will notice that Vensim did not add a new tab to the Subscript Control. An Equivalence is treated as a Subrange. Thus `prereqtask` is a Subrange of `task` and does not get a separate tab. If you want to edit the equation for `prereqtask` you need to use the **Choose** button in the Equation Editor.

Multiple Equations for a Subscripted Variable

A task can have a prerequisite, but it can not be a prerequisite of itself. For the 3 tasks (design, prototype, build), we could write out a set of equations (3 for each of the 3 tasks, 9 equations total) that tell us which task is a prerequisite of another. For example, for the task prototype:

```
is a prerequisite[prototype,design] = 1
is a prerequisite[prototype,prototype] = 0
is a prerequisite[prototype,build] = 0
```

These three equations tell us that for the task prototype, design is prerequisite (value = 1) but prototype and build are not prerequisites (value = 0). These three equations for prototype can be combined into a more compact form:

```
is a prerequisite[prototype,prereqtask] = 1,0,0
```

Note that the order of prerequisite tasks is the same as tasks, so that the value 1 refers to design, 0 refers to prototype, etc. However, we need three equations in total (although you can write one equation with a 2 dimensional array to describe the prerequisites — see the Reference Manual).

✍ Click on the variable `is a prerequisite`. Position the cursor just after the variable name.

✍ Click on the dropdown arrow for the list box on the **Subscripts** tab (currently says **Range**). Select the subscript range **task - elements** from the dropdown list. The three tasks appear in the list box below.

✍ Click on the subscript `design` appearing in the list box. This enters the subscript into the equation at the cursor placement. Position the cursor just after the word `design` appearing in the equation and *before* the square closing bracket "] ".

✍ Click on the drop-down arrow for the list box (currently says **task - elements**). Select **Range** from the drop-down list. The list box now says **Range** and the two subscript ranges are listed in the box below.

✍ Click on the range `prereqtask`.

- Move the cursor to the equation editing box (after the equals sign) and type in the numbers and commas 0,0,0 and type in units, as below:

```
is a prerequisite[design,prereqtask]=
   0,0,0
Units: Dmnl
```

This is the first of three equations describing which tasks are prerequisites of the others. To make the next two equations:

- Click on the **Add Eq** button at the top right of the Equation Editor.

A second equation is added for the variable `is a prerequisite`. If you have exited and reopened the Equation Editor Vensim will automatically add the subscripts [task,task]. This is of the same form as the previous equation, since `design` is a task, and `prereqtask` maps to task. If you are just following the steps above `is a prerequisite` will appear without any subscripts.

- If there are any subscripts on `is a prerequisite`, remove them.
- Fill in the equation as below by either typing in the new subscripts, or selecting them from the subscript list box.

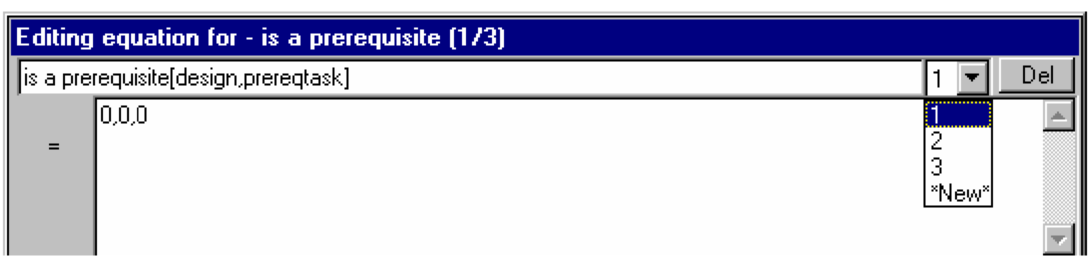
```
is a prerequisite[prototype,prereqtask]=
   1,0,0
```

Note that the units will already be in place. There is only one units definition for a variable.

- Click on the drop down box at the top right that currently says **2** (for equation 2) and click on ***New*** then fill in the equation as below, either typing in the new subscripts, or selecting them from the subscript list box:

```
is a prerequisite[build,prereqtask]=
   1,1,0
Units: Dmnl
```

You can navigate between these three equations by clicking on the same drop down arrow number list that is used to select ***New***. As you add equations, the equation number will be added to this list and the Title bar will display this as well. Do not hit **Del** unless you want to delete the current equation.



- Click on **OK** to close the Equation Editor.

NOTE There is a more compact way to write 2 dimensional arrays. The following single equation has the same meaning as the three equations you wrote for `is a prerequisite`.

```
is a prerequisite[task,prereqtask] = 0,0,0;1,0,0;1,1,0;
```

We did not use this equation because we wanted to demonstrate how to write multiple equations for one variable. For this example we could have used the 2 dimensional array, but in many models it is not possible to do so. The multiple equation concept applies not only to Constants but to all types of model variables.

Logical Functions and Operators

The task is completed when the work is done for that task. In a more complete model, we would want to allow a task to be done at some fraction of ***task size*** for the purposes of determining prerequisite readiness.

A logical IF THEN ELSE function checks the relational operator >= (greater than or equal to). It returns a YES result when the operator is true, switching to the THEN part of the equation, or NO when the operator is false, switching to the ELSE part of the equation.

✍ Click on ***task is done***. Enter the equation below:

```
task is done[task]=
   IF THEN ELSE( Work Done[task] >= task size[task], 1, 0)
Units: Dmnl
```

The last two equations use more complicated logical functions. The values for ***is a prerequisite[task]***, ***task has started[task]*** and ***task is done[task]*** are binary (0 or 1). The IF THEN ELSE logic will return a value of YES if the variable in question has a nonzero value (and move to THEN), and a NO if the variable has a value of zero2 (moving to ELSE). Using variables in IF THEN ELSE expressions like we are doing here makes sense when the variables are formulated to take on 0 or 1 as values. Using an equation that began IF THEN ELSE(morale,... would not be a sensible formulation.

The first equation describes whether the task is active or not, and uses the logical operators :AND: and :NOT: If the task has started *AND* the task is *NOT* done, then the task is active (value = 1), else the task is not active (value = 0).

✍ Click on ***task is active***, write the equation as below, then click **OK**.

```
task is active[task]=
   IF THEN ELSE(task has started[task] :AND:
         :NOT: task is done[task], 1, 0)
Units: Dmnl
```

VMIN Function

The last equation also requires some explanation. In order to start a task, any prerequisite task must first be finished. We could write a set of 9 equations, "hard-wiring" the relationships that define the prerequisites. A shorter method allows us to specify the relationships in one equation.

✍ Click on ***task has started***, write the equation as below, then click **OK**.

```
task has started[task]=
```

```
   VMIN( IF THEN ELSE(
   is a prerequisite[task,prereqtask!],
   task is done[prereqtask!],
   1))
Units: Drawing
```

This equation uses the VMIN function, which operates on subscripts marked with an exclamation mark !. Because there are 3 tasks and 3 prerequisite tasks, this equation takes on 9 separate values.

For example, let us look at the *`design`* task. Design has no prerequisites, the value for:

```
is a prerequisite[design,prereqtask] = 0,0,0
```

is zero for all the prerequisite tasks. The IF THEN ELSE function evaluates the 3 values of *`is a prerequisite[design]`* and returns zero, so it fails the logical test and moves to the ELSE statement, which returns a value of 1. This means that the *`design`* task can start as soon as the model simulates, it has no prerequisite tasks.

On the other hand, let us look at *`build`*. Build has both design and prototype as prerequisite tasks, as in:

```
is a prerequisite[build,prereqtask] = 1,1,0
```

The IF THEN ELSE function evaluates *`is a prerequisite[build,prereqtask]`* for the three prerequisites and returns 1 in two cases. For each case, it moves to the THEN statement where it looks at *`task is done[prereqtask]`* for the particular *`prereqtask`* (let's say *`prototype`*). If the prototype work is finished, the *`task is done`* variable returns 1, otherwise it returns 0. If it returns 0 for any of the prerequisite tasks, the VMIN function also returns 0, which prevents that task from starting. Once all the prerequisite tasks return 1, VMIN will return 1 and the task can finally start.

✍ Select **Model>Check Model** (Ctrl + T).

✍ Select **Model>Units Check…** (Ctrl + U).

Your model should check out OK. If not, compare equations (and structure) to the equations above.

Simulation and Analysis

✍ Double click on the **Runname** editing box and type a name for the dataset, e.g., *sub3*

✍ Click the **Simulate** button.

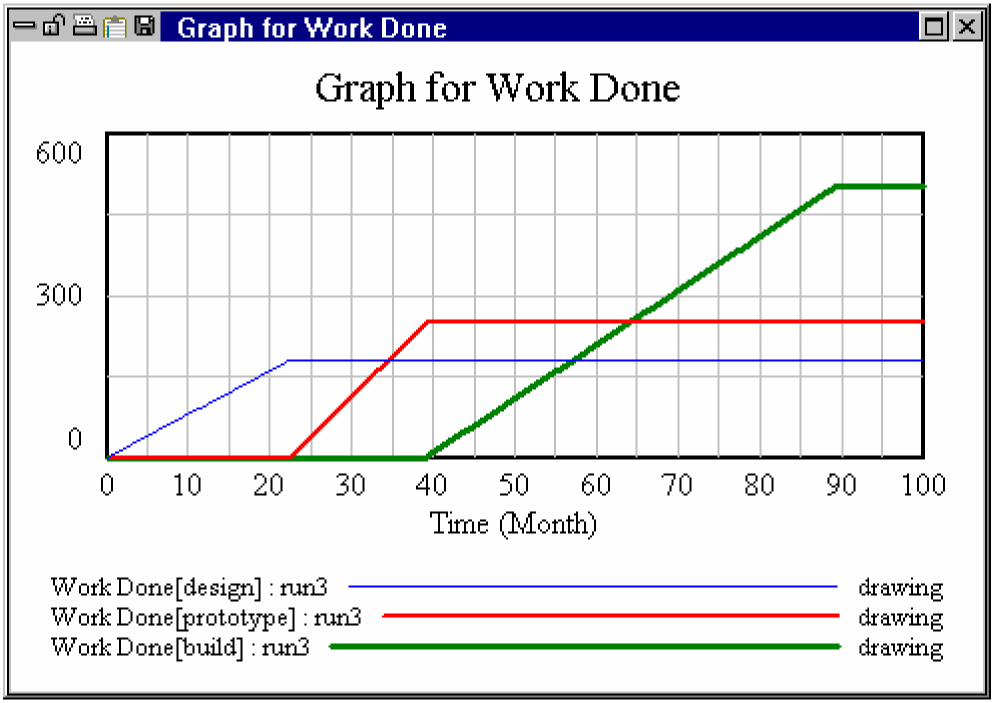
After the model simulates,

✍ Double click on *`Work Done`*, then click on the **Graph** tool.

Output is generated showing the behavior of *`Work Done`* for one subscript: *`design`*.

✍ Click the **Subscript Control** button on the Toolbar. The Subscript Control opens. Click on the button **All**.

✍ Click on the **Graph** tool.



Now we can see that each task, except design, starts only after its prerequisite has finished.

18 Optimization

Optimization is supported in Vensim Professional and DSS only.

Optimization Overview

To optimize is, quite simply, to achieve the best. Optimization can be used to validate and estimate parameters (calibration), or to select among alternative policies (policy optimization).

In order to use optimization, you will need to define what is good and what is bad — called the payoff. The payoff is a measure, reported at the end of the simulation, stating numerically how good the simulation was. The payoff collapses your entire model, over the entire time it was simulated, into a single number. The payoff definition file has the extension *.vpd*. A detailed description of the payoff function is contained in the Reference Manual.

After defining the payoff, you need to select which Constants to vary in order to maximize the payoff. The selections are stored in the optimization control file, which has the extension *.voc*. This file also contains information about how to optimize, and under what circumstances to stop.

At the end of an optimization, the results will be reported and are also written to a file called *runname.out*. Depending on what options are selected, other files may also be created. The file *runname.out* contains the values of the Constants that optimize the payoff.

NOTE Optimization can be interrupted by clicking on the **Stop** button or pressing the **Esc** key and will shut down cleanly for later resumption.

Model Calibration (electric.mdl)

In this section, we will calibrate a diffusion model of the conversion of US households to electricity to fit real data from the US Department of Commerce. The quantity of households (*`Total US Households`*) increases exponentially over time and this drives the increase in quantities of both *`Non Electric Households`* and *`Electric Households`*. *`Non Electric Households`* are making contacts with *`Electric Households`*, discovering electricity, and eventually being converted to *`Electric Households`*. Any new households (*`net household additions`*) that are built are added to both *`Non Electric Households`* and *`Electric Households`* according to the split determined by *`fraction electric`*. As the number of *`Non Electric Households`* dwindles to zero, the *`fraction electric`* becomes one and all the new households built are *`Electric Households`*.

Validation of a model rests in part on comparing model behavior to time series data collected in the "real world." When a model is structurally complete and simulates properly, calibration of the model can proceed. Calibration involves finding the values of model Constants that make the model generate behavior curves that best fit the real world data.

It is possible to manually alter model Constants to try and achieve a better fit between the real world data and simulation output. For a complex model with many Constants to optimize and many variables or datasets to fit, this is very time consuming. Using optimization, Vensim will automatically vary the Constants of your choice and look for the best fit between the simulation output and your real world data.

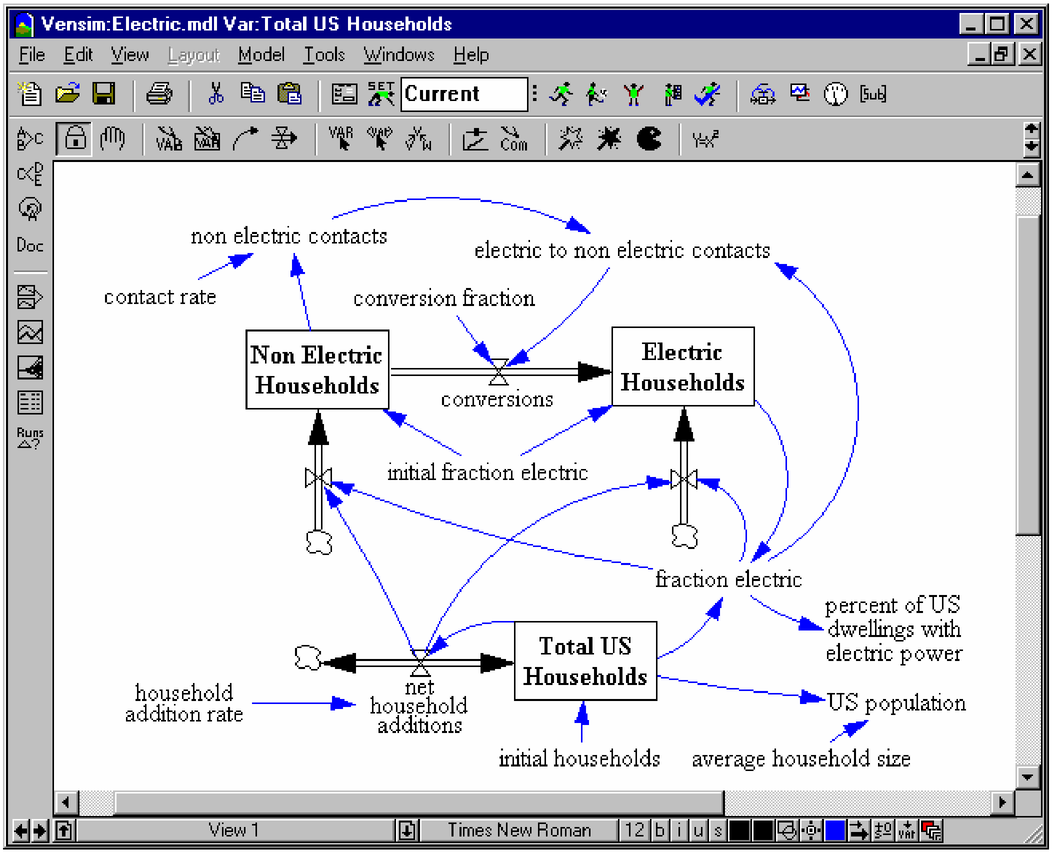
For example, we might not know (with any accuracy) what the growth fraction is for new households (`household addition rate`). We select this parameter (a Constant) for the Vensim optimizer to calibrate. The optimizer will automatically search through many values for this parameter to find the best fit of model data with the real world data.

EITHER

✍ Open the model *electric.mdl* contained in the *guide\chap18\calibrat* directory.

OR

✍ Sketch the model and enter the equations as shown below. The Time Bounds are INITIAL TIME = 1900, FINAL TIME = 1979, TIME STEP = 0.125, Save results every TIME STEP with units Year.



Electric.mdl Equations

```
average household size=
   4
Units: Person/Household

contact rate=
   20
Units: Contact/Household/Year

conversion fraction=
   0.01
Units: Household/Contact

conversions=
   electric to non electric contacts * conversion fraction
Units: Household/Year

Electric Households= INTEG (
```

```
      conversions+ fraction electric * net household additions,
            initial fraction electric * Total US Households)
   Units: Household
   electric to non electric contacts=
      non electric contacts * fraction electric
   Units: Contact/Year
   fraction electric=
      Electric Households/Total US Households
   Units: Dmnl
   household addition rate=
      0.02
   Units: 1/Year
   initial fraction electric=
      0.05
   Units: Dmnl
   initial households=
      2e+007
   Units: Household
   net household additions=
      Total US Households * household addition rate
   Units: Household/Year
   non electric contacts=
      Non Electric Households * contact rate
   Units: Contact/Year
   Non Electric Households= INTEG (
      - conversions + (1 - fraction electric) * net household
      additions,
            (1 - initial fraction electric) * Total US Households)
   Units: Household
   percent of US dwellings with electric power=
      fraction electric * 100
   Units: Dmnl
   Total US Households= INTEG (
      net household additions,
            initial households)
   Units: Household
   US population=
      Total US Households * average household size
   Units: Person
```

Note you will need to use the **Choose Variable...** button to enter initial value equations.

✍ Save your model.

✍ Double click on the Runname editing box and type a name for the dataset, e.g. *run1*.

✍ Click the **Simulate** button.

Comparing to Data

The dataset *run1* gives us a base run with Constants at their original values. The *guide\chap18* directory contains data with measurements of the US population and the fraction of households that are electric (connected to the electric grid). This file is called *electric.dat* and has two variables: `US population`, and `percent of US dwellings with electric power`. In order to compare these data series with variables in the current model, we need to translate the data file (*.dat*) into a Vensim dataset (*.vdf*).

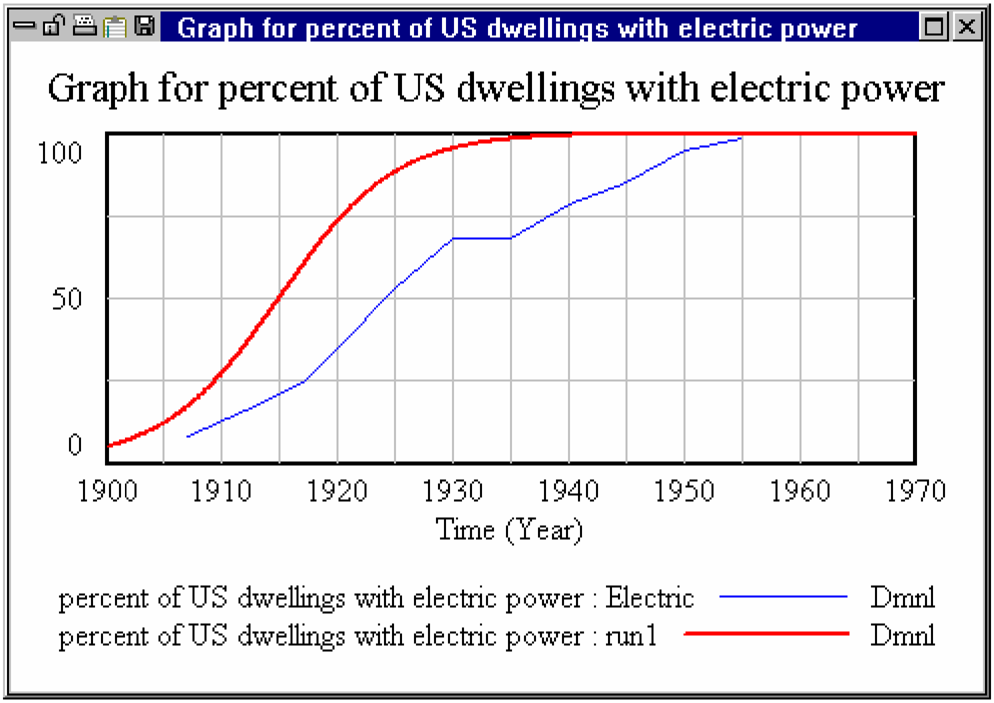
✍ Select **Model>Import Dataset…**. Choose the file *electric.dat* and click **Open**.

The message "Dat2vdf completed without error" is generated and an Output window will show that a number of values were written for each variable in the dataset. The dataset electric is loaded as the first dataset. Analysis tools will now work on this dataset.

✍ Click **OK** in the message box and close the Output window.

✍ Double click on percent of US dwellings with electric power.

✍ Click on the **Graph** tool. Two datasets, *run1* and *electric*, display lines for `percent of US dwellings with electric power`.



Note how the simulation data (*run1*) grows too fast and levels off too soon.

Setting the Payoff

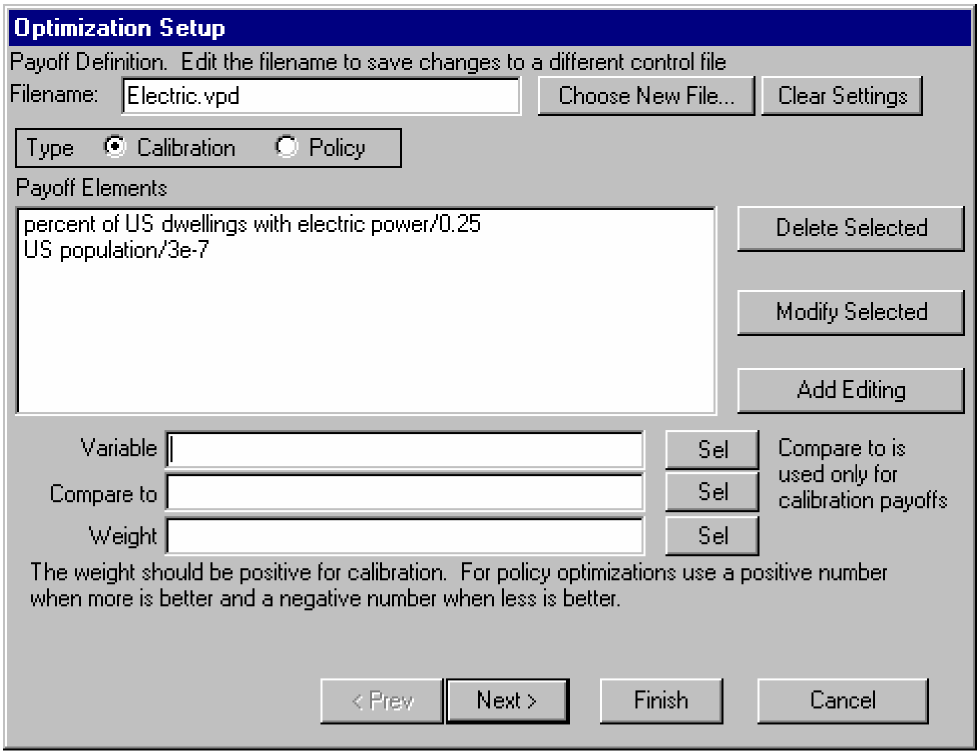
Vensim has two methods to set up optimizations: from the **Optimization** button on the Toolbar, and from the Simulation Control.

In calibration optimization, the payoff defines which variable(s) you are trying to fit. Payoff files have the extension *.vpd* (Vensim payoff definition) and by default have the same name as the model (e.g., *electric.vpd* for this model).

- Double click on the simulation Runname editing box and type a name for the dataset, e.g., *run2*.
- Click the **Optimization** button on the Toolbar. The Optimization Wizard opens with the Payoff Definition active.
- Make sure the **Calibration** radio button is selected.
- In the **Variable** editing box, type in `percent of US dwellings with electric power` (or use the **Sel** button to select it). In the **Weight** editing box, type in the number 0.25. Click on the button **Add Editing**.
- In the **Variable** editing box, type in `US population` (or use the **Sel** button to select it). In the **Weight** editing box, type in the number 3E-7 (0.0000003).

The weights act to balance the numerical sizes of the different variables. Because the optimizer is only looking at pure numbers, variables with larger values are given more importance than those with smaller values. The numbers assigned to weights are based up both the confidence we have in the data, and the units of measurement. *`percent of US dwellings with electric power`* has values around 50 and a weight set to 0.25. *`US population`*, which has a number around 200 million, has a weight that is much smaller (30 / 100 million). These weights were chosen to be approximately equal to the one over the standard deviation of the prediction error on these variables. Chapter 10 of the Reference Manual contains more discussion on the meaning of these weights.

✍ Click on the button **Add Editing**. The Payoff Setup dialog should now look like the screen below.



✍ Click **Next** to move to Optimization Control.

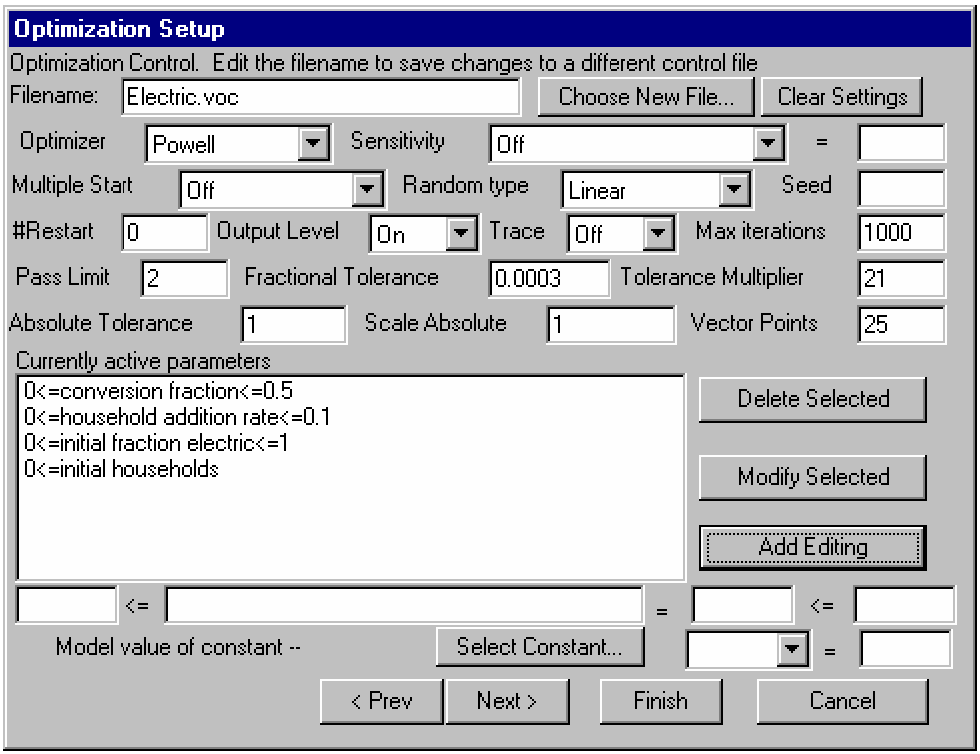
Setting Optimization Parameters

Once the payoff is defined, the Optimization Control needs to be set up. Optimization files have the extension *.voc* (Vensim optimization control) and by default have the same name as the model (e.g., *electric.voc* for this model). For this example, we will only enter the model parameters for the optimizer to vary, along with the (optional) maximum and minimum bounds for each parameter.

✍ Click on the **Select Constant...** button and select *`conversion fraction`*. Type 0 in the minimum box (left hand box, <=), type 0.5 in the maximum box (right-most box, <=), then click on the button **Add Editing**.

- Click on the **Select Constant...** button and select `household addition rate`. Type 0 in the minimum box (left hand box, <=), type 0.1 in the maximum box (right-most box, <=), then click on the button **Add Editing**.
- Click on the **Select Constant...** button and select `initial fraction electric`. Type 0 in the minimum box (left hand box, <=), type 1 in the maximum box (right-most box, >=), then click on the button **Add Editing**.
- Click on the **Select Constant...** button and select `initial households`. Type 0 in the minimum box (left hand box, <=), then click on the button **Add Editing**.

The Optimization Setup wizard should now look like:



- Click **Next**.
- Click on the **Data Sources…** button and choose the dataset *electric.vdf*. You can also simply type this name into the editing box.
- Click **Finish**.

Vensim will perform multiple simulations on the model to find the optimum Constant values in order to fit the output with the real world data. The results of the optimization are displayed in an Output window, as below:

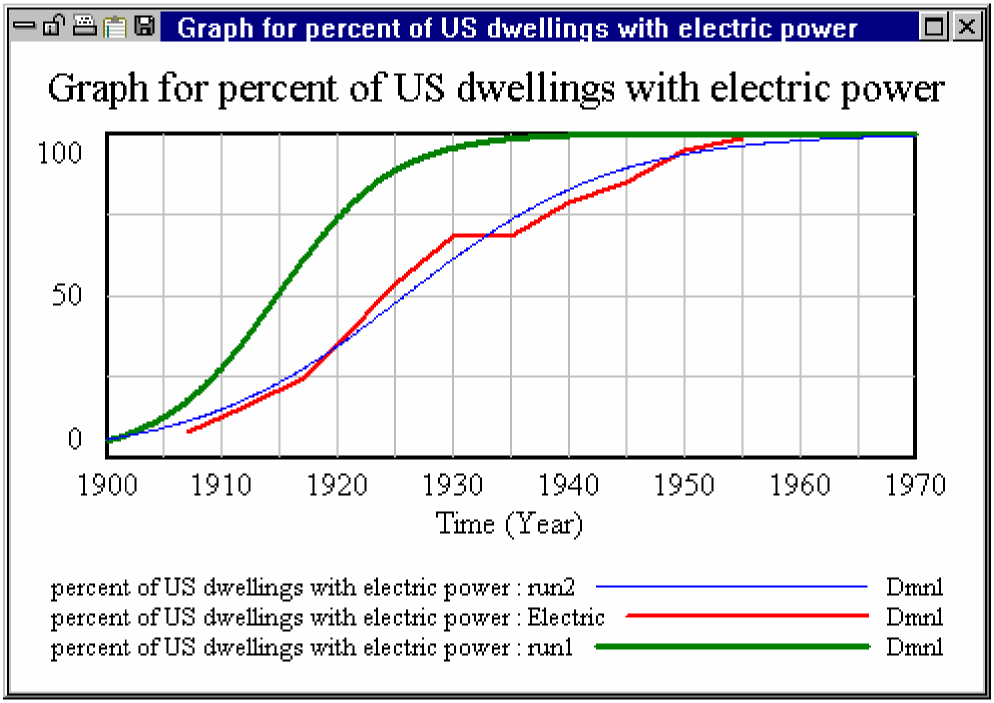
```
sim run2 -c mdlcin.tmp -o Electric.voc
Initial point of search
  CONVERSION FRACTION = 0.01
  HOUSEHOLD ADDITION RATE = 0.02
  INITIAL FRACTION ELECTRIC = 0.05
  INITIAL HOUSEHOLDS = 2e+007
Simulations = 1
Pass = 0
Payoff = -1766.51
---------------------------------
Maximum payoff found at:
  CONVERSION FRACTION = 0.00551817
  HOUSEHOLD ADDITION RATE = 0.0133057
  INITIAL FRACTION ELECTRIC = 0.0553119
 *INITIAL HOUSEHOLDS = 1.98387e+007
Simulations = 377
Pass = 3
Payoff = -16.8863
---------------------------------
```

Maximum payoff, or best fit of simulation output with real data, is displayed as a number, and the Constant values required to achieve that payoff are listed. The best fit is saved as the current simulation dataset (*run2*).

Extra information about the optimization, including the best-fit values of the constants is saved in a *.out* file of the same name as the run (here it is *run2.out*).

✍ Double click on `percent of US dwellings with electric power` if it is not selected as the Workbench Variable.

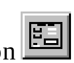
✍ Click on the **Graph** tool. You should see the three datasets, *run1*, *run2*, and *electric*, displaying lines for `percent of US dwellings with electric power`.



The simulation *run2* has a much closer fit to *electric* than does *run1*, our first attempt.

Computing Confidence Bounds

The Constant values found during optimization have some uncertainty around them and we can get a better handle on that uncertainty by testing the sensitivity of the payoff around the optimum. For this particular problem we have chosen weights so that the default payoff value sensitivity will give 95% confidence bounds. We will also use the Simulation Control to manage the optimization.

- ✍ Click on the **Simulation Control** button  on the Toolbar.
- ✍ Type a new run name such as *run3*.
- ✍ Click on the **Advanced** tab.
- ✍ Click on **Ed...** to the right of **Optimization Control**.
- ✍ The **Sensitivity** is set to **Off**. Click on the dropdown arrow to the right and select **Payoff Value**. Enter the value **4** in the box after the = sign. (Payoff Value = 4). Click on **OK**.
- ✍ Click on the **Changes** tab. In the **Load Changes From** field type *run2.out*.

The output of an optimization is designed so that it can be read as a constants input file. This will allow the optimizer to start from the value found in the last optimization.

- ✍ Click on the **Optimize** button.

✍ When the optimization is finished use the menu **File>Edit File...** to open the file *sensitiv.tab*.

This file contains information on the 95% confidence bounds for the parameter estimates. These should look something like:

```
0.00526005 <= CONVERSION FRACTION = 0.00551817 <=   0.00579413
0.0128454 <= HOUSEHOLD ADDITION RATE =0.0133057 <=    0.0137479
0.0483586 <= INITIAL FRACTION ELECTRIC =0.0553119 <=  0.0632539
1.94407e+007 <= INITIAL HOUSEHOLDS = 1.98387e+007 <= 2.02301e+007
```

✍ Select **File>Close** to close *sensitiv.tab*.

Optional — Modifying the Model

✍ Select the **Equations** tool. Modify the constants in the model to reflect the values given by the optimization. You will probably only want the first two or three significant figures.

The values from the optimizer should be, or be close to:

```
conversion fraction = 0.0055
household addition rate = 0.013
initial fraction electric = 0.055
initial households = 2.0e+007
```

It is important to note that getting a model to fit the data well does not prove the model right. When a model fits the data it means the model cannot be rejected because of the data. It is important to look for other ways to find deficiencies with a model. Reality Checks, described in Chapter 9 of the *Modeling Guide* are a very powerful tool for doing this.

Policy Optimization (stock.mdl)

This section describes the optimization of an inventory and sales supply chain to maximize profits. The model *stock.mdl* has an inventory that is drained by sales. Carrying too little inventory leads to loss of sales (and loss of profits) but carrying too much inventory costs more to stock and also leads to lower profits. To maximize profits, you need to balance the amount of inventory. You need to restock inventory when it is low, and build it to some maximum level.

Inventory Stock Model

✍ Open the model *stock.mdl* in the directory *guide\chap18\policy*

✍ Select **Model>Import Dataset…**. Choose the file *orders.dat* and click **Open**.

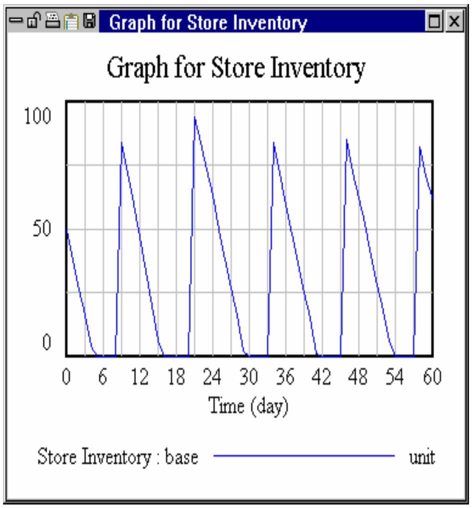
✍ Click **OK** to the conversion message and close the Output window.

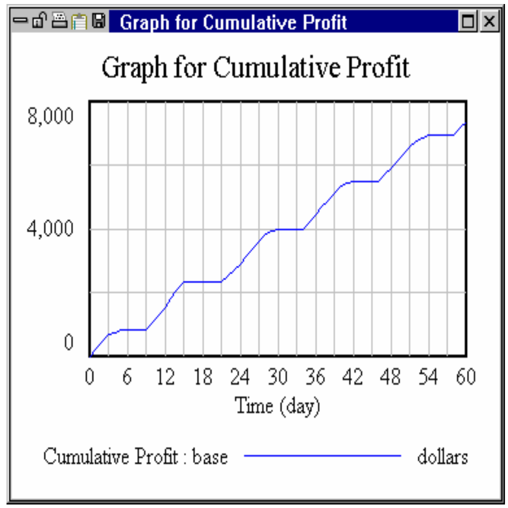
✍ Double click the **Runname** editing box and type a name for the dataset, e.g., *base*.

✍ Click on the **Set Up a Simulation** button.

✍ Click on the dataset editing box right of the integration button that reads **Euler** and type in *orders*.

- Click on the **Simulate** button.
- Click on the **Control Panel** button and then on the **Datasets** tab. Unload the dataset *orders* by double clicking on it.
- Double click on the Level `Store Inventory` then click on the **Graph** tool.
- Double click on the variable `Cumulative Profit` then click on the **Graph** tool.
- Click the **Output Windows** button and drag the top graph off the bottom graph.





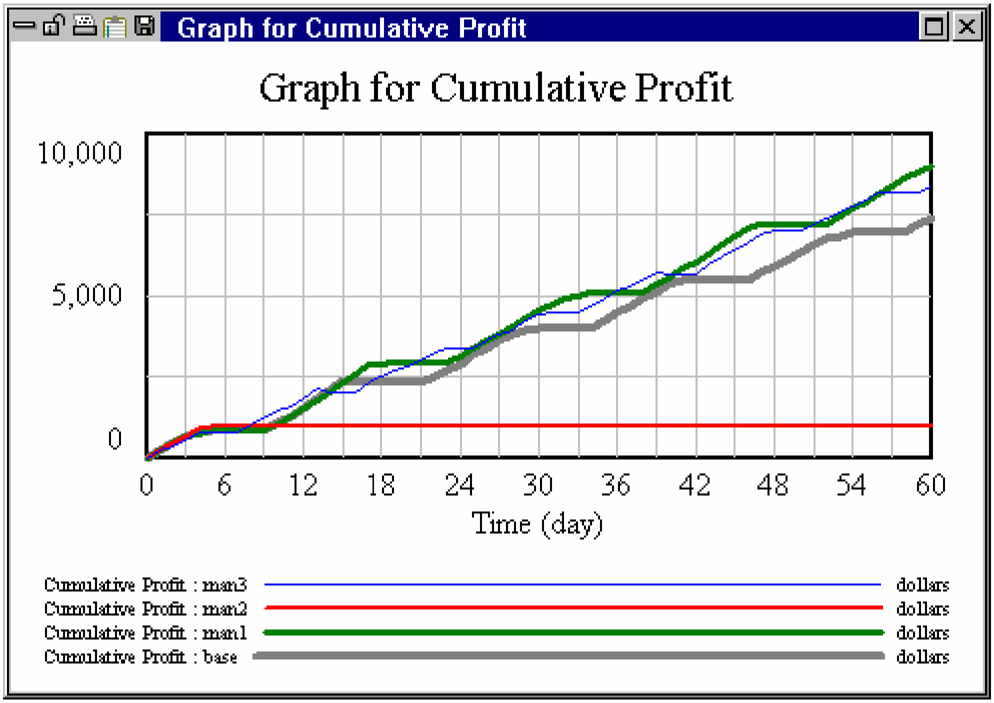
This is our base case, ordering when inventory falls to 20 units (`min inventory value`) and building up to a maximum of 100 units (`max inventory value`). Note, however, that `Inventory` falls to zero for some time before being restocked.

Optional — Manual Optimizing

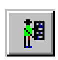
As an exercise, you can try to manually optimize the model.

- Click the **Set Up a Simulation** button. Type a name for the dataset, e.g., *manual*.
- Click on the Constants `min inventory value` and `max inventory value` and change their values.
- Click the **Simulate** button.
- Select the variable `Cumulative Profit` into the workbench and click on the **Graph** tool.

Compare the final value of `Cumulative Profit` in this run to the base run. If you want, continue by trial and error to try and maximize `Cumulative Profit`. You will get a series of runs similar to below:

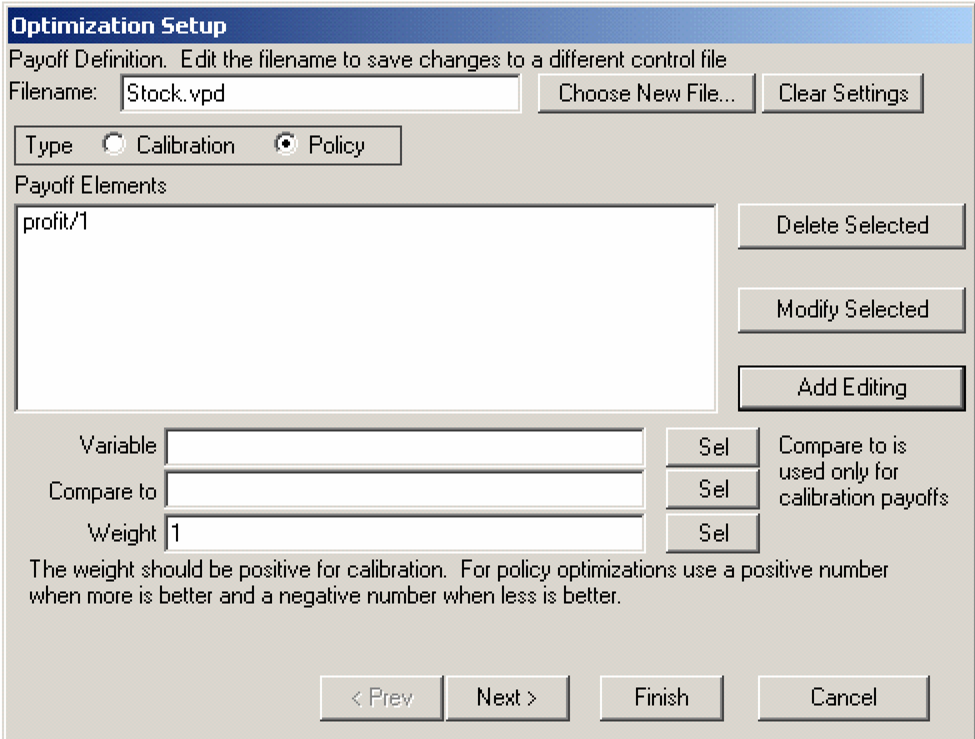


Setting the Payoff

- Double click the **Runname** editing box and type a name for the dataset, e.g., *optimize*.
- Click on the **Optimization** button  on the Toolbar. The Payoff dialog opens.
- Click on the option button for **Policy** next to **Type.** Click on the **Sel** button next to **Variable** and select `profit` from the list, then click **OK** to the variable selection dialog (or just type profit into the editing box).
- Click on the **Weight** editing box and type the number 1. Click on the button **Add Editing**.

Positive weights designate more is good (more profits); negative weights designate more is bad. If more than one parameter set in the payoff, you need to balance (or weight) your parameters appropriately. See the calibration example earlier in this chapter or Chapter 10 of the Reference Manual for more information on setting up payoffs.

Your **Payoff Setup** dialog should appear as below:



✍ Click **Next**.

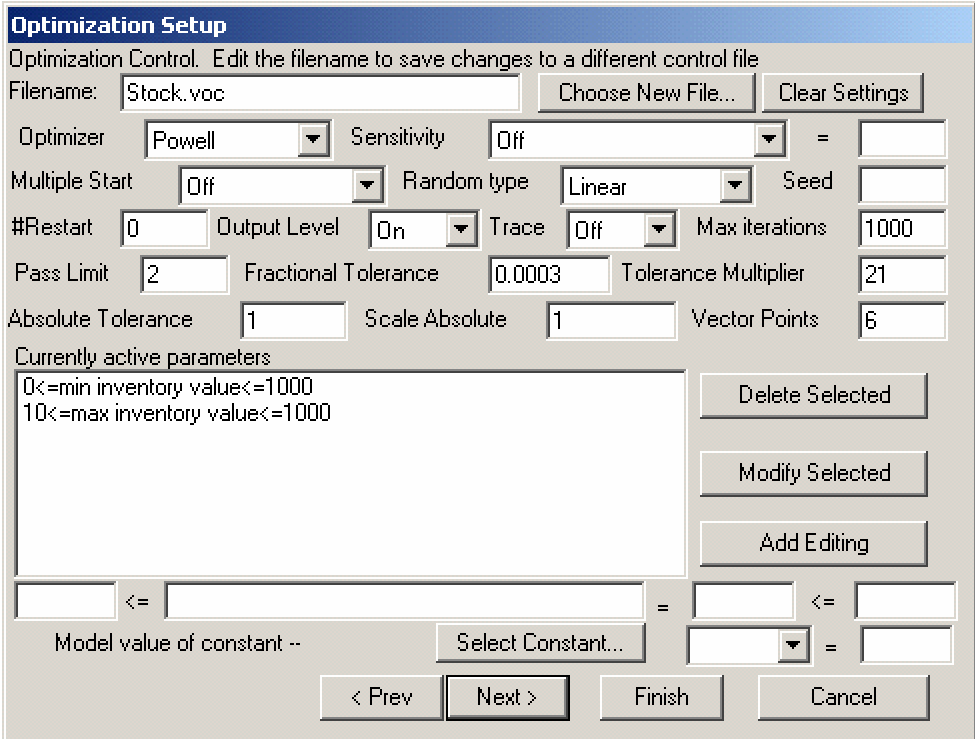
Note that policy optimizations maximize the integral of the payoff. Thus using ***profit*** in the payoff is the same as maximizing ***Cumulative Payoff*** at the end of the simulation.

Setting Optimization Parameters

✍ Click on the **Select Constant...** button and choose ***min inventory value*** from the list. Type 0 in the minimum box (left hand box, <=), type 1000 in the maximum box (right hand box, <=), then click on the button **Add Editing**.

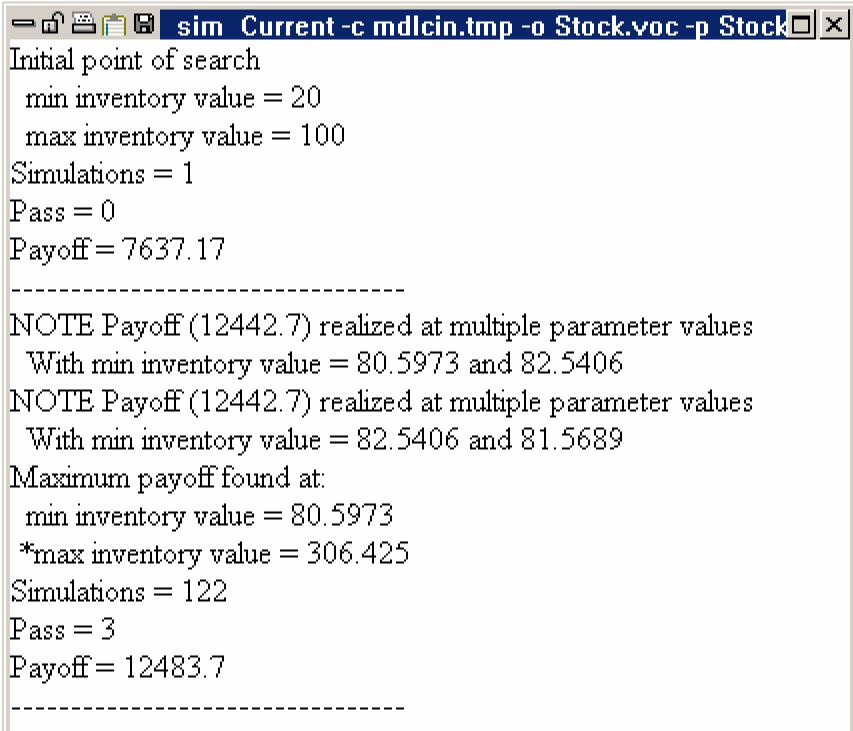
✍ Click on the **Select Constant...** button and choose ***max inventory value*** from the list. Type 10 in the minimum box (left hand box, <=), type 1000 in the maximum box (right hand box, <=), then click on the button **Add Editing**.

Your **Optimization Control** should appear as below:



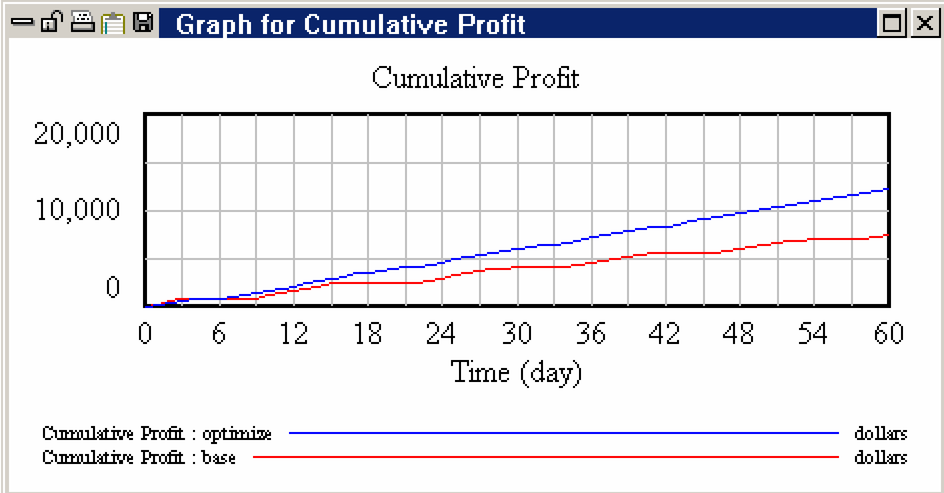
✍ Click on **Finish**.

The optimizer runs, generates an output file, and loads the optimized run.



- ✍ Click the **Control Panel** button then click the **Datasets** tab. Unload all runs except *base* and *optimize*.
- ✍ Select the variable ***`Cumulative Profit`*** into the workbench and click on the **Graph** tool and on the **Table** tool.

Compare the final value of ***`Cumulative Profit`*** in the *optimize* run to value in the *base* run.



Table

Time (day)		56	57	58	59	60
"Cumulative Profit"	Runs:	optimize	base			
Cumulative Profit		11,367	11,591	11,817	12,024	12,213
		6,963	6,963	6,963	7,173	7,364

The graph shows the behavior of the optimum run, the table shows the values for the maximum `Cumulative Profit` found by the optimizer.

19 Sending Models to Others

You can easily send the models you create to others. Even if they don't have Vensim they can use the Vensim Model Reader to view and simulate the model. Also many models can be read by less powerful version of Vensim. Thus, even if you are using Vensim DSS, users of Vensim PLE may be able to work with the models you are developing.

If you are using Vensim DSS there are a number of additional options for packaging and publishing models. These are described in more detail in Chapter 13 of the DSS Reference Supplement.

Vensim Model Reader

The Model Reader is an application that allows read-only access to models created with Vensim. You can distribute the Model Reader installation program free with your models so that others can simulate and analyze the models without using Vensim itself. If you have a Vensim CD the model reader installer is the file named *venred32.exe* in the *windows* directory or *Vensim Model Reader Installer* on the Macintosh. You can also download the model reader from our website.

The Model Reader will work with models created with Vensim PLE through DSS. Models created with Vensim DSS, except those that make use of external functions, will also work with the Model Reader. It is also not possible to access models through DDE connections using the model reader, that capability is available only in Vensim DSS.

The Model Reader does not have Sensitivity Simulation or Optimization capabilities, except when opened via a Venapp. Also the simulation functionality such as resuming from another run available through the simulation control dialog in Vensim Professional and DSS is not available unless this is set up in a Venapp. You can specify data sources from the Toolbar, and changes files from the Changes menu in the Vensim Model Reader.

SyntheSim works with the Vensim Model Reader, though you cannot override the behavior of variables.

Preparing the Model

There are two ways to prepare a model for use by others, including those who only have access to the Vensim Model Reader. The first is to use the File>Publish command in order to create a package of the model and supporting files. The second is to simply save the model to a binary format. Independent of which of these you do there are some things that are helpful for making it easier for other people to work with your model.

First, it is a good idea to include your name and the contact info you want to give people. You can embed this information as a comment on the first sketch view. Alternative you can use the model notes (use the command **Model>Settings** then go to the **Info/Password** tab). In this dialog fill in the notes and check the **Display model notes when model is opened** Checkbox. This information will then be displayed in a separate dialog when users open the model.

It is helpful to have units of measure and comments for each variable. These are displayed when user's hover over the model variables and can make it much easier to understand a model.

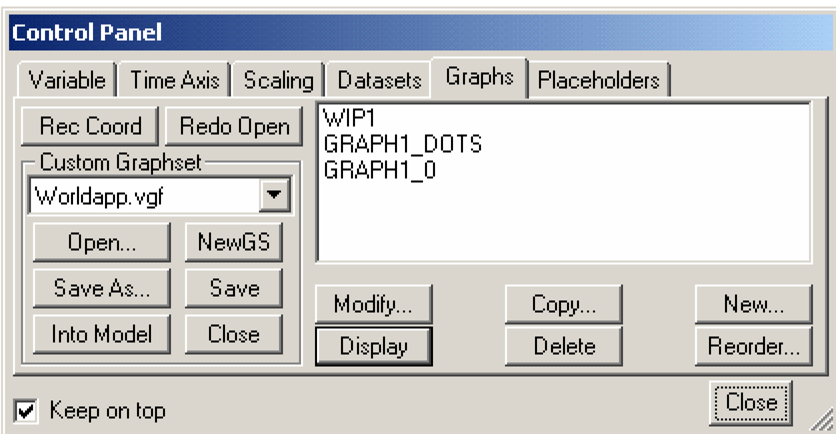
If your model has more than one sketch view than it is helpful to add a navigation page as the first page. The navigation page can contain your contact info and a navigation button for each view in the model. While page up and page down will work in the Vensim Model Reader, the navigation buttons make it much easier. If you do this you should probably put a navigation button on each view that returns to the first view.

It can also be very useful to set up a simple control panel that contains the important constants that you want users to change and some important results – possibly as imbedded custom graphs. The example in Chapter 12 does this.

Custom Graphs and Tables

This does not apply to Vensim PLE or PLE Plus.

If you have created custom graphs or tables in a separate file you will need to bring these into the model for others to use them. To see if this is the case go to the **Graphs** tab of the control panel:



If the name displayed under **Custom Graphset** is not *Default (or if the **Into Model**) button is active you will need to bring the custom graphs and tables you are working with into the model. Click on the **Into Model** button to do this.

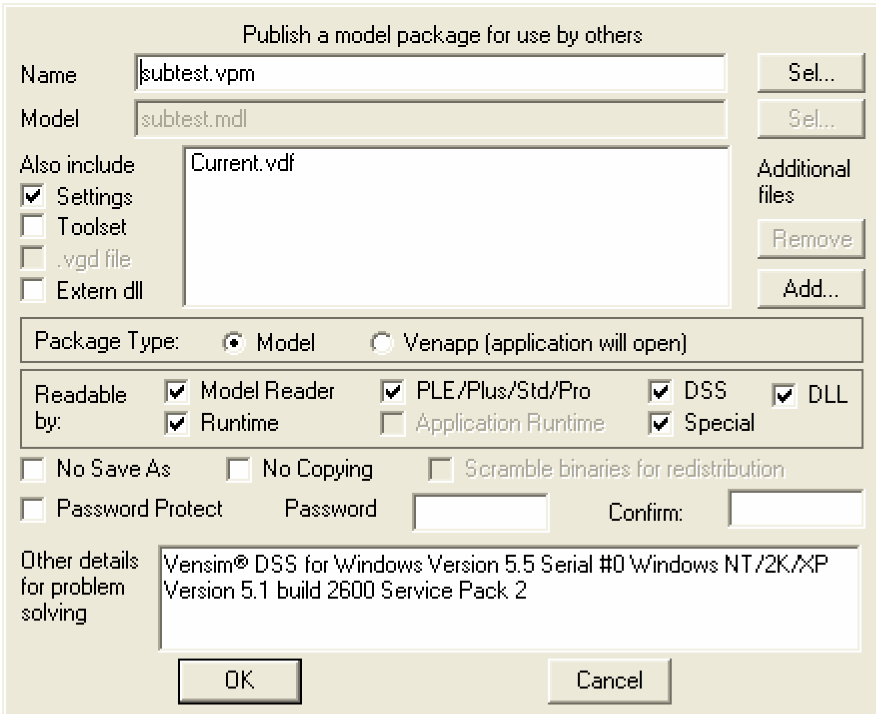
Publishing a Packaged Model

The Vensim Model Reader can only read packaged model (.vpm) files and binary format (.vmf) models. Both of these are binary file formats that can only be read by Vensim.

NOTE Vensim versions prior to 5.5 will not be able to open packaged models!

Publishing a Package

To publish a model use the menu item File>Publish when the model is in the foreground. You will see the model publication dialog:



Name is the name of the package that will be created. By default this is just the name of the model you have open with extension *.vpm* which stands for Vensim Packaged Model. If you are packaging a Venapp (DSS Only) the extension is *.vpa* for Vensim Packaged Application.

Also Include indicated other items you may want to include in the package. Depending on the configuration of Vensim you are using and other conditions some of these may be grayed.

? **Settings**, if checked, will include your .ini file, normally contained in the Windows directory or the User Preference Directory. This is useful if you need to send us a package to troubleshoot a problem, but is otherwise not used.

? **Toolset**, if checked, includes the currently loaded analysis toolset (only if a named Toolset is loaded – On PLE and PLE Plus this is always gray).

? **.vgd file**, if checked, includes the current Vensim Graph definition file if a named file (instead of *Default) is being used. This is always gray for PLE and PLE Plus.

? **Extern dll**, if checked, includes the external function library currently in use. This is gray except when you are using DSS with an external function library loaded. Check this only if the model uses external functions and you have permission from the author of the library to provide it to others.

The above files will be stored in the package as if they were in the same directory as the model, though they might be in a different location on your computer. Only the Extern DLL will actually be extracted if it is to be used when the package is opened and it will be extracted to the directory the package is in.

Additional Files lists additional files to include. When this dialog opens all the files associated with the model (loaded runs, changes files, the data file, optimization control files and so on) will be included in the list. To add something to the list click on the **Add…** button (on Windows you can use Shift+Click or Control+Click to select multiple files). To remove something from the list highlight it and click on the **Remove** button. Note that Excel files used as data sources are not automatically added to the list.

If you include runs (*.vdf*) files in the list when the package is used these files will be treated as read only. If the user attempts to simulate to a file with the same name as an included file they will receive an error message and asked for a different file name.

Archive Type specifies the type of archive. This is available only on Vensim DSS. A **Model** archive includes a model and supporting files. A **Venapp** archive includes a Venapp *(.vcd*) file and supporting materials (normally including the model). When a Venapp archive is open the Venapp is executed.

Readable by: gives you control over what applications can open the archive. For example if you want to control access to your model you it readable only by the Model Reader. **Model Reader** refers to the Vensim Model Reader. **PLE/PLUS/Std/Pro** includes all of Vensim PLE, Vensim PLE Plus, Vensim Standard and Vensim Professional. **DSS** refers to Vensim DSS. **DLL** refers to the Vensim DLL – this is available only to Vensim DSS users. **Runtime** refers to Vensim Runtime. **Application Runtime** refers to Vensim Application Runtime – this is available only for users of Vensim DSS when the Archive type of Venapp is chosen. **Special** refers to applications such as Sable that use a Vensim engine to deliver content.

No Save As, if checked, will prevent anyone who opens the archive from saving the model as a different name and making changes to it. This forces all Vensim Applications to behave in a manner that is similar to the Vensim Model Reader.

No Copying, if checked, prevents anyone from copying model structure. This is stronger than **No Save As**. The archive user can still see model structure in the Views presented and using the Document tool, but Edit>Copy will not work.

Scramble binaries for redistribution, if checked, creates an archive for use with licensed redistributable Vensim components. This option is available only if you have an Application Redistribution License.

Password Protect, if checked, causes the archive to require a password to be opened. Enter the password you would like to use and its confirmation. Anyone opening the archive will be queried for this password. If the password is lost the archive can't be opened.

NOTE Password protection prevents the curious from getting access to your model, but will not deter someone sufficiently determined to look at it. Security and protection of intellectual property are serious concerns and remain your responsibility.

Other details for problem solving lists information about your copy of Vensim and your computer. This is useful if you are sending us an archive for trouble shooting – otherwise it is ignored. You can add any comments you want in this.

Settings: Load… queries for a file which has settings for this dialog that have previously been saved. This allows you to set up the publication settings once, save them, and then load them at a later date. Note that when you load settings they do not need to be for the packaging of the same model you have open, but they must be for a model in the same directory or publication will fail.

Settings: Save… queries for a file to save the current dialog settings to. This allows you to preserve dialog setup for use at a later date. If you have specified a password you will be asked if you want to save that password. If you do the password will written in plain text into the form file. The form file does not form part of the published package, so this is not a problem as long as you do not let other people access that file.

You can open a packaged model with any Vensim application subject to the choice of the one who packaged the model. Some models may use functionality not available in the Vensim configuration you are using and it may not be possible to simulate these. Only Vensim DSS and the Vensim Model Reader can be used to open a packaged application.

Binary Format

To save a model to a binary format use **File>Save As** and then select the type **Binary Format Models** and type in a new filename. The extension that will be added to this name is *.vmf*. You can also save a model as a binary format model by simple changing the *.mdl* extension to *.vmf* in the **File>Save As** dialog.

You can open a *.vmf* file with any Vensim configuration. These are somewhat larger than the pure text format *.mdl* files and can only be opened with Vensim. You can open *.mdl* files with any text editor which makes them more useful for backup and archival storage.

Opening a Packaged Model

You open a packaged model just as you would open any Vensim model. Packaged models are listed when you use File>Open Model and double clicking on them in Windows Explorer will open them. When a packaged model is open, however, its appearance may be somewhat different.

If the model is marked as not allowing Save As then there will be no sketch tools shown across the top toolbar. File>Save and Views>As Text will not work.

If Save As is allowed, the model will appear normally. File Save will open the Save As dialog. You can make changes to the model but the changes must be made outside the package. The package itself is read only.

If the package contains runs you will not be able to use that run name. You can load and unload the runs from the control panel, but they will not appear as files on the operating system.

If the package contains a toolset then in Standard, Professional and DSS you will be asked if you want to load that toolset. In the Vensim Model Reader that toolset will automatically be loaded, this assures that any embedded graphs appear as expected. In Vensim PLE and PLE Plus the toolset is ignored as these configurations do not support toolset modification.

If the package contains sensitivity or optimization control files using the wizards to run these will read the packaged files, then write any changes you make to an operating system file. If you reopen the package you will start again from the packaged control files. Packages are read only and designed to always give the same results when opened.

If you open a packaged model and save that model to another name the runs in the package will still be available to you. However, if you close and reopen Vensim on the model that you saved the packaged model to those runs will no longer be available.

Only one packaged models can be opened at one time. This means if you open a second packaged model the model associated with the first package will be closed. It is also not possible to publish a packaged model or the files contained in the package.

If you happen to have a disk file with the same name as a file in the packaged model, the file in the packaged model will be used. It is good practice to open packaged models in folders that do not have many other files to avoid any confusion that this can cause.

Using models with Vensim PLE

In many cases you may wish to give people models that they can modify. Vensim PLE can be used to open models developed in other Vensim configurations. Since not all Vensim features are supported in Vensim PLE not all models will work.

Subscripts: Models that use subscripts will open in Vensim PLE with an error message. You will be able to look at the model structure but not simulate the model.

Unsupported Functions: Models that use unsupported functions will open in Vensim PLE but will not simulate. Using **Model>Check** will display which functions are not supported.

Navigation Comments: Navigation comments will be visible in Vensim PLE but can't be used to navigate.

There are no restrictions on model size in any of the Vensim configurations. Making very large models in Vensim PLE is not practical because of the single view restriction, but large models created in other configurations will open in Vensim PLE.

Email and Web Notes

Vensim models in *.mdl* format are plain text files. While this makes them easy to archive, it can have some unexpected side effects when making them available to other people. There are a couple of things to be aware of when emailing or putting these models on web sites.

Some email programs will automatically wrap lines in text attachments. This can cause problems when those line wraps occur in the middle of a variable name or in the sketch definitions. The typical symptom of this is either a syntax error message, or sketch views that are clearly incomplete. If you open the file in a text editor (such as notepad) you should be able to see where line breaks are added. To prevent this type of problem you may want to zip files before sending them to people.

Sometimes, text files placed on a web site get an extra blank line added between every line. This is typically because of incompatibility in text file formats between Unix, Windows or Macintosh file systems. It is a good idea to check the files you put up on a web site to see if this is a problem. Similar problems sometimes occur when text attachments to email open in a web browser. Try to save the attachment to see if this works better.

Both of these problems can be avoided by publishing the model to a model package.

20 Reference Modes

NOTE This chapter is in preliminary form.

A Reference Mode is a graph of a variable over time that shows the important characteristic of the variable for a problem statement being worked on. For example, if the problem statement is that the price of a commodity goes up and down in a cyclic manner the reference mode could simply be a graph of the price of the product. This could be the actual price over some historic time period, an approximation of that based on recollections, or an idealization of that behavior that is a sine wave. In addition to representing things that have happened, reference modes can be developed for things in the future. For example if you are creating a new product to bring to market you might have a reference mode of continued growth that is what you want to have happen and one of growth and collapse that is what you fear might happen.

Reference Modes can really help to clarify a problem statement. To draw a reference mode you need to specify the range of time over which things need to be investigated. As simple as this sounds, it is very helpful to deciding what to include and what to leave out of a model. Reference modes also help to identify which variables are the mo st important to include in a model. If there is no clear reference mode for a variable it may be possible to leave that variable out of the model.

Reference modes are also helpful in abstracting away elements of behavior that are not central to a problem. For example, if you are looking at the dynamics of the business cycle you might not want to have to include growth dynamics. This type of abstraction can be very helpful in keeping models simple and insightful. Care must be taken, however, if you want to compare model behavior to data that has been detrended or otherwise modified as the theoretical foundations for doing this are quite limited.

In this chapter we will work through how Vensim's Reference Model tool can be used to add reference modes to model diagrams and compare the behavior of a simulation model with reference modes. We will also show how to enter an exogenous driving variable quickly using the Reference Mode tool. Because you can enter exogenous drivers, and show expected behavior the Reference Mode tool provides an excellent platform for doing mental simulation exercises and we will show an example of how to use it this way.

Conceptualization with Reference Modes

Reference modes can be very helpful in conceptualizing your model and capturing information from others. This can best be shown by working through a simple example. We will consider the problem of keeping a factory running in the face of reductions in the labor force responsible for maintaining the factory. Our basic dynamic hypothesis is that decreased workforce will tend have a stronger effect than expected as repair requirements take away from the ability to perform maintenance.

We start by putting up variables that clearly are important to understanding this problem.

✍ Start a new model (using the default time bounds).

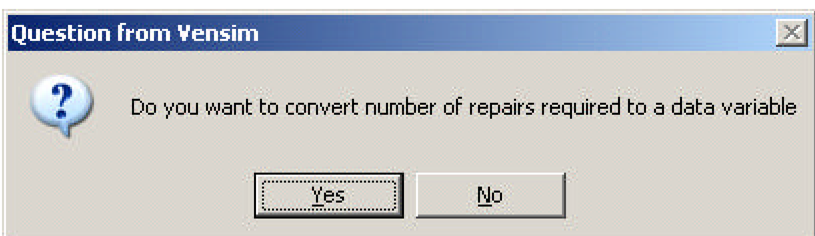
✍ Add the variables ***number of repairs required*** and ***total maintenance workforce*** to the diagram. You should have something like this:

number of repairs
required

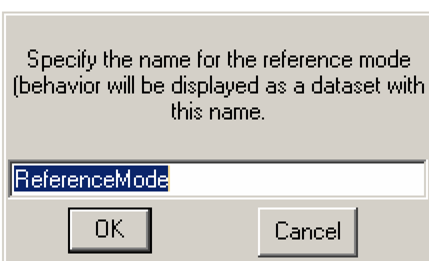
total maintenance
workforce

✍ Select the Reference Mode editing tool  then click on ***number of repairs required***.

✍ Respond No to the query (this will be explained later):



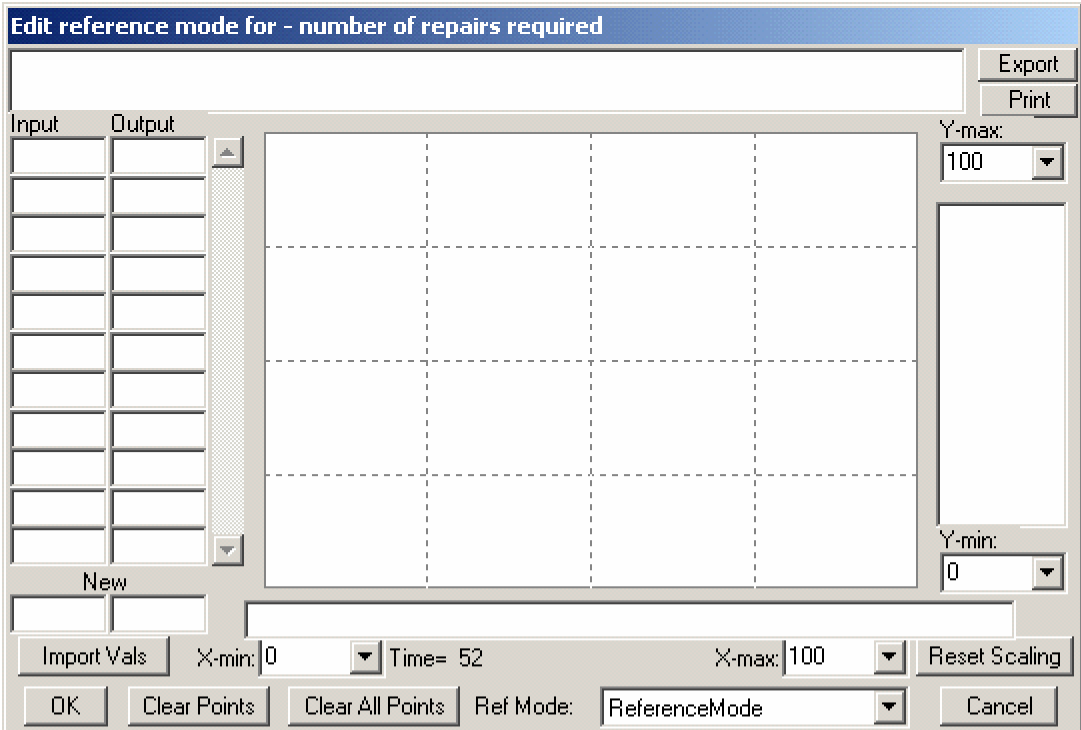
You will be asked to name the reference mode:



The default name of "*ReferenceMode*" and is fine. You can choose a different name and it is possible to have as many as eight different reference modes associated with a model.

✍ Click "OK" in the Reference Mode Name dialog.

This will open a dialog box that is almost the same as the Lookup editor:



Unlike the Lookup editor the x-axis is interpreted as the Time axis, and there is also a dropdown at the bottom (labeled RefMode:) that allows you to move between different reference modes (you can define up to eight) and create new ones. For this example we will only use a single reference mode.

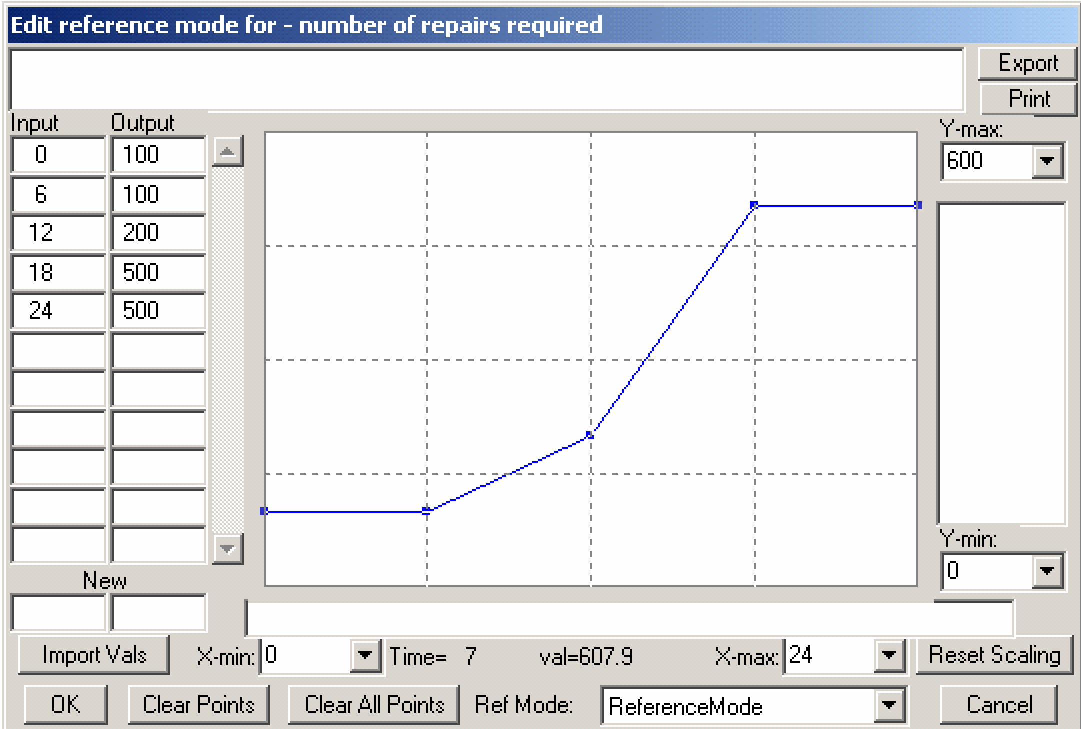
The time axis model runs from 0 to 100. This is not appropriate for our problem, and we want to set it to run for 24 months.

✍ Click on Cancel.

✍ Select the menu item Model>Settings and in the Model Settings dialog change `FINAL TIME` to be 24. Click on OK.

✍ Click on number of repairs required again to open the Reference Mode editing dialog. (Again answer no to making this a data variable).

✍ Enter the pairs (0,100), (6,100), (12,300), (18,500), (24,500). Your dialog should look like:



✍ Click on OK to close the dialog.

✍ Click on ***total maintenance workforce***.

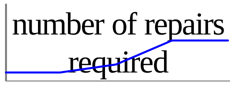
✍ Click on Yes to make this a Data variable.

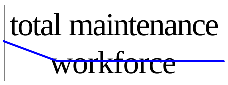
A Data variable is a time varying model input, also called an exogenous variable. When you say yes to making ***total maintenance workforce*** a data variable this means Vensim will look to a data file to find values for it and not compute it. If you have not specified a data file for simulation, Vensim assigns the first reference mode file you create to be a data file. You can change this later from the toolbar in simulation setup mode or from the advanced tab of the simulation control dialog. We make the workforce a data variable because the purpose of the model is to understand what happens when this changes.

✍ The workforce started at 10, but was cut in half over 6 months. Enter the pairs (0,10), (6,5), (24,5) to represent this.

✍ Clock on OK to close the dialog.

You model should look like:

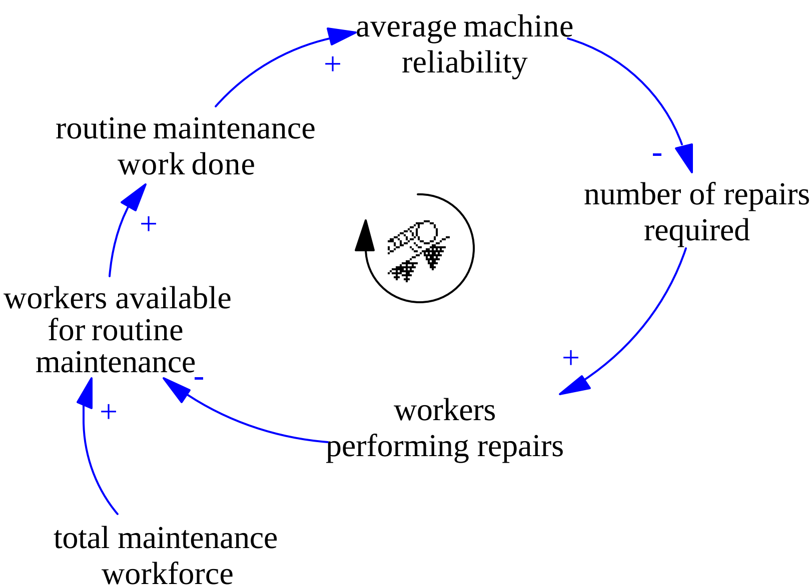




Closing the Loop

We now have an input and a result and the two have reference modes that are somewhat different. If these two are both valid, then it is clear that some dynamic structure is required to connect them. We fill in our dynamic hypothesis by adding more variables.

✍ Complete the diagram shown below, or open the model *guide\chap20\complete\repair01.mdl.*

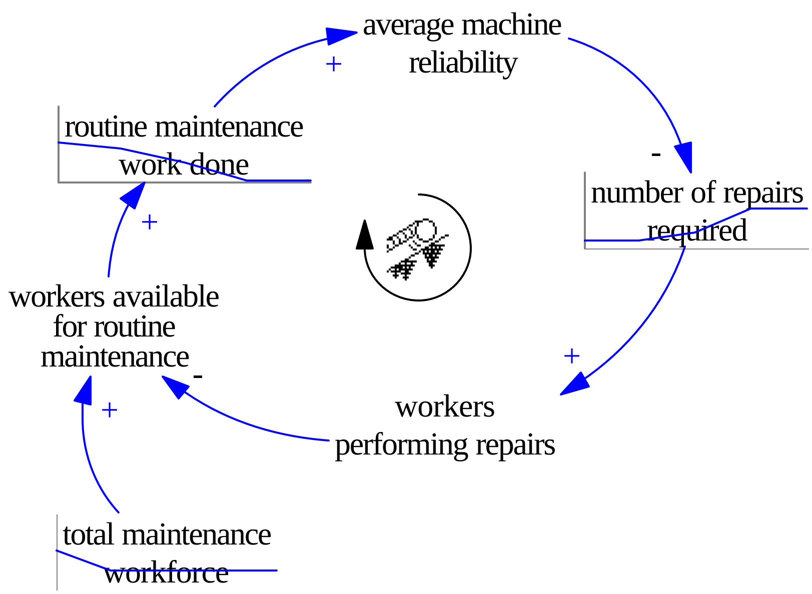


✍ Select the Reference Mode editing tool and click on `routine maintenance work done`.

This time you will not be asked if you want to make `routine maintenance work done` a data variable. This is because the input into it clearly indicates that it is endogenously computed. The same is also true of `number of repairs required` now, but wasn't when we originally added the reference mode for it.

✍ Add the pairs (0,200), (6,175), (12,100), (18,0), (24,0) and click on OK.

Your diagram should look like:

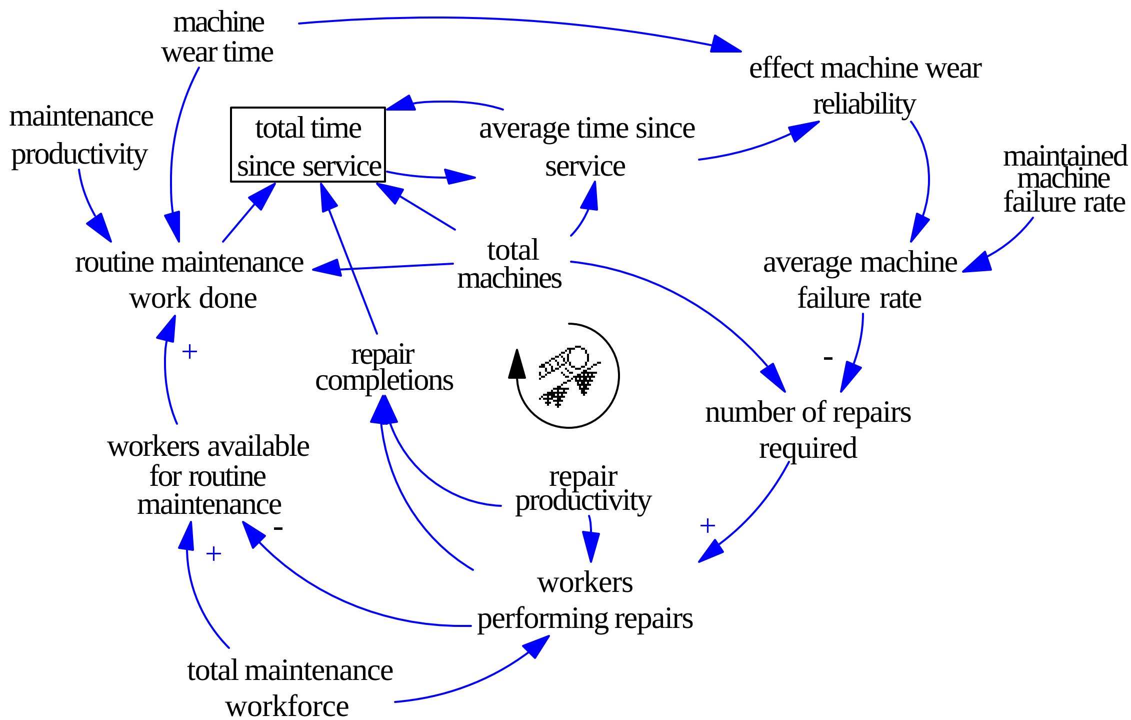


This represents our dynamic hypothesis and he reference modes help to tell the story. You can, if you want, fill in reference modes for the other variables. This is actually an excellent exercise in trying to understand how things will behave. It should, however, never be forgotten that the reference modes are not being generated by the model, but simply made up by the modeler.

Simulating

To create a simulation model, it is necessary to add a little bit more structure. You may, if you prefer, also change to a stock and flow notation. Using stock and flow notation is not strictly necessary, and we will not do that here. We will, however, mark the Level variable (in fact there is only one) with boxes.

Build the following diagram, or open the model *guide\chap20\complete\repair02.mdl*.



This model has a somewhat complicated construct that is a variation of the coflow. We measure the average time since machines have been maintained by using the variable *`total time since service`*. This is the total number of machine hours elapsed since service for all the different machines. With each unit of time this increases by *`total machines`*, but it is decreased when a machine is repaired or routinely serviced. The amount that it decreases depends on how long the repaired or serviced machine had gone without service. Our simplifying assumption for this is that this is just the average machine age. A more complete treatment could differentiate the repair and maintenance age distributions, but would make the model quite a bit more complicated.

The equation for total time since service is thus:

```
total time since service= INTEG (total machines –
   (repair completions + routine maintenance work done) *
         average time since service,
   total machines * machine wear time)
```

The initialization in this is used to put the model near an equilibrium at the beginning of the simulation. To get the average age of machines we just divide total time since service by the total machines (the number of machines). This average time is then used to drive reliability.

You will notice in this model that we changed from *`average machine reliability`* to *`average machine failure rate`*. The main reason for this is that is seems to be easier to formulate the equation for it. Reliability is basically 1-unreliability so it could have been done as in the original causal loop diagram but probably would not have been harder to undrestand.

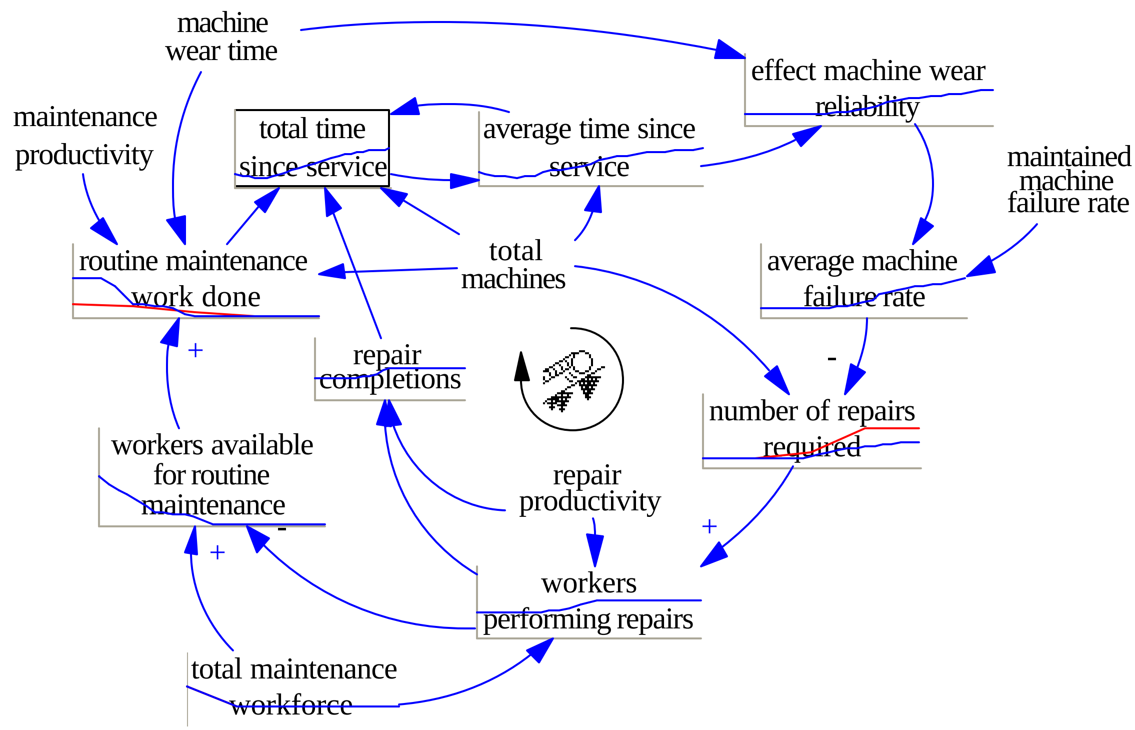
The full equations for this model are:

```
average machine failure rate=
    maintained machine failure rate *
        effect machine wear reliability
Units: 1/Month

average time since service=
    total time since service/total machines
Units: Month

effect machine wear reliability= WITH LOOKUP (
    average time since service/machine wear time,
    ((0,0.9),(1,1),(1.5,1.1),(2,1.5),(3,3),(4,6) ))
Units: Dmnl

machine wear time=2
Units: Month

maintained machine failure rate=0.1
Units: 1/Month

maintenance productivity=120
Units: Machine/Person/Month

number of repairs required=
    average machine failure rate * total machines
Units: Machine/Month

repair completions=
    workers performing repairs * repair productivity
Units: Machine/Month

repair productivity=30
Units: Machine/Person/Month

routine maintenance work done=
    MIN(total machines/machine wear time,
        workers available for routine maintenance *
        maintenance productivity)
Units: Machine/Month

total machines=1000
```

```
Units: Machine

total maintenance workforce
Units: Person

total time since service= INTEG (total machines –
   (repair completions + routine maintenance work done) *
        average time since service,
   total machines * machine wear time)
Units: Machine*Month

workers available for routine maintenance=
   total maintenance workforce –
        workers performing repairs
Units: Person

workers performing repairs=
   MIN(
        number of repairs required/repair productivity,
        total maintenance workforce)
Units: Person
```

Note that the equation for total maintenance workforce is blank. Blank equations are used to indicate data variables. If you followed the instructions earlier in this chapter you would not have needed to add that equation directly as it would have been put in place by Vensim. The units, however, will still need to be added.

If you simulate this model and then display behavior (the B key will toggle the display of behavior on and off) you should see:



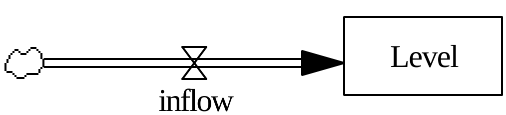
We are partially replicating the two reference modes that were entered. While the behavior is not exactly the same, it is similar. Given scaling and other issues with reference modes it is common for there to be some differences with simulated behavior

We have developed a model from conceptualization through simulation using reference modes to help in the process. The reference modes add a concrete representation of behavior early in the model development process and also make it easy to profile exogenous inputs.

Mental Integration

The Reference Mode editing tool is also an effective means of working on mental integration skills. Integration, or accumulation, is the basic building block of dynamic behavior and building intuition for how accumulation works indifferent circumstances can be helpful for improving understanding of dynamic problems.

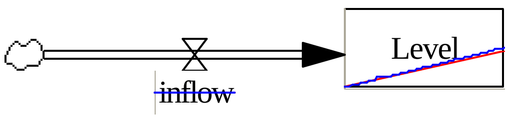
✍ Create the simple model shown below using the default time bounds (or open guide\chap20\complete\MentalSimulation01.mdl).



- Click on the Reference Mode editing tool and click on inflow. Answer yes to the question on whether you want to turn this into a data variable.
- Name the Reference Mode MentalSimulation1 and then enter the pairs (0,10), (100,10).
- Click on OK to close the dialog.

With this inflow what would you expect the value of Level to do?

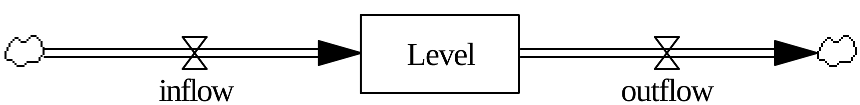
- Select the Equation editor and enter 0 as the initial value for Level.
- Click on Level with the reference mode editing tool and then enter what you would expect it to do.
- Simulate the model. You should get something like:



- Select the Reference Mode editing tool and then click on inflow again. Use the pairs (0,0), (50,100), (0,0).
- Click on Level and draw the behavior that you expect to see. Simulate and see how your guess compares with what happens.
- Repeat this type of experimentation with different inflow patterns such as ((0,0), (100,100)), ((0,100),(100,0)) and ((0,100),(50,0),(100,100)).

Even with only a single inflow mental simulation is surprisingly tricky. When a second rate is added the problem becomes even more difficult.

- Add and outflow to the model as shown below (or open the model guide\chap20\complete\MentalSimulation02.mdl).



- Select the Reference Mode editing tool and click on `outflow`. Answer Yes to turning this into a data variable. Set the `outflow` pattern to (0,0), (50,100), (100,0).
- Set the `inflow` pattern to (0,100), (50,0), (100,100).
- Select the Equation Editor and change the equation for `Level` to be

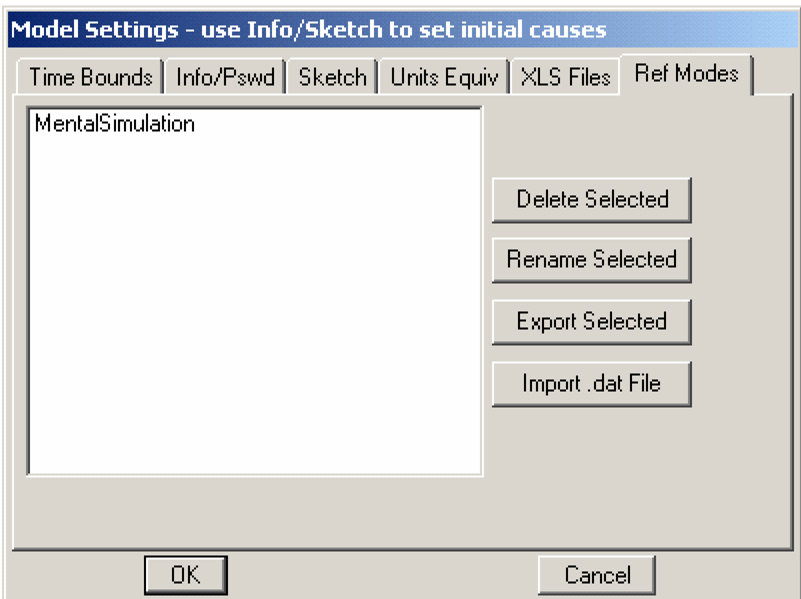
`Level = INTEG(inflow-outflow,0)`

✍ Enter the pattern of behavior you expect for Level.

Even for simple input patterns, with both an inflow and outflow the behavior of the Level may surprise you. Try entering more complex patterns for *`inflow`* and *`outflow`* and see if you get what you expect.

Managing Reference Modes

You can have up to eight distinct reference modes in a model. These can be managed from the "Ref Modes" tab of the Model Settings dialog:



Click on the reference mode you want to change.

Delete Selected will delete the selected reference mode from the model. The dataset displaying that reference mode will not be changed, but can be deleted from the Datasets tab of the Control Panel.

Rename Selected will rename the selected reference mode. The old name will still exist as a dataset, but can be deleted from the Dataset tab of the Control Panel.

Export Selected will create a *.dat* file with the information in the Reference Mode. See Chapter 9 of the Reference Manual for more information on *.dat* format files.

Import .dat File allows you to take the information in a *.dat* format file and include it as a Reference Mode. See Chapter 9 of the Reference Manual for more information on *.dat* format files.

Backward Compatibility

Versions of Vensim prior to 5.2 did not support Reference Modes. You will still be able to open models containing Reference Modes in earlier versions of Vensim but the reference mode information will be lost. This may happen even if you do not save the model in the earlier Vensim version.

Appendix Information Resources

This appendix contains a list of suggested resources in the field of System Dynamics and computer simulation. This list is by no means complete, but provides a place to start and gives pointers to other resources. Almost all books (and many audio and video tapes) published in the field of System Dynamics can be purchased from Pegasus Communications. Their address and phone number appear later in this appendix.

System Dynamics Books

System Dynamics is a field that resulted from the pioneering efforts of Jay W. Forrester to apply the engineering principles of feedback and control to social systems. One of the earliest and still one of the best references in this field is *Industrial Dynamics* by Jay W. Forrester (The MIT Press,[1] Cambridge, MA, 1961).

A recent and comprehensive book on system dynamics is *Business Dynamics: Systems Thinking and Modeling for a Complex World* by John D. Sterman (Irwin McGraw Hill, Boston: 2000). This book works through both conceptual issues and a number of sample models, all of which are available in an included CD. All the models can be run with Vensim PLE. More information is available at http://www.mhhe.com/business/opsci/sterman.

A textbook which gives a straightforward and very readable presentation of how to build models is *Introduction to System Dynamics Modeling with DYNAMO*, by G.P. Richardson and A.L. Pugh (The MIT Press, Cambridge, MA, 1981) available from Pegasus Communications.

Another useful resource, containing a number of interesting real world examples is *System Enquiry, A System Dynamic Approach* by Eric F. Wolstenholme (John Wiley & Sons, New York 1990).

A very popular and readable book covering many conceptual issues in thinking systemically about problems is *The Fifth Discipline* by Peter M. Senge (Doubleday, New York, 1990).

Modeling for Learning Organizations. Eds John D. W. Morecroft and John D. Sterman, Productivity Press, Portland, OR, 1994. Covers some of the recent work in modeling businesses and other organizations, and includes case studies and reviews of simulation software.

Urban Dynamics by Jay W. Forrester covers urban development, with emphasis on job training programs, housing and employment issues. The urban model in *Urban Dynamics* is included with Vensim.

Global Modeling

An interesting and controversial work *World Dynamics* by Jay W. Forrester (The MIT Press, Cambridge, MA, 1971; second edition, 1973, available from Pegasus Communications) discusses growth in a finite world. The World model contained in this book is included in the models supplied with Vensim.

Following up on *World Dynamics* was the widely read book *The Limits to Growth* by Donella H. Meadows, Dennis L. Meadows, Jørgen Randers and William W. Behrens, III (Universe Books, New York 1972) a follow on work *Beyond the Limits* by Donella H. Meadows, Dennis L. Meadows and Jørgen Randers (Chelsea Green Publishing Company, Post Mills Vermont 1992), and a revised version *Limits to Growth – the 30-year Update*, by Donella H. Meadows, Dennis L. Meadows and Jørgen Randers (Chelsea Green Publishing Company, White River Junction Vermont 2004 http://www.chelseagreen.com) Detailed discussion of the model used in these works is presented in *Dynamics of Growth in a Finite World* by Dennis L. Meadows et al (The MIT Press, Cambridge MA 1974). The model used for the most recent book is included with Vensim.

Environmental Modeling

Modeling the Environment: An Introduction so System Dynamics Modeling of Environmental Systems by Andrew Ford is an introduction to both system dynamics modeling techniques and their application to environmental modeling. More information is available at http://www.wsu.edu/~forda/.

Living in the Environment: 9th Edition by G. Tyler Miller Jr. (Wadsworth Publishing Company, Belmont CA 1996 ISBN 0-534-23898-X) is a biology and ecology textbook that includes discussion of a number of issues from a system dynamics perspective. A companion to this work, *Simulations for Miller's Living in the Environment 9th Edition*, by David Peterson (Wadsworth Publishing Company, Belmont CA 1996 ISBN 0-534-23905-6) presents a number of models demonstrating issues discussed in the book. Both are available from bookstores or from Thomson Communications, 7625 Empire Drive, Florence, KY 41042 Phone: 800-865-5840, Fax: 606-647-5013.

Roadmaps

Roadmaps is a series of writings and exercises that are intended as a self study guide to system dynamics. They have been developed by MIT undergraduates working under the direction of Jay W. Forrester. The material is available for free from the MIT system dynamics in education web site (http://sysdyn.clexchange.org) and can also be purchased for a nominal charge from the Creative Learning Exchange (contact information below). Some of the mechanical activities in Roadmaps are not applicable to Vensim PLE but most of the conceptual content is relevant.

Publishers

Pegasus Communications publishes a range of system dynamics books in addition to those already mentioned. The also publish *The Systems Thinker* and other material with a focus on learning organizations. They have a catalogue of books and other materials.

Pegasus Communications, Inc.
One Moody Street
Waltham MA 02154-5339
Phone: 781 398 9700
Fax: 781 894 7175
http://www.pegasuscom.com

The Creative Learning Exchange is a non profit organization devoted to the dissemination of materials to improve learning in K-12 education, with a special focus on system dynamics:

Creative Learning Exchange
27 Central Street
Acton, Massachusetts 01720
Phone: 978 635 9797
http://www.clexchange.org

System Dynamics Society

The System Dynamics Society is an international, nonprofit organization devoted to encouraging the development and use of system dynamics around the world. With members in over 35 countries the System Dynamics Society provides a forum in which researchers can keep up to date with applications, methodologies and tools.

The System Dynamics Society is responsible for the publication of *The System Dynamics Review*, a refereed journal containing articles on methodology and application. The International Conference of the System Dynamics Society is held annually. This meeting brings together people from around the world and is arranged to give broad exposure to the local activities of the area hosting it. A President's letter provides an informal forum to bring members up to date on society activities and business.

In addition to *The System Dynamics Review* and its annual conferences, the Society maintains supplies of past conference proceedings and other important publications in the field. An electronic bibliography of work done in the field is maintained and updated periodically. The "Beer Game," a production distribution board game illustrating important principles relating local decision making to system behavior is available along with video tape and instructional material.

The System Dynamics Society is open to all individuals. For more information contact:

Roberta Spencer, Executive Director
The System Dynamics Society
Milne 300, University at Albany Phone: (518) 442 3865
135 Western Avenue Fax: (518) 442-3398
Albany NY 12222 Email: system.dynamics@albany.edu
http://www.systemdynamics.org

Internet Resources

There is a support and discussion forum for Vensim and other topics in system dynamics. You can access this using the Help menu from Vensim (Help>Online Form. The direct link for this it http://www.ventanasystems.co.uk/forumand you can also reach it through http://www.vensim.com/forum.

World Wide Web

The World Wide Web contains many sites of interest to system dynamics and other simulation modeling. The sites listed here all contain links to other sites.

The Vensim Home Page (http://www.vensim.com) contains information about Vensim and has a number of downloadable programs including Vensim PLE.

The system-dynamics mailing list (see below) home page (http://www.vensim.com/sdmail.html) contains information about system dynamics and also provides access to the system dynamics bibliography.

Mailing Lists

system-dynamics

The system-dynamics mailing list is a moderated internet mailing list for discussions of issues in system dynamics. When someone posts a message to the list, it is rebroadcast to all list subscribers. To join the mailing list send an e-mail message to listserv@listserv.albany.edu containing the following text in the body (the subject line is ignored):

```
subscribe sdmail your full name
```

do not include your email address - the one in your From: or Reply-to: heading will be used.

K-12

The K-12 mailing list for the use of system dynamics in primary and secondary education. For more information visit http://www.clexchange.org/bb/k12_discussion.htm.

Distance Eduction

For a more complete academic grounding in system dynamics there is an excellent distance system dynamics program offered by the Worcester Polytechnic Institute (WPI). This is a distance program in system dynamics that includes both introductory courses, applications courses and a number of advanced topic courses. There are both certificate and degree options. See http://www.wpi.edu/Academics/ADLN/Programs/SD/ for more information.

Support and Licenses

Support and Services

Maintenance

Support for, and new versions of, Vensim are provided on a subscription basis. Your initial licensing of Vensim includes one year of maintenance. At the end of that time you must renew your maintenance to continue to receive new releases and technical support. Maintenance gives you access to all new versions of Vensim for the configuration you have licensed. These can be downloaded from our web site

http://www.vensim.com

If you choose not to renew maintenance you may, unless you have a limited time version, continue to use the software, but you will not have access to new releases or technical support.

Technical Support

Technical support is available for problems relating to the installation and use of the Vensim Software. Technical support does not cover issues in conceptualization and formulation which can best be addressed through training or consulting.

Training

Training is available both for the use of Vensim and the art and practice of building system dynamics models. Public workshops are offered periodically and workshops can also be delivered in-house when that is appropriate.

Consulting

Ventana Systems, Inc. provides a full range of consulting services. We can provide you with the expertise you need to address difficult business problems.

Professional and DSS Support

Technical support is available to by email, fax and telephone to users with current maintenance.

	Product Sales	Technical Support	Training	Consulting
Email	sales@vensim.com	support@vensim.com	training@vensim.com	consulting@vensim.com
Phone	800 836 7461	508 651 0432	800 836 7461	978 456 3069
	508 651 0432		508 651 0432	
Fax	508 650 5422	508 650 5422	508 650 5422	938 456 9064

International Support

If you purchased Vensim though one of our international distributors you can contact them directly with any support questions you may have. All are knowledgeable in both the use of Vensim and the practice of system dynamics.

PLE and PLE Plus Support

Ventana Systems provides email support for questions relating to the use of the Vensim PLE and PLE Plus software. Send questions or problems to **plesupport@vensim.com**.

IMPORTANT NOTE Telephone support is not available for Vensim PLE or PLE Plus.

Software License

Vensim® Software License

This License applies to all versions or configurations of Vensim software that have been supplied to you and is a legally binding agreement between you as a Vensim User (either on behalf of yourself or on behalf of an entity of which you are an authorized representative) and Ventana Systems, Inc. ("Ventana").If you are using a Vensim configuration provided at no cost, it is still subject to this License. If you are installing an upgrade, this License replaces the original license, which is terminated.

1. GRANT OF LICENSE. Ventana grants you a non-exclusive right only to install one copy, including all components, of the Vensim software program (the "Software") and to use the installed Software under the following conditions. You may either: (a) install the Software on a single computer allowing access at a single physical point by a single individual (which may include you and other individual users) at any one time or (b) install it only for your own use on one to three computers providing that only one installation is in use at any one time and all installations are used exclusively by you. You may use the Software in presentations and face-to-face or virtual meetings provided that you are present and active in the discussion. No other right, including any right to copy, use, display, enhance, modify, or distribute the Software, is granted by this License. The right to use the Software terminates upon the violation of any provision of this License.

2. RESTRICTIONS ON USE. You may not (a) use the Software to provide web or other network services to anyone who does not hold a valid license to the Software, (b) use the Vensim Model Reader for the development of applications using the Vensim DLL, or (c) use any limited time evaluation version of the Software for any purpose other than evaluation of the Software during that period for suitability of use.

3. TERM AND MAINTENANCE. Provided that you are not in violation of any material provision of this License, your License to use the Software is granted in perpetuity, except that your License for a limited time version of the Software expires at the end of the specified time. If the Software you have is not a limited time version you may continue to use the Software that you have installed for as long as you like. Access to more recent versions of the Software, bug fixes, and technical support may require the payment of maintenance or upgrade fees.

4. PROPRIETARY RIGHTS. The Software is proprietary to Ventana and contains valuable trade secrets of Ventana. All copyright, patent, trade secret, trademark, and other intellectual property rights in the Software are and shall remain the valuable property of Ventana. You may not reverse engineer, decompile, or disassemble the Software. You agree to take all necessary steps to ensure that the provisions of this License are not violated by you or by any person under your control or in your service. Any updates, revisions, enhancements, additions, or conversions to the Software supplied by Ventana shall be subject to this License.

5. REPRODUCTION AND COPYRIGHT. The Software is protected by United States copyright laws and international treaty provisions. Once installed, you may not copy, allow anyone else to copy, or otherwise reproduce any part of the Software or installed support files without the prior written consent of Ventana, except that you may make one copy solely for backup or archival purposes, and you may copy or use vensim.jar or simdll32.dll as noted in Paragraph 13. You may keep copies of the installation files for archival or backup purposes. You may provide the installation files for Vensim PLE and the Vensim Model Reader to other individuals, but publication of these installation programs using the Web, a CD-ROM, or any other medium is prohibited without the prior written consent of Ventana. You may make copies of the documentation for your own use and that of others holding a license to the Software. You may not remove or omit any copyright or other proprietary notices from the Software or documentation.

6. PUBLIC RESEARCH; RESTRICTIONS. If this Software has been made available to you for restricted use in conducting publicly available research, your License is conditioned on your compliance with (and you must certify in writing upon Ventana's request that your use will conform to) the following restrictions: (a) you may use the Software only for work that is to be published or otherwise made publicly available; (b) all models and supporting electronic documentation used with the Software must be made publicly available without charge beyond distribution expenses; (c) all such models must be made publicly available without restriction or under a free license (such as the GNU general public license) that does not put any constraints on manner of use; (d) you may not use of the Software on any research that is classified or otherwise restricted by any government entity; (e) you may not use the Software on any research for which the full results are not publicly available; and (f) you may use the Software with data that is not publicly available only if all results are reported in a manner that is consistent with the standards of peer reviewed scientific journals.

7. EDUCATIONAL USE; RESTRICTIONS. If this Software has been made available to you as a qualifying person for restricted educational use, your license is conditioned on your compliance with (and you must certify in writing upon Ventana's request that you meet) the following conditions: (a) you must be (i) a school, college or university licensed by an appropriate state authority, whose primary purpose is to provide instruction to an enrolled body of students (an "Eligible Institution") or (ii) a student at an Eligible Institution or (iii) a faculty member at an Eligible Institution, and (b) your use of the Software must conform to the following restrictions: (i) you may not use the Software for non-educational or commercial purposes or use the Software in the provision of consulting services for compensation, (ii) all course materials developed using the Software will be made available to students at the Eligible Institution, (iii) all use of the Software will conform to the standards of educational use at the Eligible Institution, (iv) all results of all research conducted using the Software must be published or otherwise made publicly available, and (v) you may not use the Software on work that is classified, proprietary or restricted in any way. Research being substantially conducted by a student in order to complete a course or degree requirement and that is in conformance with (iii) is exempt from (iv) and (v).

8. NETWORK INSTALLATIONS AND MULTIPLE LICENSES You may install the Software on a network server provided that its use through the network conforms to the grant of license in Paragraph 1. If multiple licenses have been acquired, each use must conform to the terms in Paragraph 1. If Vensim PLE is installed for educational use, it may be installed on a server with no limitations on the number of simultaneous users provided that all users qualify as educational users as defined in Paragraph 5(a)(i)-(iii).

9. EDUCATIONAL LAB PACK If an Educational Lab Pack consisting of five or more licenses has been acquired, then you may either (1) use the Software in conformance to the use restrictions in Paragraph 1 or (2) install the Software on a network server that permits simultaneous use by a number of people up to the number of licenses you have obtained. All users for an Educational Lab Pack must qualify as educational users as defined in Paragraph 5(a)(i)-(iii).

10. TRANSFER OF LICENSE. Your License may be transferred to another individual by removing all copies of the Software from your computers and transferring all installation programs and documentation to that individual. If the transfer is to another organization, Ventana must be informed. In the case of multiple license purchases, all Licenses must be transferred together.

11. LIMITATIONS ON EXPORT. Export of the Software to other countries may be restricted and requires the permission of Ventana. Export of the Software to any country for which the United States of America maintains a trade embargo is prohibited.

12. MODELS AND RESULTS. The Software may be used to create simulation models and generate results. Ownership in, and responsibility for, the models and results created using the Software remains entirely with you.

13. REDISTRIBUTION OF SUPPORTING PROGRAM COMPONENTS. If you have purchased a License to Vensim DSS, you may provide the supporting Java class (vensim.jar) to others for use with applications you have developed, either as is or incorporated into your own application files. You may also provide compiled model equations (simdll32.dll on Windows). You may not provide the Vensim DLL or any other Vensim components to others.

14. WARRANTY. Ventana warrants that the Software will perform substantially in accordance with the accompanying documentation for a period of ninety (90) days from the date of receipt. VENTANA EXPRESSLY DISCLAIMS ALL OTHER REPRESENTATIONS AND WARRANTIES INCLUDING ANY REPRESENTATION OR WARRANTY WITH RESPECT TO THE MERCHANTABILITY OF THE SOFTWARE OR THE FITNESS OF USE OF THE SOFTWARE FOR ANY PURPOSE.

15. LIMITATION OF LIABILITY. Ventana's entire liability and your exclusive remedy shall be, at Ventana's option, either (a) return of the price paid or (b) repair or replacement of the Software. In no event will Ventana be liable for special, incidental, indirect or consequential damages or damages from loss of use, data or profits, or cost of procurement of substitute goods or services. In no event will Ventana’s total liability for any damages in any action in any form exceed the license fees paid for the licensed program that caused the damages.

16. GENERAL. This License will be governed by the laws of the Commonwealth of Massachusetts.

By clicking on the "Yes" button you acknowledge that you have read this License and understand it, and agree to be bound by its terms and conditions; and you agree that it is the complete statement of the agreement between us and supersedes any proposal or prior agreement, oral or written, and any other communications between us, relating to the subject matter of this License.

Index

T

U

V

W

X

Y