

# witnessOPTIMIZER

## WITNESS Optimizer Training Course Notes

### Course Objectives

To gain the maximum value from your simulation package it is essential that you be trained to use the package efficiently and effectively. This course is designed to give you that training.

The objectives of this course will enable you to do the following:

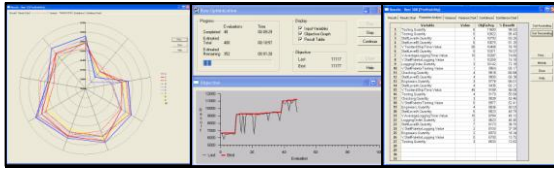
- to summarize uses of the Witness Optimizer Module
- to create models suited for the Witness Optimizer Module
- to understand the methods used to achieve an optimal solution
- to interpret data output from the Witness Optimizer Module

The overall goal of the course is to provide you with the capability to write WITNESS simulation models and update existing models to gain the best results from the Witness Optimizer Module.

Hands-on operations, conventional lecturing and group discussions are some of the various instructional techniques used to deliver the course and reinforce material. Instruction is provided by a Lanner Consultant/Trainer who has had considerable simulation project experience.

Prior WITNESS knowledge and usage are prerequisites for the WITNESS Optimizer course.

NOTES



## **Course Content**

### Introduction

- 1) Introduction to WITNESS Optimizer
  - a) An overview of how the optimizer is used
  - b) A discussion on which models to optimize
  
- 2) Preparing a model for Optimization
  - a) Defining an Objective Function
  - b) Checking for conflict in actions statements
  - c) More complex model setup – an introduction

Exercise One – A simple Objective Function

- 3) Starting the Optimizer Module
  - a) From Within WITNESS
  - b) From Outside WITNESS
  
- 4) Optimizer Experimentation Basics
  - a) Defining experimentation parameters (variables)
    - i) Adding parameters
    - ii) Modifying parameters
    - iii) Deleting parameters
  - b) Experimentation run properties
    - i) Warm up Period and Run Length
    - ii) Runs per Evaluation and Random Number Control
  - c) Choosing an Algorithm
  - d) Running an Optimization Experiment
  - e) Results Table

Exercise Two – A first simple experiment

### NOTES



- 5) Constraints
  - a) Adding Constraints
  - b) Modifying Constraints
  - c) Deleting Constraints
  - d) Re-evaluating constrained combinations

Exercise Three – Adding a constraint

- 6) Running replications
  - a) Specifying replications
  - b) Using the Variance and Variance Chart Tabs
  - c) Using the Confidence and Confidence Chart Tabs

Exercise Four – Running Replications

- 7) Analyzing Variability

- 8) Saving and Reloading Experiments
  - a) The Answer Pool
  - b) OPT Files

- 9) Defining Other Key Performance Indicators
  - a) Setting this up in WITNESS
  - b) The Tracking Tab

Exercise Five – A Production Line Model

NOTES



- 10) Different Optimization Methods and Procedure
  - a) Adaptive Thermostatistical Simulated Annealing  
Exercise Six – A First Intelligent Experiment
  
  - b) Six Sigma  
Exercise Seven – Applying the Optimizer to Six Sigma
  
  - c) Hill Climb
  - d) All Combinations, Random Solutions, Min/Mid/Max
  - e) Adding your own algorithm
  - f) Iterative running of the optimizer

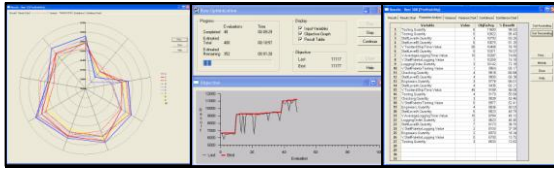
- 11) Copying and Printing Results and Output to MINITAB
  - a) Copying and Printing Results
  - b) Outputting to MINITAB

- 12) Exercise Eight – A full Case Study

Questions and Discussion

End of Course

NOTES



## 1) Introduction

### a) An Overview of how the Optimizer is used

The WITNESS Optimizer Module is geared to give you the power to be more productive with your WITNESS experimentation. It is a plug-in module to WITNESS that offers an easy-to-use experimentation and analysis tool with modern tabular and graphical reports.

It allows a variety of experimentation options including a unique algorithm, developed by Lanner in conjunction with best academic advice, which offers fast and effective performance in searching for optimal solutions. Within the algorithm, aspects of simulated annealing and tabu search are used.

Other methods include a basic hill climbing algorithm to search for local solutions, full factorial analysis and a six sigma algorithm that applies the simulated annealing algorithm but limits the amount of change from a base definition.

To begin an optimization, you must decide upon your objective, then transform that objective into an objective function. This is a normal function element within the WITNESS Model. Then, start the optimizer and decide which factors you would like to vary. Constraints may also be set with links between factors.

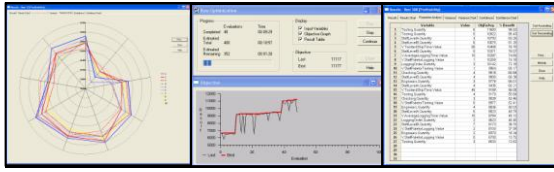
Then you must decide the run length, number of replications, and which algorithm will be used to carry out the optimization process.

As the optimization runs, numerous reports of the results are generated.

In summary, the whole optimization process can be broken down into the following 8 steps:

1. Build / Open a model for which the WITNESS Optimizer Module can be used
2. Decide upon an objective of the optimization process
3. Quantify the objective by writing a WITNESS function
4. Start the Optimizer Module
5. Decide which factors may vary
6. Determine run length, replications, and algorithm to be used
7. Optimize
8. Interpret results

NOTES



**b) A discussion on which models to Optimize**

There are many different types of simulation models built in WITNESS for many different purposes. It is usual to experiment with all models but not necessarily to define parameter ranges to vary, seeking to optimize against an objective function.

The WITNESS Optimizer is useful for that subset of models where there is a fair variety of options. If there are only 2 main designs for a facility then an optimization algorithm is strictly not necessary as only 2 sets of replications need to be run when searching for the best choice. Indeed the rule holds that if you have the time to run experiments for ALL the different options, whether 2 or 500, then this is the best choice. The optimizer or indeed the WITNESS Scenario Manager does allow you to do this, as a full factorial experiment, against a single WITNESS model. Where different models represent different design options then a good solution for proper scenario comparison is the WITNESS Scenario Manager Module.

The variety of choice needs to be structured for an optimization into a 'reasonable' number of choices. The clever simulated annealing algorithm searches for solutions by trying options and learning from the experience. The learning then guides the next choice of experiment. However if the number of solutions is too many then not enough can be learnt quickly enough by the algorithm to identify good/optimal solutions.

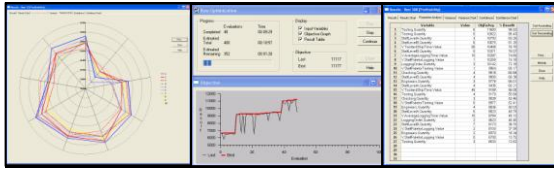
How many choices is reasonable is a difficult question as it depends on the nature of the 'solution space'(set of all possible outcomes). However most simulation experiments have relatively well behaved solution spaces and therefore they optimize fairly quickly.

The following table is general advice only with each model being different.

Number of Choices	Number of Experiments needed
20000	200 to 500
20,000,000	2,000 to 5,000
20,000,000,000	20,000 to 50,000
For every extra three 0's add one 0 to the number needed.	
Factors which add to the number needed	
<ul style="list-style-type: none"> <li>• Models with unusual behaviour – e.g. designs which 'click' with only a few settings</li> <li>• Models with few ranges – in general range parameters optimize faster than 'set value' parameters.</li> </ul>	

It is surprising how quickly the options grow. Even with just 15 parameters that can vary – each with only 5 values the number of options is  $5^{*}15$  or 30,517,578,125 which would require maybe 25,000 experiments as a minimum.

**NOTES**



All this means that the way in which the optimization problem is formulated is important. It is not good to expect the algorithm to do all the work – you normally have to simplify the choices somewhat – reducing the solution space in an intelligent way.

For some models this is not possible and the Optimizer should not be used to attempt complex scheduling where the parameters can vary enormously. However it is ideal for simple scheduling where the range of choices is reduced.

**Example - Optimizing manpower**

A model with different resource requirements may wish to optimize shift patterns.

Let us assume there are 30 people, each with a choice of 6 shift patterns.

(Lesson 1 here is that is a good idea to impose a choice of sensible shift patterns – allowing each person to have completely free choice of when to start and stop work would give many, many more options and would probably not be realistic).

Modeling each person choosing a different shift is 30 variables each with a choice of 6 options. This translates to  $6^{*}30$  choices

or 221073919720733357899776 options

However by simply changing this around to make the choice how many people are on each shift translates to  $30^{*}6$  choices

or 729000000 options

Further reducing this to a 'sensible' maximum of only up to 20 on each shift will reduce the number further to  $20^{*}6$  choices

or 64000000 options

It is further possible to constrain input values so that the total over all shifts is 30 or less which will constrain much further.

Or 592525 options (number of options evaluated by the Optimizer – see later!!)

**NOTES**



## 2) Preparing a model for Optimization

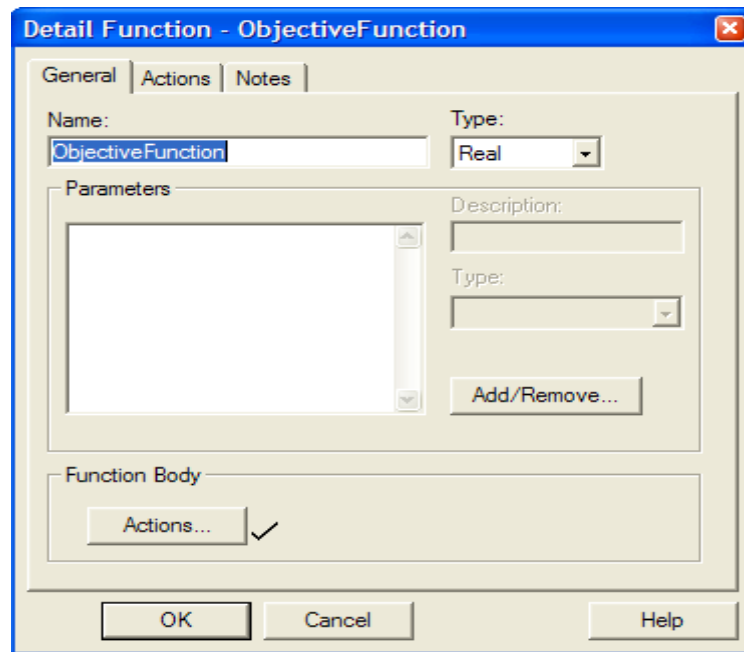
### a) Defining an Objective Function

Before a model can be optimized, an objective function must be defined. This function quantifies the aim of the optimization.

During the optimization process the model will be run with different combinations of the values that are specified in the optimization variables. At the end of each run the objective function will be called to obtain the result of that run. The value of this result determines if the combination of values used was better or worse than other combinations. The optimizer's task is to find the combination of values that provides the best result.

The objective function is a normal WITNESS function. It is defined in the same way as any other user defined function. It takes no parameters and must return a numeric result, either integer or real.

The Function Detail Dialog Box is seen below:



NOTES



In general, the objective function will be a combination of some of the current values of the optimization variables and some results of the simulation.

### Objective Function Example

In a simple model of a production process there is a choice of the number of machines and the number of staff to employ. The aim of the optimization is to produce the highest possible throughput with the lowest possible cost. The objective function could be expressed as:

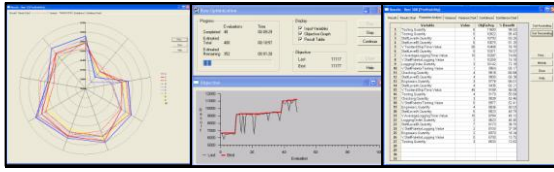
**value of throughput - cost of machines - cost of staff**

Choosing values for the value of each produced item and the relative costs of machines and staff might yield the following WITNESS expression:

**100 \* NSHIP(Item) - 1000 \* NQTY(Mch001) - 200 \* NQTY(Staff)**

This would then be used in the objective function. Note that changing the relative values of the cost and value figures would result in more or less emphasis being placed on throughput as opposed to cost and so may result in a different optimum solution.

NOTES



## b) Checking Actions Statements

There are a number of points in a WITNESS model where action statements can be executed that might change the model. The most common place for these is **Initialize Actions**. You must ensure that any such actions are not in conflict with the changes being made by the optimizer during the optimization process. Examples of actions that may cause problems are 'SET CAPACITY' and 'SET QUANTITY'.

Suppose the optimizer is varying the capacity of a buffer to determine its effect on throughput. It will be setting the capacity to different values and then running the model. If the **Initialize Actions** has a 'SET CAPACITY' statement for that buffer then it will override the required value and invalidate the result of the evaluation. In this example, the 'SET CAPACITY' statement must be removed before the optimizer is started.

This conflict does not arise when setting the value of WITNESS variables as these are handled differently – being reset by the optimizer following Initialize actions.

### Optimizer methodology for running the WITNESS model

- Open/Begin the model
- Set the parameter values that have been specified by the optimizer
- Run Initialize actions
- Reset all variable value settings specified by the optimizer
- Batch run the model for the desired run time
- Calculate the results by running the objective function (and any other tracked functions)

### NOTES



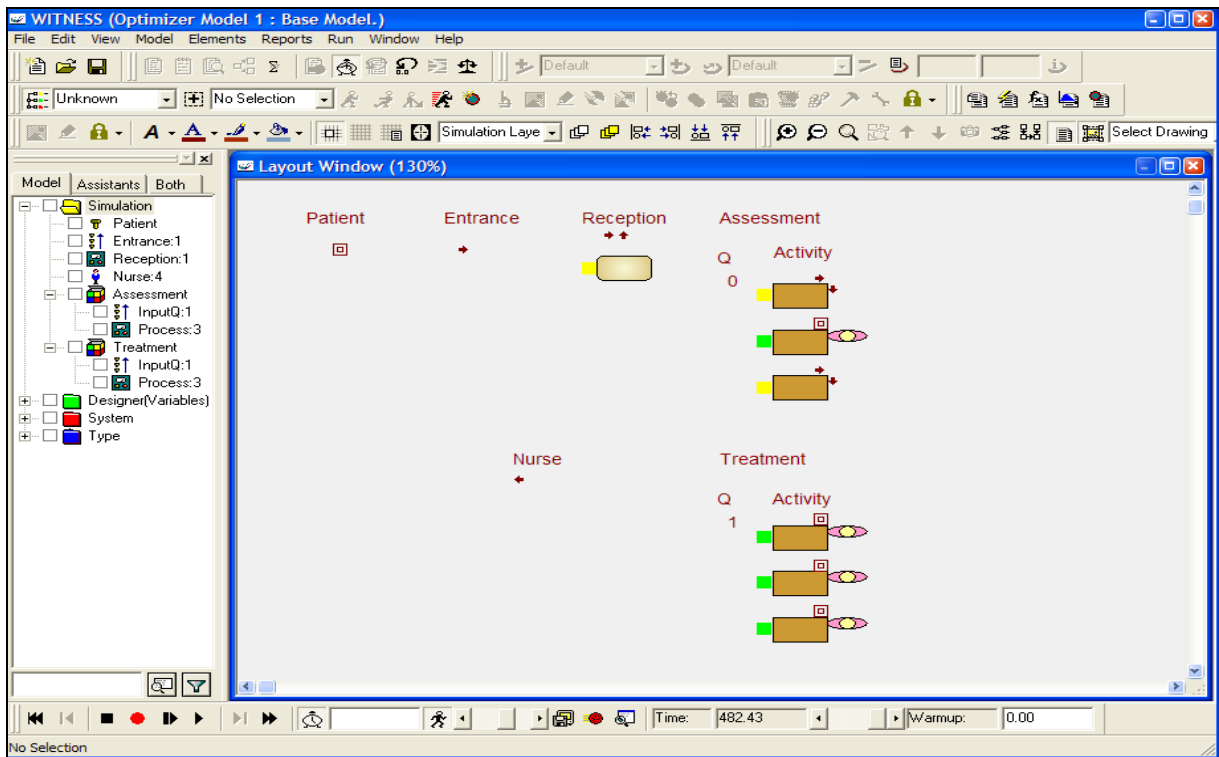
### Exercise One - A simple Objective Function

Open the model **SimpleClinic.Mod** in the Optimizer Training Course Materials Directory. This model is from the tutorial manual of the Service and Process Edition of WITNESS and represents a simple hospital clinic process.

Patients arrive according to an arrival profile. From reception they are guided to an assessment room or a treatment room. After assessment some patients will be immediately sent for treatment. After treatment the patients leave.

A key measure of success in such a model may be the amount of time that patients spend in this process. For this first simple example let us assume that cost is no object and that all that is required is to minimize the patient time in the model. Obviously there will be treatment and assessment times but the rest of the time is waiting time and this is what we would like to be as little as possible.

Run the model for a while and observe the flows and fluctuating queues.

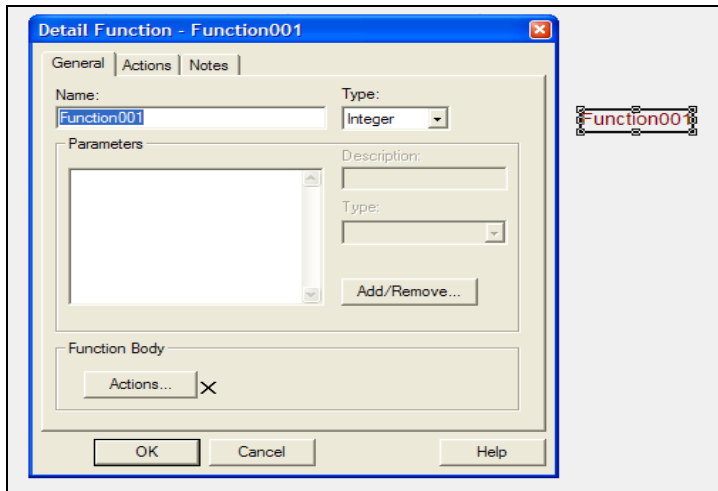


### NOTES

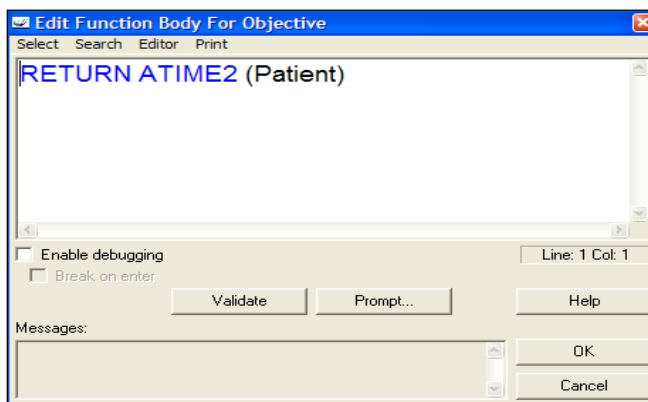


Define a new WITNESS function called Objective from the designer elements window.

- i) Open the designer elements window from the windows menu
- ii) Select the Variables Tab
- iii) Select Function to obtain the box surround
- iv) Click on the modelling window – once to place and once again to confirm placement
- v) Double Click on the new Function001 element to bring up the element detail



- vi) Change the name to Objective
- vii) Change the type to Real
- viii) Enter the actions section in the function body
- ix) Enter the following text in the box and click OK



This function will return the average time that all patients, who have left, spent in this process.

This is an ideal objective function for this model.

Save your model as **SimpleClinic with Objective.MOD** – we will use this later.

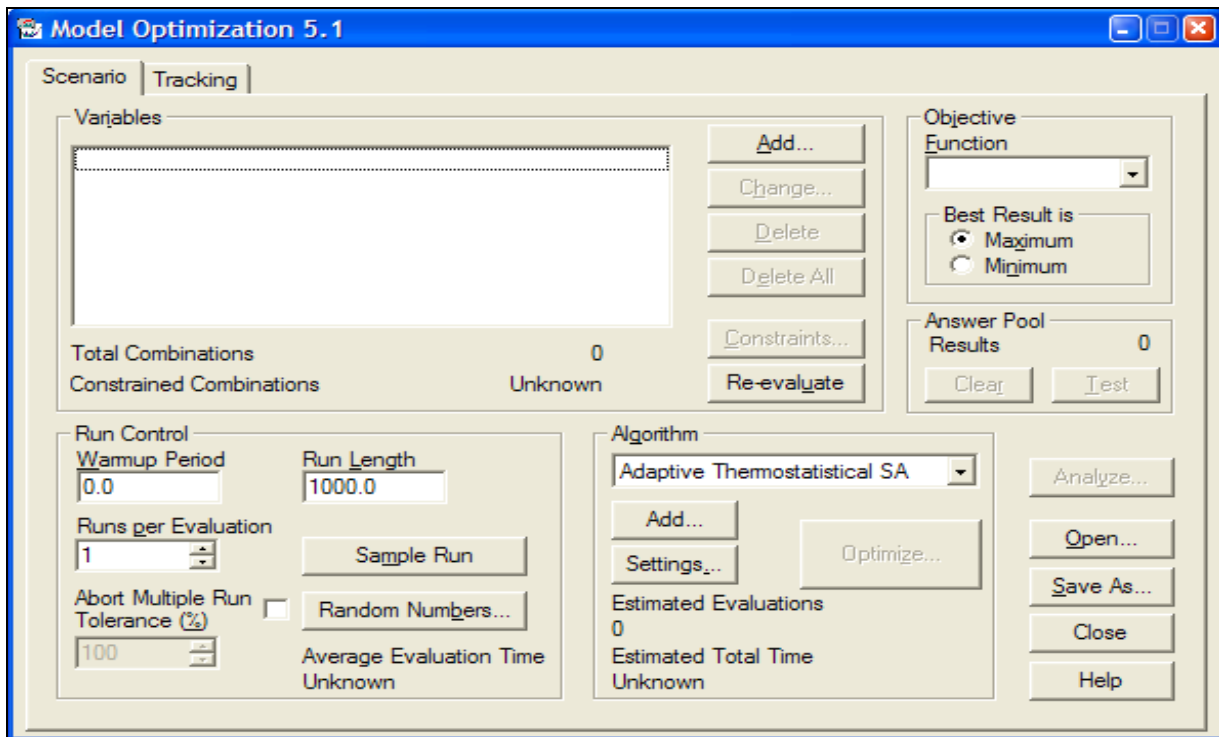
**NOTES**



### 3) Starting the Optimizer

#### a) From Within WITNESS

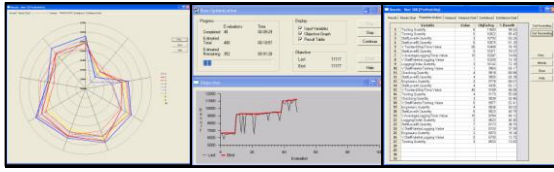
To start Optimizer, select the **Optimizer** option from the WITNESS **Model** menu. The **Model Optimization** dialog is displayed:



#### b) From Outside WITNESS

This topic is not covered fully in this course but there are methods that enable setup, controlling and running of an optimization from an application such as Excel using VBA or from a program written in VB, C, C++, C#, etc. These methods use COM control. A file detailing these methods is available on request.

#### NOTES



## 4) Optimizer Experimentation Basics

### a) Defining Experimentation Parameters

Most elements within WITNESS have factors that are automatically offered by the WITNESS Optimizer as factors that can vary. Examples include:

Part (Entity)	Maximum Arrivals, Inter Arrival Time, First Arrival At, Lot Size
Buffer (Queue)	Quantity, Capacity
Machine (Activity)	Quantity, Priority, Cycle Time/Duration
Conveyor	Quantity, Index Time, Length, Priority, Maximum Capacity
Labor (Resource)	Quantity
Track	Quantity, Zone, Capacity, Maximum Speed, Physical Length
Vehicle	Quantity, Capacity, Loaded/Unloaded Speed, Start/Stop Delay
Variable	Value
(Options are also included for other elements such as Power and Free, Continuous, etc)	

Note: Since a variable value can be changed, anywhere a variable can be used can also vary. This is extremely powerful. For example a model can be setup to employ a different control rule based on the value of a variable being 0 or 1 (say) or even read in a different dataset from a different file – again an IF condition being used in actions language in conjunction with different values of a variable

An example of code for an Output rule in a model might be:

```

IF VariableA = 0
Push to D,E
Else
Push to E,D
Endif
    
```

In the Optimizer setting the value of VariableA to be either 0 or 1 would vary this rule.

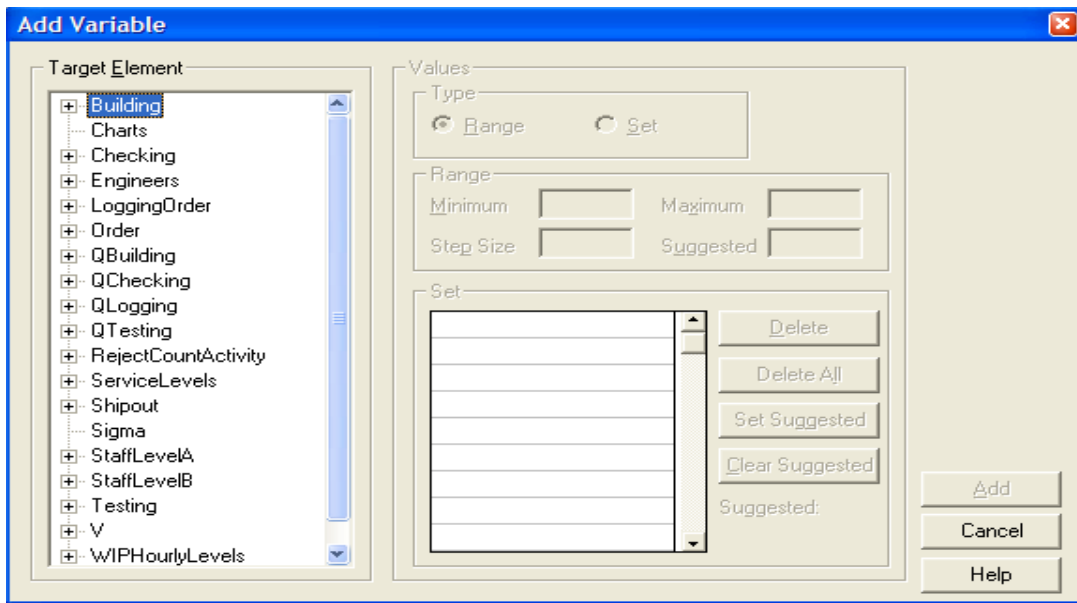
**NOTES**



### Adding Parameters (Variables)

An optimization variable is a combination of the item in the model that is to be varied and the values to be used.

To add new variables to the existing set of variables, click the Add button on the **Model Optimization** dialog. The **Add Variable** dialog is displayed:



NOTES

To specify a new variable:

- Select the target element for the variable by expanding the required model element and clicking the appropriate property.
- Specify the type of element value by clicking either the Range or the Set radio button.

### Range value type

This specifies that the target element should take values within a range. Specify a value for each of the following fields:

- Minimum: the minimum value in the range.
- Maximum: the maximum value in the range.

Step size: the interval between each allowed value within the range. If nothing is specified then a value of one is assumed.

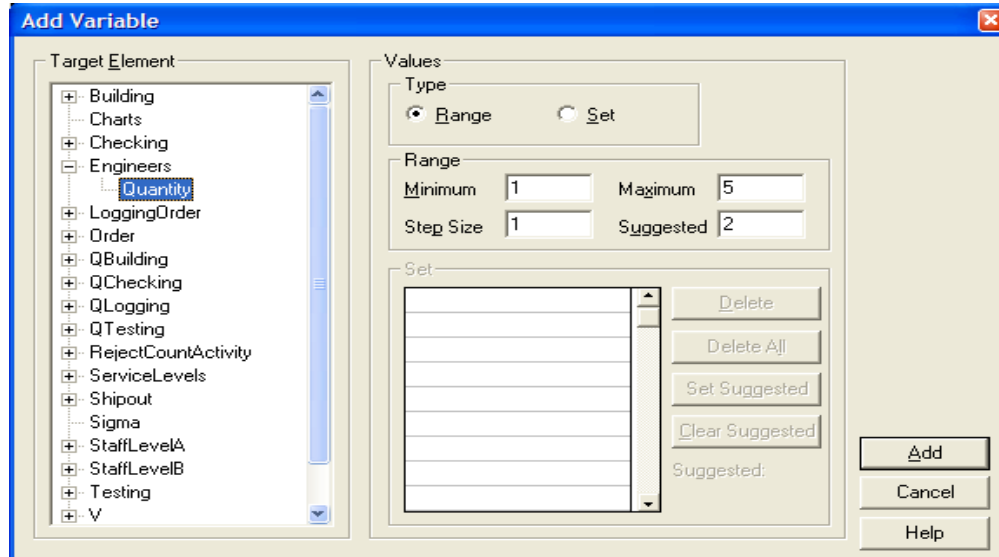
Suggested: the value that you believe is most suitable for the model. This parameter is optional.

### Set value type

This defines a set of discrete values for the target element. Add values to the set by selecting a set field and entering a value. Delete set values by using the Delete or the Delete All buttons. Specify a set value as a suggested value by selecting the value and clicking the Set Suggested button. Clear suggested values by clicking the Clear Suggested button.

Overleaf is an example **Add Variable** dialog:

**NOTES**



In this example, there will be 1, 2, 3 4 or 5 engineers, and the suggested number is 2.

Add the new variable to the optimization file by clicking the Add button. The **Add Variable** dialog closes and you are returned to the **Model Optimization** dialog. The optimization file is automatically updated and the new variable is added to the list of existing variables. You will be warned if you attempt to add a variable that is already in use.

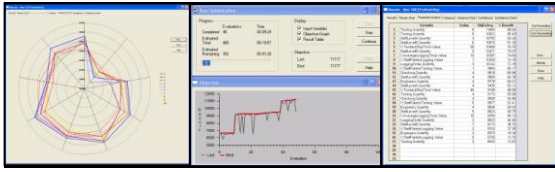
### Modifying Parameters

To modify the definition of an existing variable, select the variable to be modified and click the Change button on the **Model Optimization** dialog. Alternatively, double-click the variable to be modified. The **Change Variable** dialog is displayed, showing the current definition of the selected variable.

To modify the variable:

- If you wish to change the variable's element, select a new target element by expanding the required model element and clicking the appropriate property.
- Change the type of element value by clicking either the Range or the Set radio button.

### NOTES



**Range value type**

Specify a new value for each of the following fields:

Minimum: the minimum value in the range.

Maximum: the maximum value in the range.

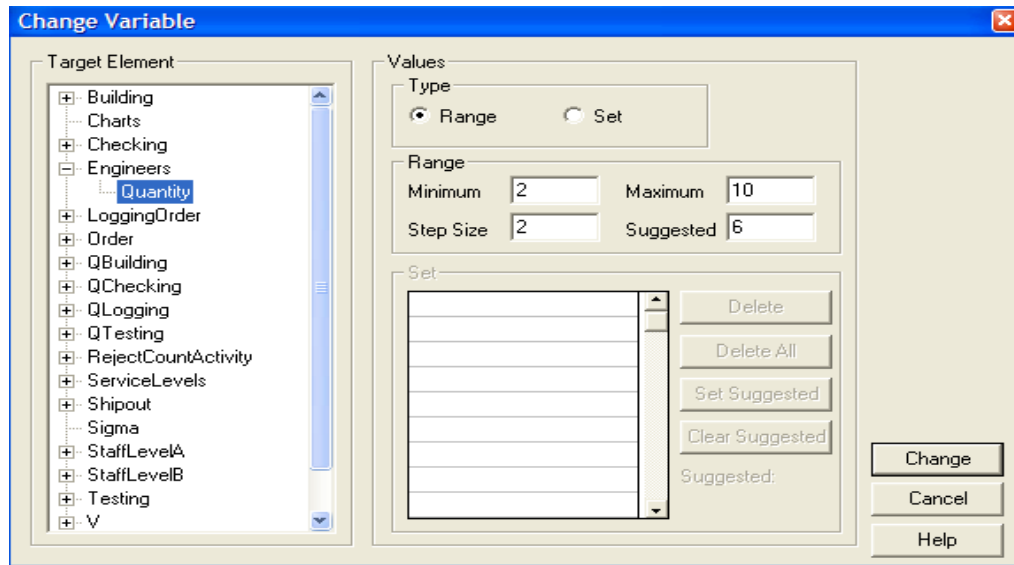
Step size: the interval between each allowed value within the range.

Suggested: the value that you believe is most suitable for the model.

**Set value type**

Change set values by selecting a set field and overwriting the existing value. Delete set values by using the Delete or the Delete All buttons. Clear a suggested value by clicking the Clear Suggested button or change it by selecting the required value and clicking the Set Suggested button.

Here is an example **Change Variable** dialog:



In this example, the variable shown in the previous **Add Variable** dialog has been changed so that there are now 2, 4, 6, 8 or 10 engineers, with the suggested number 6.

Replace the existing variable with the changed variable by clicking the Change button. The **Change Variable** dialog closes and the **Model Optimization** dialogs are automatically updated with the details of the changed variable.

**NOTES**



### Deleting Parameters

You can delete variables from the optimization scenario either singly or totally. To delete a single variable:

- Select the variable to be deleted.
- Click the Delete button on the **Model Optimization** dialog. The variable is removed permanently from the list of variables.

To delete all variables from the optimization scenario, click the Delete All button on the **Model Optimization** dialog. All variables are removed permanently from the list of variables.

NOTES



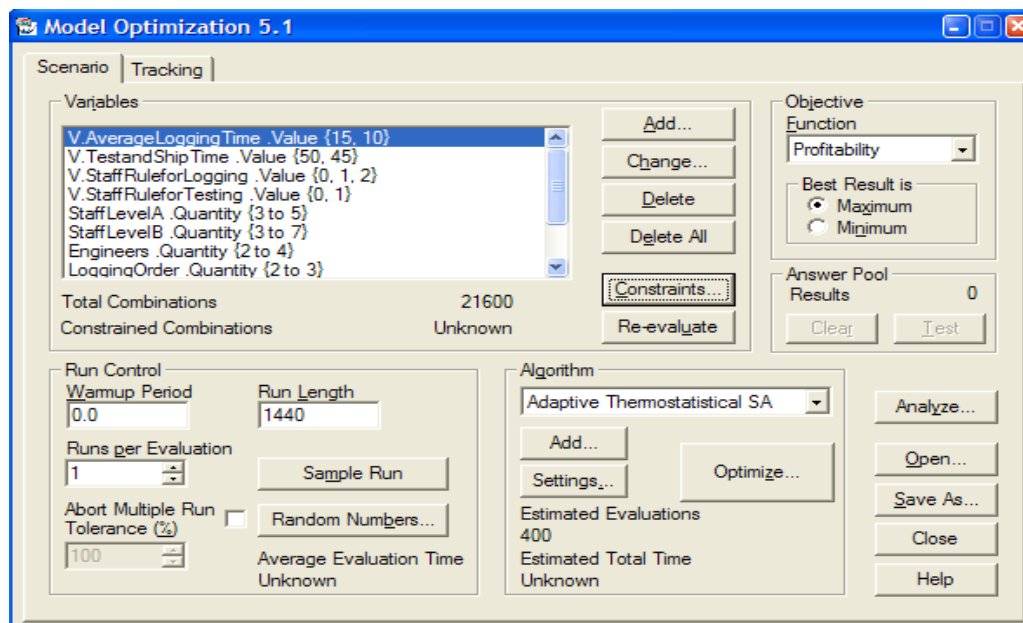
## b) Experimentation Run Properties

### Warm up Period and Run Length

#### Specifying the warm-up period

The warm-up period is the time from the start of a simulation run during which the model reaches a steady state, and after which the collection of statistics about the run will start.

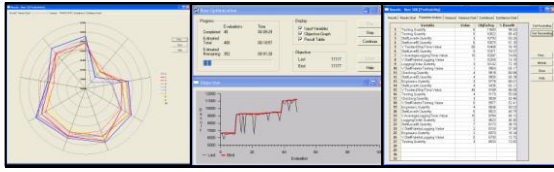
To specify a warm-up period, enter the required time in the Warm-up Period field of the **Model Optimization** dialog (refer below). The units are simulation time units and the default is zero.



#### Specifying the length of the run

Specify the length of a run by entering the value in the Run Length field of the **Model Optimization** dialog (refer above). The units are simulation time units and the value you enter depends on your model. For example if you are trying to optimize daily productivity and the simulation time equates to minutes then the run length would probably be set to 1440 (one day expressed in minutes). If your model can produce a reliable result from a shorter run then set the shorter length as this will reduce the time taken to run an optimization.

#### NOTES



## Runs Per Evaluation and Random Number Control

### Specifying the number of runs per evaluation

When evaluating a particular combination of variables the optimizer can run the WITNESS model one or more times. If more than one run is performed for each evaluation then the WITNESS random number streams are varied for each run so different results will be obtained. The overall result for the evaluation is taken as the mean of the results for each run.

Increasing the number of runs per evaluation will increase the time taken to perform an optimization but may result in a final solution that is less susceptible to random variations.

To specify the number of runs per evaluation, click the Runs per Evaluation spin button control of the **Model Optimization** dialog as required (or simply overwrite the existing number).

If you set the number of runs to be greater than one, then the results of the optimization run will include variance and confidence statistics. You can use the **Analyze Variability** facility to help you decide on the number of runs per evaluation.

NOTES

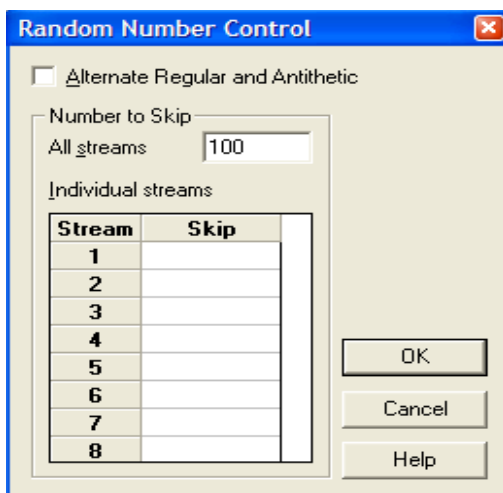


### Random number Generation

When performing more than one run for each evaluation the WITNESS random number streams are varied to produce different model situations. You can control how the streams are altered for each run using the **Random Number Control** dialog. You can specify that the random number streams alternate between regular and antithetic sampling. You can also control how many random numbers are skipped at the start of each run. This can be specified for all streams and for individual streams. For more details on random number streams refer to the main WITNESS documentation.

To control the generation of random numbers for the optimization run:

1. Click the Random Numbers button on the **Model Optimization** dialog. The **Random Number Control** dialog is displayed:



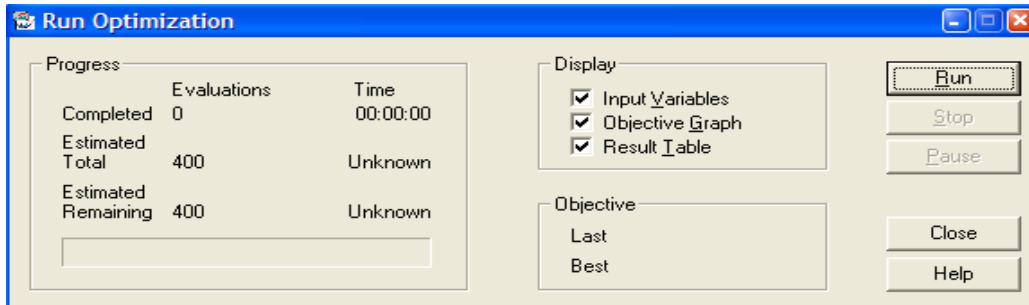
2. Select the Alternate Regular and Antithetic check box or leave unselected. If you select this check box, regular sampling is applied to one run, then antithetic sampling applied to the next run, and so on.
3. Specify the number of samples to be discarded for all random number streams. The default number is 50. This means that on the first run no numbers are skipped, on the second run the first 50 numbers are skipped from each stream, on the third run the first 100 numbers are skipped from each stream, and so on.
4. Specify the number of samples to be discarded for individual random number streams. This value overrides any value specified for all streams.
5. Click the OK button. The **Random Number Control** dialog closes and you are returned to the **Model Optimization** dialog.

NOTES



### d) Beginning the Simulation Experiments

Once a method has been chosen the Optimize button can be pressed. This takes you to the Run Optimization Control dialog.



This dialog has check boxes – these will be used to display different displays whilst the experiments are run:

- In the **Input Variables** window, which lists all variables in the current optimization scenario and shows their values as different combinations are evaluated.
- In the **Objective** window, which graphically displays the result of each evaluation.
- In the **Results** dialog, which contains tables that list details of the best results of the optimization run, and charts that graphically summarize these same results.

To initiate the optimization run, click the Run button on the **Run Optimization** dialog. Depending on how you specified the display of results, the **Input Variables** window, the **Objective Graph** window and the **Result** dialog are automatically displayed.

At any stage during the execution of the optimization run, you can click the Stop button to halt the run. The **Input Variables** window closes. You can also click the Pause button to temporarily halt the run; the button title changes to Continue, allowing you to resume the run at any time.

#### NOTES



### Algorithm Random Numbers

A number of the algorithms make use of random numbers. The most obvious one is **Random Solutions** but **Adaptive Thermostatistical SA** and **Hill Climb** also use random numbers to determine choices.

These random numbers are separate from the WITNESS random number streams and are produced by the optimizer's own pseudo random number generator.

When the **Run Optimization** dialog is displayed the random number generator is reset to a base value. This means that when the first time the Run button is clicked the optimization algorithms start out with the same random numbers. In this way the optimization run is always reproducible.

However, if the optimization run is stopped and restarted, or re-run after it has finished, then the random number is randomized. This means that a second optimization run can differ from the first. When the **Run Optimization** dialog is closed and subsequently opened again the random number generator is again reset to the base value.

### NOTES



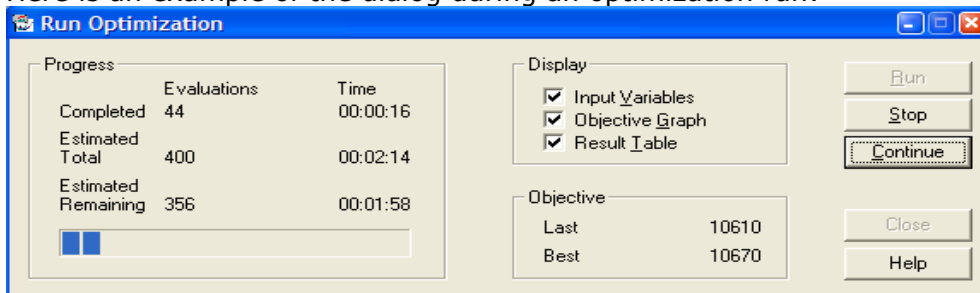
**e) Viewing the Run and Results**

You can view progress as the optimizer runs from any of the open windows and dialogs.

**Run Optimization dialog**

In addition to stating the total number of runs and the estimated time to completion, this dialog dynamically updates the number of runs that have been completed and the time taken, and the number remaining to be completed and the time to be taken. It also dynamically updates the last result and the current best result displayed in the Objective area.

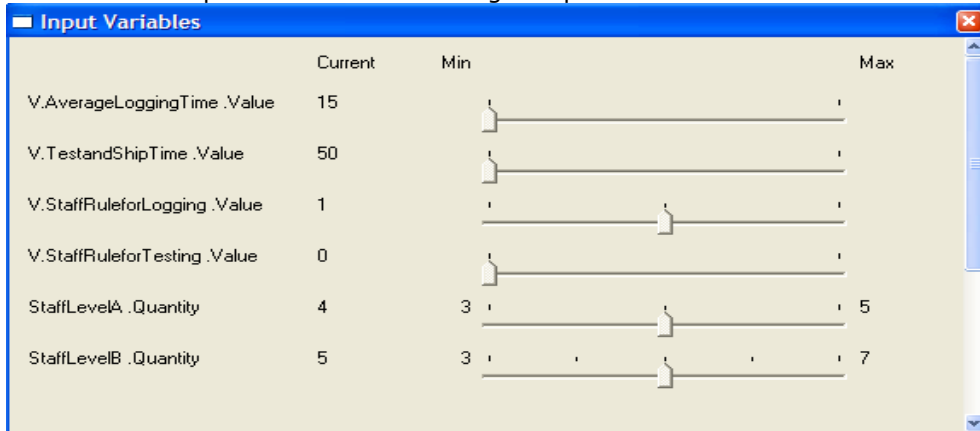
Here is an example of the dialog during an optimization run:



**Input variables window**

This window lists all variables in the current optimization file. As the run is executing, it dynamically displays the value that each variable is taking. The column titled Current lists the current value of each variable. Min and Max indicate the range of each variable defined as having a range, with a pointer indicating the current value.

Here is an example of the window during an optimization run:



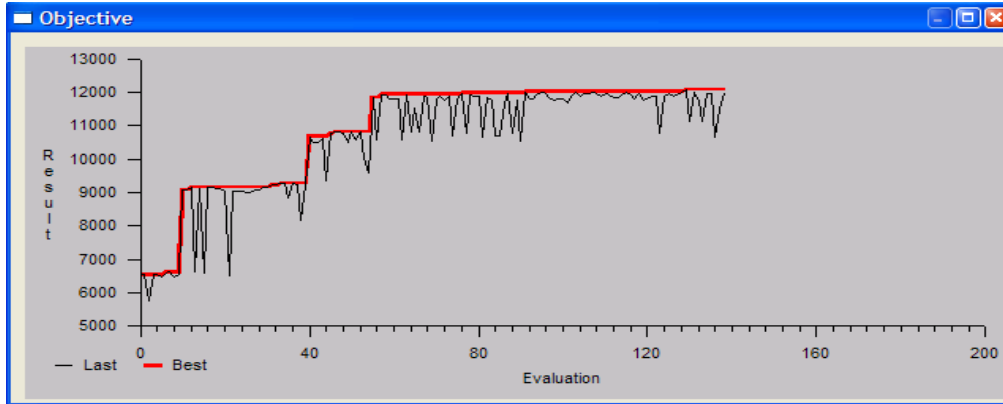
When the run completes, this window automatically closes.

**NOTES**



**Objective graph window**

This window graphically displays the result of each evaluation, and identifies the best result. Here is an example of the window during an optimization run:



Note that all the tables and charts displayed by the optimizer during the run are separate program items – each window will have a separate display on the taskbar and clicking on the WITNESS display screen will make all of the other windows disappear. They can be brought back by using the taskbar buttons.

Whilst the optimizer runs the experiments the WITNESS window time display will update as normal – this gives a good idea of model progress especially for long model runs.

**NOTES**



**Results Table**

The **Results** table contains tabs that list details of the best results of the optimization run, and charts that graphically summarize these same results. You specify the number of results to be kept on the **Model Optimization** dialog.

As the run progresses, the results table and chart are continuously updated. The number of results listed is increased until the number of best results is reached, and the list of best results is updated as new results replace those that are less successful.

The nature of some of the optimization algorithms means that they can attempt to evaluate a combination of variables more than once. If this happens the previously determined result is used, rather than running the WITNESS model again. These duplicate runs are not shown in the results table even if there are empty rows or worse results that could be replaced.

If you have set the number of runs per evaluation (in the Run Control area of the **Model Optimization** dialog) to greater than 1, the **Results** dialog also contains a Variance table and chart, and a Confidence table and chart. These are similarly updated as the run progresses. Details of how to use these tables is covered in chapter 6.

Here is an example of the **Results** dialog during an optimization run:

Evaluation	Profitability	V AverageLogging Time .Value	V TestandShip Time .Value	V .StaffRuleforLogging .Value	V .StaffRuleforTesting .Value	StaffLevelA .Quantity	StaffLevelB .Quantity	Engineers .Quantity	L
1	0	6540	10	45	2	0	4	7	4
2	1	6538	15	45	2	0	4	7	4
3	2	5770	10	50	2	0	4	7	4
4	3	6540	10	45	0	0	4	7	4
5	4	6540	10	45	2	1	4	7	4
6	5	6480	10	45	2	0	5	7	4
7	6	6600	10	45	2	0	4	6	4
8	7	6620	10	45	2	0	4	7	3
9	8	6490	10	45	2	0	4	7	4
10	9	6540	10	45	2	0	4	7	4
11	10	9080	10	45	2	0	4	7	4
12	11	9080	10	45	2	1	4	7	4
13	12	9160	10	45	2	1	4	7	3
14	13	6640	10	45	2	1	4	7	2
15	14	9100	10	45	2	1	5	7	3
16	15	6580	10	45	2	1	5	7	2
17	16	9160	10	45	2	1	5	6	3
18	17	9160	10	45	2	0	5	6	3
19	18	9100	10	45	2	0	5	7	3
20	19	9100	10	45	1	0	5	7	3
21	20	9050	10	45	1	0	5	7	3
22	21	6540	10	45	1	0	5	7	3

In this example the Objective Function is "Profitability" and is shown in the second column.

The buttons on the right hand side of the screen enable manipulation of the table including ordering the rows (e.g. click at the top of column 2 and press sort descending and this will bring the best result to the top)

**NOTES**



To use the Results table to modify the model parameters:

1. Click the **Results** dialog to bring it to the top, and if necessary, click the Results tab to display the Results table. If required, expand the window so that the complete table is displayed.
2. Select any vertical column and click either the Sort Ascending or Sort Descending buttons to reorder the evaluations. For example, if you select the Result column and click the Sort Ascending button, the evaluations will be ordered starting with the lowest numerical result. If you have defined the best result as Minimum (from the **Model Optimization** dialog), the topmost result will be the best.
3. Select the best evaluation and click the Set Suggested button to update the optimization scenario by setting the suggested value of each variable to the value it had in the selected evaluation. You will be prompted to override any existing suggested values.
4. Select the best evaluation and click the Set Model button to update the WITNESS model with the selected evaluation's parameters. You can choose whether the model should be left at time zero or whether to run it to the end of the run.

**Note** If any of the target elements are WITNESS variables, there is no permanent place in the variable definition to store they value of the variable. If you wish to alter the model to permanently reflect the selected evaluation then you will need to add statements to **Initialize Actions** to set the WITNESS variables accordingly.

You can also view the Results chart by clicking on the Results Chart tab.

**NOTES**



**The Parameter Analysis screen**

You can only access the Parameter Analysis table whilst the Optimizer is not running an algorithm. When the algorithm has stopped, the Parameter Analysis table shows an assessment of the effect that each value of each parameter has on the Objective Function.

Calculation: For each parameter value tried, the average Objective Function value is calculated and reported. This is a simple average of all the experiments carried out with that parameter value that have been recorded in the results table. (You can change the number of results recorded in the table on the Model Optimization dialog's Tracking page).

The effect calculated is then also reported as a % Benefit column in the table – this is calculated as the improvement in value from the worst value of the Objective Function in the table.

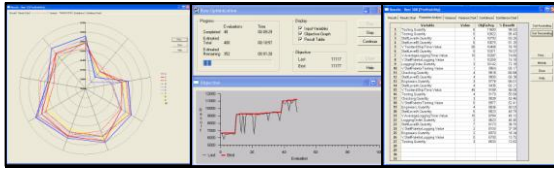
In this way, the parameters are ranked for effect on the model. The best results from this screen will come where the number of experiments using each parameter value option are roughly equal, as in this case all correlation effects are also taken into account. When using the heuristic algorithms there may be very few experiments with some parameter values, which can skew the results.

Example Table:

	Variable	Value	ObjFnAvg	% Benefit	Number Of Experiments
1	Engineers.Quantity	4	5258	52.05	3
2	StaffLevelA.Quantity	8	4431	28.15	3
3	StaffLevelA.Quantity	9	4371	26.41	3
4	Engineers.Quantity	3	4338	25.45	3
5	StaffLevelA.Quantity	10	4311	24.68	3
6	Engineers.Quantity	2	3518	1.74	3
7					

In this experiment it can be seen that the changes in value of the quantity of engineers is much more significant than the changes in the quantity of staff level A. This is seen by the difference in % benefit between the different levels.

**NOTES**



### Exercise Two – A first simple experiment

Open the model created in Exercise One – **SimpleClinic with Objective.Mod**

Select Optimize from the Model Menu.

Using the methods from chapter 4 above add the following parameters as variables that can change – they should all be range parameters

Parameter	Lower Limit	Upper Limit	Step value
Nurse.Quantity	1	5	1
Assessment.Process.Quantity	1	5	1
Treatment.Process.Quantity	1	5	1

- After adding these parameters it can be seen that there are 125 different experiments that can be run
- Set a run time of 10,000 time units and choose the algorithm “All Combinations”
- Make sure that the objective function (“Objective”) is chosen and that “Minimize” is the option chosen beneath this.
- Then run the optimization by pressing the Optimize button and then the run button
- Watch the results build up as the optimization runs
- At the end of the run select the objective column on the results table and use the “Sort Ascending” button to order the results
- See if your results matches the table on the next page

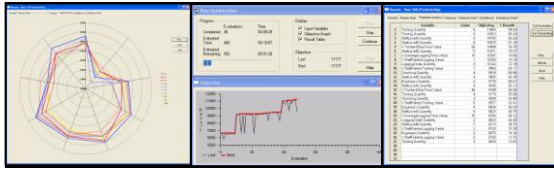
**NOTES**



	Evaluation	Objective	Nurse .Quantity	Assessment .Process .Quantity	Treatment .Process .Quantity
1	119	25.128	5	4	5
2	124	25.437	5	5	5
3	114	25.592	5	3	5
4	109	25.658	5	2	5
5	118	26.937	5	4	4
6	123	27.149	5	5	4
7	108	27.633	5	2	4
8	113	27.822	5	3	4
9	104	27.824	5	1	5
10	103	28.277	5	1	4
11	94	35.78	4	4	5
12	89	36.073	4	3	5
13	99	36.91	4	5	5
14	93	37.237	4	4	4
15	88	37.975	4	3	4
16	98	38.379	4	5	4
17	84	38.992	4	2	5
18	83	40.45	4	2	4
19	78	42.885	4	1	4
20	79	44.834	4	1	5
21	77	77.765	4	4	2

Unsurprisingly the experiments all show the best results include 5 nurses. The results also indicate that 4 or 5 treatment rooms is an important factor and that really any number of assessment rooms provide substantially the same result.

NOTES



## 5) Constraints

Constraints enable you to reduce the number of possible combinations within a model in a controlled way. You can do this for two reasons. The first is to formulate additional conditions that apply to the scenario. For example, if optimization variables allow between 3 and 5 staff of one type, and between 3 and 7 of another type, you can apply a constraint condition that at any time there is to be no more than a total of 10 staff.

The second is to quantify your experience of what contributes to a good solution. For example, suppose you have a variable for the number machines of a certain type and another variable for the number of staff who can operate such machines. If you know from experience that an operator can practically manage a maximum of three machines then you can apply a constraint that the number of machines is never greater than three times the number of operators.

Constraints are always expressed as linear equalities or inequalities combining two or more of the optimization variables. They are of the form:

$$a*v1 + b*v2 + c*v3..... \leq d$$

or

$$a*v1 + b*v2 + c*v3..... = d$$

or

$$a*v1 + b*v2 + c*v3..... \geq d$$

where v1, v2 & v3 are variables and a, b, c & d are constants

The first example above would have the form:

$$staff1 + staff2 \leq 10$$

The second example can be expressed initially as:

$$machines / 3 \leq operators$$

manipulating this to a linear inequality we get:

$$machines - (operators * 3) \leq 0$$

You can manage constraints in the following ways:

- Add a constraint
- Modify an existing constraint.
- Delete a single constraint or all constraints.

To manage constraints, click the Constraints button on the **Model Optimization** dialog. The **Optimization Constraints** dialog is displayed, with the list of variables displayed in the Factors list. If any constraints have already been defined, they will be listed in the Constraints list. When you select a constraint, its factors and condition are automatically displayed.

NOTES



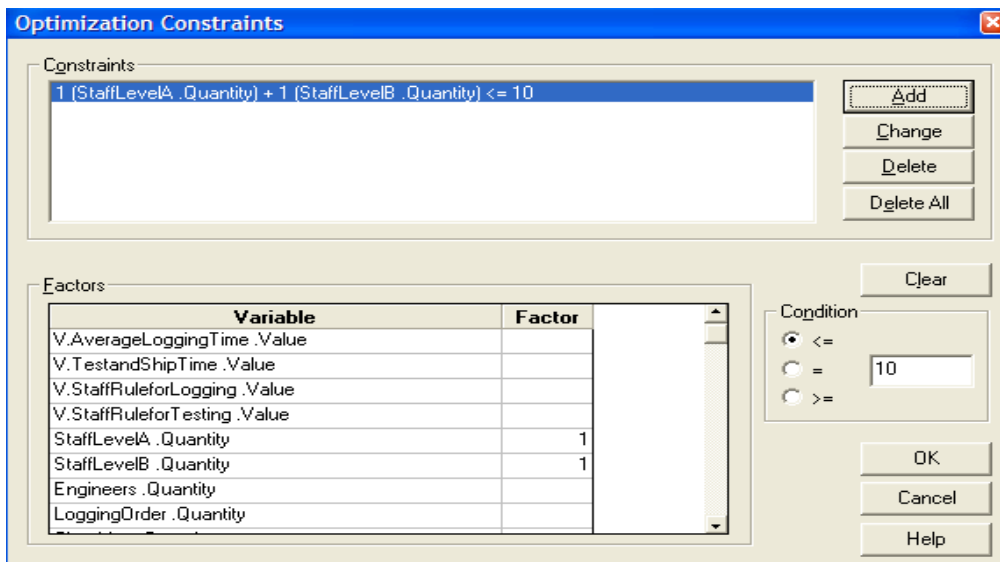
**a) Adding constraints**

To add constraints to one or more of the variables in the current optimization file:

1. Clear existing factors and condition by clicking the Clear button.
2. For each variable to be included in the constraint enter the appropriate factor next to the variable in the factor list. You must enter at least one factor.
3. Enter the constraint condition by selecting the required radio button and entering a numeric string. The default condition is = 0.
4. Click the Add button. The constructed constraint is added to the Constraints list.

Repeat the procedure to add additional constraints.

Here is an example constraint that limits the total of two types of staff to 10:



NOTES



### b) Changing constraints

To change a constraint:

- From the Constraints list, select the constraint to be modified. The variable(s) and its factor(s) are displayed in the Factors list, and the condition is displayed.
- Modify the existing constraint factor(s) and/or the existing constraint condition or click the Clear button, then enter the new factor(s) and condition.
- Click the Change button. The variable and its modified constraint displayed in the Constraints list are automatically updated.

### c) Deleting constraints

To delete a single constraint, select the constraint to be deleted and click the Delete button. The constraint is removed from the Constraints list. To delete all constraints click the Delete All button. You are prompted to remove all constraints from the Constraints list.

When you have completed adding, modifying and deleting constraints, click the OK button. The **Optimization Constraints** dialog closes, you are returned to the **Model Optimization** dialog. The **Model Optimization** dialog will now show the number of constrained combinations as unknown since it has not re-evaluated the constraints you have added or changed.

### d) Re-evaluating constrained combinations

After adding, modifying and deleting variables and their constraints, you can determine the resulting number of constrained combinations. To do this, click the Re-evaluate button on the **Model Optimization** dialog. All possible combinations of variables are evaluated against the constraints to arrive at a figure for the number of constrained combinations. For very large combinations this process can take some time and therefore it is not done automatically every time you adjust the constraints.

It is not necessary to re-evaluate the constrained combinations before running an optimization; the constraints will be checked as the optimization proceeds. However, it is useful to know the number of constrained combinations as this can help with choosing an algorithm and with estimating more accurately the time it will take to perform an optimization.

NOTES

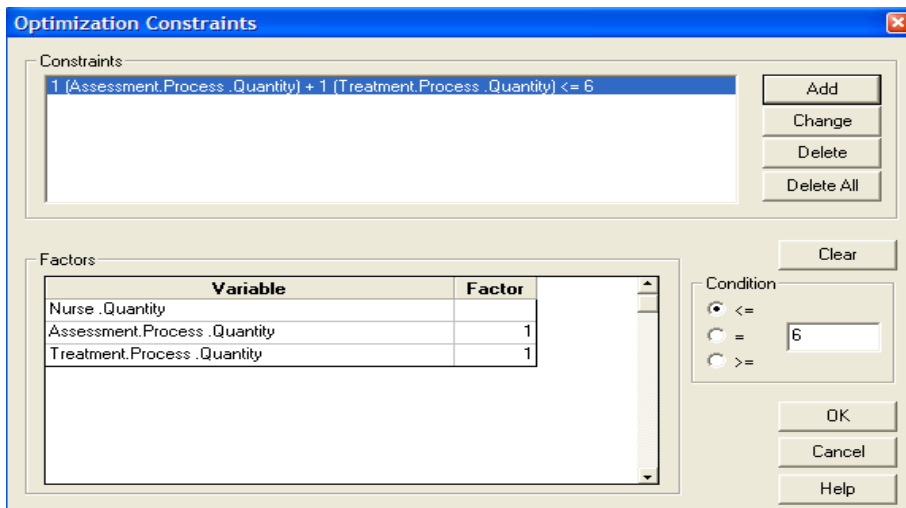


### Exercise Three – Adding a constraint

Use the model created in exercise 2.

In the modeling scenario it is decided that there are only 6 possible rooms available for treatment and assessment. Add a constraint to the optimization.

Add a constraint to the model using the method in chapter 5 that ensures no more than 6 rooms are used in any experiment.



Re-evaluate how many constrained options there are. There should be 75.

Then run the experiment again, sort as before, and see if your results match those overleaf.

#### NOTES



Results - Best 50 [Objective]

Results | Results Chart | Parameter Analysis

	Evaluation	Objective	Nurse .Quantity	Assessment .Process .Quantity	Treatment .Process .Quantity
1	68	27.633	5	2	4
2	64	27.824	5	1	5
3	63	28.277	5	1	4
4	53	40.45	4	2	4
5	48	42.885	4	1	4
6	49	44.834	4	1	5
7	62	72.765	5	1	3
8	47	72.765	4	1	3
9	67	82.394	5	2	3
10	71	85.185	5	3	3
11	52	106	4	2	3
12	56	116.612	4	3	3
13	33	546.502	3	1	4
14	34	560.951	3	1	5
15	38	598.028	3	2	4
16	32	598.988	3	1	3

Sort Ascending  
Sort Descending  
Set Model  
Set Suggested  
Print...  
Minitab  
Close  
Help

Note that the total number of rooms used never exceeds 6.

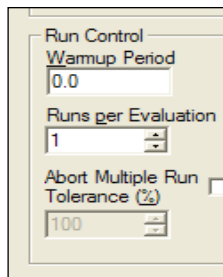
NOTES



## 6) Running replications

### a) Specifying replications

To set a number of replications use the up and down arrows or edit the number field called "Runs per Evaluation" on the main optimizer control screen



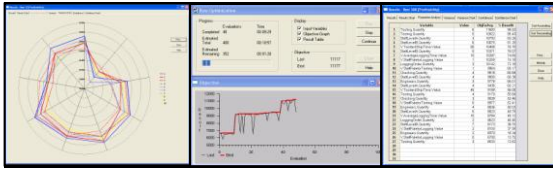
The number of replications needed for a model will depend on the variation inherent in the run. The more varied the model results the more replications needed to be confident of the result.

If a model reaches an equilibrium position and is a non-terminating model then a single long run may be all that is needed. Other models need replications. To some extent this can be done iteratively with a small number of replications indicating whether a larger number may be of use.

There is much material in standard textbooks (such as "Simulation" by Professor Stewart Robinson – chapter 9) to help determine how long to run/how many replications to run for a simulation model. We cannot include all the material here. However if a number of replications is used with a suitable model then the WITNESS Optimizer will use a standard statistical test (a t-test) to determine confidence intervals for key statistics. (see section c below).

There is also an analysis tool built in to the optimizer to help you assess model variability. This is included in section 7 below.

NOTES



**b) Using the Variance and Variance Chart Tabs**

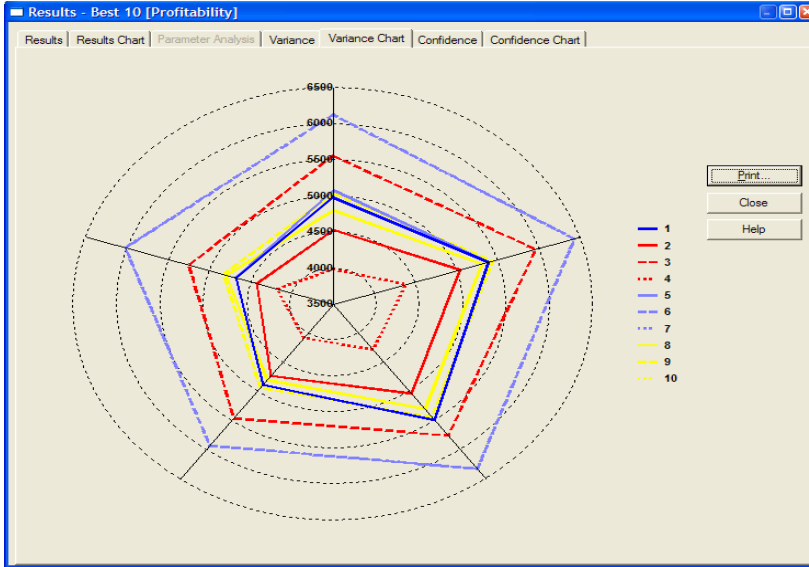
1. Click the **Results** dialog to bring it to the top, and click the Variance tab to display the Variance table. If required, expand the window so that the complete table is displayed. Here is an example of the table during an optimization run:

	Evaluation	Mean	Std. Dev.	Best	Run 1	Run 2	Run 3
1	0	6506.667	57.735	6540	6540	6440	6540
2	1	6538	0	6538	6538	6538	6538
3	2	5670	100	5770	5770	5670	5570
4	3	6506.667	57.735	6540	6540	6440	6540
5	4	6506.667	57.735	6540	6540	6440	6540
6	5	6446.667	57.735	6480	6480	6380	6480
7	6	6566.667	57.735	6600	6600	6500	6600
8	7	6586.667	57.735	6620	6620	6520	6620
9	8	6456.667	57.735	6490	6490	6390	6490
10	9	6473.333	57.735	6540	6540	6440	6440
11	10	9013.333	57.735	9080	9080	8980	8980
12	11	9013.333	57.735	9080	9080	8980	8980
13	12	9093.333	57.735	9160	9160	9060	9060
14	13	6706.667	115.47	6840	6640	6640	6840
15	14	9033.333	57.735	9100	9100	9000	9000
16	15	6646.667	115.47	6780	6580	6580	6780
17	16	9093.333	57.735	9160	9160	9060	9060
18	17	9093.333	57.735	9160	9160	9060	9060
19	18	9033.333	57.735	9100	9100	9000	9000

2. Select any vertical column and click either the Sort Ascending or Sort Descending buttons to reorder the evaluations. For example, if you select the column displaying the mean value of the run results, and click the Sort Ascending button, the evaluations will be ordered starting with the lowest numerical result. If you have defined the best result as Minimum (from the **Model Optimization** dialog), the topmost result will be the best.

The Set Suggested and Set Model button function in the same way as on the **Results** table.

NOTES



An example of a variance chart showing the top ten sets of readings from the variance table (note this changes when the table is sorted).

**c) Using the Confidence Table and Chart tabs**

1. Click the **Results** dialog to bring it to the top, and click the Confidence tab to display the Confidence table. If required, expand the window so that the complete table is displayed. Here is an example of the table during an optimization run:

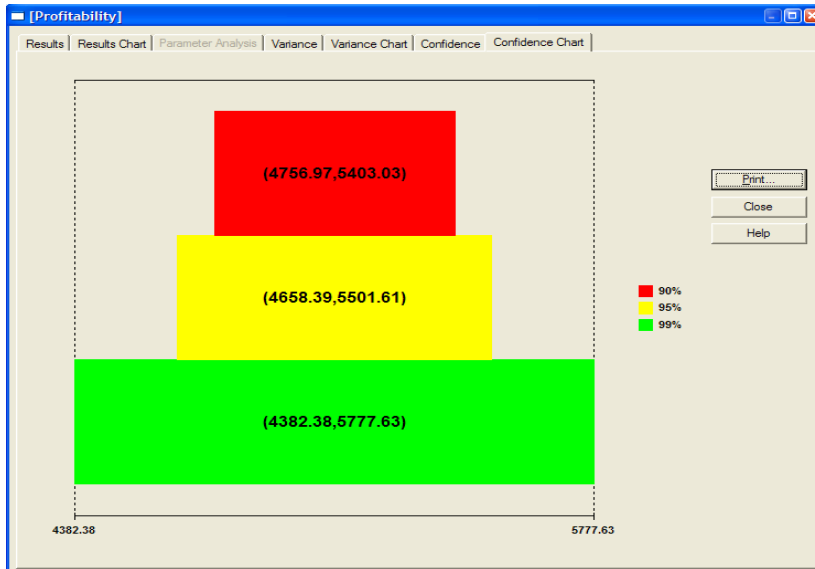
	Evaluation	Mean	90% Min	90% Max	95% Min	95% Max	99% Min	99% Max
1	0	6600	6455.536	6744.464	6411.451	6788.549	6288.013	6911.987
2	1	6658	6487.6	6828.4	6435.6	6880.4	6290	7026
3	2	5810	5612.472	6007.528	5552.193	6067.807	5383.414	6236.586
4	3	6600	6455.536	6744.464	6411.451	6788.549	6288.013	6911.987
5	4	6600	6455.536	6744.464	6411.451	6788.549	6288.013	6911.987
6	5	6540	6395.536	6684.464	6351.451	6728.549	6228.013	6851.987
7	6	6660	6515.536	6804.464	6471.451	6848.549	6348.013	6971.987
8	7	6680	6535.536	6824.464	6491.451	6868.549	6368.013	6991.987
9	8	6550	6405.536	6694.464	6361.451	6738.549	6238.013	6861.987
10	9	6600	6426.958	6773.042	6374.152	6825.848	6226.294	6973.706
11	10	9140	8966.958	9313.042	8914.152	9365.848	8766.294	9513.706
12	11	9140	8966.958	9313.042	8914.152	9365.848	8766.294	9513.706
13	12	9220	9046.958	9393.042	8994.152	9445.848	8846.294	9593.706
14	13	6920	6631.073	7208.927	6542.902	7297.098	6296.026	7543.974
15	14	9160	8986.958	9333.042	8934.152	9385.848	8786.294	9533.706
16	15	6860	6571.073	7148.927	6482.902	7237.098	6236.026	7483.974
17	16	9220	9046.958	9393.042	8994.152	9445.848	8846.294	9593.706
18	17	9220	9046.958	9393.042	8994.152	9445.848	8846.294	9593.706
19	18	9160	8986.958	9333.042	8934.152	9385.848	8786.294	9533.706

**NOTES**



2. Select any vertical column and click either the Sort Ascending or Sort Descending buttons to reorder the evaluations. For example, if you select the 90% Minimum column, and click the Sort Ascending button, the evaluations will be ordered starting with the lowest numerical result. If you have defined the best result as Minimum (from the **Model Optimization** dialog), the topmost result will be the best.

The Set Suggested and Set Model button function in the same way as on the **Results** table. You can also view the Confidence chart by clicking on the Confidence Chart tab. The chart displayed is for the row that is selected on the confidence table.



These confidence intervals use the Student's T Test / Statistic. This assumes that the samples come from a Normal distribution. It is your responsibility to confirm the validity of this assumption in any particular instance.

Note – The sorting of the results tab, the variance tab and the confidence tab is not linked. Each sort order is maintained individually.

NOTES



### Exercise Four – Running Replications

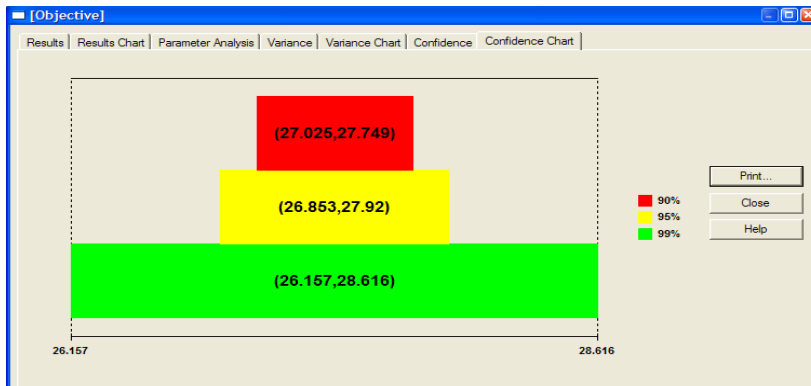
Use the model created and the optimization experiment set up in exercise 3

Change the number of replications (Runs per evaluation) to 3 and run the experiment again and check your results against the table below – remembering to sort the column in the variance table by the mean column.

(If you get prompted as to whether you should clear the answer pool – answer YES – this is covered later in the course).

Results	Results Chart	Parameter Analysis	Variance	Variance Chart	Confidence	Confidence Chart	
	Evaluation	Mean	Std. Dev.	Best	Run 1	Run 2	Run 3
1	68	27.387	0.215	27.239	27.633	27.288	27.239
2	64	28.186	0.318	27.824	27.824	28.417	28.319
3	63	28.507	0.293	28.277	28.277	28.408	28.837
4	53	37.26	2.764	35.581	40.45	35.75	35.581
5	48	42.283	1.217	40.882	42.885	40.882	43.081
6	49	46.046	1.059	44.834	44.834	46.79	46.513
7	62	68.115	5.117	62.585	72.765	62.585	68.996

Variance of each run shown



Confidence intervals fairly narrow – however beware only 3 results!!

### NOTES



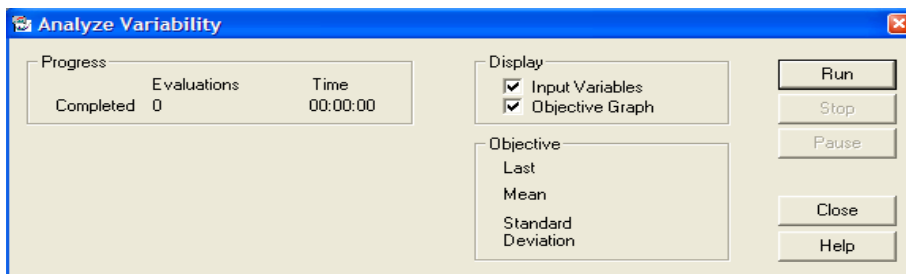
## 7) Analyzing Variability

There is an additional tool provided in the WITNESS Optimizer to help you assess the variability of a model. This is the Analyze button on the main Optimizer dialog.

As stated above a stable model will be affected less than an unstable model by the random number changes. This button runs the model repeatedly at a chosen level and charts the variability found.

**Chosen values:** If a suggested value is set then this is used for each parameter setting. Otherwise for a range the midpoint is used and for set values a value is chosen at random (but not changed for each run).

1. Click the Analyze button on the **Model Optimization** dialog. The **Analyze Variability** dialog is displayed:



2. Check the Input Variables and Objective Graph checkboxes as required.

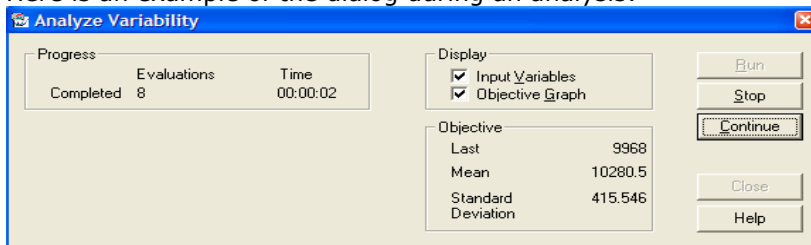
To initiate the analysis, click the Run button on the **Analyze Variability** dialog. At any stage during the execution of the analysis, you can click the Stop button to halt the analysis. The **Input Variables** window if open will close. You can also click the Pause button to halt the analysis temporarily; the button title changes to Continue, allowing you to resume the analysis at any time.

### NOTES

**Viewing progress**

You can view progress from any of the open windows and dialogs.

Here is an example of the dialog during an analysis:

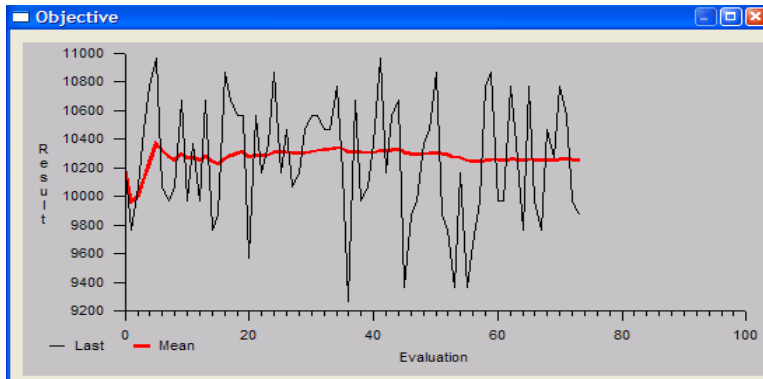


**Input variables window**

This window lists all variables in the current optimization scenario and the values they have been set to for the analysis. (as for the optimization experiments)

**Objective graph window**

This window graphically displays the result of each evaluation and the mean of all results to that point. Here is an example of the window during an analysis:



The mean line indicates the Moving Average of the results up to that point. When the line 'settles down' it indicates stability after that number of experiments. Note that this is rough analysis but useful nonetheless.

**Interpretation**

The key results are the mean and the standard deviation from the main dialog and the visual appreciation given in the objective chart graph. Obviously the higher the variation the more replications need to be run. Again attention is drawn to simulation textbooks (e.g. "Simulation" by Professor Stewart Robinson – Chapter 9)

**NOTES**



## 8) Saving and Reloading Experiments

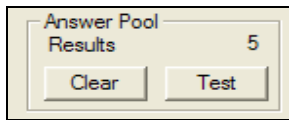
### a) The Answer Pool

The results of any optimization run are automatically stored in an answer pool when you run any experiments. The answer pool is not erased when you start a new optimization run unless it needs to be. Reasons why it may need to be include e.g. the addition of a new model parameter where the results tables would not know the parameter value for the experiments run to date.

The main purposes of the answer pool are:

- To speed up subsequent runs of the optimization scenario. OPTIMIZER only runs WITNESS if the results for a selected set of optimization variable values are not found in the answer pool.
- To allow you to suspend an optimization run and resume it later, which allows you to back up the results so far and to carry out other tasks on your computer.

If you change the model in such a way that would change the results of experiments run to date then you **MUST** manually clear the answer pool before running any experiments. This is done from the main optimizer dialog using the clear button. If you are unsure whether the model has changed then you can use the test button to automatically run an experiment at random and compare the results with those from the answer pool. This is not a foolproof test as a single successful test cannot guarantee that all answers are the same.



When an experiment is saved (see b below) the answer pool is saved with the experiment setup. This is a huge timesaver in recreating results as and when needed.

If the main optimizer dialog is quitted without saving then the answer pool is lost.

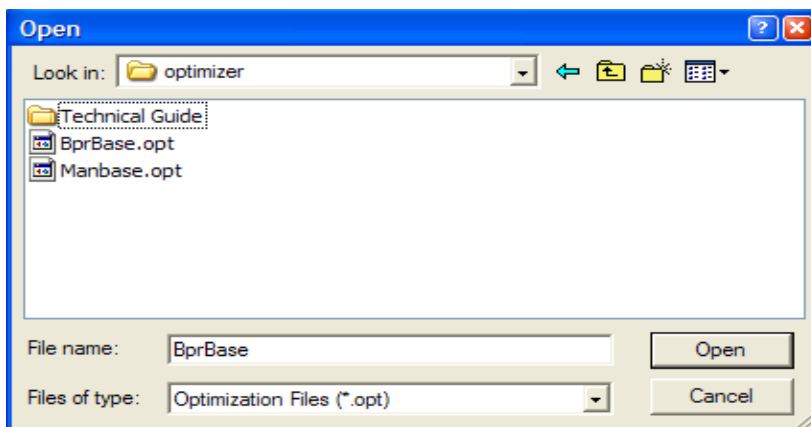
NOTES



### b) Saving and Restoring Optimizer experiments

To save an optimizer experiment use the Save As button on **the Model Optimization** dialog. Experiments are saved to a .OPT file as standard. This file stores all the settings from the main optimizer dialog and the answer pool.

To select and open a saved optimization scenario, click the Open button on **the Model Optimization** dialog. The standard Windows **Open** dialog is displayed:



A .OPT file does not store the results pages from an experiment – these must be recreated by running an experiment again – however this is quick and easy as the results tables will be populated from the answer pool and the model will not need to be run.

#### NOTES



## 9) Defining Other Key Performance Indicators

### a) Setting this up in WITNESS

It is possible to define other results than the objective function to be recorded in the results tables (and answer pool). This is done by defining other functions in WITNESS as normal function elements (just as for the answer pool).

All other functions which are defined and tracked are shown in the results table in italics in columns to the far right of the table. In addition details of Variance and Confidence intervals may also be obtained for the tracked functions – simply select the function of interest on the variance tab in the results. After a choice is made the name of the function selected will be shown in the title bar of the results window when viewing the appropriate tabs (Variance, variance Chart, Confidence, Confidence Chart).

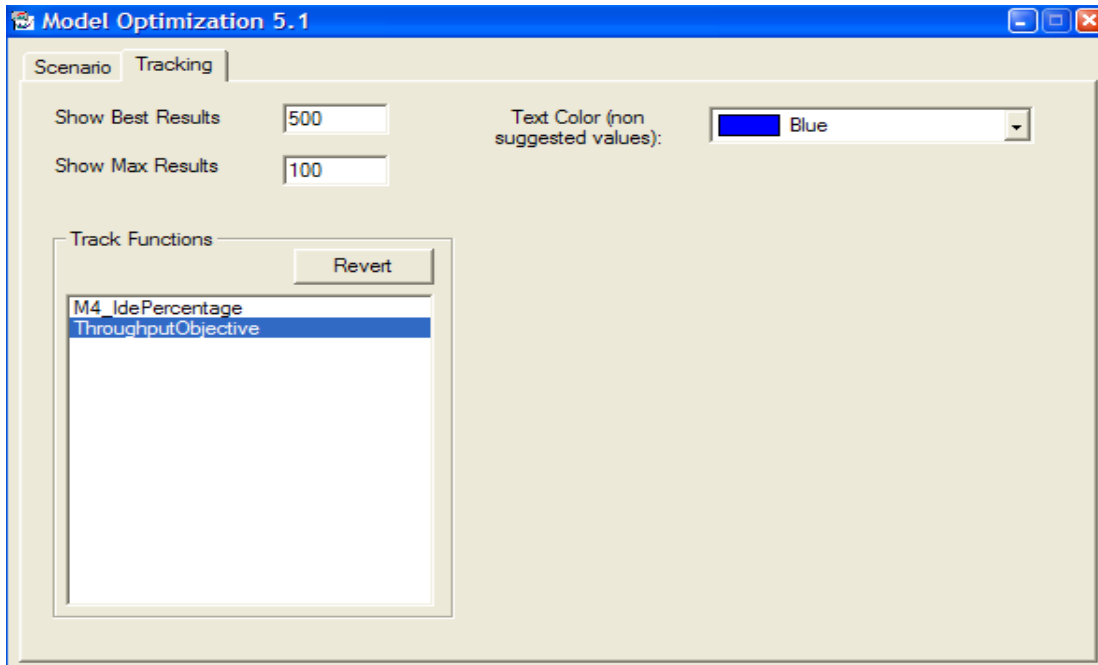
	Evaluation	Mean	Std. Dev.	Best	Run 1	Run 2	Run 3
1	0	551.667	2.309	553	553	553	549
2	1	551.667	2.309	553	553	553	549
3	2	551	2	553	551	553	549
4	3	551.667	2.517	554	552	554	549
5	4	554	2.646	556	555	556	551
6	5	549	1	550	548	550	549
7	6	551.333	2.082	553	552	553	549
8	7	555.333	4.041	559	556	559	551
9	8	556.667	4.163	560	560	558	552
10	9	555.667	4.163	559	557	559	551
11	11	555.333	1.155	556	556	556	554
12	12	559.333	3.215	563	557	563	558
13	13	554	2	556	554	556	552
14	14	551.333	2.082	553	552	553	549
15	15	551	2	553	551	553	549
16	17	552	2.646	554	553	554	549
17	18	554.333	2.082	556	555	556	552
18	19	550.333	1.528	552	552	550	549
19	21	554	2.646	556	555	556	551

NOTES



**b) The Tracking Tab**

The tracking tab (the second tab of the main optimizer dialog) offers the following options:



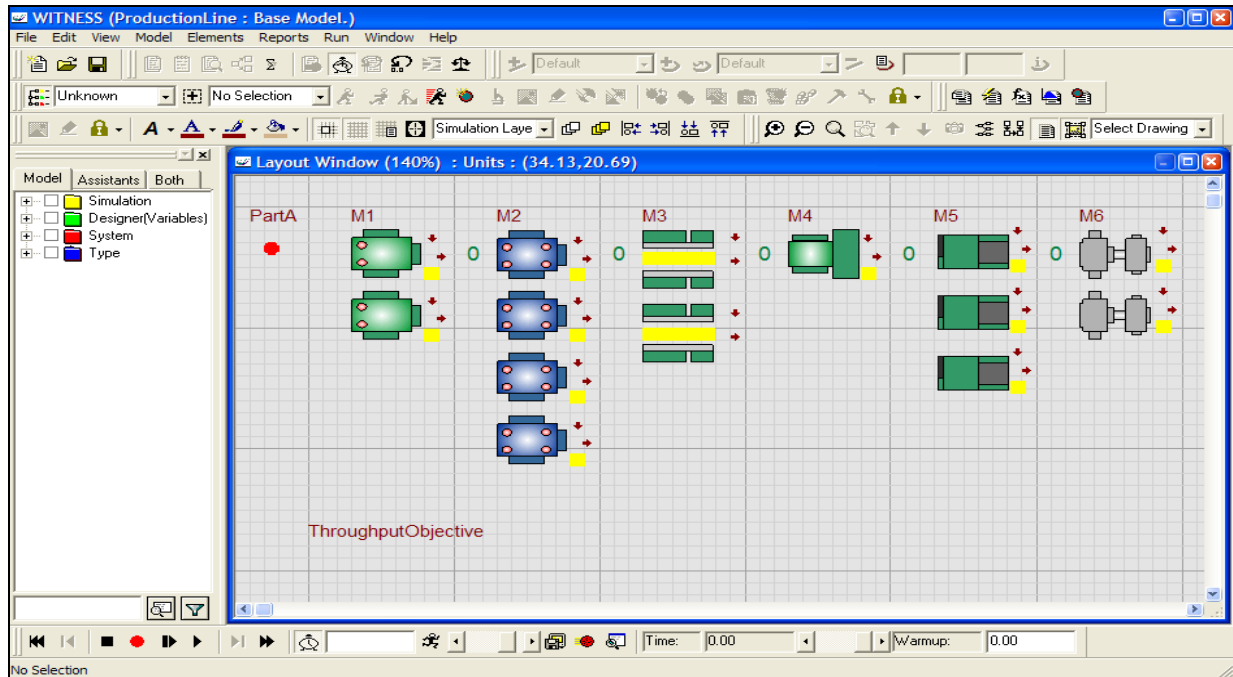
- Alteration of the number of best results in the results tables (the number of rows of results created)(between 1 and 10,000)
- Alteration of the number of maximum results showed on the Objective Function Chart (between 100 and 1,000)
- Selection of additional functions to track and record on the results screen. Simply click on a function to select or deselect. Click and drag to select or deselect more than one function. (Note the objective function chosen cannot be deselected).
- The text color to display non-suggested values in the results tables (see later in the Six Sigma algorithm topic in section 10).

NOTES



## Exercise Five – A Production Line Model

Open the model **ProductionLine.mod** from the Optimizer Training Course Materials Directory.



This is a model where PartA is being manufactured sequentially through M1, M2, M3, M4, M5 and M6 machines. There are differing quantities of each machine.

M1 pulls parts into production from an unlimited supply of raw materials and each other machine position is preceded by a buffer of limited capacity.

Each machine has a different variable cycle time and breakdown frequency and duration.

There is an objective function defined for this line which is simply the throughput achieved in a model run.

### NOTES



Look at the objective function in the model.

It simply returns the number of goods shipped.

```

Edit Function Body For ThroughputObjective
Select Search Editor Print
RETURN NSHIP (PartA)
    
```

Initially this model is to be used to optimize the buffer capacities in front of each set of machines. The total space is limited by the factory size so the problem is how best to carve up what is available.

There are other factors of interest too. With the optimum configuration what is the free time on machine M4. This is not expected to be a bottleneck and, if free, could be used for other vital test procedures.

Create another function for the model called M4\_IdlePercentage and return the idle percentage for this machine (SUTIL(M4,1))

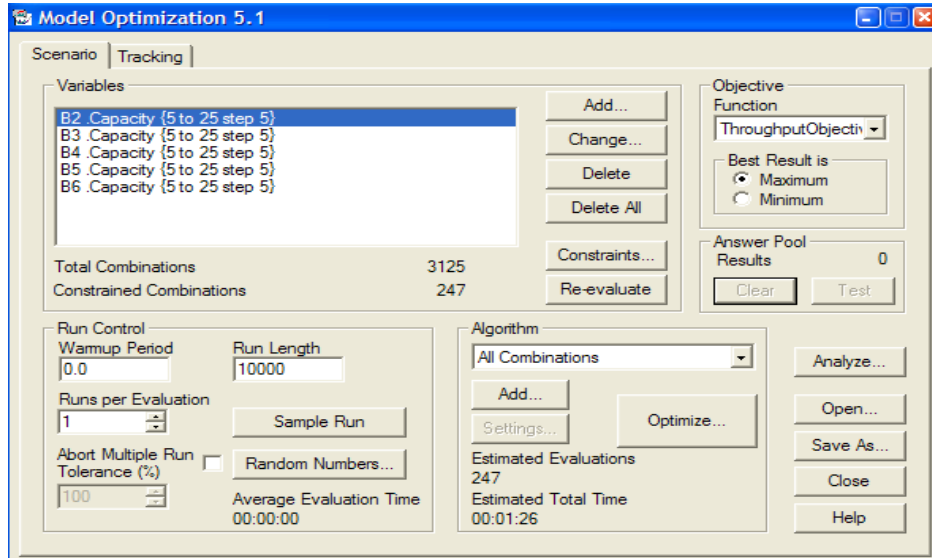
Save the model as ProductionLinePlus.mod and then begin the optimizer.

Setup an experiment where:

- i) Each of the buffer capacities (B2 to B6) can vary in steps of 5 between 5 and 25
- ii) There is a constraint on the total bufferage available due to space of 50
- iii) You are recording all the potential results in the results table and tracking the additional function created.
- iv) Set the experiment to use the All Combinations Method (Don't worry we will get to the optimization algorithms soon!!)

When you evaluate the number of constrained combinations it will show that there are 247 to be evaluated. (see overleaf)

NOTES



How long should the experiment be run for? You could come to a judgement about this in a number of ways e.g.

- By using the analyze button and looking at the graph variability for different run lengths
- By looking at the variability in the model (the random bits)

The data for this model is shown overleaf. It shows that all the distributions have a fairly narrow range (danger signals here often include distributions such as a negative exponential where the value sampled can tend to infinity for rare occurrences). Here all is uniform samples (plus a single tightly constrained normal). The most variation is the breakdown on M4 (1000 to 2000 interval).

Normally you may decide to play it safe and run the model for say 1,000,000 time units. For a model such as this that would definitely be long enough. However we now come up against another constraint – how long can you afford to wait for the results!!

(Continue on the page after next)

**NOTES**



Model Detail – from WITNESS Documentor Report

ELEMENT NAME: M1

Cycle Time: UNIFORM (10,35)

Breakdown

Breakdown type: Busy Time  
 Down Interval: UNIFORM (10,500)  
 Repair Time: 90.0

ELEMENT NAME: M2

Cycle Time: UNIFORM (40,68)

Breakdown

Breakdown type: Busy Time  
 Down Interval: UNIFORM (200,300)  
 Repair Time: 55.0

ELEMENT NAME: M3

Cycle Time: UNIFORM (10,45)

Breakdown type: Busy Time  
 Down Interval: 100.0  
 Repair Time: UNIFORM (10,30)

ELEMENT NAME: M4

Cycle Time: UNIFORM (5,10)

Breakdown

Breakdown type: Busy Time  
 Down Interval: UNIFORM (1000,2000)  
 Repair Time: UNIFORM (200,300)

ELEMENT NAME: M5

Cycle Time: UNIFORM (10,60)

Breakdown

Breakdown type: Busy Time  
 Down Interval: 250.0  
 Repair Time: 80.0

ELEMENT NAME: M6

Cycle Time: NORMAL (24,2)

Breakdown

Breakdown type: Busy Time  
 Down Interval: UNIFORM (200,500)  
 Repair Time: UNIFORM (100,190)

NOTES



Type in 1,000,000 into the run time field (we will assume a zero warm-up time here).

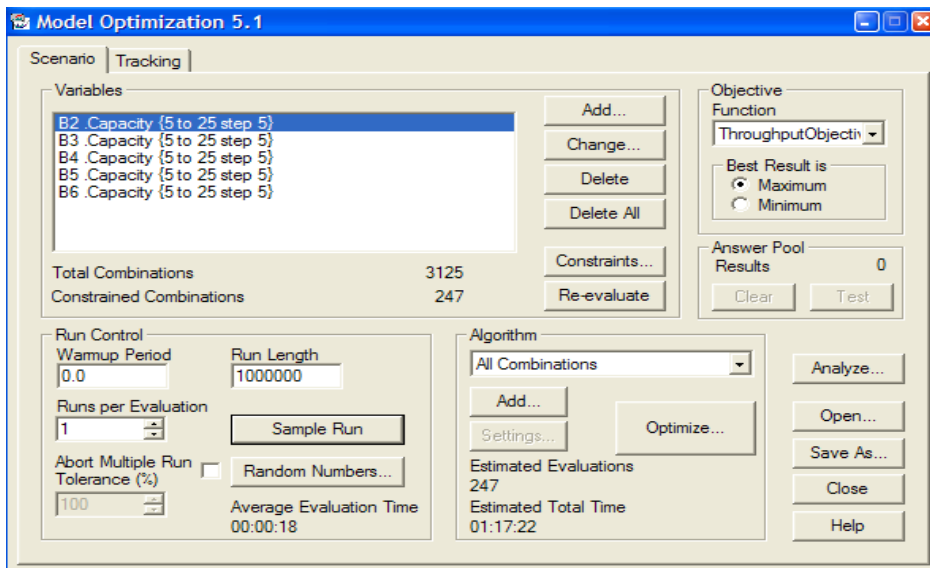
**Executing a sample run**

After setting the warm-up period, run length or runs per evaluation, the Average Evaluation Time and hence the Estimated Total Time will be set to Unknown. For these times to be re-calculated and displayed, click the Sample Run button. This performs one evaluation of the model according to the current settings, and times how long it takes.

When performing a sample run, the variables are set to their suggested values. If a variable does not have a suggested value then the mid-point value is used. If you have defined any constraints then these values may not meet the constraints. If this happens you will be warned. You can choose to carry on and perform the sample run or cancel it. If you wish to perform the sample run with valid values then you should ensure that all the variables have suggested values that meet the constraints.

Usually it will not matter if the sample run is performed with values that do not meet the constraints as the real time taken to perform a run will not generally be affected.

Then use the Sample Run button to check how long the computer thinks it will take. All computers will be different but an example may be seen below:



The computer is predicting a run time of 18 seconds per experiment which would take 1 hr 17 minutes to complete. This is probably OK in the real world but too long for a training class so please set the run time to 10,000 in this instance and run the whole experiment.

**NOTES**



Your answers should match those below:

Results - Best 500 [ThroughputObjective]								
Results   Results Chart   Parameter Analysis								
	Evaluation	ThroughputObjective	B2 .Capacity	B3 .Capacity	B4 .Capacity	B5 .Capacity	B6 .Capacity	M4_IdePercentage
1	82	553	5	10	15	10	10	32
2	36	551	5	5	15	5	15	30
3	37	551	5	5	15	5	20	32
4	39	551	5	5	15	10	10	40
5	40	551	5	5	15	10	15	44
6	42	551	5	5	15	15	10	48
7	46	551	5	5	20	5	15	30
8	48	551	5	5	20	10	10	40
9	70	551	5	10	10	5	15	32
10	71	551	5	10	10	5	20	33
11	74	551	5	10	10	10	15	42
12	80	551	5	10	15	5	15	26
13	100	551	5	15	10	5	15	30
14	149	551	10	5	15	5	15	26
15	151	551	10	5	15	10	10	32
16	169	551	10	10	10	5	15	30
17	73	550	5	10	10	10	10	40
18	76	550	5	10	10	15	10	48
19	102	550	5	15	10	10	10	36
20	139	550	10	5	10	5	15	34
21	140	550	10	5	10	5	20	34
22	142	550	10	5	10	10	10	41
23	143	550	10	5	10	10	15	44
24	145	550	10	5	10	15	10	49
25	171	550	10	10	10	10	10	36
26	204	550	15	5	10	5	15	34
27	206	550	15	5	10	10	10	41
28	21	549	5	5	10	5	15	33
29	22	549	5	5	10	5	20	34
30	23	549	5	5	10	5	25	34
31	25	549	5	5	10	10	10	41

Note the tracked result in the final column.

Now the results here are quite close and therefore this may indicate that to be sure of settings the model should perhaps be run for longer or with a larger number of replications. Obviously though when you are looking at factors such as buffer capacities a variety of settings can also be roughly in the same area – in this case the difference between top and bottom is 553 to 529 – a difference of over 4.5% - quite substantial for many manufacturers.

The results show you need a reasonably good buffer before B4 and B6 (at least 10 in these experiments) – and if you want to maximize availability of M4 then a buffer of 15 in front of M5 is advisable (this would seem reasonable to stop M4 being blocked from outputting).

**NOTES**



## 10) Different Optimization Methods and Procedure

### a) Adaptive Thermostatistical Simulated Annealing

This is the most sophisticated of the optimization algorithms. It will intelligently search for an optimal answer constrained only by the time that you tell it to take (specified in number of experiments).

Simulated Annealing is a modern heuristic algorithm. An evaluation (simulation experiment) is followed by intelligent reasoning based on the results obtained. The algorithm determines the best experiment to run next based on the results so far.

At the start of an algorithm run the experiments chosen are fairly random as not much has yet been learnt. If you have entered suggested values then these are the starting experiments chosen by the algorithm.

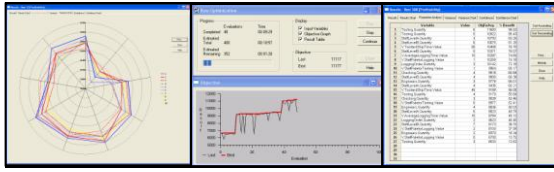
The skill of the algorithm is to adequately search the solution space (results of all experiments) to be confident of good solutions.

A simulated annealing algorithm gradually cools (lowers) the randomness of experimentation as patterns become tangible in the results. Cool too quickly and not enough of the solution space has been experienced, cool too slowly and it takes too long to get results.

The algorithm in the WITNESS Optimizer is unique and has been crafted by experts in this field with Lanner purely to suit typical simulation experiments. It adapts the cooling schedule based on the model results and freezes successful values as soon as these become clear.

For more details on the algorithm please see the optimizer helptext.

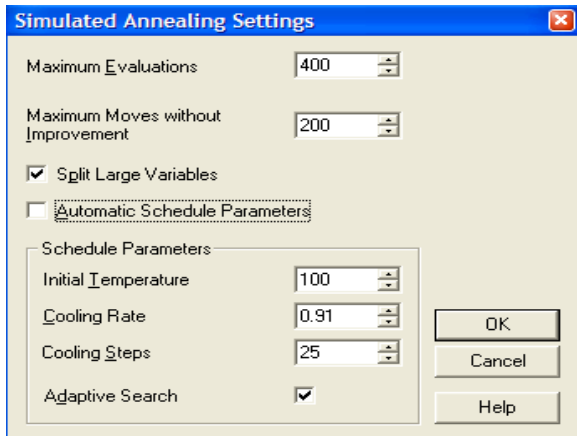
#### NOTES



If you choose Adaptive Thermostatical SA, the Settings button is enabled. Using the settings button you can:

1. Specify the maximum number of evaluations you wish to be executed. This will tell you in conjunction with the sample run button how long you may have to wait for the results.
2. Specify the **Maximum moves without improvement**. This causes the optimization to stop before it reaches the maximum evaluations if the specified number of search moves have been tried without any improvement in the best result found.
3. Allow the splitting of large range parameters. This helps the algorithm jump about large ranges more quickly enabling effective coverage. It is recommended that you always leave this setting on.
4. If you are a simulated annealing expert you may care to alter the algorithm settings. To do this you will need to understand concepts such as initial temperature, cooling rates, cooling steps. You can also choose to turn off the adaptive part of the algorithm. However the algorithm has been devised to run automatically and therefore for most users we recommend that the Automatic Schedule Parameters box is checked.

IF the total number of options for an experiment is less than the number defined in 1 above then the algorithm will not be used – even if set – all combinations will be run in its place as this is a more sensible setting as no algorithm can be certain of finding the optimal solution



Here is an example of the **Simulated Annealing Settings** dialog:

When you click the OK button the **Simulated Annealing Settings** dialog closes and you are returned to the **Model Optimization** dialog. The Estimated Total Time is automatically updated to reflect the parameters you have just set.

**NOTES**



### Exercise Six – A first Intelligent Experiment

Open the model created in exercise five.

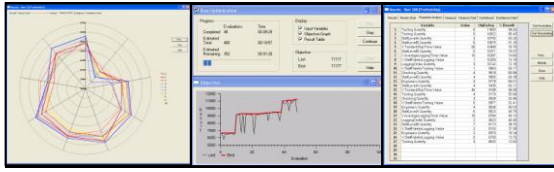
Define a wider experiment. Change the parameter settings for each buffer to vary between 2 and 30 in steps of 2. With the constraints this would now mean 51,870 experiments were we to evaluate all options. Use the intelligent algorithm to provide good results in much less time. Set the simulated annealing algorithm to do a maximum of 400 experiments (400 in each of the top two boxes in the settings dialog).

Running the experiment the algorithm identifies some good results in this time (we already know the ballpark from our earlier experiments)

	Evaluation	ThroughputObjective	B2 .Capacity	B3 .Capacity	B4 .Capacity	B5 .Capacity	B6 .Capacity	M4_IdePercentage
1	308	553	6	4	14	12	8	44
2	309	553	6	4	14	14	8	48
3	314	553	6	4	14	16	8	49
4	315	553	8	4	14	16	8	47
5	319	553	8	4	14	14	8	43
6	320	553	8	4	14	12	8	39
7	322	553	10	4	14	14	8	43
8	332	553	10	4	16	12	8	39
9	335	553	10	4	14	12	8	39
10	39	552	6	4	22	10	8	35
11	296	552	6	4	12	12	8	44
12	298	552	8	4	12	12	8	42
13	299	552	8	4	12	14	8	48
14	300	552	6	4	12	14	8	49
15	301	552	6	4	12	16	8	40

It really is quite remarkable how the algorithm achieves these results with so few experiments (less than 1% of the options) and it performs without the experimenter having to think about the problem (the next experiment to try). Of course we do not advocate not thinking about the problem at all (there may be other creative solutions to add in to the model) but this intelligent helper will work on its own.

NOTES



**b) Six Sigma**

The Six Sigma Algorithm essentially applies the same simulated annealing algorithm to the problem. However it also limits the amount of change to an existing process / model.

The name Six Sigma Algorithm has been used here but the algorithm applies equally to any process improvement methodology. It will attempt to identify the most effective changes that can be made to a process.

To use the Six Sigma algorithm you need to specify suggested values for each model parameter (variable) that you define. In this case the suggested value is used to represent the current value of that parameter.

When viewing the results screens this is where the color setting from the tracking tab is useful. It clearly identifies the parameter changes that have occurred in a run.

**Exercise Seven – Applying the Optimizer to Six Sigma**

From the results in Exercise Six select the result with 553 output that has the maximum idle time for M4 and then press the Set Model Button. (you do not need to run the model when prompted).

Exit from the optimizer results and the optimizer dialog back to the model.

Save the model as ProductionLineIdealBuffers.mod

Validate that the model buffer capacities have indeed been set and run the model to 10,000 to confirm the 553 output.

Now we need to prepare the model to investigate other improvements that could be made.

Some model changes have been made for you to save time – please open SixSigmaProduction.mod. A factor has been added to each machine cycle time so that we can experiment easily to see the effect of reducing each machine cycle time.

- M1 has its cycle time adjusted by Factor(1)
- M2 has its cycle time adjusted by Factor(2)

And so on.

**NOTES**

The leading black belt Six Sigma practitioner in the organization has identified this line for a potential Six Sigma project. There is a choice of seven different projects:

- i) To improve the performance of machine M1 (investigating and improving breakdowns, planned maintenance, plc programming, etc). Improvements could be up to 10%
- ii) Likewise for each of machines M2 to M6
- iii) There is also an option to investigate the practicality of increasing buffer B5 to take a higher capacity (currently 16 – possibly up to 18, 20 or even 22)

Open up the optimizer dialog and set each of these parameters in a new experiment. Set the options for each factor to be 0.9 to 1.0 (step 0.2) (suggested 1.0 – the current value). Note that these factors can be set all at once even though the Optimizer will treat them as 6 separate experimentation variables.

Set the options for the buffer too (16 to 18 step 2 – suggested/current =16)

Select the objective function to maximize, set the run time to 10,000, track the extra function and then select the Six Sigma method with the settings of 400 maximum experiments (note option to do all limited change options too!).

Run the optimizer and observe the results coming into the results table:

Results - Best 500 [ThroughputObjective]											
Results   Results Chart   Parameter Analysis											
	Evaluation	ThroughputObjective	B5 .Capacity	Factor[1]	Factor[2]	Factor[3]	Factor[4]	Factor[5]	Factor[6]	M4_IdePercentage	
1	0	553	16	1	1	1	1	1	1	49	
2	1	553	18	1	1	1	1	1	1	49	
3	2	551	16	0.98	1	1	1	1	1	48	
4	3	552	16	1	0.98	1	1	1	1	42	
5	4	555	16	1	1	0.98	1	1	1	45	
6	5	548	16	1	1	1	0.98	1	1	45	
7	6	552	16	1	1	1	1	0.98	1	49	
8	7	556	16	1	1	1	1	1	0.98	49	
9	8	559	16	1	1	1	0.98	1	0.98	47	
10	9	548	18	1	1	1	0.98	1	1	49	
11	10	548	16	0.98	1	1	0.98	1	1	47	
12	11	552	16	1	0.98	1	0.98	1	1	42	
13	12	548	16	1	1	1	0.98	0.98	1	47	
14	14	552	18	1	0.98	1	1	1	1	47	
15	15	552	18	1	1	1	1	0.98	1	49	
16	17	554	16	1	1	0.98	1	0.98	1	46	
17	18	554	16	1	1	0.98	0.98	1	1	45	
18	19	557	16	1	1	0.98	0.98	1	1	47	

The first run is at the suggested / current settings. Subsequently the algorithm tries a single change on each parameter as a starting set of experiments and then moves on to more complex combinations. Note changed values in blue.

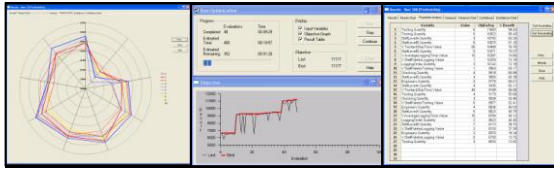
**NOTES**



When the algorithm has finished order the results and see the best options. It is clear that it would be best to speed up Machines M3 and M6 in order to improve production. A Six Sigma project would be best targeted at these machining operations.

Results - Best 500 [ThroughputObjective]										
Results   Results Chart   Parameter Analysis										
	Evaluation	ThroughputObjective	B5 Capacity	Factor[1]	Factor[2]	Factor[3]	Factor[4]	Factor[5]	Factor[6]	M4_IdePercentage
1	115	577	16	1	1	0.9	1	1	0.9	46
2	114	574	16	1	1	0.92	1	1	0.9	46
3	101	573	16	1	1	0.92	1	1	0.94	47
4	103	573	16	1	1	0.92	1	1	0.96	47
5	122	571	16	1	1	0.9	1	1	0.92	46
6	130	570	16	1	1	0.94	1	1	0.9	46
7	59	568	16	1	1	0.96	1	1	0.94	48
8	86	568	16	1	1	0.96	1	1	0.92	47
9	91	568	16	1	1	0.94	1	1	0.92	46
10	97	568	16	1	1	0.92	1	1	0.92	47
11	108	568	16	1	1	0.94	1	1	0.96	46
12	13	567	16	1	1	0.96	1	1	0.96	47
13	105	567	16	1	1	0.92	1	1	0.98	47
14	12	566	16	1	1	0.98	1	1	0.96	46
15	120	565	16	1	1	0.93	1	1	0.98	46

NOTES



### c) Hill Climb

- The hill climb algorithm is also a heuristic algorithm but a very simple one. Unlike the Simulated Annealing algorithm it can easily get trapped in what are called Local Optima and not seek out a more global optimum.
- It is fast and aggressive optimization and hence often worth a try with a few experiments to see how well it can do. If the solution space is one gigantic gently sloped mountain it may even find a global optimum (or if you are lucky with the starting position chosen).
- Again the settings allow you to specify how many experiments are to be run.

### d) All Combinations, Random Solutions, Min/Mid/Max

- All Combinations runs a full factorial experiment. This is the most sensible option if there is the time to wait for all the experiments to be run.
- Random solutions are just that – each parameter on each run is chosen at random. This can be useful to assess just how wrong or right you could be – i.e. the extents of the solution space. In the settings dialog you can set how many random solutions you want to be generated.
- Min/Mid/Max tests the extremes of your parameter range settings. Each range parameter is run at each of these 3 values - all set parameters are run at all levels.

NOTES



**e) Adding your own algorithm**

It is possible to use your own algorithm in the WITNESS Optimizer. It may be that you wish to run a particular design of experiment (e.g. a partial factorial experiment) or indeed try an alternative heuristic for optimization (e.g. a genetic algorithm method). Simply use you the optimizer to set parameters to vary, constraints and any objective function as normal and then use your own algorithm made available from the pull down list of choices. The results grids will be populated from your experiments with all the attendant tables, charts and analyses.

To setup your own algorithm in the list of choices use the Add button on the main Optimizer dialog and select the COM DLL containing the algorithm.

A DLL may contain more than one algorithm, and each COM class within the DLL that implements an algorithm may support other interfaces as well as the IWitnessOptimizerAlgorithm interface which support the Optimizer.

The IWitnessOptimizerAlgorithm interface consists of several methods and properties that must be implemented e.g.

- Property get: Name
- Method: EstimateIterations
- Method: DoOptimisation

The COM Object model allows access to all the information set in a "Settings" dialog that may be defined together with all parameter settings and associated information such as constraints, objective function, answer pool, etc.

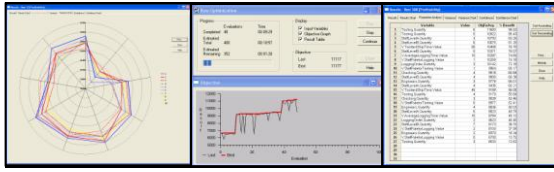
Examples of code are included in the WITNESS/Optimizer/Technical Guide directory.

**f) Iterative running of the optimizer**

It is common to run the optimizer iteratively – using different algorithms in sequence. For example you may choose to:

- i) Run Mid/Mid/Max to determine whether any effects are purely linear (i.e. indicating that more/less is better for a particular scenario in all cases in which case the parameter can be set and not treated as a parameter any more.
- ii) Run the simulated annealing algorithm
- iii) Set the suggested values to the best results found in ii) and run a hill climb algorithm to check the immediate local area of the solution space.

NOTES



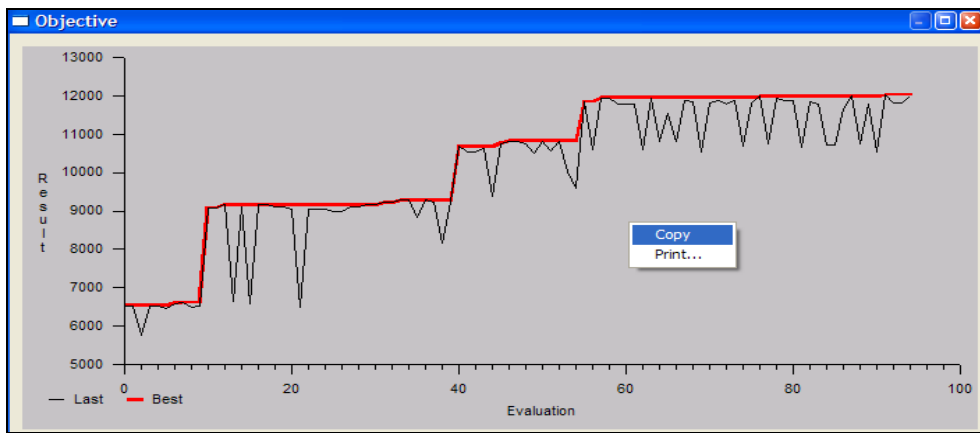
## 11) Copying and Printing Results & Output to MINITAB

### a) Copying and Printing Results

All the information contained in the results tables and charts may be easily copied or printed. It may also be output to MINITAB for further analysis.

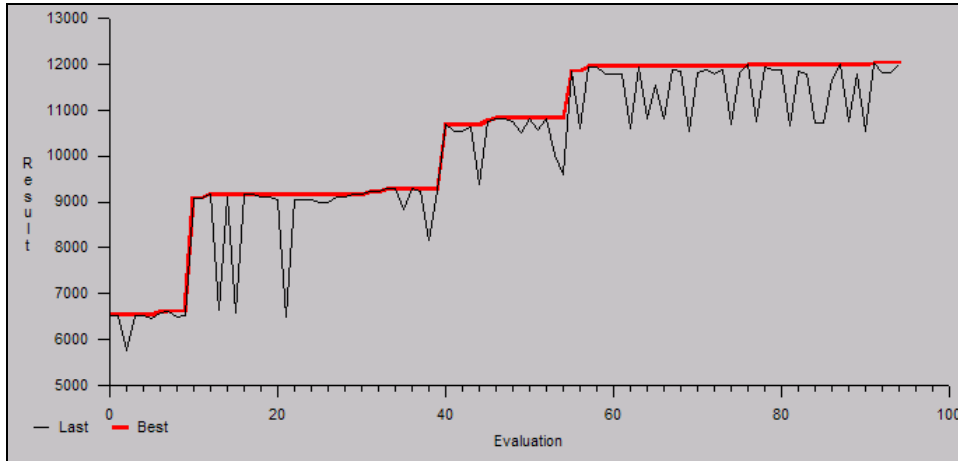
To copy or print the results in chart or graph form:

1. Select the region of the table or the chart you wish to copy or print and either use the menu commands or right mouse menu to copy or print.



2. When pasting a chart into another application you can either paste the figures from the chart (normal paste option) or paste the picture using paste special options (device independent bitmap or metafile options). The result of a bitmap paste is shown overleaf.

NOTES



Example of a Bitmap paste from a chart

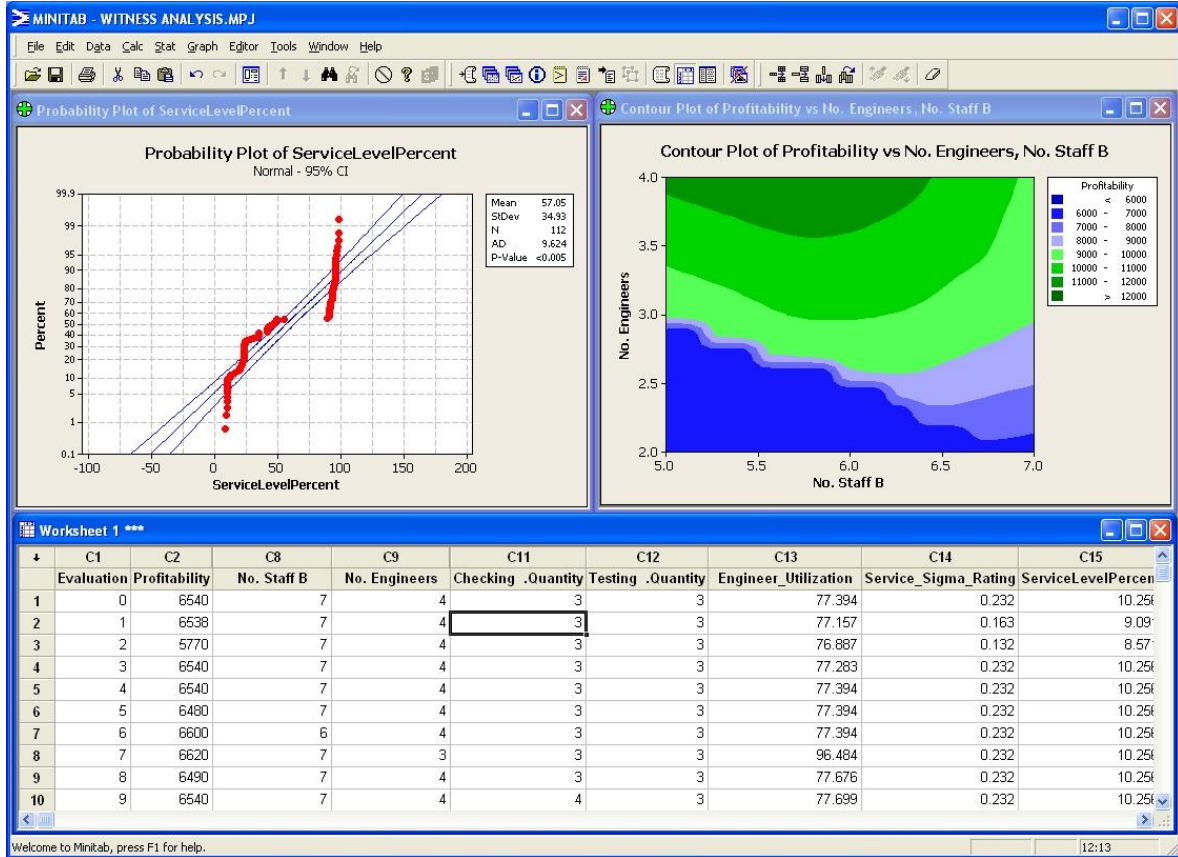
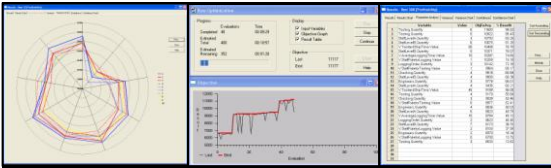
**b) Outputting to MINITAB**

To use this feature you will need to have MINITAB loaded on your computer (this is not included with the WITNESS Optimizer and must be obtained separately).

To output to MINITAB simply make your selection and press the MINITAB button on any results table screen. MINITAB will automatically be opened (if not already) and a new worksheet created containing the results and the column headings set to the titles from the Optimizer screen.

MINITAB enables several extended analyses of results and is popular for results presentation in Six Sigma projects. An example of a screen from MINITAB analyses is shown overleaf.

**NOTES**



- An example of a MINITAB analysis screen showing results from an Optimizer experiment

NOTES



## 12) Exercise Eight – A Full Case Study

This case study is designed as additional course value and can be delivered as part of a two day course. For one day course attendees the model and notes are provided largely for additional self study.

Additional concepts are introduced in this case study. These include:

- i) the use of soft constraints in the objective function. The type of constraint defined in the course above is one where certain combinations are invalid. Sometimes it is better to define certain combinations or indeed outcomes of a model as not invalid, but only undesirable. For example if one has a budget to spend on \$50,000 and the best solution costs \$50,100 then would you not want to know? In circumstances such as these it is best to include a penalty value in the objective function when certain constraints are broken.

E.g. for the above example the objective function could have another term added such as:

Return Calculated objective – (Spend-50000)

Expressing the constraint in these terms also teaches the clever optimization algorithm as, not only does the algorithm know that a constraint is broken but it is fed a value of how badly the constraint is broken.

Note in the above example that this expression will not just penalize spend over \$50,000 but will reward underspend. If this is not wanted then an 'IF' condition should be used.

- ii) The use of complex use of variables to affect WITNESS behaviour. The variables set in the optimizer are used extensively in the Initialize actions section of the WITNESS model in order to set model parameters. In some cases simply such as in the use of expressions such as SET QUANTITY and in some cases using complex functions such as SETINFO.

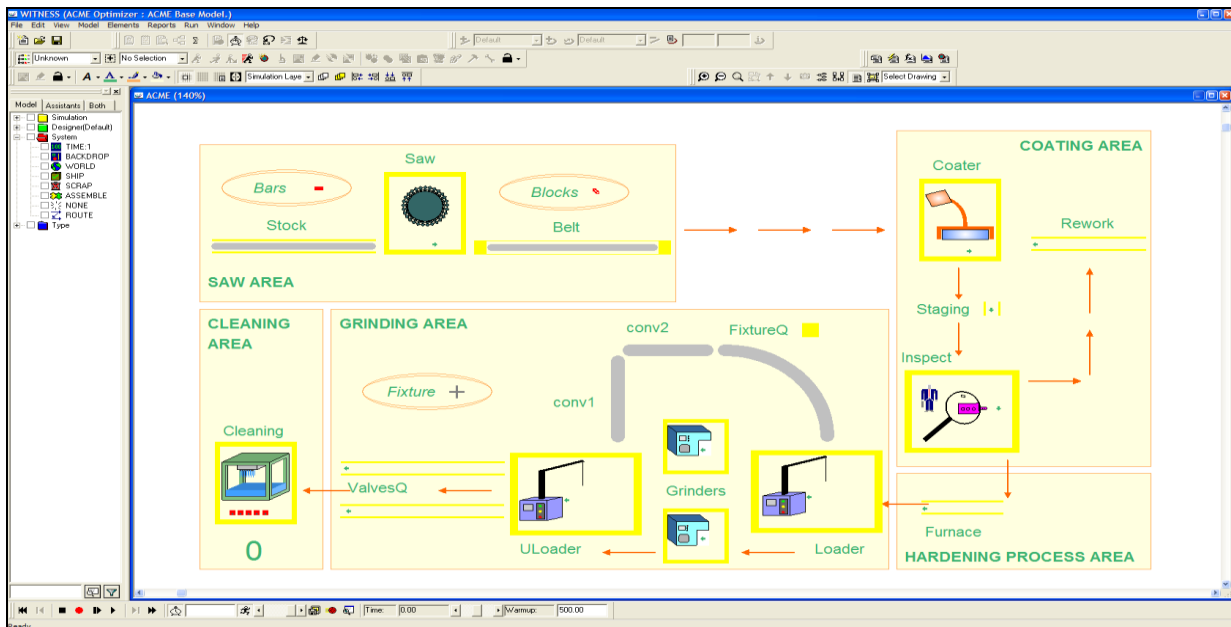
There is a start model and a finished model available for you so that you can spend as much time on this problem as you wish. If time is pressing simply open up the finished model (ACME Optimizer.mod) and view the options set up in the objective functions (2 options) and the actions programmed in Initialize Actions.

NOTES



**Overview**

You have been employed as a member of the production improvement task force for the ACME Valve Manufacturing Company. ACME has been experiencing a severe backlog in sales orders due to antiquated machinery and poor production planning. The CEO of the company will tolerate the situation no longer. He has given you the simulation model that was built to try to increase output as a reference.



**Objective**

Output is currently around 130 to 160 valves in a 5,000 minute production run. The CEO would like for you to use the WITNESS Optimizer Module in order to increase throughput. Below are your objectives and the next page gives a list of potential capital improvements and their associated cost.

Your objective is to increase total throughput by at least 100% with a budget of \$100,000. The duration of the production run is 5,000 minutes which includes a warm-up period of 500 minutes. The next page is a list of changes that can be implemented.

**NOTES**



**LIST OF ALLOWABLE CHANGES**

**Costs(\$)**

**Buffers:**

Per Part capacity in the Stock Buffer (up to 9 extra parts) 1000

**Saw:**

Identical new saw 40,000  
 Cycle time decrease per 10%(up to 20%) 5,000  
 Decrease repair time per 20%(up to 40%) 7,000

**Belt Conveyor:**

Convert belt to roller 10,000  
 New conveyor of either type 20,000  
 Cycle time decrease per 10%(up to 20%) 5,000

**Coating Machine:**

Increase batch capacity per Part 10,000  
 Set-up decrease to double number of operations 5,000

**Inspector:**

New Inspector 30,000  
 Decrease rejection rate by 50% 20,000

**Hardening Process:**

Cut process time by 10 minutes 20,000

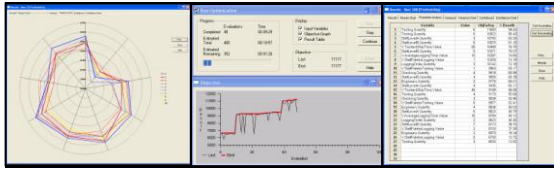
**Grinder:**

New grinder 80,000

**Cleaner:**

Additional cleaner 16,500  
 Cycle time decrease per 10%(up to 20%) 10,000

**NOTES**



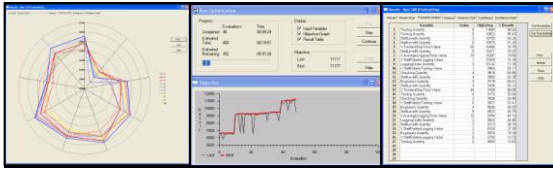
1. Start **WITNESS** and open **ACMEBASE.MOD**.
2. Run the model and become familiar with the different parts of the production process.
3. Examine the detail of the model with respect to the things that may vary in the list above. Decide how the optimization is to be structured, bearing in mind that there is a constraint. In the first instance treat this as a hard constraint that must be set for the optimizer as such. Have some thoughts and then read the advice paragraph in the box below:

Note: There is more than one way to accomplish this. The most effective way is as follows: You will need to create a variable for every change that needs to take place (~30 variables). In order to create a valid and understandable constraint equation, the variables will have to have zero as their default value, and increment by steps of one. Only in this way will the equation be able to be of the form  $a*v1 + b*v2 + c*v3..... \leq 100,000$ .

A neat way to define variables and other items such as functions for the optimization is to create a new module in the WITNESS model and then define all else within this. See overleaf for some hints on how to use this structure.

4. Set the model up for optimization by defining any extra elements that are necessary and also defining a function that quantifies simply the objective of the optimization process. (This is a simple function – do not overcomplicate here!!). The work to effect all the changes in the WITNESS model is extensive in this case as we have deliberately included some more difficult examples. Your class trainer will give guidance where necessary – if self studying then please refer to the completed model **Acme Optimizer.mod** and the Optimizer file **ACME Optimizer.OPT** for details. This work can be done in stages – if you wish to try just one or two simple options first then please do this to enable you to see simpler evaluations first. Additionally some hints on the modeling are overleaf.
5. Check existing model Action statements for anything that may conflict with the **Optimizer**.
6. Save the model as **ACMEOPT1.MOD**.

NOTES



### Hints on Model Structure

*The following expressions could form part of the model/initialize actions in the WITNESS model – based on values input as Optimizer paramters*

The following is an example of how the cycle time field for the Saw operation could appear to allow for a 10% or 20% decrease in Cycle time using a simple 0,1,2 value variable. (Current value =6)

$$6 * (1 - \text{Opt\_Vars.Saw\_CT} / 10)$$

The following is an example of how in initialize actions the buffer capacity of buffer Stock could be set to a different capacity based on a simple EXTRA capacity variable called BufferCapacity with values set from 0 to x

SET CAPACITY OF Stock TO 10 + BufferCapacity

Some of the most complex changes may require the use of a SETINFO command to make the change. For example the following code may be used to set the conveyor to fixed or queueing in model initialization

```

DIM InfoSTR AS STRING
InfoSTR = "DETAIL\nSELECT\nBelt\nTYPE: "
IF Belt_Type = 0
    InfoSTR = InfoSTR + "Fixed;\n"
ELSE
    InfoSTR = InfoSTR + "Queueing;\n"
ENDIF
InfoSTR = InfoSTR + "END Belt\nEND SELECT\nEND DETAIL"
SETINFO (InfoSTR)
    
```

Not all model changes are made in initialize actions. To alter routing logic it may be necessary to define variables that are set up in Actions on Finish which then provide the routing destination OR use a function in the routing rule itself.

### NOTES



**Stage 2: Start the Optimizer and Define an Optimization Scenario**

1. Open up **ACMEOPT1.MOD**.
2. Start the **Optimizer**.
3. Define an Optimization Scenario for the ACME model
  - Add the variables needed according to the plan set in stage 1
  - Set the constraint equation that will keep the cost under \$100,000. Note again how the structure of this has required the structure of the parameters to be as they are – simple values that can be assigned 0 for the as-is situation and a linear scale above this.
  - Input run-length and warm-up period according to the data given.
  - Analyze the variability in the model to determine how many runs per evaluation should be used.
  - Set up the Optimization to skip 100 random numbers after each run.
  - Evaluate how many constrained combinations there are -
4. Choose an Optimization method (remember iterative methods)
5. Save the model setup as an OPT file
6. Evaluate the results

NOTES



**Stage 3: Trying Soft Constraints**

1. Open up **ACMEOPT1.MOD**.
2. Add a function to the model to calculate the total money spent out of the \$100,000
3. Define another objective function that includes the same expression of output as before but adjusts the result by a factor based on expense. If the cost is over the limit of £\$100,000 penalize the value by 1 for every \$1,000 overspend.
4. Open the optimization file from stage 2 and remove the hard constraint
5. Re-run the experiments and see if a little more money might be worth it or whether the soft constraints allow the algorithm to 'learn' more about the situation.
6. The exact results obtained may depend on the nature of the programming of the problem into the model – even the order of definition of the optimizer variables will affect the running of the algorithm. This type of problem, where the allowed solution space is quite highly constrained can be problematical – usually the algorithm will work more robustly when the soft constraint method is used. This is due to the difficulty the algorithm has in making combinations of changes to match the constraints – it works more effectively when allowed to roam more freely.

You should have been able to identify solutions that allow output to reach the region of 350 to 400 in the model run time defined and that results in the mid 400's are possible with another 20% to 50% more spend.

The optimization file **ACME Optimizer Soft.OPT** is setup to show a single replication run where the top results obtained in 2000 experiments are:

Evaluation	Obj Modified	Opt_Vars Saw Qty Value	Opt_Vars Saw CT Value	Opt_Vars Saw RT Value	Opt_Vars Belt Type Value	Opt_Vars Belt Qty Value	Opt_Vars Belt CT Value	Opt_Vars Coating Batch	Opt_Vars Coating Stops	Opt_Vars Insp Qty Value	Opt_Vars Insp RR Value	Opt_Vars Hard CT Value	Opt_Vars Grind Qty Value	Opt_Vars Cleaner Qty	Opt_Vars Cleaner CT	Opt_Vars Buffer Capap[2]	Obj	Opt_Vars Expenses
1710	473	0	0	0	1	0	0	2	0	2	1	0	0	0	1	0	6	505 132000
1837	471	0	0	0	1	0	0	2	0	2	1	0	0	1	0	8	505 134000	
1838	470	0	0	0	1	0	0	2	0	2	1	0	0	1	0	9	505 135000	
1711	469	0	0	0	1	0	0	2	1	2	1	0	0	1	0	6	506 137000	
1715	469	0	0	0	1	0	0	2	0	2	1	0	0	1	1	6	511 142000	
1598	469	0	0	0	1	0	1	2	0	2	1	0	0	1	0	6	506 137000	
1707	468	0	1	0	1	0	0	2	0	2	1	0	0	1	0	6	505 137000	

In the table the modified objective is the soft constrained one, the obj column is the number shipped and the expenses column is the expenditure.

**NOTES**