
Guide to Using

RISKOptimizer

**Simulation Optimization for
Microsoft Excel**

**Version 5.7
September, 2010**

**Palisade Corporation
798 Cascadilla St.
Ithaca, NY USA 14850
(607) 277-8000
(607) 277-8001 (fax)
<http://www.palisade.com> (website)
sales@palisade.com (e-mail)**

Copyright Notice

Copyright © 2010, Palisade Corporation.

Trademark Acknowledgments

Microsoft, Excel and Windows are registered trademarks of Microsoft, Inc.

IBM is a registered trademark of International Business Machines, Inc. Palisade, RISKOptimizer, TopRank, BestFit and RISKview are registered trademarks of Palisade Corporation.

RISK is a trademark of Parker Brothers, Division of Tonka Corporation and is used under license.

Table of Contents

Chapter 1: Introduction	1
Introduction	3
Installation Instructions	11
Chapter 2: Background	15
What Is RISKOptimizer?	17
Traditional Optimization vs. Simulation Optimization	25
Chapter 3: RISKOptimizer: Step-by-Step	33
Introduction	35
The RISKOptimizer Tour	37
Chapter 4: Example Applications	59
Introduction	61
Budget Allocation	63
Capacity Planning	65
Class Scheduler	67
Hedging with Futures	69
Job Shop Scheduling	71
Portfolio Balancing	73
Portfolio Mix	75
Portfolio Risk	77

Salesman Problem	79
Yield Management.....	81
Chapter 5: RISKOptimizer Reference Guide	83
Model Definition Command.....	85
Optimization Settings Command – General Tab.....	113
Optimization Settings Command – Runtime Tab.....	117
Optimization Settings Command – View Tab.....	121
Optimization Settings Command – Macros Tab	123
Start Optimization Command.....	125
Utilities Commands.....	127
RISKOptimizer Watcher	131
Chapter 6: Optimization	141
Chapter 7: Genetic Algorithms	153
Introduction	155
History	155
A Biological Example.....	158
A Digital Example	159
Chapter 8: Simulation and Risk Analysis	163
Introduction	165
What is Risk?.....	165
Modeling Uncertainty in RISKOptimizer	171
Analyzing a Model with Simulation	173

Chapter 9: RISKOptimizer Extras	175
Adding Constraints	177
Improving Speed.....	187
How RISKOptimizer's Optimization is Implemented.....	189
Appendix A: Automating RISKOptimizer	193
Appendix B: Troubleshooting / Q&A	195
Troubleshooting / Q&A	197
Appendix C: Additional Resources	199
Glossary	207
Index	217

Chapter 1: Introduction

Introduction	3
Why RISKOptimizer?	3
Traditional Optimization Problems.....	3
Optimization of Uncertain Models	4
Modeling Uncertainty.....	4
Optimization Using Simulation	5
Simulation Results	6
Custom Applications Using RISKOptimizer	6
Applications of Simulation Optimization Using RISKOptimizer	6
Before You Begin.....	7
What the Package Includes.....	7
About This Version	7
Working with your Operating Environment	8
If You Need Help	8
Before Calling	8
Contacting Palisade.....	9
Student Versions	10
RISKOptimizer System Requirements	10
Installation Instructions	11
General Installation Instructions	11
Removing RISKOptimizer from Your Computer.....	11
The DecisionTools Suite.....	12
Setting Up the RISKOptimizer Icons or Shortcuts.....	13
Macro Security Warning Message on Startup	13
Other RISKOptimizer Information	14
RISKOptimizer Readme	14
RISKOptimizer Tutorial	14
Learning RISKOptimizer	14

Introduction



RISKOptimizer combines simulation and optimization to allow the optimization of models that contain uncertain factors. RISKOptimizer, through the application of powerful genetic algorithm-based optimization techniques and Monte Carlo simulation, can find optimal solutions to problems which are "unsolvable" for standard linear and non-linear optimizers. RISKOptimizer combines the simulation technology of @RISK, Palisade's risk analysis add-in, and Evolver, Palisade's genetic algorithm - based solver. Users familiar with @RISK and either Evolver or Excel's built in Solver should be able to use RISKOptimizer with little difficulty.

The **RISKOptimizer User's Guide**, which you are reading now, offers an introduction to RISKOptimizer and the principles behind it, then goes on to show several example applications of RISKOptimizer's unique genetic algorithm and simulation technologies. This complete manual may also be used as a fully-indexed reference guide, with a description and illustration of each RISKOptimizer feature.

Why RISKOptimizer?

RISKOptimizer opens up a whole new spectrum of problems to optimization. With RISKOptimizer, optimal solutions can be found when problems contain variables outside your control whose values are not known. Current optimizers such as **Solver** (a linear and non-linear optimizer included with Excel) and **Evolver** (a genetic algorithm-based optimizer from Palisade Corporation) cannot find optimal solutions when ranges of possible values are entered for uncertain factors in a model.

Traditional Optimization Problems

Traditional Excel-based optimization problems analyzed using Solver or Evolver are comprised of:

- An output or "target" cell that you wish to minimize or maximize
- A set of input or "adjustable cells" whose values you control
- A set of constraints that need to be met, often specified using expressions such as $COSTS < 100$ or $A11 \geq 0$

During an optimization in Solver or Evolver, the adjustable cells are changed across allowable ranges you specify. For each possible set of adjustable cell values the model is recalculated, and a new value for the target cell is generated. When the optimization is complete, an optimal solution (or combination of adjustable cell values) is found. This solution is the combination of adjustable cell values which yields the best (i.e., minimum or maximum) value for the target cell while satisfying the constraints you've entered.

Optimization of Uncertain Models

When a model has uncertain elements, however, both Solver and Evolver cannot generate optimal solutions. In the past, many optimization models just ignored uncertainty, making models unrealistic but optimizable. If an attempt was made to find optimal values by using simulation, a "brute-force" approach was employed to search possible adjustable cell values on an iterative basis. This involved running an initial simulation, changing one or more values, rerunning the simulation, and repeating this process until what looked like an optimal solution was found. This is a lengthy process, and it is usually not clear how to change the values from one simulation to the next.

With RISKOptimizer the uncertainty present in a model may be included and reliable optimal solutions which take that uncertainty into account can be generated. RISKOptimizer uses simulation (from @RISK) to deal with the uncertainty present in the model and uses genetic algorithms (from *Evolver*) to generate possible values for the adjustable cells. The result of this "simulation optimization" is the combination of values for the adjustable cells which minimizes or maximizes a statistic for the simulation results for the target cell. You may, for example, wish to find the combination of adjustable cell values which maximizes the mean of the target cell's probability distribution, or minimizes the standard deviation.

Modeling Uncertainty

For modeling uncertainty, RISKOptimizer allows you to describe the possible values for any spreadsheet element using any of the probability distribution functions available in @RISK. A value of 10, for example, in a spreadsheet cell could be replaced with the @RISK function =RiskNormal(10,2). This would specify that the possible values for the cell are described by a probability distribution with a mean of 10 and a standard deviation of 2. As in @RISK, probability distributions can be correlated using @RISK functions such RiskCorrmat and DepC.

Optimization Using Simulation

When optimizing, RISKOptimizer runs a full simulation for each possible trial solution that is generated by the GA-based optimizer. In each iteration of a trial solution's simulation, probability distribution functions in the spreadsheet are sampled and a new value for the target cell is generated. At the end of a simulation, the result for the trial solution is the statistic for the distribution of the target cell which you wish to minimize or maximize. This value is then returned to the optimizer and used by the genetic algorithms to generate new and better trial solutions. For each new trial solution, another simulation is run and another value for the target statistic is generated.

As in traditional optimizers, constraints that need to be met can be entered in RISKOptimizer. Constraints can either be checked each iteration of a simulation (an **"iteration" constraint**) or at the end of each simulation (a **"simulation" constraint**). Iterations constraints are typically traditional Solver or Evolver style constraints, such as *A11>1000*. Simulation constraints are constraints that reference a statistic on the distribution of simulation results for any cell in your model you specify. A typical simulation constraint could be *"Mean of A11>1000"* or the mean of the distribution of simulation results for cell A11 must be greater than 1000. As in Evolver, constraints can be hard or soft, and a violated hard constraint causes a trial solution to be rejected.

As large numbers of simulations are being run by RISKOptimizer, two important techniques are used to minimize runtimes and generate optimal solutions as quickly as possible. First, RISKOptimizer uses convergence monitoring to determine when a sufficient number of iterations have been run (but not too many). This insures that the resulting statistic from the target cell's probability distribution is stable, and that any statistics from output distributions referenced in constraints are stable. Secondly, RISKOptimizer uses Evolver's genetic operators to generate trial solutions that move toward an optimal solution as quickly as possible.

Simulation Results

RISKOptimizer comes with a set of simulation statistics functions which can be used to return simulation results directly to your spreadsheet. The function *RiskMean(cell reference)*, for example, returns the mean of the simulated distribution for the entered cell directly to a worksheet cell or formula. In addition, any model built in RISKOptimizer can be directly simulated in @RISK, Palisade Corporation's risk analysis and simulation add-in for Excel, when you want to get detailed graphics and statistics on the best model solution found by RISKOptimizer. Because RISKOptimizer's simulation is based on @RISK, no changes to a RISKOptimizer model are required to simulate it in @RISK!

Custom Applications Using RISKOptimizer

RISKOptimizer comes with a complete macro language for building custom applications which use RISKOptimizer's capabilities. RISKOptimizer's custom functions can be used in Visual Basic for Applications (VBA) for setting up and running optimizations and displaying the results from optimizations. For more information on this programming interface, see the RISKOptimizer Developer Kit help document, available via the RISKOptimizer Help menu.

Applications of Simulation Optimization Using RISKOptimizer

The availability of optimization for uncertain models allows the solution of many previously "unoptimizable" problems. As a rule, any model that has uncertain elements can be optimized through the combination of simulation and optimization, including:

- ◆ Selection of optimal production and capacity levels for new products with uncertain market conditions
- ◆ Identification of optimal inventory levels with uncertain demand
- ◆ Portfolio allocation for risk minimization
- ◆ Identification of optimal product mix from a factory where product markets are geographically distributed and demand levels are uncertain
- ◆ Determining optimal levels for options purchases when hedging
- ◆ Yield management when the same product is sold at different prices under different restrictions
- ◆ Scheduling with uncertain task times

Before You Begin

Before you install and begin working with RISKOptimizer, make sure that your RISKOptimizer package contains all the required items, and check that your computer meets the minimum requirements for proper use.

What the Package Includes

RISKOptimizer ships with the @RISK Industrial version, and the DecisionTools Suite Industrial version. The @RISK Industrial CD-ROM contains the RISKOptimizer Excel add-in, several RISKOptimizer examples, and a fully-indexed RISKOptimizer on-line help system, in addition to @RISK for Excel files contained with @RISK Industrial for Excel. The DecisionTools Suite Industrial version contain all of the above plus additional applications.

About This Version

This version of RISKOptimizer can be installed as a 32-bit program for Microsoft Excel 2000 or higher.

Working with your Operating Environment

This User's Guide assumes that you have a general knowledge of the Windows operating system and Excel. In particular:

- ◆ *You are familiar with your computer and using the mouse.*
- ◆ *You are familiar with terms such as icons, click, double-click, menu, window, command and object.*
- ◆ *You understand basic concepts such as directory structures and file naming.*

If You Need Help

Technical support is provided free of charge for all registered users of RISKOptimizer with a current maintenance plan, or is available on a per incident charge. To ensure that you are a registered user of RISKOptimizer, **please register online at <http://www.palisade.com/support/register.asp>.**

If you contact us by telephone, please have your serial number and User's Guide ready. We can offer better technical support if you are in front of your computer and ready to work.

Before Calling

Before contacting technical support, please review the following checklist:

- *Have you referred to the on-line help?*
- *Have you checked this User's Guide and reviewed the on-line multimedia tutorial?*
- *Have you read the README.WRI file? It contains current information on RISKOptimizer that may not be included in the manual.*
- *Can you duplicate the problem consistently? Can you duplicate the problem on a different computer or with a different model?*
- *Have you looked at our site on the World Wide Web? It can be found at **<http://www.palisade.com>**. Our Web site also contains the latest FAQ (a searchable database of tech support questions and answers) and RISKOptimizer patches in our Technical Support section. We recommend visiting our Web site regularly for all the latest information on RISKOptimizer and other Palisade software.*

Contacting Palisade

Palisade Corporation welcomes your questions, comments or suggestions regarding RISKOptimizer. Contact our technical support staff using any of the following methods:

- *Email us at support@palisade.com.*
- *Telephone us at (607) 277-8000 any weekday from 9:00 AM to 5:00 PM, EST. Follow the prompt to reach technical support.*
- *Fax us at (607) 277-8001.*
- *Mail us a letter at:*

**Technical Support
Palisade Corporation
798 Cascadilla St.
Ithaca, NY 14850 USA**

If you want to contact Palisade Europe:

- *Email us at support@palisade-europe.com.*
- *Telephone us at +44 1895 425050 (UK).*
- *Fax us at +44 1895 425051 (UK).*
- *Mail us a letter at:*

**Palisade Europe
31 The Green
West Drayton
Middlesex
UB7 7PN
United Kingdom**

If you want to contact Palisade Asia-Pacific:

- *Email us at support@palisade.com.au.*
- *Telephone us at + 61 2 9252 5922 (AU).*
- *Fax us at + 61 2 9252 2820 (AU).*
- *Mail us a letter to:*

**Palisade Asia-Pacific Pty Limited
Suite 404, Level 4
20 Loftus Street
Sydney NSW 2000
Australia**

Regardless of how you contact us, please include the product name, version and serial number. The exact version can be found by selecting the Help About command on the RISKOptimizer menu in Excel.

Student Versions

Telephone support is not available with the student version of RISKOptimizer. If you need help, we recommend the following alternatives:

- ◆ *Consult with your professor or teaching assistant.*
- ◆ *Log on to <http://www.palisade.com> for answers to frequently asked questions.*
- ◆ *Contact our technical support department via e-mail or fax.*

RISKOptimizer System Requirements

System requirements for RISKOptimizer include:

- *Pentium PC or faster with a hard disk.*
- *Microsoft Windows 2000 SP4 or higher.*
- *Microsoft Excel Version 2000 or higher.*

Installation Instructions

RISKOptimizer is an add-in program to Microsoft Excel. By adding additional commands to the Excel menu bars, RISKOptimizer enhances the functionality of the spreadsheet program.

General Installation Instructions

The Setup program copies the RISKOptimizer system files into a directory you specify on your hard disk. To run the Setup program in Windows 2000 or higher:

- 1) *Insert the @RISK Industrial version or DecisionTools Suite Industrial version CD-ROM in your CD-ROM drive*
- 2) *Click the Start button, click Settings and then click Control Panel*
- 3) *Double-click the Add/Remove Programs icon*
- 4) *On the Install/Uninstall tab, click the Install button*
- 5) *Follow the Setup instructions on the screen*

If you encounter problems while installing RISKOptimizer, verify that there is adequate space on the drive to which you're trying to install. After you've freed up adequate space, try rerunning the installation.

If you wish to remove RISKOptimizer (along with @RISK Industrial or the DecisionTools Suite Industrial version) from your computer, use the Control Panel's Add/Remove Programs utility and select the entry for @RISK or the DecisionTools Suite.

Removing RISKOptimizer from Your Computer

The DecisionTools Suite

RISKOptimizer can be used with the DecisionTools Suite, a set of products for risk and decision analysis available from Palisade Corporation. The default installation procedure of RISKOptimizer puts RISKOptimizer in a subdirectory of a main “Program Files\Palisade” directory. This is quite similar to how Excel is often installed into a subdirectory of a “Microsoft Office” directory.

One subdirectory of the Program Files\Palisade directory will be the RISKOptimizer directory (by default called RISKOptimizer5). This directory contains the RISKOptimizer add-in program file (RISKOPT.XLA) plus example models and other files necessary for RISKOptimizer to run. Another subdirectory of Program Files\Palisade is the SYSTEM directory which contains files needed by every program in the DecisionTools Suite, including common help files and program libraries.

Setting Up the RISKOptimizer Icons or Shortcuts

In Windows, setup automatically creates a RISKOptimizer command in the Programs menu of the Taskbar. However, if problems are encountered during Setup, or if you wish to do this manually another time, follow these directions:

- 1) Click the Start button, and then point to Settings.
- 2) Click Taskbar, and then click the Start Menu Programs tab.
- 3) Click Add, and then click Browse.
- 4) Locate the file RISKOPT.EXE and double click it.
- 5) Click Next, and then double-click the menu on which you want the program to appear.
- 6) Type the name "RISKOptimizer", and then click Finish.

Macro Security Warning Message on Startup

Microsoft Office provides several security settings (under **Tools>Macro>Security**) to keep unwanted or malicious macros from being run in Office applications. A warning message appears each time you attempt to load a file with macros, unless you use the lowest security setting. To keep this message from appearing every time you run a Palisade add-in, Palisade digitally signs their add-in files. Thus, once you have specified **Palisade Corporation** as a trusted source, you can open any Palisade add-in without warning messages. To do this:

- Click **Always trust macros from this source** when a Security Warning dialog (such as the one below) is displayed when starting RISKOptimizer.



Other RISKOptimizer Information

Additional information on RISKOptimizer can be found in the following sources:

RISKOptimizer Readme

This file contains a quick summary of RISKOptimizer, as well as any late-breaking news or information on the latest version of your software. View the Readme file by selecting the Windows Start Menu/ Programs/ Palisade DecisionTools/ Readmes and clicking on RISKOptimizer 5.0 – Readme. It is a good idea to read this file before using RISKOptimizer.

RISKOptimizer Tutorial

The RISKOptimizer on-line tutorial provides first-time users with a quick introduction of RISKOptimizer and genetic algorithms. The presentation takes only a few minutes to view. See the Learning RISKOptimizer section below for information on how to access the tutorial.

Learning RISKOptimizer

The quickest way to become familiar with RISKOptimizer is by using the on-line RISKOptimizer Tutorial, where experts guide you through sample models in movie format. This tutorial is a multi-media presentation on the main features of RISKOptimizer.

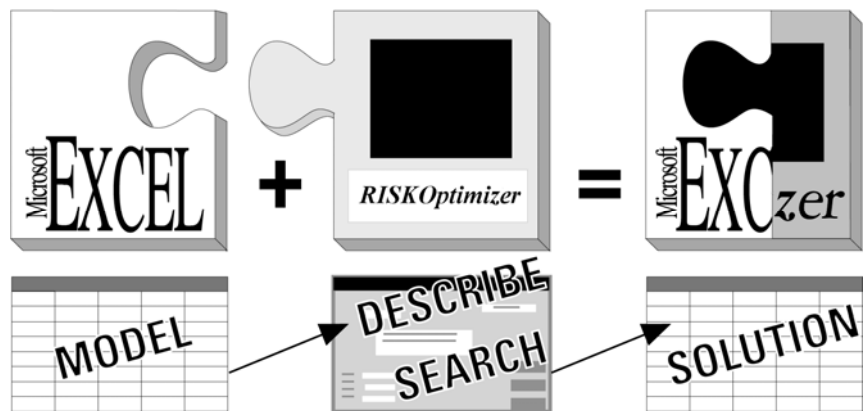
The tutorial can be run by selecting the **RISKOptimizer Help menu Getting Started Tutorial command**.

Chapter 2: Background

What Is RISKOptimizer?	17
How does RISKOptimizer work?	18
Genetic Algorithms	18
Probability Distributions and Simulation.....	18
What Is Optimization?	19
Why Build Excel Models?.....	20
Modeling Uncertainty in Excel Models	21
Using Simulation to Account for Uncertainty	21
Why Use RISKOptimizer?.....	22
More Accurate, More Meaningful	22
More Flexible.....	23
Easier to Use	23
Traditional Optimization vs. Simulation Optimization	25
Traditional Spreadsheet Optimization Process.....	25
Simulation Optimization Process	26
Figure 2-1	27
Each Step of an Optimization with RISKOptimizer	28
Entering Probability Distributions	28
Identify the Target Cell and Statistic.....	29
Entering Adjustable Cells.....	29
Entering Constraints	30
Setting the Optimization and Simulation Options.....	30
Running the Optimization.....	31

What Is RISKOptimizer?

The RISKOptimizer software package provides users with an easy way to find optimal solutions to models that include uncertainty. Simply put, RISKOptimizer finds the best inputs that produce a desired simulation output. You can use RISKOptimizer to find the right mix, order, or grouping of variables that produces the highest expected value for profits, the lowest risk (i.e., the minimum variance) for profits, or the largest expected value for goods from the least amount of materials. RISKOptimizer is an add-in to the Microsoft Excel spreadsheet program; users set up a model of their problem in Excel, then call up RISKOptimizer to solve it.



You must first model your problem in Excel, then describe it to the RISKOptimizer add-in.

Excel provides all of the formulas, functions, graphs, and macro capabilities that most users need to create realistic models of their problems. RISKOptimizer provides the interface to describe the uncertainty in your model and what you are looking for, and provides the engines that will find it. Together, they can find optimal solutions to virtually any problem that can be modeled.

How does RISKOptimizer work?

RISKOptimizer uses a proprietary set of *genetic algorithms* to search for optimum solutions to a problem, along with *probability distributions* and *simulation* to handle the uncertainty present in your model.

Genetic Algorithms

Genetic algorithms are used in RISKOptimizer to find the best solution for your model. Genetic algorithms mimic Darwinian principles of natural selection by creating an environment where hundreds of possible solutions to a problem can compete with one another, and only the “fittest” survive. Just as in biological evolution, each solution can pass along its good “genes” through “offspring” solutions so that the entire population of solutions will continue to evolve better solutions.

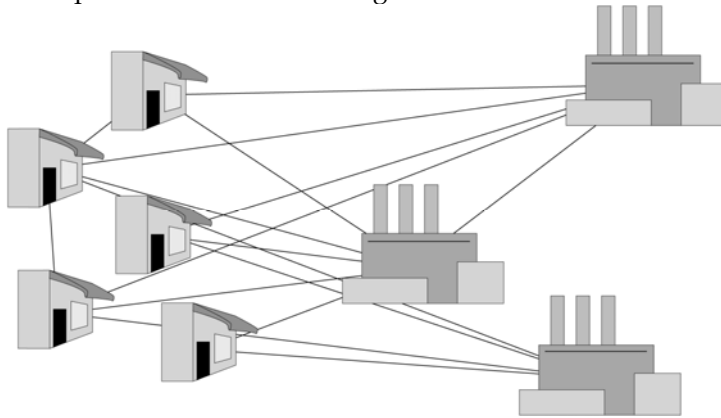
As you may already realize, the terminology used when working with genetic algorithms is often similar to that of its inspiration. We talk about how “crossover” functions help focus the search for solutions, “mutation” rates help diversify the “gene pool”, and we evaluate the entire “population” of solutions or “organisms”. To learn more about how RISKOptimizer’s genetic algorithm works, see [Chapter 7 - Genetic Algorithms](#).

Probability Distributions and Simulation

Probability distributions and simulation are used in RISKOptimizer to handle the uncertainty present in the variables in your model. These capabilities are taken from @RISK, Palisade Corporation's risk analysis add-in for Excel. Probability distributions are used to describe the range of possible values for uncertain elements in your model and are entered using probability distribution functions such as *RiskTriang(10,20,30)*. This would specify that a variable in your model could take a minimum value of 10, a most likely value of 20 and a maximum value of 30. Simulation is then used to generate a distribution of possible outcomes for each possible trial solution that is generated by the optimizer.

What Is Optimization?

Optimization is the process of trying to find the best solution to a problem that may have many possible solutions. Most problems involve many variables that interact based on given formulas and constraints. For example, a company may have three manufacturing plants, each manufacturing different quantities of different goods. Given the cost for each plant to produce each good, the costs for each plant to ship to each store, and the limitations of each plant, what is the optimal way to adequately meet the demand of local retail stores while minimizing the transportation costs? This is the sort of question that optimization tools are designed to answer.



Optimization often deals with searching for the combination that yields the most from given resources.

In the example above, each proposed solution would consist of a complete list of what goods made by what manufacturing plant get shipped in what truck to what retail store. Other examples of optimization problems include finding out how to produce the highest profit, the lowest cost, the most lives saved, the least noise in a circuit, the shortest route between a set of cities, or the most effective mix of advertising media purchases. An important subset of optimization problems involves scheduling, where the goals may include maximizing efficiency during a work shift or minimizing schedule conflicts of groups meeting at different times. To learn more about optimization, see [Chapter 6 - Optimization](#).

When a problem includes uncertainty, traditional solvers will fail because they have no capabilities for dealing with the uncertainty present in a model. In the above example, what if the demand of local retail stores is uncertain - that is, you don't know exactly what

quantities of products will be demanded by each store? With a traditional solver, you would just assume a quantity for demand from each store. This would allow the model to be optimized; however, the assumed demand levels would make your model an inaccurate depiction of what will actually occur. With RISKOptimizer, you don't have to assume a level for demand. You describe the possible values for demand using a probability distribution and then use RISKOptimizer's built-in simulation capabilities to include all possible values for demand in your optimization results.

When RISKOptimizer is used, the best solution generated by the optimizer is not a single maximum or minimum value for the objective or "target cell" in the model you are trying to optimize, but a maximum or minimum simulation statistic for the objective. Each simulation run by RISKOptimizer generates a distribution of possible results for your objective. This distribution has a variety of statistics, such as a mean, standard deviation, minimum, etc. In the above example, you might want to find the combination of inputs that maximizes the mean of the distribution for profit or minimizes its standard deviation.

To learn more about simulation, see [Chapter 8 - Simulation](#).

Why Build Excel Models?

To increase the efficiency of any system, we must first understand how it behaves. This is why we construct a working model of the system. Models are necessary abstractions when studying complex systems, yet in order for the results to be applicable to the "real-world," the model must not oversimplify the cause-and-effect relationships between variables. Better software and increasingly powerful computers allow economists to build more realistic models of the economy, scientists to improve predictions of chemical reactions, and business people to increase the sensitivity of their corporate models.

In the last few years computer hardware and software programs such as Microsoft Excel, have advanced so dramatically that virtually anyone with a personal computer can create realistic models of complex systems. Excel's built-in functions, macro capabilities and clean, intuitive interface allow beginners to model and analyze sophisticated problems. To learn more about building a model, see [Chapter 9 - RISKOptimizer Extras](#).

Modeling Uncertainty in Excel Models

Variables are the basic elements in your Excel models that you have identified as being important ingredients to your analysis. If you are modeling a financial situation, your variables might be things like Sales, Costs, Revenues or Profits. If you are modeling a geologic situation, your variables might be things like Depth to Deposit, Thickness of Coal Seam, or Porosity. Each situation has its own variables, identified by you.

In some cases, you may know the values your variables will take in the time frame of your model — they are certain or what statisticians call "deterministic". Conversely, you may not know the values they will take — they are uncertain, or "stochastic". If your variables are uncertain you will need to describe the nature of their uncertainty. This is done with probability distributions, which give both the range of values that the variable could take (minimum to maximum), and the likelihood of occurrence of each value within the range. In RISKOptimizer, uncertain variables and cell values are entered as probability distribution functions, for example:

RiskNormal(100,10)

RiskUniform(20,30)

RiskExpon(A1+A2)

RiskTriang(A3/2.01,A4,A5)

These "distribution" functions can be placed in your worksheet cells and formulas just like any other Excel function.

Using Simulation to Account for Uncertainty

RISKOptimizer uses simulation, sometimes called Monte Carlo simulation, to do a Risk Analysis on each possible solution generated during an optimization. Simulation in this sense refers to a method whereby the distribution of possible outcomes is generated by letting a computer recalculate your worksheet over and over again, each time using different randomly selected sets of values for the probability distributions in your cell values and formulas. In effect, the computer is trying all valid combinations of the values of input variables to simulate all possible outcomes. This is just as if you ran hundreds or thousands of "what-if" analyses on your worksheet, all in one sitting.

In each iteration of the simulation, probability distribution functions in the spreadsheet are sampled and a new value for the target cell is generated. At the end of a simulation, the result for the trial solution is the statistic which you wish to minimize or maximize for the distribution of the target cell. This value is then returned to the optimizer and used by the genetic algorithms to generate new and better trial solutions. For each new trial solution, another simulation is run, and another value for the target statistic is generated.

Why Use RISKOptimizer?

When you are dealing with large numbers of interacting variables, and you are trying to find the best mix, the right order, or the optimum grouping of those variables, you may be tempted to just take an “educated guess”. A surprising number of people assume that any kind of modeling and analysis beyond guessing will require complicated programming, or confusing statistical or mathematical algorithms. A good optimized solution might save millions of dollars, thousands of gallons of scarce fuel, months of wasted time, etc. Now that powerful desktop computers are increasingly affordable, and software like Excel and RISKOptimizer are readily available, there is little reason to guess at solutions, or waste valuable time trying out many scenarios by hand.

***More Accurate,
More
Meaningful***

RISKOptimizer allows you to use the entire range of Excel formulas and probability distributions to build more realistic models of any system. When you use RISKOptimizer, you do not have to “compromise” the accuracy of your model because the algorithm you are using can not handle real world complexities. Traditional “baby” solvers (statistical and linear programming tools) force users to make assumptions about the way the variables in their problem interact, thereby forcing users to build over-simplified, unrealistic models of their problem. They force them to assume values for uncertain variables because the optimizer cannot handle ranges of possible values for uncertain model components. By the time users have simplified a system enough that these solvers can be used, the resulting solution is often too abstract to be practical. Any problems involving large amounts of variables, non-linear functions, lookup tables, if-then statements, database queries, or stochastic (random) elements cannot be solved by these methods, no matter how simply you try to design your model.

More Flexible

There are many solving algorithms which do a good job at solving small, simple linear and non-linear types of problems, including hill-climbing, baby-solvers, and other mathematical methods. Even when offered as spreadsheet add-ins, these general-purpose optimization tools can only perform numerical optimization. For larger or more complex problems, you may be able to write specific, customized algorithms to get good results, but this may require a lot of research and development. Even then, the resulting program would require modification each time your model changed.

Not only can RISKOptimizer handle numerical problems, it is the only commercial program in the world that can solve most combinatorial problems. These are problems where the variables must be shuffled around (permuted) or combined with each other. For example, choosing the batting order for a baseball team is a combinatorial problem; it is a question of swapping players' positions in the lineup. Complex scheduling problems are also combinatorial. The same RISKOptimizer can solve all these types of problems and many more that nothing else can solve. RISKOptimizer's unique *genetic algorithm* and *simulation* technology allows it to optimize virtually any type of model; any size and any complexity.

Easier to Use

In spite of its obvious power and flexibility advantages, RISKOptimizer remains easy to use because an understanding of the complex genetic algorithm techniques it uses is completely unnecessary. RISKOptimizer doesn't care about the "nuts and bolts" of your problem; it just needs a spreadsheet model that can evaluate how good different scenarios are. Just select the spreadsheet cells that contain the variables and tell RISKOptimizer what you are looking for. RISKOptimizer intelligently hides the difficult technology, automating the "what-if" process of analyzing a problem.

Although there have been many commercial programs developed for mathematical programming and model-building, spreadsheets are by far the most popular, with literally millions being sold each month. With their intuitive row and column format, spreadsheets are easier to set up and maintain than other dedicated packages. They are also more compatible with other programs such as word processors and databases, and offer more built-in formulas, formatting options, graphing, and macro capabilities than any of the stand-alone packages. Because RISKOptimizer is an add-in to Microsoft Excel, users have access to the entire range of functions and development tools to easily build more realistic models of their system.

Traditional Optimization vs. Simulation Optimization

RISKOptimizer combines simulation and optimization to allow the optimization of models with uncertain factors. The optimizer uses the results from successive runs of the simulation model to guide its search for better, more optimal solutions. This section provides background information on how simulation and optimization work together in RISKOptimizer.

Traditional Spreadsheet Optimization Process

In the traditional process for optimizing a spreadsheet using an optimization add-in such as Solver or Evolver, the following steps are undertaken:

- 1) **An output or “target” cell that you wish to minimize or maximize is identified.**
- 2) **A set of input or “adjustable” cells whose values you control are identified, and ranges of possible values for those cells are described.**
- 3) **A set of constraints that need to be met, often specified using expressions such as $COSTS < 100$ or $A11 \geq 0$ are entered.**
- 4) **An optimization is run, in which the spreadsheet is recalculated successive times using different possible values for the adjustable cells.**
- 5) **During this process:**
 - a) Each recalculation generates a new "answer" or value for the target cell.
 - b) The optimizer uses this new target cell value to select the next set of values for the adjustable cells it will try.
 - c) Another recalculation is performed, providing another new answer that the optimizer can use for identifying a new set of values for the adjustable cells.

This process in 5) repeats over and over again, as the optimizer moves towards identifying an optimal solution - that is, the set of values for the adjustable cells that minimizes or maximizes the target cell value.

Simulation Optimization Process

Simulation optimization using RISKOptimizer follows many of the same steps as the traditional spreadsheet optimization process outlined here. However, changes are made to 1) **allow uncertainty to be entered in the spreadsheet** and 2) **to use simulation, instead of a simple recalculation of the spreadsheet**, to provide the new target cell "answer" that provides feedback to the optimizer for guiding its selection of a new set of values for the adjustable cells.

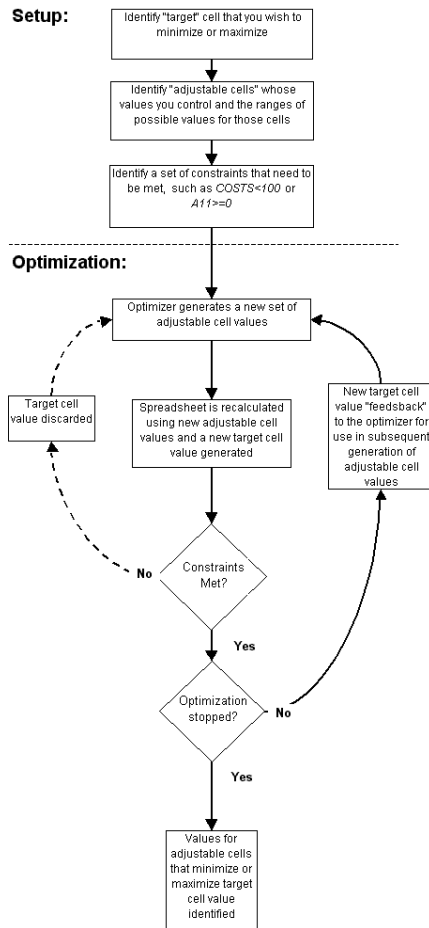
The new process for simulation optimization using RISKOptimizer is described below, with differences from traditional spreadsheet optimization shown in bold:

- 1) **Probability distribution functions are used to describe the range of possible values for the uncertain elements in the model.**
- 2) An output or "target" cell is identified **and the simulation statistic (mean, standard deviation, etc.) for the cell that you wish to minimize or maximize is selected.**
- 3) A set of input or "adjustable" cells whose values you control are identified and ranges of possible values for those cells are described.
- 4) A set of constraints that need to be met, often specified using expressions such as $COSTS < 100$ or $A11 \geq 0$ are entered.
Additional constraints based on simulation statistics (i.e., 95th Percentile of $A11 > 1000$) can also be entered.
- 5) An optimization is run, in which the spreadsheet is **simulated** successive times, **with each simulation** using different possible values for the adjustable cells. During this process:
 - a) **Each simulation generates a new distribution of possible values for the target cell. The statistic you wish to minimize or maximize is calculated from this distribution.**
 - b) **The optimizer uses this new statistic** for the target cell to select the next set of values for the adjustable cells it will try.
 - c) **Another simulation** is performed, providing another new **statistic** that the optimizer can use for identifying a new set of values for the adjustable cells

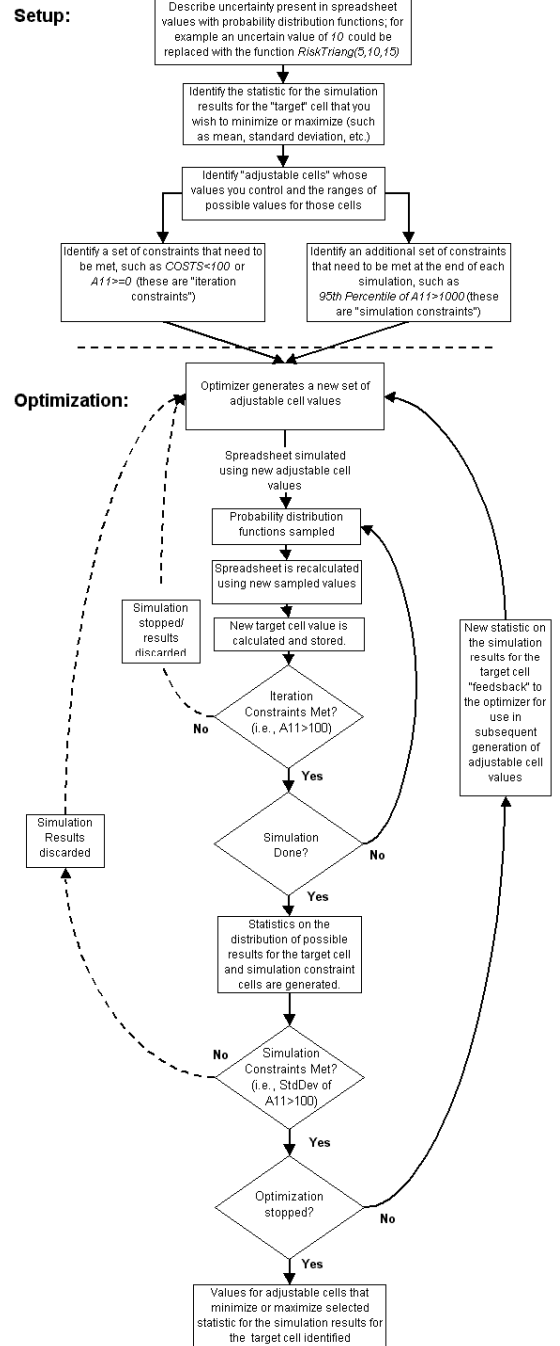
This process in 5) repeats over and over again, as the optimizer moves towards identifying an optimal solution - that is, the set of values for the adjustable cells that minimizes or maximizes the statistic for the simulation results for the target cell. Figure 2-1 outlines both the traditional optimization and simulation optimization processes.

Figure 2-1

Traditional Spreadsheet Optimization:



Spreadsheet Simulation Optimization:



Each Step of an Optimization with RISKOptimizer

Each step in the simulation optimization process used by RISKOptimizer is detailed here:

Entering Probability Distributions

Probability distributions are used in RISKOptimizer to describe the uncertainty present in the components of a model. For example, you could enter *RiskUniform(10,20)* to a cell in your worksheet. This specifies that the values for the cell will be generated by a uniform distribution with a minimum of 10 and a maximum of 20. This range of values replaces the single "fixed" value required by Excel. In traditional spreadsheet optimization, no uncertainty can be added to a model so probability distributions are not used.

In RISKOptimizer, a simulation of your model is run for each possible combination of input values generated by the optimizer. Distribution functions are used by RISKOptimizer during these simulations for sampling sets of possible values. Each iteration of a simulation uses a new set of values sampled from each distribution function in your worksheet. These values are then used in recalculating your worksheet and generating a new value for your target cell.

As with Excel functions, distribution functions contain two elements, a function name and argument values which are enclosed in parentheses. A typical distribution function is:

RiskNormal(100,10)

Like Excel functions, distribution functions may have arguments which are references to cells or expressions. For example:

RiskTriang(B1,B2*1.5,B3)

In this case the cell value would be specified by a triangular distribution with a minimum value taken from cell B1, a most likely value calculated by taking the value for cell B2 and multiplying it by 1.5 and a maximum value taken from cell B3.

Distribution functions also may be used in cell formulas, just as are Excel functions. For example, a cell formula could read:

B2: 100+RiskUniform(10,20)+(1.5*RiskNormal(A1,A2))

For more information on entering probability distributions, see Reference: Distribution Functions in the @RISK manual or Help.

Identify the Target Cell and Statistic

In both RISKOptimizer and in a traditional spreadsheet optimization, a target cell is identified. This is the cell whose value you are trying to minimize or maximize, or the cell whose value you are trying to make as close as possible to a pre-set value. Typically this is the "result" of your model - *profit, the model's grand total, etc.* - but it can be any cell in the spreadsheet. The cell needs to have a formula in it that will return different values as the values in your adjustable cells change.

In RISKOptimizer, you are not minimizing or maximizing the actual value in the target cell; you are **minimizing or maximizing a "statistic" associated with the simulation results for the target cell.** During an optimization, RISKOptimizer will run successive simulations, each with a different set of adjustable cell values. Each simulation generates a distribution of possible outcomes for the target cell. You are looking for the set of adjustable cell values that, for example, maximizes the mean of the target cell's distribution, or minimizes its standard deviation.

In RISKOptimizer you have more options on what to minimize or maximize (*mean, standard deviation, minimum, etc.*) because - for each solution tried by the optimizer - the associated simulation does not just generate a single answer. The simulation generates a full distribution of possible outcomes for the target cell, with a minimum value, a maximum, a mean, a standard deviation and so on. A traditional optimization generates just one thing - a new target cell value - for each solution tried by the optimizer and this value is the only possible selection for minimizing or maximizing.

Entering Adjustable Cells

Adjustable cells are entered in a similar fashion in both traditional spreadsheet optimization and RISKOptimizer. For each cell which can be changed during an optimization, a minimum possible value and a maximum possible value are entered.

Since the optimizer used by RISKOptimizer is based on Evolver, the entry of adjustable cells in RISKOptimizer has the same options as in Evolver. This includes mutation rate, solving method and genetic operators. For more information on entering adjustable cells, see the section "*Adjustable Cell Ranges*" in Chapter 5: RISKOptimizer Reference.

Entering Constraints

In RISKOptimizer, as in traditional spreadsheet optimization, hard constraints that must be met can be entered. In traditional spreadsheet optimization, hard constraints are tested for each trial solution. If they are not met, the solution is discarded.

In RISKOptimizer, a full simulation is run for each trial solution. Each simulation is comprised of a number of iterations, or individual recalculations of the spreadsheet using new samples from probability distributions in the model. A **hard constraint** can be tested:

- ◆ **Each iteration of each simulation** (an *iteration constraint*). If an iteration results in values which violate the hard constraint, the simulation is stopped (and trial solution rejected) and the next trial solution and its associated simulation begins.
- ◆ **At the end of the simulation** (a *simulation constraint*). This type of constraint is specified in terms of a simulation statistic for a spreadsheet cell; for example the *Mean of A11>1000*. In this case, the constraint is evaluated at the end of a simulation. A simulation constraint, as opposed to an iteration constraint, will never cause a simulation to be stopped prior to completion.

A second form of constraints - "soft constraints" can also be used in RISKOptimizer. The resulting penalties from soft constraints are calculated at the end of a simulation. Any penalty calculated is added to (or subtracted from) the target statistic which is being minimized or maximized.

For more information on entering constraints, see the section "**Constraints**" in [Chapter 5: RISKOptimizer Reference](#).

Setting the Optimization and Simulation Options

In RISKOptimizer, as in traditional spreadsheet optimization, a variety of options are available for controlling how long an optimization runs. However, RISKOptimizer adds new options for controlling how long each simulation runs for each trial solution.

RISKOptimizer will search for better solutions and run simulations until the selected optimization stopping options are met. You might have RISKOptimizer run a specified number of minutes, run until it has generated a specific number of trial solutions or run until the best simulation statistic for the target cell has not changed for a given number of trials.

You can also specify how long each trial solution's simulation will run. You may select to have each simulation run a specified number of iterations or, alternatively, let RISKOptimizer determine when to stop each simulation. When you select to have RISKOptimizer decide stop each simulation it will stop simulating when distributions generated for both 1) the target cell of the optimization and 2) cells referenced in simulation constraints are stable and the statistics of interest converge.

Running the Optimization

When RISKOptimizer runs an optimization the spreadsheet is **simulated** successive times, **with each simulation** using different possible values for the adjustable cells. During this process:

- 1) **The optimizer generates a set of values for the adjustable cells.**
- 2) **The spreadsheet is simulated with the adjustable cells set to the values generated by the optimizer .** In each iteration of the simulation all distribution functions in the spreadsheet are sampled and the spreadsheet is recalculated, generating a new value for the target cell. If any iteration constraints are not met after an iteration's recalculation, the simulation stops and the optimizer generates a new trial solution to be simulated.
- 3) **At the end of each simulation a new distribution of possible values for the target cell is generated. The statistic you wish to minimize or maximize is calculated from this distribution.** If any simulation constraints are not met, the trial solution and simulation results are discarded and the optimizer generates a new trial solution to be simulated
- 4) The optimizer uses the new **statistic** for the target cell calculated in the simulation to select the next set of values for the adjustable cells it will try.
- 5) Another **simulation** is performed, providing another new **statistic** that the optimizer can use for identifying a new set of values for the adjustable cells

This process repeats itself over and over again, as the optimizer moves towards identifying an optimal solution - that is, the set of values for the adjustable cells that minimizes or maximizes the statistic for the target cell.

Chapter 3:

RISKOptimizer: Step-by-Step

Introduction	35
The RISKOptimizer Tour	37
Starting RISKOptimizer	37
The RISKOptimizer Toolbar.....	37
Opening an Example Model.....	37
Describing Uncertainty in the Model.....	39
The RISKOptimizer Model Dialog	41
Selecting the Statistic for the Target Cell	41
Adding Adjustable Cell Ranges.....	42
Entering the Min-Max Range for Adjustable Cells	42
Selecting a Solving Method.....	44
Constraints	45
Iteration and Simulation Constraints	45
Adding a Constraint.....	46
Simple Range of Values and Formula Constraints	47
Other RISKOptimizer Options	50
Optimization Stopping Conditions	50
Simulation Stopping Conditions.....	51
Logging Simulation Data	52
Running the Optimization	53
The RISKOptimizer Watcher	54
Stopping the Optimization.....	55
Summary Report.....	56
Placing the Results in Your Model.....	57

Introduction

In this chapter, we will take you through an entire RISKOptimizer optimization one step at a time. If you do not have RISKOptimizer installed on your hard drive, please refer to the installation section of [Chapter 1: Introduction](#) and install RISKOptimizer before you begin this tutorial.

We will start by opening a pre-made spreadsheet model, and then we will define the problem to RISKOptimizer using probability distributions and the RISKOptimizer dialogs. Finally we will oversee RISKOptimizer's progress as it is searching for solutions, and explore some of the many options in the RISKOptimizer Watcher. For additional information about any specific topic, see the index at the back of this manual, or refer to [Chapter 5: RISKOptimizer Reference](#).

NOTE: *The screens shown below are from Excel 2007. If you are using other versions of Excel, your windows may appear slightly different from the pictures.*

The problem-solving process begins with a model that accurately represents your problem. Your model must be able to evaluate a given set of input values (adjustable cells) and produce a numerical rating of how well those inputs solve the problem (the evaluation or "fitness" function). Your model also needs to include probability distributions that describe the range of possible values for any uncertain elements. As RISKOptimizer searches for solutions, the simulation of the fitness function provides feedback, telling RISKOptimizer how good or bad each guess is, thereby allowing RISKOptimizer to breed increasingly better guesses. When you create a model of your problem, you must pay close attention to the fitness function, because RISKOptimizer will be doing everything it can to maximize (or minimize) the simulation results for this cell.

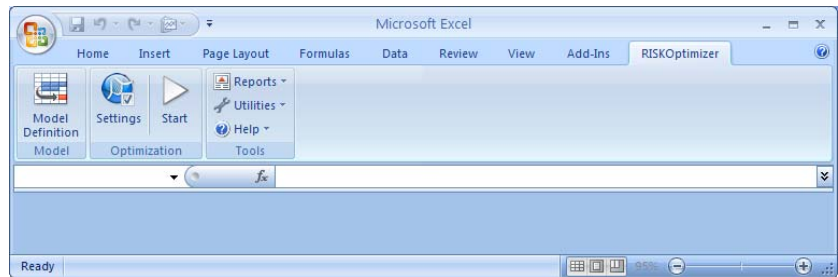
The RISKOptimizer Tour

Starting RISKOptimizer

To start RISKOptimizer, either: 1) *click the RISKOptimizer icon in your Windows desktop*, or 2) *select Palisade DecisionTools then RISKOptimizer 5.0 in the Windows Start menu Programs entries*. Each of these methods starts both Microsoft Excel and RISKOptimizer.

The RISKOptimizer Toolbar

When RISKOptimizer is loaded, a new RISKOptimizer toolbar or ribbon is visible in Excel. This toolbar contains buttons which can be used to specify RISKOptimizer settings and start, pause, and stop optimizations.



Opening an Example Model

To review the features of RISKOptimizer, you'll examine an example model that was installed when you installed RISKOptimizer. To do this:

- 1) *Open the AIRLINES.XLS worksheet from your RISKOPTIMIZER5\EXAMPLES directory.*

	A	B	C	D	E	F	G	H	I	J	K	L
3												
4		Seats Available	19									
5												
6	Full Fare	Ticket Price per Seat	\$195									
7	(fully refundable)	% No Shows	20.00%									
8		Demand for Full Fare Reservations	8									
9												
10	Discount	Ticket Price per Seat	\$85									
11	(\$80 change fee)	% No Shows	10.00%									
12		Demand for Discount Reservations	25									
13												
14		Maximum Reservations Accepted	28									
15		Percentage Sold at Full Fare	48.59%									
16												
17		Full Fare Reservations Accepted	8									
18		Discount Reservations Accepted	14									
19		Full Fare Passengers to Board	6									
20		Discount Passengers to Board	13									
21												
22		Cost of Bumping	\$150									
23												
24		Ticket Revenue	\$2,325									
25		Cost of Bumping Passengers	\$0									
26		Profit	\$2,325									
27												
28												
29												
30												
31												

This example sheet contains a yield management model which identifies the optimal number of full and discount fare seats to sell on a given flight. It also identifies the optimal number of reservations to accept in excess of the number of available seats – the classic “overbooking” problem. There’s just one catch to this standard optimization problem -- some estimates in the model are uncertain or "stochastic". This includes the number of passengers that will actually show up to board the flight, the number of reservations that will be demanded in each fare category and the cost of bumping a passenger (i.e., sometimes a \$100 travel voucher will suffice, while sometimes a free roundtrip is necessary). Traditionally, single point estimates are used for these items, allowing a normal optimization to be performed. But what if your estimates aren’t right? You might end up taking too few reservations, sending seats out empty, or overbooking too much. You could sell too many discount seats – lowering your profit. You might also set aside too many full fare seats, resulting in half-filled planes. RISKOptimizer will solve this optimization problem while allowing you to account for the uncertainty inherent to your model!

With the Airlines example, first you will describe the uncertainty present in your model using probability distributions. You will then use the RISKOptimizer dialogs to setup your optimization problem. Then, RISKOptimizer will run to identify the optimal number of full and discount fare reservations to maximize profit while keeping risk at acceptable levels.

Describing Uncertainty in the Model

In RISKOptimizer, probability distributions are used to describe the range of possible values for the uncertain elements in your model. A probability distribution specifies the minimum and maximum values for an uncertain factor and relative probabilities of values between the minimum and the maximum.

In RISKOptimizer, probability distributions are entered using probability distribution functions. These are custom RISKOptimizer functions that can be entered in the cells and formulas in your spreadsheet just as are standard Excel functions. For example, the function:

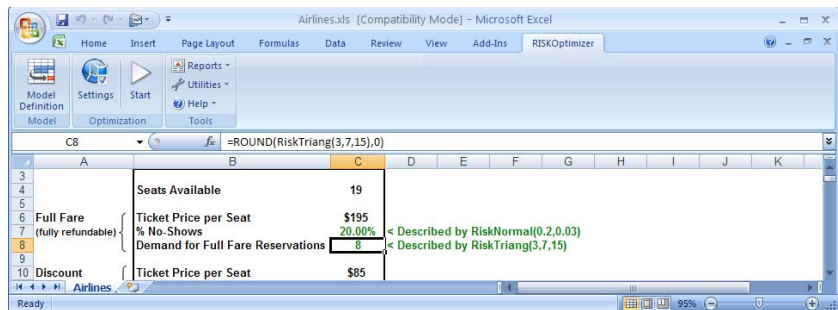
- ♦ **RiskTriang(10,20,30)** specifies a triangular distribution with a minimum possible value of 10, a most likely of 20 and a maximum of 30.

In the Airlines model there are five uncertain factors, each described by probability distributions. The first of these is:

- ♦ **Demand for Full Fare Reservations (in cell C8)**, described by the probability distribution *RiskTriang(3,7,15)*. This function specifies that the number of full fare reservations that will be demanded could be as low as three, as high as 15 with a most likely value of 7.

To enter this probability distribution:

- 1) *Select cell C8.*
- 2) *Enter the formula =ROUND(RiskTriang(3,7,15),0). The Excel ROUND function simply takes the sample returned by the RiskTriang function and rounds it to the nearest integer. (You can't have 5.65 reservations demanded!)*



The other distributions in the model, listed below, are already entered in the Airlines.XLS. You can move to the cell where each is located and review them if you like.

- ◆ **% No Shows - Full Fare reservations (in cell C7).** This is described by *RiskNormal(.2,.03)*, meaning on average 20% of full fare reservations made do not show up for the flight. The actual no show percentage will vary around 20% as described by a normal distribution with a mean of .2 and a standard deviation of .03.
- ◆ **% No Shows - Discount Fare reservations(in cell C11).** This is described by *RiskNormal(.1,.01)*, meaning on average 10% of discount fare reservations made do not show up for the flight. The actual no show percentage will vary around 10% as described by a normal distribution with a mean of .1 and a standard deviation of .01. More discount reservations show up as compared with full fare as there is a \$75 charge to change discount tickets vs. none for fully refundable full fare tickets.
- ◆ **Demand for Discount Fare Reservations(in cell C12),** described by the probability distribution *RiskTrigen(12,20,40,10,90)*. This function specifies that the number of discount reservations demanded is described by a triangular probability distribution whose 10th percentile is 12, most likely value is 20 and 90th percentile is 40.
- ◆ **Cost of Bumping (in cell C23),** described by the probability distribution *RiskDiscrete({100,150,200,250},{0.1,0.4,0.4,0.1})*. This specifies that the cost per passenger bumped can be \$100, \$150, \$200 or \$250, as sometimes passengers will volunteer to get off an overbooked flight for a \$100 travel voucher, while other times greater compensation is necessary.

For more information on these and other probability distributions, see Reference: Distribution Functions in the @RISK manual or Help.

With probability distributions describing uncertainty entered in your model, you can now set up the optimization using the RISKOptimizer dialogs.

The RISKOptimizer Model Dialog

To set the RISKOptimizer options for this worksheet:

- 1) *Click the RISKOptimizer Model icon on the RISKOptimizer toolbar (the one on the far left).*

This displays the following RISKOptimizer Model dialog box:

RISKOptimizer - Model

Optimization Goal:

Cell:

Statistic:

Adjustable Cell Ranges

Minimum	Range	Maximum	Values
---------	-------	---------	--------

Buttons: Add..., Delete, Group

Constraints

Description	Formula	Type
-------------	---------	------

Buttons: Add..., Edit..., Delete

Buttons: OK, Cancel

The RISKOptimizer Model Dialog is designed so users can describe their problem in a simple, straightforward way. In our tutorial example, we are trying to find the number of reservations for full fare and discount seats that should be accepted in order to maximize overall total profit.

Selecting the Statistic for the Target Cell

"Profit" in cell C27 in the Airlines.XLS model is what's known as the target cell. This is the cell whose simulation statistic you are trying to minimize or maximize, or the cell whose simulation statistic you are trying to make as close as possible to a pre-set value. To specify the simulation statistic for the target cell:

- 1) *Set the "Optimization Goal" option to "Maximum."*
- 2) *Enter the target cell, \$C\$27, in the "Cell" field.*

- 3) *Select "Mean" from the dropdown "Statistic" list to select the Mean as the simulation statistic to maximize.*

Cell references can be entered in RISKOptimizer dialog fields two ways: 1) You may click in the field with your cursor, and type the reference directly into the field, or 2) with your cursor in the selected field, you may click on Reference Entry icon to select the worksheet cell(s) directly with the mouse.

Adding Adjustable Cell Ranges

Now you must specify the location of the cells that contain values which RISKOptimizer can adjust to search for solutions. These variables are added and edited one block at a time through the *Adjustable Cells Dialog*. The number of cells you can enter in the Adjustable Cells dialog depends on the version of RISKOptimizer you are using.

- 1) *Click the "Add" button in the "Adjustable Cell Ranges" section.*
- 2) *Select C14 as the cell in Excel you want to add as an adjustable cell.*

Entering the Min-Max Range for Adjustable Cells

Most of the time you'll want to restrict the possible values for an adjustable cell range to a specific minimum-maximum range. In RISKOptimizer this is known as a "range" constraint. You can quickly enter this min-max range when you select the set of cells to be adjusted. For the Airlines example, the minimum possible value for reservations accepted in this range is 19 and the maximum is 30. To enter this range constraint:

- 1) *Enter 19 in the Minimum cell and 30 in the Maximum cell.*
- 2) *In the Values cell, select Integer from the drop-down list*

The dialog box shows the following settings:

- Optimization Goal: Maximum
- Cell: =C27
- Statistic: Mean

Under Adjustable Cell Ranges, there is a table with the following data:

Minimum		Range		Maximum	Values
19	<=	=C14	<=	30	Integer

Buttons on the right: Add..., Delete, Group.

Now, enter a second cell to be adjusted:

- 1) Click Add to enter a second adjustable cell.
- 2) Select cell C15.
- 3) Enter 0 as the Minimum and 1 as the Maximum.

The dialog box shows the following settings:

- Optimization Goal: Maximum
- Cell: =C27
- Statistic: Mean

Under Adjustable Cell Ranges, there is a table with the following data:

Minimum		Range		Maximum	Values
19	<=	=C14	<=	30	Integer
0	<=	=C15	<=	1	Any

Buttons on the right: Add..., Delete, Group.

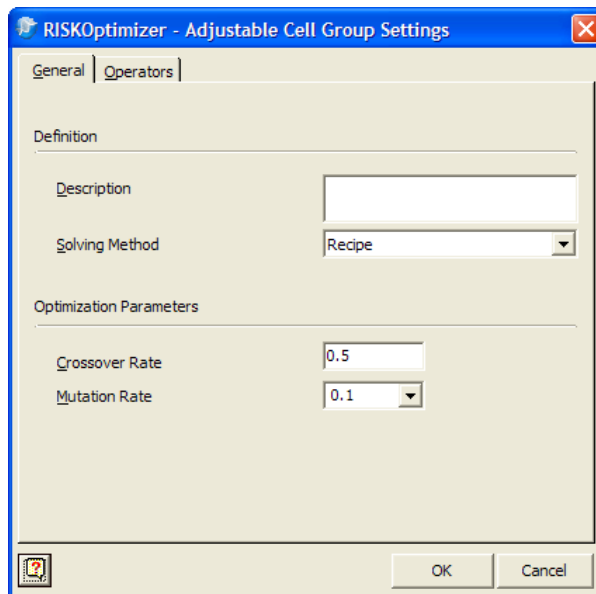
This specifies the last adjustable cell, C15, representing the percentage of total reservations that will be allotted to full fare seats.

If there were additional variables in this problem, we would continue to add sets of adjustable cells. In RISKOptimizer, you may create an unlimited number of groups of adjustable cells. To add more cells, click the “Add” button once again.

Later, you may want to check the adjustable cells or change some of their settings. To do this, simply edit the min-max range in the table. You may also select a set of cells and delete it by clicking the “Delete” button.

Selecting a Solving Method

When defining adjustable cells, you can specify a *solving method* to be used. Different types of adjustable cells are handled by different solving methods. Solving methods are set for a Group of adjustable cells and are changed by clicking the “*Group*” button and displaying the **Adjustable Cell Group Settings** dialog box. Often you'll use the default “recipe” solving method where each cell's value can be changed independently of the others. Since this is selected as the default method, you don't have to change it.



The “recipe” and “order” solving methods are the most popular and they can be used together to solve complex combinatorial problems. Specifically, the “recipe” solving method treats each variable as an ingredient in a recipe, trying to find the “best mix” by changing each variable’s value independently. In contrast, the “order” solving method swaps values between variables, shuffling the original values to find the “best order.”

Constraints

RISKOptimizer allows you to enter constraints which are conditions that must be met for a solution to be valid. In this example model there are two additional constraints that must be met for a possible set of values for *maximum number of reservations accepted* and *% full fare seats* to be valid. These are in addition to the range constraints we already entered for the adjustable cells. They are:

- ◆ **Profit must always be >0.**
- ◆ **Standard deviation of the simulation results for profit must be <400.**

Each time RISKOptimizer generates a possible solution to your model it will run a simulation for that solution. Each simulation will involve hundreds or thousands of iterations or recalculations of the spreadsheet. In each iteration, a value is sampled from each probability distribution in the model, the model is recalculated using these new sampled values and a new value for the target cell is generated. At the end of a trial solution's simulation a probability distribution for the target cell is generated using target cell values calculated for each iteration.

Iteration and Simulation Constraints

RISKOptimizer can check your constraints either:

- ◆ **After each iteration of a simulation (an "iteration" constraint)**
- ◆ **At the end of each simulation (a "simulation" constraint)**

In the Airlines model, "**Profit must always be >0**" is an iteration constraint, while "**Standard deviation of the simulation results for profit must be <400**" is a simulation constraint. In other words, after each iteration of a simulation RISKOptimizer will check to insure that Profit is greater than 0; if it is not the trial solution will be discarded. If a simulation completes successfully (i.e., Profit > 0 for all iterations) the standard deviation of the probability distribution for profit will be checked to insure that it is less than 400; if it is not the trial solution will be discarded.

Constraints are displayed in the bottom *Constraints* section of the RISKOptimizer Model dialog box. Two types of constraints can be specified in RISKOptimizer:

- ◆ **Hard.** These are conditions that must be met for a solution to be valid (i.e., a hard iteration constraint could be $C10 \leq A4$; in this case, if a solution generates a value for C10 that is greater than the value of cell A4, the solution will be thrown out)
- ◆ **Soft.** These are conditions which we would like to be met as much as possible, but which we may be willing to compromise for a big improvement in fitness or target cell result. (i.e., a soft constraint could be $C10 < 100$. In this case, C10 could go over 100, but when that happens the calculated value for the target cell would be decreased according to the penalty function you have entered).

Adding a Constraint

To add a constraint:

- 1) ***Click the Add button in the Constraints section of the main RISKOptimizer dialog.***

This displays the Constraint Settings dialog box, where you enter the constraints for your model.

The screenshot shows the 'RISKOptimizer - Constraint Settings' dialog box. It has a blue title bar with a close button. The dialog is divided into several sections: 'Description' (a text box), 'Constraint Type' (with radio buttons for 'Hard (Discards Solutions that Do Not Meet the Constraint)' and 'Soft (Disfavors Solutions that Do Not Meet the Constraint)'), 'Penalty Function' (a text box containing the formula '=100*(EXP(DEVIATION/100)-1)'), 'Definition' (with a dropdown for 'Entry Style' set to 'Simple', and input fields for 'Minimum' (0), 'Range to Constrain' (with a dropdown set to '<='), and 'Maximum' (0)), 'Statistic to Constrain' (a dropdown set to 'Value'), and 'Evaluation Time' (with radio buttons for 'Every Iteration of Each Simulation (Iteration Constraint)' and 'Only at the End of Each Simulation (Simulation Constraint)'). At the bottom are 'OK' and 'Cancel' buttons.

Simple Range of Values and Formula Constraints

Two formats – *Simple* and *Formula* – can be used for entering constraints. The Simple Range of Values format allows constraints to be entered using simple $<$, $<=$, $>$, $>=$ or $=$ relations. A typical Simple Range of Values constraint would be $0 < \text{Value of A1} < 10$, where A1 is entered in the *Cell Range* box, 0 is entered in the *Min* box and 10 is entered in the *Max* box. The operator desired is selected from the drop down list boxes. With a Simple Range of Values format constraint, you can enter just a Min value, just a Max or both.

A formula constraint, on the other hand, allows you to enter any valid Excel formula as a constraint, such as $A19 < (1.2 * E7) + E8$. For each possible solution RISKOptimizer will check whether the entered formula evaluates to TRUE or FALSE to see if the constraint has been met. If you want to use a boolean formula in a worksheet cell as a constraint, simply reference that cell in the *Formula* field of the Constraint Settings dialog box.

To enter the constraints for the Airlines model you'll specify two new constraints. First, enter the *Simple Range of Values* format hard constraint for Profit > 0:

- 1) Enter "*Profit > 0*" in the description box.
- 2) In the *Range to Constrain* box, enter C27.
- 3) Select the > operator to the right of the *Range to Constrain*.
- 4) Clear the default value of 0 in the *Maximum* box
- 5) To the left of *Range to Constrain*, clear the operator by selecting a blank from the drop down list
- 6) Click "*Every Iteration of Each Simulation*" and click OK. This specifies that you must always insure that Profit is greater than 0, no matter how many reservations are taken.

RISKOptimizer - Constraint Settings

Description: Profit>0

Constraint Type:

- ☒ Hard (Discards Solutions that Do Not Meet the Constraint)
- ☐ Soft (Disfavors Solutions that Do Not Meet the Constraint)

Penalty Function: [Empty Box]

Definition:

Entry Style: Simple

Range to Constrain: [Empty] =C27 [Empty] > [Empty]

Statistic to Constrain: Value

Evaluation Time:

- ☒ Every Iteration of Each Simulation (Iteration Constraint)
- ☐ Only at the End of Each Simulation (Simulation Constraint)

OK Cancel

- 7) Click OK to enter this constraint.

Now, enter the simulation constraint:

- 1) Click Add to display the Constraint Settings dialog box again.
- 2) Enter "StdDev of Profit <400" in the description box.
- 3) In the Range to Constrain box, enter C27.
- 4) Select the < operator to the right of the Cell Range.
- 5) Enter 400 in the Max box.
- 6) To the left of Range to Constrain, clear the operator by selecting a blank from the drop down list
- 7) Click the Statistic to Constrain dropdown list and select "Standard Deviation".
- 8) Click OK.

Your Model dialog with the completed constraints section should look like this.

The screenshot shows the RISKOptimizer - Model dialog box. The Optimization Goal is set to Maximum, the Cell is =C27, and the Statistic is Mean. The Adjustable Cell Ranges section contains two entries: "Recipe: Max Reservations Accepted" with a minimum of 19, a range of =C14, and a maximum of 30 (Integer); and "Recipe: % Full Fare Seats" with a minimum of 0, a range of =C15, and a maximum of 1 (Any). The Constraints section contains two entries: "StdDev Profit <400" with the formula =RiskStdDev(\$C\$27) < 400 (Hard); and "Profit > 0" with the formula = \$C\$27 > 0 (Hard). The dialog box has buttons for Add..., Delete, Group, OK, and Cancel.

Minimum		Range		Maximum	Values
- Recipe: Max Reservations Accepted					
19	<=	=C14	<=	30	Integer
- Recipe: % Full Fare Seats					
0	<=	=C15	<=	1	Any

Description	Formula	Type
StdDev Profit <400	=RiskStdDev(\$C\$27) < 400	Hard
Profit > 0	= \$C\$27 > 0	Hard

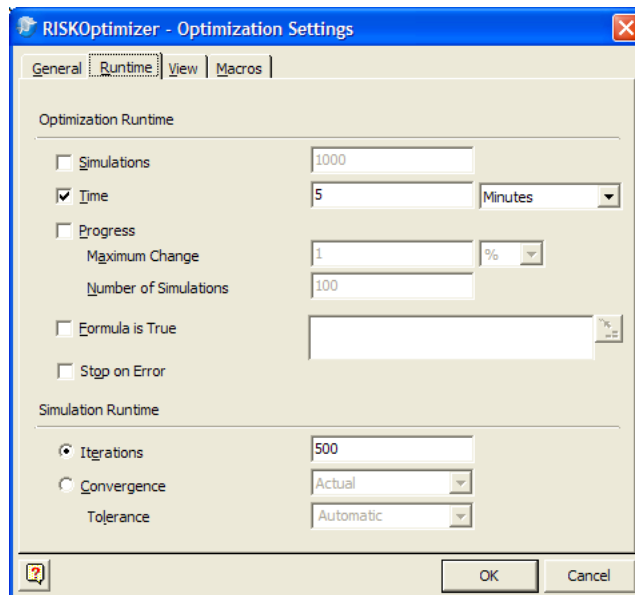
Other RISKOptimizer Options

Options such as **Update Display**, **Random Number Seed**, **Optimization Stopping Conditions** and **Simulation Stopping Conditions** are available to control how RISKOptimizer operates during an optimization. Let's specify some stopping conditions and update display settings.

Optimization Stopping Conditions

RISKOptimizer will run an optimization as long as you wish. The stopping conditions allow RISKOptimizer to automatically stop when either: a) *a certain number of scenarios or “trials” have been examined*, b) *a certain amount of time has elapsed*, c) *no improvement has been found in the last n scenarios*, d) *the entered Excel formula evaluates to TRUE*, or e) *an Error value is calculated for the target cell*. To view and edit the stopping conditions:

- 1) *Click the Optimization Settings icon on the RISKOptimizer toolbar.*
- 2) *Select the Runtime tab.*



In the Optimization Settings dialog you can select any combination of these optimization stopping conditions, or none at all. If you select more than one stopping condition, RISKOptimizer will stop when any one of the selected conditions are met. If you do not select any stopping conditions, RISKOptimizer will run forever, until you stop it manually by pressing the “stop” button in the RISKOptimizer toolbar.

Simulation Stopping Conditions

Simulations	Time	Progress	Formula is True
This option sets the number of simulations that you would like RISKOptimizer to run. RISKOptimizer runs a simulation for one complete set of variables or one possible solution to the problem.	RISKOptimizer will stop after the specified amount of time has elapsed. This number can be a fraction (4.25).	This stopping condition is the most popular because it keeps track of the improvement and allows RISKOptimizer to run until the rate of improvement has decreased. For example, RISKOptimizer could stop if 100 simulations have passed and we still haven't had any change in the best scenario found so far.	RISKOptimizer will stop if the entered Excel formula evaluates to TRUE in a simulation.

- 1) *Set Minutes = 5 to allow RISKOptimizer to run for five minutes.*

RISKOptimizer runs a full simulation of your model for each trial solution it generates. You can specify how long to run each of these simulations using Simulation Stopping Conditions. You can run each simulation for a fixed number of iterations, or, alternatively, let RISKOptimizer determine when to stop each simulation.

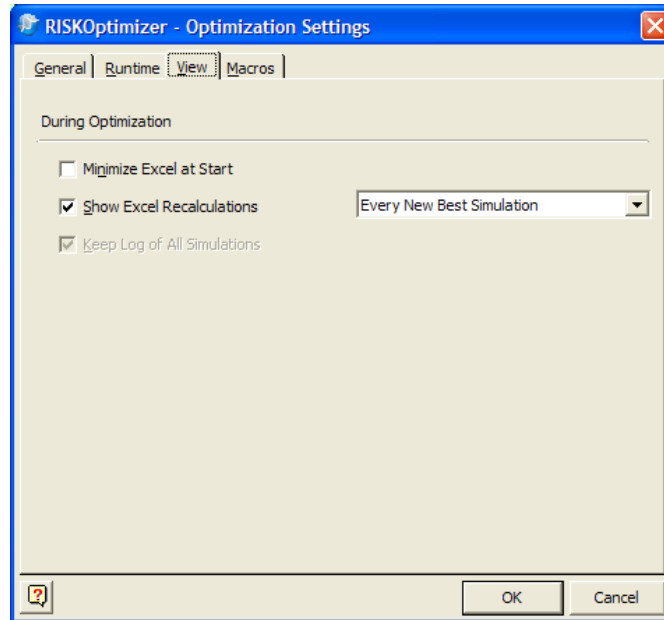
Iterations	Stop on Actual Convergence	Stop on Projected Convergence
This option allows you to run each simulation for a fixed number of iterations. In this case, RISKOptimizer will run the specified number of iterations for each simulation that is performed for each trial solution generated by RISKOptimizer (unless stopped prematurely when an iteration constraint is not met).	This option instructs RISKOptimizer to stop each simulation when the distributions generated for both 1) the target cell of the optimization and 2) cells referenced in simulation constraints are stable and the statistics of interest converge. The amount of variation allowed in a statistic when it is marked as "converged" is set by the <i>Tolerance</i> option.	This option instructs RISKOptimizer to stop each simulation when it can project internally that the distributions generated for both 1) the target cell of the optimization and 2) cells referenced in simulation constraints are stable. RISKOptimizer projects convergence based on the results of prior simulations that have been run during the optimization.

- 1) *Set Iterations = 500 to have RISKOptimizer run a quick simulation for each trial solution.*

Logging Simulation Data

RISKOptimizer can display an ongoing description of each simulation run during an optimization, including the value of the target statistic calculated, basic statistics on the simulated distribution of target cell values, the adjustable cell values used and whether constraints were met. To view this log during an optimization:

- 1) *Click the View tab and select "Keep Log of All Simulations" in the Optimization Settings dialog.*

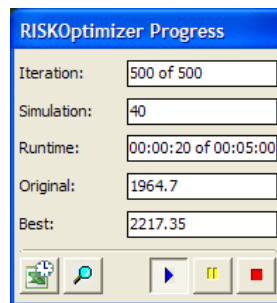


Running the Optimization

Now, all that remains is to optimize this model to determine the maximum number of reservations in each fare category which maximizes your profit. To do this:

- 1) *Click OK to exit the Optimization Settings dialog.*
- 2) *Click the Start Optimization icon*

As RISKOptimizer begins working on your problem, you will see the current best values for your adjustable cells - *total # of reservations accepted and % of reservations which are full fare* - in your spreadsheet. The best mean for *Profit* is shown in blue with an arrow pointing at the target cell.



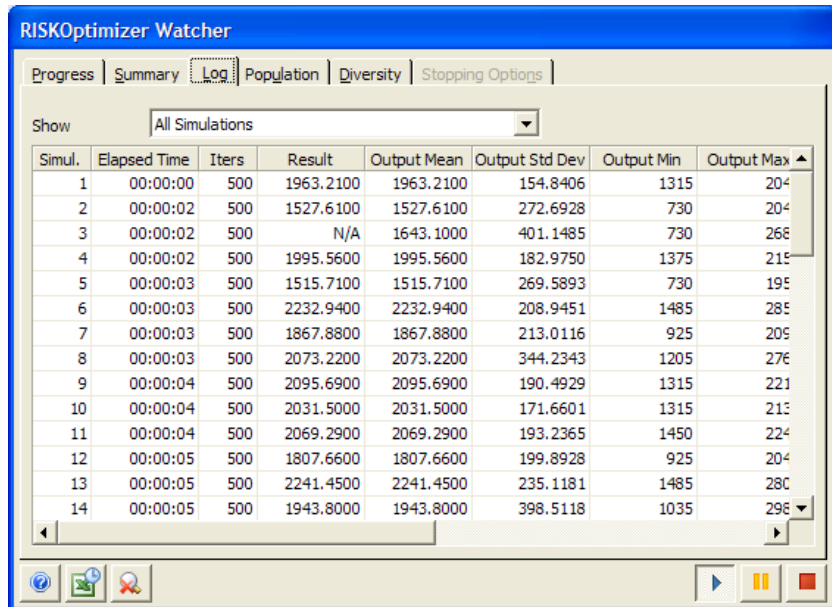
During the run, the Progress window displays: 1) the best solution found so far, 2) the original value for the selected simulation statistic for the target cell when the RISKOptimizer optimization began, 3) the number of simulations of your model that have been executed and number of those simulations which were valid; i.e., all constraints were met and 4) the time that has elapsed in the optimization.

Any time during the run you can click the **Display Excel Updating Options** icon to see a live updating of the screen each simulation.

The RISKOptimizer Watcher

RISKOptimizer can also display a running log of the simulations performed for each trial solution. This is displayed in the RISKOptimizer Watcher while RISKOptimizer is running. The RISKOptimizer Watcher allows you to explore and modify many aspects of your problem as it runs. To view a running log of the simulations performed:

- 1) *Click the Watcher (magnifying glass) icon in the Progress window to display the RISKOptimizer Watcher*
- 2) *Click the Log tab.*



Simul.	Elapsed Time	Iters	Result	Output Mean	Output Std Dev	Output Min	Output Max
1	00:00:00	500	1963.2100	1963.2100	154.8406	1315	204
2	00:00:02	500	1527.6100	1527.6100	272.6928	730	204
3	00:00:02	500	N/A	1643.1000	401.1485	730	268
4	00:00:02	500	1995.5600	1995.5600	182.9750	1375	215
5	00:00:03	500	1515.7100	1515.7100	269.5893	730	195
6	00:00:03	500	2232.9400	2232.9400	208.9451	1485	285
7	00:00:03	500	1867.8800	1867.8800	213.0116	925	205
8	00:00:03	500	2073.2200	2073.2200	344.2343	1205	276
9	00:00:04	500	2095.6900	2095.6900	190.4929	1315	221
10	00:00:04	500	2031.5000	2031.5000	171.6601	1315	213
11	00:00:04	500	2069.2900	2069.2900	193.2365	1450	224
12	00:00:05	500	1807.6600	1807.6600	199.8928	925	204
13	00:00:05	500	2241.4500	2241.4500	235.1181	1485	280
14	00:00:05	500	1943.8000	1943.8000	398.5118	1035	298

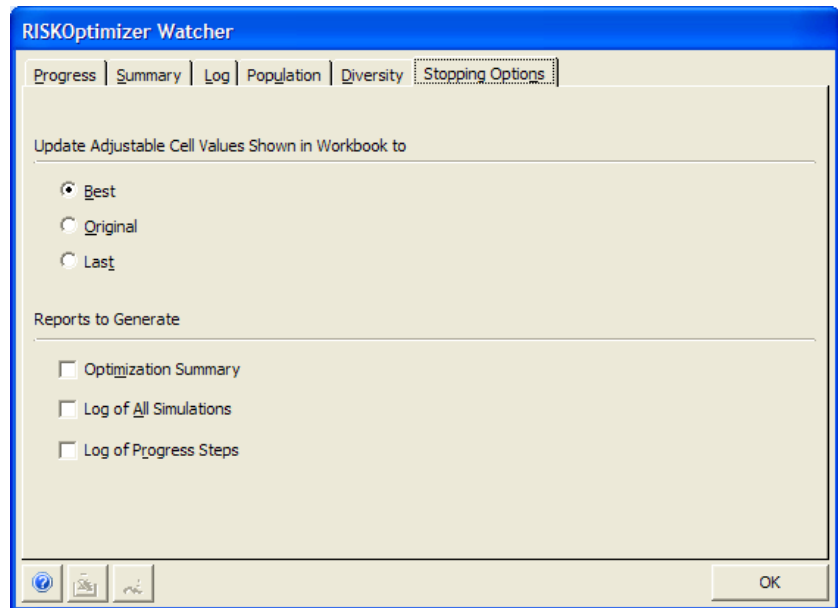
In this report the results of the simulation run for each trial solution is shown. The column for *Result* shows by simulation the value of the target cell's statistic that you are trying to maximize or minimize - in this case, the mean of Profit in \$C\$27. Columns for *Output Mean*, *Output StdDev*, *Output Min* and *Output Max* describe the probability distribution for the target cell Profit that was calculated by each simulation. The columns for \$C\$14 and \$C\$15 identify the values used for your adjustable cells. Columns for *StdDev Profit<400* and *Profit>0* show whether your constraints were met in each simulation.

Stopping the Optimization

After five minutes, RISKOptimizer will stop the optimization. You can also stop the optimization by:

- 1) *Clicking the Stop icon in the RISKOptimizer Watcher or Progress windows.*

When the RISKOptimizer process stops, RISKOptimizer displays the Stopping Options tab which offers the following choices:



These same options will automatically appear when any of the stopping conditions that were set in the RISKOptimizer Optimization Settings dialog are met.

Summary Report

RISKOptimizer can create an optimization summary report that contains information such as date and time of the run, the optimization settings used, the value calculated for the target cell and the value for each of the adjustable cells.

Book2 - Microsoft Excel						
<div> <div>Model Definition</div> <div>Settings</div> <div>Start</div> <div>Reports</div> <div>Utilities</div> <div>Help</div> <div>Tools</div> </div>						
A1	fx					
1	RISKOptimizer: Optimization Summary					
2	Performed By: Test					
3	Date: Monday, February 09, 2009 1:27:48 PM					
4	Model: Airfield.xls					
5						
6	Goal					
7	Cell to Optimize Airlines!\$C\$27					
8	Statistic to Optimize Mean					
9	Type of Goal Maximum					
10						
11	Results					
12	Valid Simulations 892					
13	Total Simulations 892					
14	Original Value \$1,964					
15	+ soft constraint penalties \$0					
16	= result \$1,964					
17	Best Value Found \$2,249					
18	+ soft constraint penalties \$0					
19	= result \$2,249					
20	Best Simulation Number 8					
21	Time to Find Best Value 0:00:04					
22	Reason Optimization Stopped Stop button pressed					
23	Time Optimization Started 2/9/2009 13:23					
24	Time Optimization Finished 2/9/2009 13:27					
25	Total Optimization Time 0:03:56					
26	Adjustable Cell Values Airlines!\$C\$14					
27	Original 19					
28	Best 28					
29	Adjustable Cell Values Airlines!\$C\$15					
30	Original 30.00%					
31	Best 50.15%					
32						
33	Constraints					
34	Description StdDev Profit<400					
35	Definition =RiskStdDev(Airlines!\$C\$27) < 400					
36	Constraint Type Hard					
37	Evaluation Time Simulation					
38	Satisfied for % of Simulations 100.00%					
39	Satisfied for % of Valid Simulations 100.00%					
40	Description Profit>0					
41	Definition =Airlines!\$C\$27 > 0					
42	Constraint Type Hard					
43	Evaluation Time Iteration					
44	Satisfied for % of Simulations 100.00%					
45	Satisfied for % of Valid Simulations 100.00%					

This report is useful for comparing the results of successive optimizations.

Placing the Results in Your Model

To place the new, optimized mix of production levels for the Airlines to each of the sixteen tasks in your worksheet:

- 1) Click on the "Stop" button.
- 2) Make sure the "Update Adjustable Cell Values Shown in Workbook to" option is set to "Best"

You will be returned to the AIRLINES.XLS spreadsheet, with all of the new variable values that created the best solution. **Remember, the best solution is a mean of simulation results for Profit and this is not the same as the value shown for a simple recalculation of Profit that uses the best variable values.** The best mean is shown in the blue box with the arrow that points at Profit.

Variable	Value	Description
Seats Available	19	
Ticket Price per Seat	\$195	
% No-Shows	20.00%	< Described by RiskNormal(0.2,0.03)
Demand for Full Fare Reservations	8	< Described by RiskTriang(3,1,15)
Ticket Price per Seat	\$85	
% No-Shows	10.00%	< Described by RiskNormal(0.1,0.01)
Demand for Discount Reservations	25	< Described by RiskTriang(12,20,35,10,50)
Maximum Reservations Accepted	28	< Adjustable between 19 and 30
Percentage Sold at Full Fare	50.15%	< Adjustable between 0% and 100%
Full Fare Reservations Accepted	8	
Discount Reservations Accepted	14	
Full Fare Passengers to Board	6	
Discount Passengers to Board	13	
Cost of Bumping	\$150	< Described by RiskDiscrete((150,200,250,300),(0.1,0.3,0.4,0.2))
Ticket Revenue	\$2,325	
Cost of Bumping Passengers	\$0	
Profit	\$2,325	< Maximize mean of distribution for Profit, but Profit must always be >0 and StdDev of Profit distribution must be <400

Mean (Best Simulation) = 2248.96

IMPORTANT NOTE: Although in our example you can see that RISKOptimizer found a solution which yielded a total profit of 2248.96, your result may be higher or lower than this. RISKOptimizer may have also found a different combination of Maximum Reservations Accepted and Percentage Sold at Full Fare that produced the same total score. These differences are due to an important distinction between RISKOptimizer and all other problem-solving algorithms: it is the random nature of RISKOptimizer's genetic algorithm engine that enables it to solve a wider variety of problems, and find better solutions.

When you save any sheet after RISKOptimizer has run on it (even if you “restore” the original values of your sheet after running RISKOptimizer), all of the RISKOptimizer settings in the RISKOptimizer dialogs will be saved along with that sheet. The next time that sheet is opened, all of the most recent RISKOptimizer settings load up automatically. All of the other example worksheets have the RISKOptimizer settings pre-filled out and ready to be optimized.

NOTE: If you want to take a look at the Airlines model with all optimization settings pre-filled out, open the example model AIRYIELD.XLS

Chapter 4: Example Applications

Introduction	61
Budget Allocation	63
Capacity Planning	65
Class Scheduler	67
Hedging with Futures	69
Job Shop Scheduling	71
Portfolio Balancing	73
Portfolio Mix	75
Portfolio Risk	77
Salesman Problem	79
Yield Management	81

Introduction

This chapter explains how RISKOptimizer can be used in a variety of applications. These example applications may not include all of the features you would want in your own models, and are most effective as idea generators and templates. All examples illustrate how RISKOptimizer finds solutions by relying on the relationships that already exist in your worksheet, so it is important that your worksheet model accurately portrays the problem you are trying to solve.

All Excel worksheet examples can be found within your RISKOPTIMIZER5 directory in a sub-directory called "EXAMPLES".

Each example comes with all RISKOptimizer settings pre-selected, including the target cell, adjustable cells, solving methods and constraints. You are encouraged to examine these dialog settings before optimizing. By studying the formulas and experimenting with different RISKOptimizer settings, you can get a better understanding of how RISKOptimizer is used. The models also let you replace the sample data with your own "user" data. If you decide to modify or adapt these example sheets, you may wish to save them with a new name to preserve the original examples for reference.

Budget Allocation

A senior executive wants to find the most effective way to distribute funds among the various departments of the company to maximize profit. Below is a model of a business and its projected profit for the next year. The model estimates next year's profit by examining the annual budget and making assumptions about, for example, how advertising affects sales. Uncertain sales estimates include probability distributions to reflect ranges of possible values. This is a simple model, but it illustrates how you can set up any model and use RISKOptimizer to feed inputs into it to find the best output.

Example file:	budget.xls
Goal:	Allocate the annual budget among five departments to maximize next year's profits.
Solving method:	budget
Similar problems:	Allocate any scarce resource (such as labor, money, gas, time) to entities that can use them in different ways or with different efficiencies.

Corporate Budget (\$ thousands)		Sales Estimate		Next Year	
Advertising	\$1,000	marginal cost	0.07749814	sales(units)	1634.931191
Marketing	\$220	can make	16774	production co	485.4931191
Production	\$1,600	ad saturation	200	price	3.3
Salary	\$1,500	ad multiplier	2.44948974	gross income	5395.272931
Operations	\$800	demand	1634.93119	total budget	\$5,120
total	\$5,120			Profit (\$ thousands)	\$275.27*
(must be kept constant)					

How the Model Works

The file "budget.xls" models the effects of a company's budget on its future sales and profit. Cells C4:C8 (the variables) contain the amounts to be spent on each of the five departments. These values total the amount in cell C10, the total annual budget for the company. This budget is set by the company and is unchangeable.

Cells F6:F10 compute an estimate of the demand for the company's product next year, based on the advertising and marketing budgets. The amount of actual sales is the minimum of the calculated demand and the supply. The supply is dependent upon the money allocated to the production and operations departments. The uncertain estimates in the model are included in probability distributions used in the sales estimate calculations in cells F6 to F10.

How to Solve It

Maximize the profit in cell I16 by using the “budget” solving method to adjust the values in cells C4:C8. Set the independent ranges for each of the adjustable cells for the budget for each department, to keep RISKOptimizer from trying negative numbers, or numbers which would not make suitable solutions (e.g., all advertising and no production) for the departmental budget.

The “budget” solving method works like the “recipe” solving method, in that it is trying to find the right “mix” of the chosen variables. When you use the budget method, however, you add the constraint that all variables must sum up to the same number as they did before RISKOptimizer started optimizing.

Capacity Planning

This model uses RISKOptimizer to select the capacity level for a new plant in order to maximize profits. In the model ZooCo is thinking of marketing a new drug used to make hippos healthier. A standard simulation model is used to generate the distribution of NPV for the production of the new drug. However, it is necessary to decide on what capacity of plant to build. What capacity level maximizes risk adjusted NPV?

Example file:	capacity.xls
Goal:	Maximize the mean of the simulated distribution for NPV by changing plant capacity.
Solving method:	recipe
Similar problems:	Business analyses combining traditional simulation models with user-controlled decision variables.

Capacity Planning
 This model uses RISKOptimizer to select the capacity level for a new plant in order to maximize profits. In the model ZooCo is thinking of marketing a new drug used to make hippos healthier. A standard simulation model is used to generate the distribution of NPV for the production of the new drug. However, it is necessary to decide on what capacity of plant to build. What capacity level maximizes risk adjusted NPV?

At the beginning of the current year there are 1,000,000 hippos that may use the product as shown in cell B34. Each hippo will use the drug (or a competitor's drug) at most once a year. The number of hippos is forecasted to grow by an average of 5% per year, and we are 95% sure that the number of hippos will grow each year by between 3% and 7% (modeled using probability distributions in cells C34 to F34). We are not sure what use of the drug will be during year 1, but our worst case guess is 20% use, most likely use is 40% and best case use is 70% (modeled using probability distribution in cell B35). In later years, we feel the fraction of hippos using our drug (or a competitor's) will remain the same, but in the year after a competitor enters, we lose 20% of our share for each competitor who enters. It costs \$3.50 to build one unit of annual capacity and \$0.30 per year to operate one unit of capacity (whether or not we use the capacity to produce the drug). Any capacity level between 100,000 and 500,000 units can be built.

Use the "recipe" solving method to adjust cell I26. Maximize the simulated mean of B45.

This example was adapted from *Financial Models Using Simulation and Optimization* by Wayne Winston, published by Palisade Corporation.

ZooCo, Inc.					
27 Price	\$ 2.20	Compet %age	0.2	Capacity	100000
28 Unit Var Cost	\$ 0.40	Year 1 Market Size	1000000	Unit op cost	0.3
29 Interest Rate	0.1	Year 1 worst share	0.2	Unit Building cos	3.5
30 Entrant Prob	0.4	Year 1 most likely	0.4		
31		Year 1 best share	0.7		
32					
33 Year	1	2	3	4	5
34 Market Size	1000000	1050000	1102500	1157625	1215506.25
35 hippo	0.433333333	0.346666667	0.277333333	0.277333333	0.277333333
36 Competitors(beg					
37 inning of year)	0	1	2	2	2
38 Entrants	1	1	0	0	0
39 Unit Sales	100000	100000	100000	100000	100000
40 Revenues	\$ 220,000	\$ 220,000	\$ 220,000	\$ 220,000	\$ 220,000
41 Costs	\$ 40,000	\$ 40,000	\$ 40,000	\$ 40,000	\$ 40,000
42 Building cost	\$ 350,000				
43 Fixed op cost	\$ 30,000	\$ 30,000	\$ 30,000	\$ 30,000	\$ 30,000
44 Profits	\$ (200,000)	\$ 150,000	\$ 150,000	\$ 150,000	\$ 150,000
45 NPV	\$250,436				
46 Utility	0.45445356				

How the Model Works

At the beginning of the current year there are 1,000,000 hippos that may use the product as shown in cell B34. Each hippo will use the drug (or a competitor's drug) at most once a year. The number of hippos is forecasted to grow by an average of 5% per year, and we are 95% sure that the number of hippos will grow each year by between 3% and 7% (modeled using probability distributions in cells B34 to F34). We are not sure what use of the drug will be during year 1, but our worst case guess is 20% use, most likely use is 40% and best case use is 70% (modeled using probability distribution in cell B35). In later years, we feel the fraction of hippos using our drug (or a competitor's) will remain the same, but in the year after a competitor enters, we lose 20% of our share for each competitor who enters. It costs \$3.50 to build one unit of annual capacity and \$0.30 per year to operate one unit of capacity (whether or not we use the capacity to produce the drug). Any capacity level between 100,000 and 500,000 units can be built.

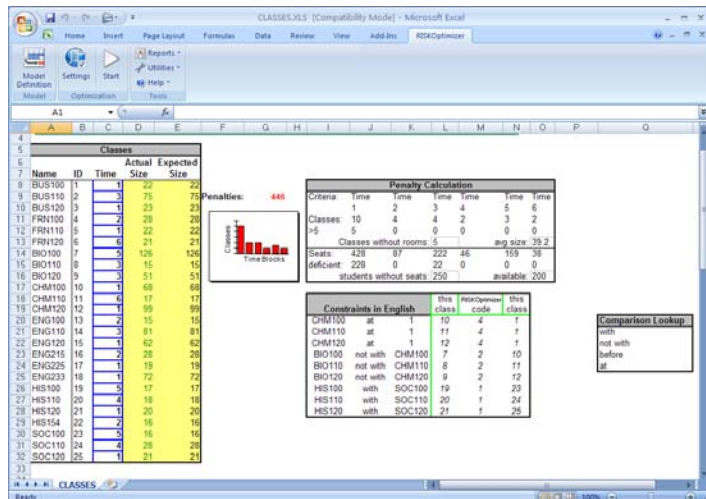
How to Solve It

Use the recipe solving method for cell I26. Maximize the simulated mean of B45.

Class Scheduler

A university must assign 25 different classes to 6 pre-defined time blocks. Since the schedule must be developed prior to student registration, the actual number of students per class is uncertain. Each class lasts exactly one time block. Normally, this would allow us to treat the problem with the “grouping” solving method. However, there are a number of constraints that must be met while the classes are being scheduled. For example, biology and chemistry should not occur at the same time so that pre-medical students can take both classes in the same semester. To meet such constraints, we use the “schedule” solving method instead. The “schedule” solving method is like the “grouping” method, only with the constraint that certain tasks must (or must not) occur before (or after or during) other tasks.

Example file:	classes.xls
Goal:	Assign 25 classes to 6 time periods to minimize the mean of the simulated distribution for the number of students who get squeezed out of their classes. Meet a number of constraints regarding which classes can meet when.
Solving method:	schedule
Similar problems:	Any scheduling problem where all tasks are the same length and can be assigned to any of a number of discrete time blocks. Also, any grouping problem where constraints exist as to which groups certain items can be assigned.



How the Model Works

The “classes.xls” file contains a model of a typical scheduling problem where many constraints must be met. The range of possible values for each class is given by the probability distributions entered in the range D8:D32 labeled “Actual Size”. Cells C8:C32 assign the 25 classes to the 6 time blocks. There are only five classrooms available, so assigning more than five classes to one time block means that at least one of the classes cannot meet.

Cells L20:N28 contain the constraints; to the left of the constraints are English descriptions of the constraints. You can use either the number code or the English description as the constraint. The list of constraint codes for scheduling problems can be found in more detail in the “Solving Methods” section of [Chapter 5: RISKOptimizer Reference](#).

Each possible schedule is evaluated by calculating both a) the number of classes which cannot meet, and b) the number of students who cannot sit at their classes because the classrooms are full. This last constraint keeps RISKOptimizer from scheduling all the large classes at the same time. If only one or two large classes meet during a time block, the larger classrooms can be used for them.

Cells J11:M11 uses the DCOUNT Excel function to count up how many classes are assigned to each time block. The section right below cells J12:M12 computes how many classes did not get assigned a room for that time block. All the classes that are without rooms are totaled in cell L13.

If the number of seats required by a given class exceeds the number of seats available, cells J15:M15 calculate by how much, and the total number of students without seats is calculated in cell L16. In cell G9, this total number of students without seats is added to the average class size, and multiplied by the number of classes without rooms. This way, we have one cell which combines all penalties such that a lower number in this cell always indicates a better schedule.

How to Solve It

Minimize the mean of the simulated distribution for the penalties in G9 by changing cells C8:C32. Use the “schedule” solving method. When this solving method is chosen, you will see a number of related options appear in the lower “options” section of the dialog box. Set the number of time blocks to 6, and set the constraints cells to L20:N28.

Hedging with Futures

It is June 8, 2000. GlassCo needs to purchase 500,000 gallons of heating oil on November 8, 2000. The current spot price of oil is \$0.42 per gallon. Oil prices are assumed to follow a Lognormal random variable with Mean = .08 and StdDev = .30. The risk free rate is 6%. We are hedging the price risk inherent in our future oil purchase by buying oil futures that expire on December 8, 2000. How many futures should we buy?

Example file:	oil.xls
Goal:	Find the number of future contracts to buy to protect against price changes of a future purchase.
Solving method:	recipe
Similar problems:	Risk minimization models where the objective is to minimize standard deviation of the target

Hedging with Futures

It is June 8. GlassCo needs to purchase 500,000 gallons of heating oil on November 8. The current spot price of oil is \$0.42 per gallon. Oil prices are assumed to follow a Lognormal random variable with Mean = .08 and StdDev = .30. The risk free rate is 6%. We are hedging the price risk inherent in our future oil purchase by buying oil futures that expire on December 8. How many futures should we buy?

The model tries to ensure that the cost of purchasing 500,000 gallons of heating oil five months in the future is as predictable as possible by using futures contracts to protect against price swings. The uncertain factors in the model are the future spot price of oil (cell B13) and the future Oil Futures price (cell B15). The first thing we need to do is choose an adjustable cell. For this model we want to adjust cell B12 - the number of future contracts "long" or purchased - to minimize the standard deviation of total cost in cell B20. The minimum number of contracts that can be purchased is 0 and the maximum is 600,000.

This example was adapted from *Financial Models Using Simulation and Optimization* by Wayne Winston, published by Palisade Corporation.

Future Expires December 8	
June 8 oil price per gallon	\$0.42000
r	0.06
Volatility	0.3
Oil drift	0.08
Sigma of percentage variation of future from mean	0.05
Future duration	0.5
December futures price on June 8	\$0.43769
Gallons bought	500000
Number long	100000
November 8 spot oil price	\$0.42617
Mean of November 8 price of	
December future	\$0.42631
Actual November 8 futures price	\$0.42631
Bottom line	
Cost of buying oil	\$213,084.94
Revenue from futures	\$ 42,830.61
Cost of buying futures on June 8	\$ 43,769.00
Total cost	\$214,023.33

How the Model Works

The model tries to insure that the cost of purchasing 210,000 gallons of heating oil five months in the future is as predictable as possible by using futures contracts to protect against price swings. The uncertain factors in the model are the future spot price of oil (cell B13) and the future Oil Futures price (cell B15).

How to Solve It

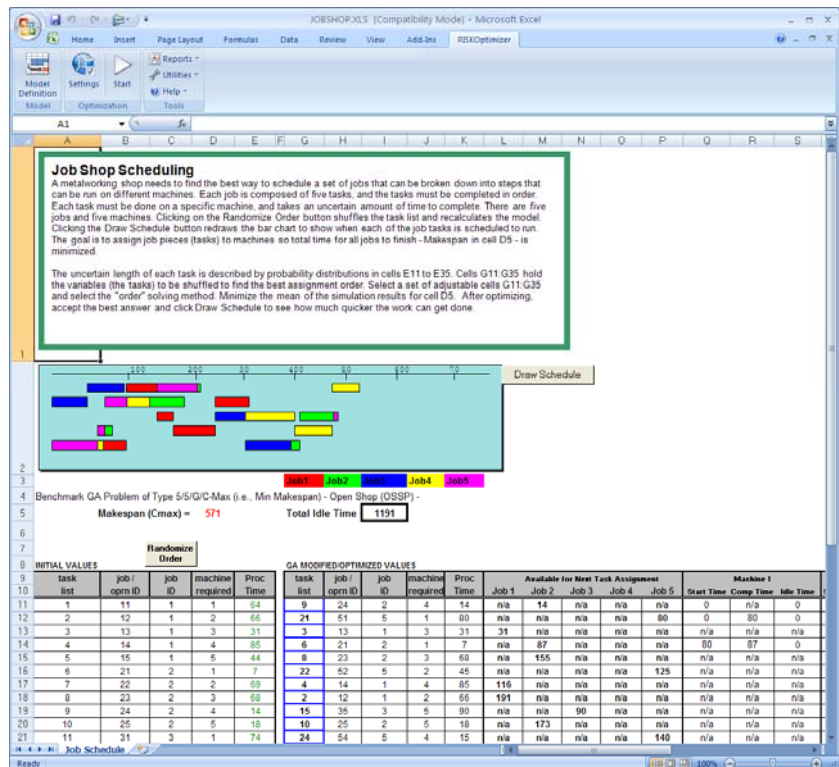
The first thing we need to do is choose an adjustable cell. For this model we want to adjust cell B12 - the # of *future contracts "long" or purchased* - to minimize the standard deviation of total cost in cell B23. The minimum number of contracts that can be purchased is 0 and the maximum is 600,000.

Job Shop Scheduling

A metalworking shop needs to find the best way to schedule a set of jobs that can be broken down into steps that can be run on different machines. Each job is composed of five tasks, and the tasks must be completed in order. Each task must be done on a specific machine, and takes an uncertain amount of time to complete. There are five jobs and five machines.

Clicking the Draw Schedule button at the top of the sheet will redraw the bar chart to show when each of the job tasks is scheduled to run.

Example file:	jobshop.xls
Goal:	Assign job pieces (tasks) to machines so total time for all jobs to finish is minimized.
Solving method:	order
Similar problems:	Scheduling or project-management problems



How the Model Works

The uncertain length of each task is described by probability distributions in cells E11 to E35. Cell D5 computes the makespan, or how much time elapses between the start of the first scheduled task and the end of the last scheduled task. This total time is what we wish to minimize. Cells G11:G35 hold the variables (the tasks) to be shuffled to find the best assignment order. The equations on the sheet figure out how soon each task can run on the machine that it needs.

How to Solve It

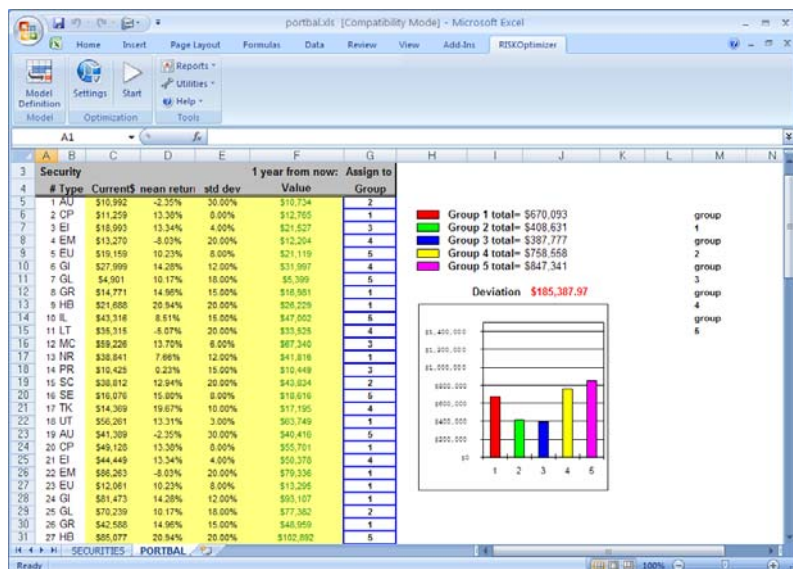
Select a set of adjustable cells G11:G35 and select the order solving method. Minimize the mean of the simulation results for cell D5.

Portfolio Balancing

A broker has a list of 80 securities of different types that will be worth a different and uncertain amount of money in the future. The broker wants to group these securities into five packages (portfolios) that will be as close to each other in total value as possible one year from now.

This is an example of a general class of problems called bin packing problems. Packing the holds of a cargo ship so that each hold weighs as much as the others is another example. If there are millions of small items to be packaged into a few groups, such as grains of wheat into ship holds, a roughly equal distribution can be estimated without a big difference in weight. However, several dozen packages of different weights and/or sizes can be packed in very different ways, and efficient packing can improve the balance that would be found manually.

Example file:	portbal.xls
Goal:	Break a list of securities up into five different portfolios whose future values are as close as possible to each other.
Solving method:	grouping
Similar problems:	Create teams that have roughly equivalent collective skills. Pack containers into holds of a ship so that the weight is evenly distributed.



How the Model Works

The “portbal.xls” file models a typical grouping assignment. Column A contains identification numbers to specific securities, and column B identifies the class of each security (the worksheet SECURITIES gives information on each class of security). Columns C, D and E give the current dollar value of each security and the mean and standard deviation of the return for the next year for the security (as determined by the security's class). Column F calculates the value of the security one year from now using a rate of return sampled from a probability distribution that uses the shown mean and standard deviation. Column G assigns each security to one of the five portfolios. When setting a grouping or bin packing type of problem and using the grouping solving method, you must be sure that before you start RISKOptimizer each group (1-5) is represented in the current scenario at least once.

Cells J6:J10 use “DSUM()” formulas to calculate the total value of each of the five portfolios. Thus, cell J6, for example, calculates DSUM of all the values in column F that have been assigned to group 5 (in column G).

Cell J12 computes the standard deviation among the total portfolio values using the “STDEV()” function. This provides a measure of how close in total value to each other the portfolios are. The graph shows the total value of each portfolio, with a reference line drawn at the goal number where each portfolio would be if they were all even.

How to Solve It

Minimize the mean of the simulation results for cell J12 by adjusting the cells in G5:G84. Use the “grouping” method and make sure the values 1, 2, 3, 4, and 5 each appear at least once in column G.

The “grouping” solving method tells RISKOptimizer to arrange variables into x groups, where x is the number of different values in the adjustable cells at the start of an optimization.

Portfolio Mix

A young couple has assets in many different types of investments, each with its own yield, potential growth, and risk. Their goal is to pick the combination of investments that maximizes total return while keeping risk to an acceptable level.

Example file:	portmix.xls
The Goal:	Find the optimal mix of investments to maximize your profit, given your current risk/return needs.
Solving method:	budget

Asset Category	Portfolio Weight	Potential Capital Growth	Current Yield	Total Return
Money Market	6.00%	0.0%	6.0%	6.0%
Domestic Taxable Bond	2.60%	0.0%	9.0%	9.0%
Balanced	2.60%	4.0%	6.0%	10.0%
Growth & Income	12.00%	8.0%	4.0%	10.0%
Growth	24.92%	9.0%	2.0%	11.0%
Aggressive Growth	20.79%	11.0%	1.0%	12.0%
International Stock	25.00%	11.0%	1.0%	12.0%
Gold	6.00%	4.5%	2.5%	7.0%
Portfolio Total	100.00%			

Potential Capital Growth	8.38%
Current Yield	2.34%
Total Return	10.72%

How the Model Works

This is a classic financial model which attempts to balance the risk of loss against the return on investment. Each asset listed in column B has an uncertain capital growth % and a fixed yield. Total return sums the capital growth and yield. The objective is to maximize total return while keeping the standard deviation of portfolio return less than 9%.

How to Solve It

The total return in cell D33 reflects the sum of total capital growth and total yield. We maximize the mean of the simulated distribution for this cell. A hard simulation constraint is entered that specifies that the standard deviation of cell D33 must be less than .09.

Portfolio Risk

An investor wants to determine the safest way to structure a portfolio from several investments. Historical data has shown that the returns on the investments are correlated. The objective is to divide the total portfolio among the three available investments in order to achieve the desired 12% return while minimizing the risk, or standard deviation, of the portfolio return.

Example file:	corrmat.xls
Goal:	Minimize the standard deviation of portfolio return while still achieving the desired return.
Solving method:	budget
Similar problems:	any risk minimization model.

The screenshot shows the RISKOptimizer interface within Microsoft Excel. A text box titled "Portfolio Risk" provides the problem description. Below it, a table lists the characteristics of three investments. To the right, covariance and correlation matrices are displayed, along with the formula for covariance.

Portfolio Risk
 An investor wants to determine the safest way to structure a portfolio from several investments. Historical data has shown that the returns on the investments are correlated. The objective is to divide the total portfolio among the three available investments in order to achieve the desired 12% return while minimizing the risk, or standard deviation, of the portfolio return.

Each of three available investments has an uncertain return that is modeled using the probability distributions in cells E3 to E5. To correlate the returns of the three investments the RiskCorrmat function is used with the correlation matrix located in J9:L11. RISKOptimizer will adjust the percentage of the portfolio allocated to each investment. The "budget" solving method is used to insure that the total % allocated always sums to 100%.

The objective is to minimize the standard deviation of total portfolio return, while meeting the constraint that total return is greater than or equal to 12%. Minimize the standard deviation of simulation results for cell G6. Enter a hard simulation constraint that the mean of the simulation results for cell G6 needs to be greater than .12.

This example was adapted from *Spreadsheet Modeling and Decision Analysis* by Cliff Ragsdale.

	Return	Variance	Std.Dev	Risk	% Invested	Earnings
Investment A	14.00%	2.50%	15.81%	14.00%	50.00%	7.00%
Investment B	9.00%	1.50%	12.25%	9.00%	25.00%	2.25%
Investment C	8.00%	1.00%	10.00%	8.00%	25.00%	2.00%
Total					100.00%	11.25%

Covariance Matrix

	A	B	C
A	1	0.00028	-0.006
B		1	0.00125
C			1

Correlation Matrix

	A	B	C
A	1	0.014459	-0.37947
B		1	0.102062
C			1

$$r_{x,y} = \frac{Cov(x,y)}{s_x s_y}$$

How the Model Works

Each of three available investments has an uncertain return that is modeled using the probability distributions in cells E3 to E5. To correlate the returns of the three investments the RiskCorrmat function is used with the correlation matrix located in J9:L11. RISKOptimizer will adjust the percentage of the portfolio allocated to each investment. The "budget" solving method is used to insure that the total % allocated always sums to 100%.

The objective is to minimize the standard deviation of total portfolio return, while meeting the constraint that total return is greater than or equal to 12%.

How to Solve It

Minimize the standard deviation of simulation results for cell G6.
Enter a hard simulation constraint that the mean of the simulation results for cell G6 needs to be greater than .12.

Salesman Problem

A salesman is required to visit every city in the assigned territory once. What is the route with the shortest travel time possible that visits every city? This is a classic optimization problem with one twist - the travel time between each city is uncertain - and one that is extremely difficult for conventional techniques to solve if there are a large (>50) number of cities involved.

A similar problem might be finding the best order to perform tasks in a factory. For example, it might be much easier to apply black paint after applying white paint than the other way around. In RISKOptimizer, these types of problems can be best solved by the *order* solving method.

Example file:	salesman.xls
Goal:	Find the route with the shortest travel time among n cities that visits each city once.
Solving method:	order
Similar problems:	Plan the drilling of circuit board holes in the fastest way.

FileHomeInsertFormulasDataReviewViewAdd-InsRISKOptimizer

Model SettingsStart

Model OptimizationTools

ReportUtilitiesHelp

A1

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

City ID #

City Name

Total Travel Time

805.44

in hours

1

Anchorage AK

2

Calgary AB

3

Chicago IL

4

Dawson Creek BC

5

Edmonton AB

6

Halifax NS

7

Montreal PQ

8

New York NY

9

Ottawa ON

10

Prince Rupert BC

11

Quebec PQ

12

Regina SK

13

Saint John NB

14

San Francisco CA

15

Thunder Bay ON

16

Toronto ON

17

Vancouver BC

18

Whitehorse YT

19

Winnipeg MB

20

Winnipeg MB

Travel Time in hours

AK

AB

IL

BC

AB

NS

PQ

NY

ON

BC

PQ

SK

NB

CA

ON

ON

BC

1

Anchorage AK

2

Calgary AB

3

Chicago IL

4

Dawson Creek BC

5

Edmonton AB

6

Halifax NS

7

Montreal PQ

8

New York NY

9

Ottawa ON

10

Prince Rupert BC

11

Quebec PQ

12

Regina SK

13

Saint John NB

14

San Francisco CA

15

Thunder Bay ON

16

Toronto ON

17

Vancouver BC

18

Whitehorse YT

19

Winnipeg MB

20

Winnipeg MB

FileHomeInsertFormulasDataReviewViewAdd-InsRISKOptimizer

Model SettingsStart

Model OptimizationTools

ReportUtilitiesHelp

How the Model Works

The file “salesman.xls” calculates the travel time of a trip to various cities by looking up the travel times between cities in a table. The travel times between any two cities is described by a probability distribution (there are 200 probability distributions in the table). Column A contains identifying numbers for specific cities. Column B contains the names that those numbers represent (with a lookup function). The order in which the cities (and their numbers) appear from top to bottom represents the order in which the cities are visited. For example, if you entered a “9” into cell A3, then Ottawa would be the first city visited. If A4 contained “6” (Halifax), then Halifax would be the second city visited.

The travel times between cities are represented by probability distributions in the table beginning at C25. These distributions reference the table beginning at C48 that contains the actual driving distance between the cities. The distances in the table are symmetric (distance from A to B is the same as from B to A). However, more realistic models may include non-symmetric distances to represent greater difficulty of traveling in one direction (because of tolls, available transportation, headwinds, slope, etc.).

A function now must be used to calculate the length of the route between these cities. The total route length will be stored in cell G2, the cell we wish to optimize. To do this, we use the “RouteLength” function. This is a custom VBA function in Salesman.xls.

How to Solve It

Minimize the value in cell G2 by adjusting the cells in A3:A22. Use the “order” method and make sure the values 1 through 20 exist in the adjustable cells (A3:A22) before you start optimizing.

The “order” solving method tells RISKOptimizer to rearrange the chosen variables, trying different permutations of existing variables.

Yield Management

This is a yield management model which identifies the optimal number of full and discount fare seats to sell on a given flight. It also identifies the optimal number of reservations to accept in excess of the number of available seats – the classic “overbooking” problem.

Example file:	airyield.xls
Goal:	Identify the maximum number of reservations to accept in different fare categories to maximize profit.
Solving method:	recipe
Similar problems:	Any yield management problem where a variety of different prices are offered for the same product

Piedmont Commuter Airlines Flight 343			
Seats Available		19	
Full Fare (fully refundable)	Ticket Price per Seat	\$195	
	% No-Shows	20.00%	< Described by RiskNormal(0.2,0.03)
	Demand for Full Fare Reservations	8	< Described by RiskTriang(3,7,15)
Discount (\$50 change fee)	Ticket Price per Seat	\$85	
	% No-Shows	10.00%	< Described by RiskNormal(0.1,0.01)
	Demand for Discount Reservations	25	< Described by RiskTriang(12,20,35,10,90)
Maximum Reservations Accepted		19	< Adjustable between 19 and 30
Percentage Sold at Full Fare		30.00%	< Adjustable between 0% and 100%
Full Fare Reservations Accepted		6	
Discount Reservations Accepted		13	
Full Fare Passengers to Board		5	
Discount Passengers to Board		12	
Cost of Bumping		\$150	< Described by RiskDiscrete((150,200,250,300),(0.1,0.3,0.4,0.2))
Ticket Revenue		\$2,045	
Cost of Bumping Passengers		\$0	
Profit		\$2,045	< Maximize mean of distribution for Profit, but Profit must always be >0 and StdDev of Profit distribution must be <400

How the Model Works

The "airyield.xls" file is a very simple model which illustrates RISKOptimizer's use for yield management. Probability distributions are assigned to a variety of uncertain factors in the model, including the **Demand for Full Fare Reservations (in cell C8)**, the **% No Shows - Full Fare Reservations (in cell C7)**, the **% No Shows - Discount Fare Reservations (in cell C11)**, the **Demand for Discount Fare Reservations (in cell C12)**, and the **Cost of Bumping (in cell C23)**. The gross profit from the flight is calculated by calculating the total revenue from reservations in each fare category, less the cost of bumping passengers from an overbooked flight.

How to Solve It

In this model, the variables to be adjusted are located in cells C14 and C15. These cells contain the values for the maximum number of reservations accepted and the percentage of those reservations that will be allocated to full fare seats. "Profit must always be >0" is an iteration constraint, while "Standard deviation of the simulation results for profit must be <400" is a simulation constraint. The objective is to maximize the mean of the simulated distribution for profit while minimizing risk as specified by the entered constraints.

RISKOptimizer - Model

Optimization Goal: **Maximum**

Cell: **=C27**

Statistic: **Mean**

Adjustable Cell Ranges

Minimum		Range		Maximum	Values
Recipe: Max Reservations Accepted					
19	<=	=C14	<=	30	Integer
Recipe: % Full Fare Seats					
0	<=	=C15	<=	1	Any

Constraints

Description	Formula	Type
StdDev Profit < 400	=RiskStdDev(\$C\$27) < 400	Hard
Profit > 0	=\$C\$27 > 0	Hard

Buttons: Add..., Delete, Group, OK, Cancel

Chapter 5:

RISKOptimizer

Reference Guide

Model Definition Command	85
Adjustable Cell Ranges.....	89
Adjustable Cell Groups	91
Recipe Solving Method	93
Order Solving Method.....	94
Grouping Solving Method.....	94
Budget Solving Method	96
Project Solving Method.....	97
Schedule Solving Method.....	98
Crossover and Mutation Rate	100
Number of Time Blocks and Constraint Cells	102
Preceding Tasks	102
Operators.....	103
Constraints	106
Add - Adding Constraints.....	106
Simulation Constraints.....	108
Simple and Formula Constraints.....	109
Soft Constraints	109
Optimization Settings Command – General Tab	113
Optimization Settings Command – Runtime Tab	117
Optimization Runtime Options.....	118
Simulation Runtime Options	119
Optimization Settings Command – View Tab	121
Optimization Settings Command – Macros Tab.....	123
Start Optimization Command	125
Utilities Commands	127
Application Settings Command	127

Constraint Solver Command.....	128
RISKOptimizer Watcher	131
RISKOptimizer Watcher - Progress Tab	132
RISKOptimizer Watcher - Summary Tab	134
RISKOptimizer Watcher - Log Tab	135
RISKOptimizer Watcher - Population Tab.....	136
RISKOptimizer Watcher - Diversity Tab.....	137
RISKOptimizer Watcher - Stopping Options Tab	138

Model Definition Command

Defines the goal, adjustable cells and constraints for a model

Selecting the RISKOptimizer Model Definition command (or clicking the Model icon on the RISKOptimizer toolbar) displays the Model Dialog.

RISKOptimizer - Model

Optimization Goal:

Cell:

Statistic:

Adjustable Cell Ranges

Minimum	Range	Maximum	Values
---------	-------	---------	--------

Add...
Delete
Group

Constraints

Description	Formula	Type
-------------	---------	------

Add...
Edit...
Delete

OK Cancel

The RISKOptimizer Model Dialog.

The RISKOptimizer Model Dialog is used to specify or describe an optimization problem to RISKOptimizer. This dialog starts empty with each new Excel workbook, but saves its information with each workbook. That means that when the sheet is opened again, it will be filled out the same way. Each component of the dialog is described in this section.

Options in the Model dialog include:

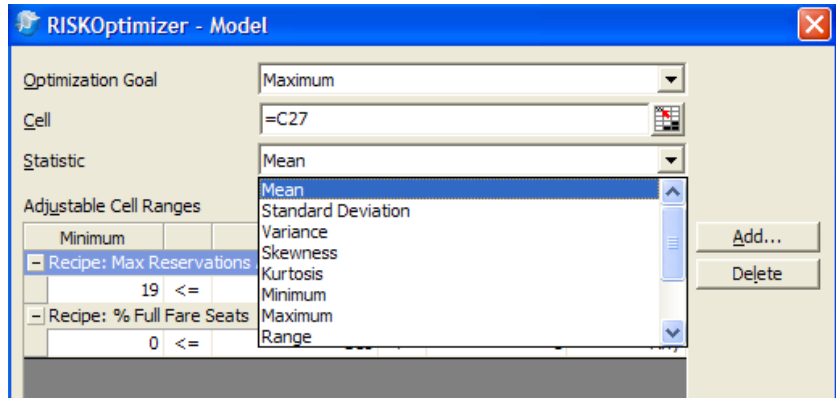
- **Optimization Goal.** The *Optimization Goal* option determines what kind of answer RISKOptimizer is to search for. If *Minimum* is selected, RISKOptimizer will look for variable values that produce the smallest possible value for the selected statistic of the simulation results for the target cell (all the way down to -1e300). If *Maximum* is selected, RISKOptimizer will search for the variable values that result in the largest possible value for the selected statistic (up to +1e300).

If *Target Value* is selected, RISKOptimizer will search for variable values that produce a value for the selected statistic as close as possible to the value you specify. When RISKOptimizer finds a solution which produces this result, it will automatically stop. For example, if you specify that RISKOptimizer should find the mean of the distribution of simulation results that is closest to 14, RISKOptimizer might find scenarios that result in a mean such as 13.7 or 14.5. Note that 13.7 is closer to 14 than 14.5; RISKOptimizer does not care whether the statistic's value is greater or less than the value you specify, it only looks at how close the value is.

- **Cell.** The cell or *target cell* contains the output of your model. A distribution of possible values for this target cell will be generated (via simulation) for each "trial solution" that RISKOptimizer generates (i.e., each combination of possible adjustable cell values). The target cell should contain a formula which depends (either directly or through a series of calculations) on the adjustable cells. This formula can be made with standard Excel formulas such as SUM() or user-defined VBA macro functions. By using VBA macro functions you can have RISKOptimizer evaluate models that are very complex.

As RISKOptimizer searches for a solution it uses the statistic for the simulation results of the target cell as a rating or "fitness function" to evaluate how good each possible scenario is, and to determine which variable values should continue cross-breeding, and which should die. In biological evolution, death is the "fitness function" that determines what genes continue to flourish throughout the population. When you build your model, your target cell must reflect the fitness or "goodness" of any given scenario, so as RISKOptimizer calculates the possibilities, it can accurately measure its progress.

- **Statistic.** The statistic entry is where you specify the statistic of the simulation results for your *target cell* that you wish to minimize, maximize, or set to a specific value. The actual statistic you wish to minimize, maximize, or set to a specific value is selected from the dropdown list.



To select the statistic for the target cell which you wish to minimize, maximize, or set to a specific value, simply select the desired statistic from the displayed dropdown list. If you wish to select a Percentile or Target for the target cell's distribution, simply:

- 1) **Select Percentile (*X* for given *P*) or Target (*P* for given *X*).**
- 2) **For Percentile (*X* for given *P*), enter the desired "*P*" value between 0 and 100 in the % field.** The value that will be minimized or maximized will be the value associated with the entered percentile; i.e., Percentile (99%) will cause RISKOptimizer to identify the combination of adjustable cell values that minimizes or maximizes the 99th percentile of the distribution of simulation results for the target cell.
- 3) **For Target (*P* for given *X*), enter the desired "*X*" value.** The value that will be minimized or maximized will be the cumulative probability associated with the entered value; i.e., Target (1000) will cause RISKOptimizer to identify the combination of adjustable cell values that minimizes or maximizes the cumulative probability of the value 1000 (as calculated using the distribution of simulation results for the target cell).

Users may choose to collect statistics inside their models by using @RISK/RISKOptimizer statistic functions like **RiskMean**. To optimize the value of such a cell, the statistic to optimize does not need to be specified, since the cell itself contains that information. In this case, select the **Value** option from the **Statistic** dropdown list, telling RISKOptimizer to optimize the value of a given cell at the end of a simulation. For example, if a user wishes to optimize the mean of cell C5, they can type =*RiskMean*(C5) in cell C6, specify C6 as the cell to optimize in the Model dialog, and select **Value** from the Statistic dropdown list. This is equivalent to specifying C5 as the cell to optimize, and selecting Mean from the Statistic dropdown list.

Adjustable Cell Ranges

The *Adjustable Cell Ranges* table displays each range which contains the cells or values that RISKOptimizer can adjust, along with the description entered for those cells. Each set of adjustable cells is listed in a horizontal row. One or more adjustable cell ranges can be included in an **Adjustable Cell Group**. All cell ranges in an Adjustable Cell Group share a common solving method, crossover rate, mutation rate and operators.

Minimum	Range	Maximum	Values
Recipe: Max Reservations Accepted			
19	<=	=C14	<= 30 Integer
Recipe: % Full Fare Seats			
0	<=	=C15	<= 1 Any

Because the adjustable cells contain the variables of the problem, you must define at least one group of adjustable cells to use RISKOptimizer. Most problems will be described with only one group of adjustable cells, but more complex problems may require different blocks of variables to be solved with different solving methods simultaneously. This unique architecture allows for highly complex problems to be easily built up from many groups of adjustable cells.

The following options are available for entering Adjustable Cell Ranges:

- **Add.** You can add new adjustable cells by clicking on the “Add” button next to the Adjustable Cells list box. Select the cell or cell range to be added, and a new row will appear in the **Adjustable Cell Ranges** table. In the table, you can enter a **Minimum** and **Maximum** value for the cells in the range, along with the type of Values to test – **Integer** values across the range, or **Any** values.
- **Minimum and Maximum.** After you have specified the location of the adjustable cells, the Minimum and Maximum entries set the range of acceptable values for each adjustable cell. By default, each adjustable cell takes on a real-number (double-precision floating point) value between -infinity and +infinity.

Range settings are constraints that are strictly enforced. RISKOptimizer will not allow any variable to take on a value outside the set ranges. You are encouraged to set more specific ranges for your variables whenever possible to improve RISKOptimizer's performance. For example, you may know that the number cannot be a negative, or that RISKOptimizer should only try values between 50 and 70 for a given variable.

- **Range.** The reference for the cell(s) to be adjusted is entered in the *Range* field. This reference can be entered by selecting the region in the spreadsheet with the mouse, entering a range name or typing in a valid Excel reference such as Sheet1!A1:B8. The **Range** field is available for all solving methods. For recipe and budget methods, however, *Minimum*, *Maximum* and *Values* options can be added to allow the entry of a range for the adjustable cells.

NOTE: By assigning tight ranges to your variables, you can limit the scope of the search, and speed up RISKOptimizer's convergence on a solution. But be careful not to limit the ranges of your variables too tightly; this may prevent RISKOptimizer from finding optimal solutions.

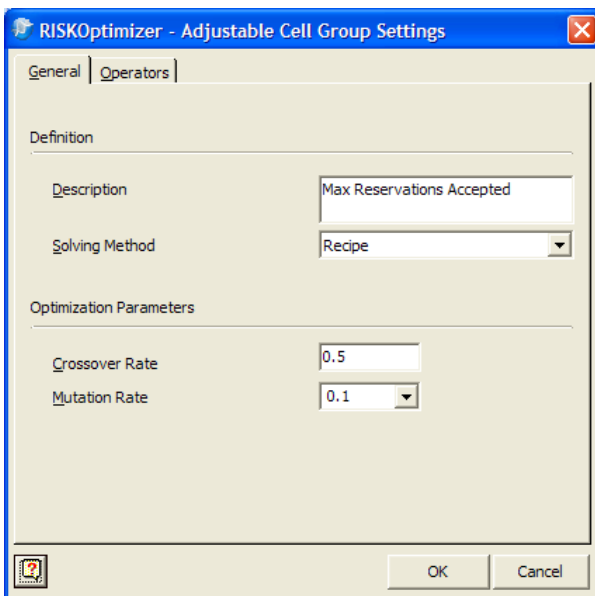
- **Values.** The Values entry allows you to specify that RISKOptimizer should treat all of the variables in the specified range as integers (e.g., 22), rather than as real numbers (e.g., 22.395). This option is only available when using the "recipe" and "budget" solving methods. The default is to treat the variables as real numbers.

Be sure to turn on the Integers setting if your model uses variables to lookup items from tables (HLOOKUP(), VLOOKUP(), INDEX(), OFFSET(), etc.). Note that the Integers setting affects all of the variables in the selected range. If you want to treat some of your variables as reals and some as integers, you can create two groups of adjustable cells instead of one, and treat one block as integers and the other block as reals. Simply "Add" a recipe group of adjustable cells, and leave the Values entry as Any. Next, "Add" another cell range, this time selecting the Integers setting and selecting only the integer adjustable cells.

Adjustable Cell Groups

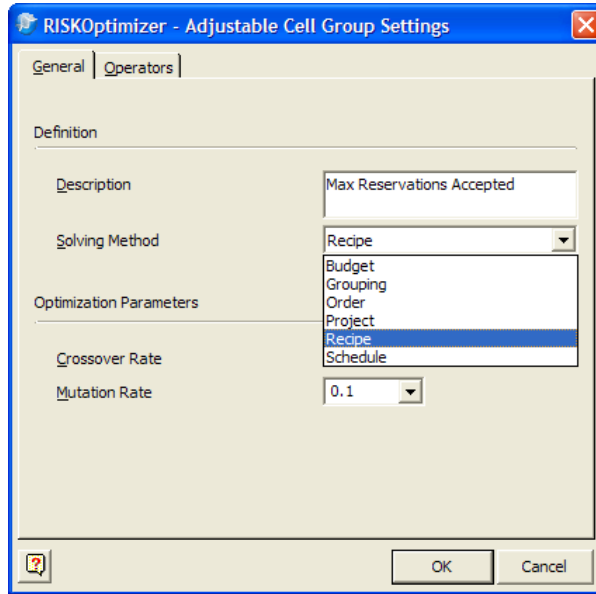
Each group of adjustable cells can contain multiple cell ranges. This allows you to build a "hierarchy" of groups of cell ranges that are related. Within each group, each cell range can have its own Min-Max range constraint.

All cell ranges in an Adjustable Cell Group share a common **solving method, crossover rate, mutation rate and operators**. These are specified in the **Adjustable Cell Group Settings dialog**. This dialog is accessed by clicking the **Group** button next to the **Adjustable Cell Ranges** table. You may create a new Group to which you can add adjustable cell ranges or edit the settings for an existing group.



Options on the **General tab** in the Adjustable Cell Group Settings dialog include:

- **Description.** Describes the group of adjustable cell ranges in dialogs and reports.
- **Solving Method.** Selects the Solving Method to be used for each of the adjustable cell ranges in the group.



When you select a range of cells to be adjusted by RISKOptimizer, you also are specifying a “solving method” you wish to apply when adjusting those adjustable cells. Each solving method is, in essence, a completely different genetic algorithm, with its own optimized selection, crossover and mutation routines. Each solving method juggles the values of your variables a different way.

The “recipe” solving method, for example, treats each variable selected as an ingredient in a recipe; each variable’s value can be changed independently of the others’. In contrast, the “order” solving method swaps values between the adjustable cells, reordering the values that were originally there.

There are six solving methods that come with RISKOptimizer. Three of the solving methods (recipe, order, and grouping) use entirely different algorithms. The other three are *descendants* of the first three, adding additional constraints.

The following section describes the function of each solving method. To get a better understanding of how each solving method is used,

Recipe Solving Method



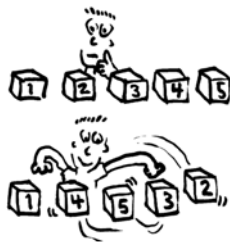
you are also encouraged to explore the example files included with the software (see [Chapter 4: Example Applications](#)).

The “recipe” solving method is the most simple and most popular type of solving method. Use recipe whenever the set of variables that are to be adjusted can be varied independently of one another. Think of each variable as the amount of an ingredient in a cake; when you use the “recipe” solving method, you are telling RISKOptimizer to generate numbers for those variables in an effort to find the best mix. The only constraint you place on recipe variables is to set the range (the highest and lowest value) that those values must fall between. Set these values in the *Min* and *Max* fields in the Adjustable Cells dialog (e.g. 1 to 100), and also indicate whether or not RISKOptimizer should be trying integers (1, 2, 7) or real numbers (1.4230024, 63.72442).

Below are examples of a set of variable values as they might be in a sheet before RISKOptimizer is called, and what two new scenarios might look like after using the recipe solving method.

Original Set of Variable Values	One Set of Possible Recipe Values	Another Set of Possible Recipe Values
23.472	15.344	37.452
145	101	190
9	32.44	7.073
65,664	14,021	93,572

Order Solving Method



The “order” solving method is the second most popular type, after “recipe”. An order is a permutation of a list of items, where you are trying to find the best way to arrange a set of given values. Unlike “recipe” and “budget” solving methods, which ask RISKOptimizer to generate values for the chosen variables, this solving method asks RISKOptimizer to use the existing values in your model.

An order could represent the order in which to perform a set of tasks. For example, you might wish to find the order in which to accomplish five tasks, numbered 1,2,3,4, and 5. The “order” solving method would scramble those values, so one scenario might be 3,5,2,4,1. Because RISKOptimizer is just trying variable values from your initial sheet, there is no Min - Max range entered for adjustable cells when the Order solving method is used.

Below are examples of a set of variable values as they might be in a sheet before RISKOptimizer is called, and what two new scenarios might look like after using the order solving method.

Original Set of Variable Values	One Set of Possible Order Values	Another Set of Possible Order Values
23,472	145	65,664
145	23,472	9
9	65,664	145
65,664	9	23,472

Grouping Solving Method



The “grouping” solving method should be used whenever your problem involves multiple variables to be grouped together in sets. The number of different groups that RISKOptimizer creates will be equal to the number of unique values present in the adjustable cells at the start of an optimization. Therefore, when you build a model of your system, be sure that each group is represented at least once.

For example, suppose a range of 50 cells contains only the values 2, 3.5, and 17. When you select the 50 cells and adjust the values using the “grouping” solving method, RISKOptimizer will assign each of the fifty cells to one of the three groups, 2, 3.5 or 17. All of the groups are represented by at least one of the adjustable cells; just like tossing each of the 50 variables in one of several “bins”, and making sure there is at least one variable in each bin. Another example would be assigning 1s, and 0s, and -1s to a trading system to indicate buy, sell and hold positions. Like the “order” solving method, RISKOptimizer is arranging existing values, so there is no min-max range or integers option to define.

NOTE: When using the “grouping” solving method, do not leave any cells blank, unless you would like 0.0 to be considered one of the groups.

You may realize that the “grouping” solving method could be approximated by using the “recipe” solving method with the integers option “on” and the ranges set from 1 to 3 (or whatever number of groups there are). The difference lies in the way a recipe and a grouping perform their search. Their *selection*, *mutation* and *crossover* routines are different; a grouping is much more concerned with the values of all the variables, because it can swap a set of variables from one group with a set of variables from another group.

Below are examples of a set of variable values as they might be in a sheet before RISKOptimizer is called, and what two new scenarios might look like after using the grouping solving method.

Original Set of Variable Values	One Set of Possible Grouping Values	Another Set of Possible Grouping Values
6	6	8
7	6	7
8	8	6
8	7	7

Budget Solving Method

A “budget” is similar to a “recipe” except that all of the variables’ values must total to a certain number. That number is the total of the variables’ values at the time an optimization is started.

For example, you might want to find the best way to distribute an annual budget among a number of departments. The “budget” solving method will take the total of the current values for the departments, and use that sum as the total budget to be optimally distributed. Below are examples of what two new scenarios might look like after using the budget solving method.

Original Set of Budget Values	One Set of Possible Budget Values	Another Set of Possible Budget Values
200	93.1	223.5
3.5	30	0
10	100	-67
10	.4	67

Many values are being tried, but the sum of all values remains 223.5.

Project Solving Method

The “project” solving method is similar to the “order” solving method except that certain items (tasks) must precede others. The “project” solving method can be used in project management to rearrange the order in which tasks are carried out, but the order will always meet the precedence constraints.

A problem modeled using the *Project* solving method will be much easier to work with and understand if the adjustable cells containing the task order are in a single column, rather than in a row. This is because the solving method expects the preceding tasks cells to be arranged vertically rather than horizontally, and it will be easier to examine your worksheet if the adjustable cells are also vertical.

After you have specified the location of the adjustable cells, you should specify the location of the preceding tasks cells in the *Preceding Tasks* section of the dialog. This is a table of cells that describes which tasks must be preceded by which other tasks. The solving method uses this table to rearrange the order of variables in a scenario until the precedence constraints are met. There should be one row in the preceding tasks range for each task in the adjustable cells. Starting in the first column of the preceding tasks range, the identifying number of each task on which that row’s task depends should be listed in separate columns.

This Item	Must Comes After These...			
1	6	9		
2	1	6		3
3	1			
4	9	12		
5				
6	9	1		2
7	3	4		
8				
9	12	3		1

Example of how to set up precedents for Project solving method.

The precedence tasks range should be specified as being n rows by m columns, where n is the number of tasks in the project (adjustable cells), and m is the largest number of preceding tasks that any one task has.

Below are examples of a set of variable values as they might be in a sheet before RISKOptimizer is called, and what two new scenarios might look like after using the Project solving method, with the constraint that 2 must always come after 1, and 4 must always come after 2.

Original Set of Variable Values	One Set of Possible Project Values	Another Set of Possible Project Values
1	1	1
2	3	2
3	2	4
4	4	3

Schedule Solving Method

A schedule is similar to a grouping; it is an assignment of tasks to times. Each task is assumed to take the same amount of time, much as classes at a school are all of the same length. Unlike a grouping, however, the Adjustable Cell Group Settings Dialog for the “schedule” solving method lets you directly specify the number of time blocks (or groups) to be used. Notice when you select the “schedule” method, several related options appear in the lower portion of the dialog box.

Optimization Parameters

Crossover Rate: 0.5

Mutation Rate: 0.1

Constraint Cells: =L20:N28

Number of Time Blocks: 6

In the *Optimization Parameters* section, you will notice that you can also have a constraint cell range attached to it. This range can be of any length, but must be exactly three columns wide. Eight kinds of constraints are recognized:

- 1) *(with)* The tasks in the 1st & 3rd columns must occur in the same time block.
- 2) *(not with)* The tasks in the 1st & 3rd columns must not occur in the same time block.
- 3) *(before)* The task in the 1st column must occur before the task in the 3rd column.
- 4) *(at)* The task in the 1st column must occur in the time block in the 3rd column.

- 5) *(not after)* The task in 1st column must occur at the same time or before the task in the 3rd column.
- 6) *(not before)* The task in 1st column must occur at the same time or after the task in the 3rd column.
- 7) *(not at)* The task in the 1st column must not occur in the time block in the 3rd column.
- 8) *(after)* The task in the 1st column must occur after the task in the 3rd column.

Either a numeric code (1 through 8) or the description (*after, not at, etc.*) can be entered for a constraint. (Note: All language versions of the RiskOptimizer will recognize the English description entered for a constraint as well as the its translated form). All of the constraints specified in your problem will be met. To create constraints, find an empty space on your worksheet and create a table where the left and right columns represent tasks, and the middle column represents the type of constraints. A number from 1 to 8 represents the kind of constraint listed above. The cells in the constraint range must have the constraint data in them before you start optimizing.

This Task	Constraint	This Task
5	4	2
12	2	8
2	3	1
7	1	5
6	2	4
9	3	1

Below are examples of a set of variable values as they might be in a sheet before RISKOptimizer is called, and what two new scenarios might look like after using the Schedule solving method.

Original Set of Variable Values	One Set of Possible Schedule Values	Another Set of Possible Schedule Values
1	1	1
2	1	3
3	3	1
1	1	2
2	2	2
3	3	2

NOTE: When you select the schedule solving method, integers starting from 1 are always used (1,2,3...), regardless of the original values in the adjustable cells.

Crossover and Mutation Rate

One of the most difficult problems with searching for optimal solutions, when your problem has seemingly endless possibilities, is in determining where to focus your energy. In other words, how much computational time should be devoted to looking in new areas of the “solution space”, and how much time should be devoted to fine-tuning the solutions in our population that have already proven to be pretty good?

A big part of the genetic algorithm success has been attributed to its ability to preserve this balance inherently. The structure of the GA allows good solutions to “breed”, but also keeps “less fit” organisms around to maintain diversity in the hopes that maybe a latent “gene” will prove important to the final solution.

Crossover and *Mutation* are two parameters that affect the scope of the search, and RISKOptimizer allows users to change these parameters before, and also during the evolutionary process. This way, a knowledgeable user can help out the GA by deciding where it should focus its energy. For most purposes, the default crossover and mutation settings (.5 and .1 respectively) do not need adjustment. In the event that you wish to fine-tune the algorithm to your problem, conduct comparative studies, or just to experiment, here is a brief introduction to these two parameters:

- **Crossover.** The crossover rate can be set to between 0.01 and 1.0, and reflects the likelihood that future scenarios or “organisms” will contain a mix of information from the previous generation of parent organisms. This rate can be changed by experienced users to fine-tune RISKOptimizer’s performance on complex problems.

In other words, a rate of 0.5 means that an offspring organism will contain roughly 50% of its variable values from one parent and the remaining values from the other parent. A rate of 0.9 means that roughly 90% of an offspring organism’s values will come from the first parent and 10% will come from the second parent. A Crossover rate of 1 means that no crossover will occur, so only clones of the parents will be evaluated.

The default rate used by RISKOptimizer is 0.5. Once RISKOptimizer has started solving a problem, you can change the crossover rate by using the RISKOptimizer Watcher (see the RISKOptimizer Watcher section in this chapter).

- **Mutation Rate.** The mutation rate can be set to between 0.0 and 1.0, and reflects the likelihood that future scenarios will contain some random values. A higher mutation rate simply means that more mutations or random “gene” values will be introduced into the population. Because mutation occurs after crossover, setting the mutation rate to 1 (100% random values) will effectively prevent the crossover from having any effect, and RISKOptimizer will generate totally random scenarios.

If all the data of the optimal solution was somewhere in the population, then the crossover operator alone would be enough to eventually piece together the solution. Mutation has proven to be a powerful force in the biological world for many of the same reasons that it is needed in a genetic algorithm: it is vital to maintaining a diverse population of individual organisms, thereby preventing the population from becoming too rigid, and unable to adapt to a dynamic environment. As in a genetic algorithm, it is often the genetic mutations in animals which eventually lead to the development of critical new functions.

For most purposes, the default mutation setting does not need adjustment, but can, however, be changed by experienced users to fine-tune RISKOptimizer’s performance on complex problems. The user may wish to boost the mutation rate if RISKOptimizer’s population is fairly homogenous, and no new solutions have been found in the last several hundred trials. Typical setting changes are from .06 to .2. Once RISKOptimizer has started solving a problem, you can change the mutation rate dynamically by using the RISKOptimizer Watcher (see the RISKOptimizer Watcher section later in this chapter).

By selecting *Auto* from the drop down list in the Mutation rate field, auto-mutation rate adjustment is selected. Auto-mutation rate adjustment allows RISKOptimizer to increase the mutation rate automatically when an organism “ages” significantly; that is, it has remained in place over an extended number of trials. For many models, especially where the optimal mutation rate is not known, selecting Auto can give better results faster.

For more information on these options, see the *Schedule Solving* method in the *Solving Methods* section of this chapter.

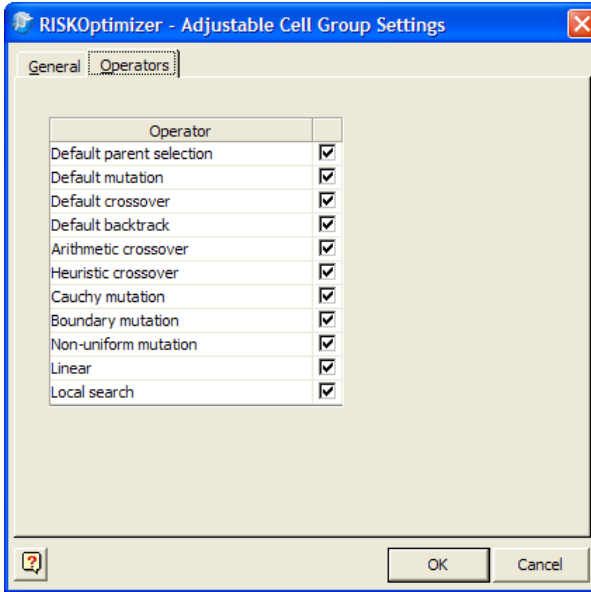
Number of Time Blocks and Constraint Cells

Preceding Tasks

For more information on these options, see the *Project Solving* method in the *Solving Methods* section of this chapter.

Operators

RISKOptimizer includes selectable genetic operators when used with the Recipe solving method. Clicking the **Operators** tab in the Adjustable Cell Group Settings Dialog allows you to select a specific genetic operator (such as heuristic crossover or boundary mutation) to be used when generating possible values for a set of adjustable cells. In addition, you can have RISKOptimizer automatically test all available operators and identify the best performing one for your problem.



Genetic algorithms use genetic operators to create new members of the population from current members. Two of the types of genetic operators RISKOptimizer employs are *mutation* and *crossover*. The mutation operator determines if random changes in “genes” (variables) will occur and how they occur. The crossover operator determines how pairs of members in a population swap genes to produce “offspring” that may be better answers than either of their “parents”.

RISKOptimizer includes the following specialized genetic operators:

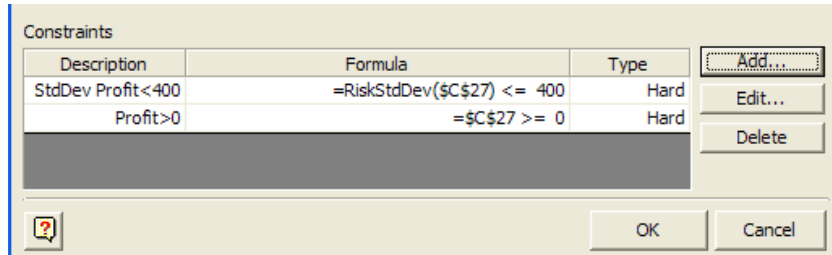
- ◆ **Linear Operators** – Designed to solve problems where the optimal solution lies on the boundary of the search space defined by the constraints. This mutation and crossover operator pair is well suited for solving linear optimization problems.
- ◆ **Boundary Mutation** – Designed to quickly optimize variables that affect the result in a monotonic fashion and can be set to the extremes of their range without violating constraints.
- ◆ **Cauchy Mutation** – Designed to produce small changes in variables most of the time, but can occasionally generate large changes.
- ◆ **Non-uniform Mutation** – Produces smaller and smaller mutations as more trials are calculated. This allows RISKOptimizer to “fine tune” answers.
- ◆ **Arithmetic Crossover** – Creates new offspring by arithmetically combining the two parents (as opposed to swapping genes).
- ◆ **Heuristic Crossover** – Uses values produced by the parents to determine how the offspring is produced. Searches in the most promising direction and provides fine local tuning.

Depending on the type of optimization problem, different combinations of mutation and crossover operators may produce better results than others. In the Operators tab of the Adjustable Cell Group Settings dialog, when using the Recipe solving method, any number of operators may be selected. When multiple selections are made, RISKOptimizer will test valid combinations of the selected operators to identify the best performing ones for your model. After a run, the *Optimization summary worksheet* ranks each of the selected operators by their performance during the run. For subsequent runs of the same model, selecting just the top performing operators may lead to faster, better performing optimizations.

NOTE: When creating multiple groups of adjustable cells, check to be sure that no spreadsheet cell is included in several different groups of adjustable cells. Each group of adjustable cells should contain unique adjustable cells because the values in the first group of adjustable cells would be ignored and overwritten by the values in the second group of adjustable cells. If you think a problem needs to be represented by more than one solving method, consider how to break up the variables into two or more groups.

Constraints

RISKOptimizer allows you to enter constraints, or conditions that must be met for a solution to be valid. Constraints you have entered are shown in the **Constraints table** in the Model Definition dialog box.

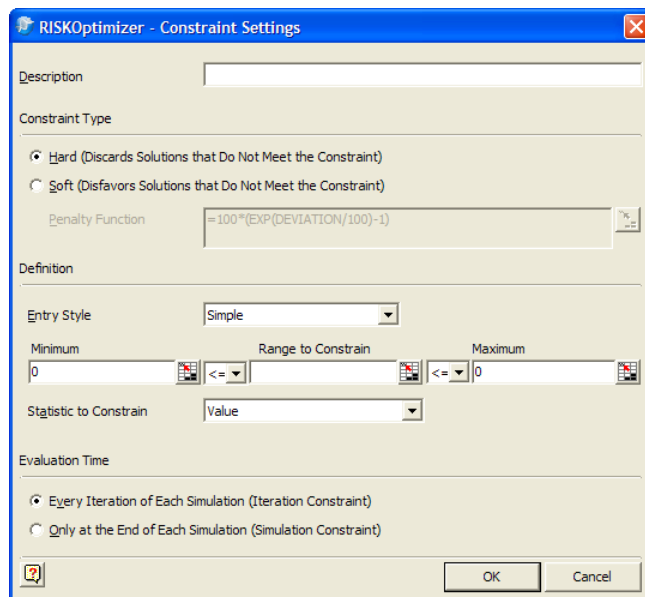


The screenshot shows a dialog box titled "Constraints" with a table containing two rows of constraints. The table has three columns: Description, Formula, and Type. To the right of the table are buttons for "Add...", "Edit...", and "Delete". At the bottom of the dialog are "OK" and "Cancel" buttons.

Description	Formula	Type
StdDev Profit < 400	=RiskStdDev(\$C\$27) <= 400	Hard
Profit > 0	=\$C\$27 >= 0	Hard

Add - Adding Constraints

Clicking the *Add* button next to the Constraints table displays the **Constraint Settings** dialog box where constraints are entered. Using this dialog box the type of constraint desired, along with its description, type, definition and evaluation time can be entered.



The screenshot shows the "RISKOptimizer - Constraint Settings" dialog box. It contains several sections: "Description" with a text field; "Constraint Type" with radio buttons for "Hard" (selected) and "Soft"; "Penalty Function" with a text field containing the formula "=100*(EXP(DEVIATION/100)-1)"; "Definition" with a dropdown for "Entry Style" set to "Simple", and fields for "Minimum" (0), "Range to Constrain" (with a dropdown set to "<="), and "Maximum" (0); "Statistic to Constrain" with a dropdown set to "Value"; and "Evaluation Time" with radio buttons for "Every Iteration of Each Simulation" (selected) and "Only at the End of Each Simulation". At the bottom are "OK" and "Cancel" buttons.

Constraint Type

Two types of constraints can be specified in RISKOptimizer:

- **Hard**, or conditions that must be met for a solution to be valid (i.e., a hard constraint could be $C10 \leq A4$; in this case, if a solution generates a value for C10 that is greater than the value of cell A4, the solution will be thrown out).
- **Soft**, or conditions which we would like to be met as much as possible, but which we may be willing to compromise for a big improvement in fitness or target cell result (i.e., a soft constraint could be $C10 < 100$; however, C10 could go over 100, but when that happened the calculated value for the target cell would be decreased based on the penalty function you have entered).

Evaluation Time

Hard constraints may be evaluated **1) each iteration of a simulation run for a trial solution (an “iteration” constraint)**, or **2) at the end of the simulation run for a trial solution (a “simulation” constraint)**.

- ♦ An **iteration constraint** is a constraint that is evaluated each iteration of a simulation run for a given trial solution. If an iteration results in values which violate the hard constraint, the simulation is stopped (and the trial solution rejected) and the next trial solution and its associated simulation begins.
- A **simulation constraint** is specified in terms of a simulation statistic for a spreadsheet cell; for example the *Mean of A11* > 1000 . In this case, the constraint is evaluated at the end of a simulation. A simulation constraint, as opposed to an iteration constraint, will never cause a simulation to be stopped prior to completion.

Simulation Constraints

A *simulation constraint* is specified in terms of a simulation statistic for a spreadsheet cell; for example the *Mean of A11>1000*. The statistic to use in the constraint is selected from the available dropdown list:

The screenshot shows a dialog box titled "Definition" for setting simulation constraints. It includes the following fields and options:

- Entry Style:** A dropdown menu set to "Simple".
- Range to Constrain:** A field containing the formula "=C27", followed by a comparison operator dropdown set to "<=" and a value field set to "400".
- Statistic to Constrain:** A dropdown menu with a list of statistics: Standard Deviation, Standard Deviation (highlighted), Variance, Skewness, Kurtosis, Minimum, Maximum, Range, and Mode.
- Evaluation Time:** Two radio button options: "Every Iteration of Each Simulation" and "Only at the End Of Each Simulation".

When a simulation constraint is used, a distribution of possible values for the *Range to Constrain* is generated during each trial solution's simulation. At the end of each simulation, the constraint is checked to see if it has been met. If the simulation constraint is a hard constraint and the constraint is not met, the trial solution is discarded. If the constraint is a soft constraint and the constraint is not met, the target cell's statistic that is being minimized or maximized is penalized according to the entered penalty function (see the next section *Soft Constraints*).

Simple and Formula Constraints

Two formats – *Simple* and *Formula* -- can be used for entering constraints. The type of information you can enter for a constraint depends on the format you select.

- **Simple Format** - The *Simple* format allows constraints to be entered using simple $<$, \leq , $>$, \geq or $=$ relations where a cell is compared with an entered number. A typical Simple constraint would be:

$0 < \text{Value of } A1 < 10$

where A1 is entered in the *Cell Range* box, 0 is entered in the *Min* box and 10 is entered in the *Max* box. The operator desired is selected from the drop down list boxes. With a simple range of values format constraint, you can enter just a Min value, just a Max or both. The entered Min and Max values must be numeric in the simple range of values constraint format.

- **Formula Format** - The *Formula* format allows you to enter any valid Excel formula as a constraint, such as $A19 < (1.2 * E7) + E8$. RISKOptimizer will check whether the entered formula evaluates to TRUE or FALSE to see if the constraint has been met

Soft Constraints

Soft Constraints are conditions which we would like to be met as much as possible, but which we may be willing to compromise for a big improvement in fitness or target cell result. When a soft constraint is not met it causes a change in the target cell result away from its optimal value. The amount of change caused by an unmet soft constraint is calculated using a penalty function that is entered when you specify the soft constraint.

RISKOptimizer - Constraint Settings

Description: StdDev Profit<400

Constraint Type:

- ☐ Hard (Discards Solutions that Do Not Meet the Constraint)
- ☒ Soft (Disfavors Solutions that Do Not Meet the Constraint)

Penalty Function: $100 * (\text{EXP}(\text{deviation}/100) - 1)$

Definition:

Entry Style: Simple

Range to Constrain: =C27 <= 400

Statistic to Constrain: Standard Deviation

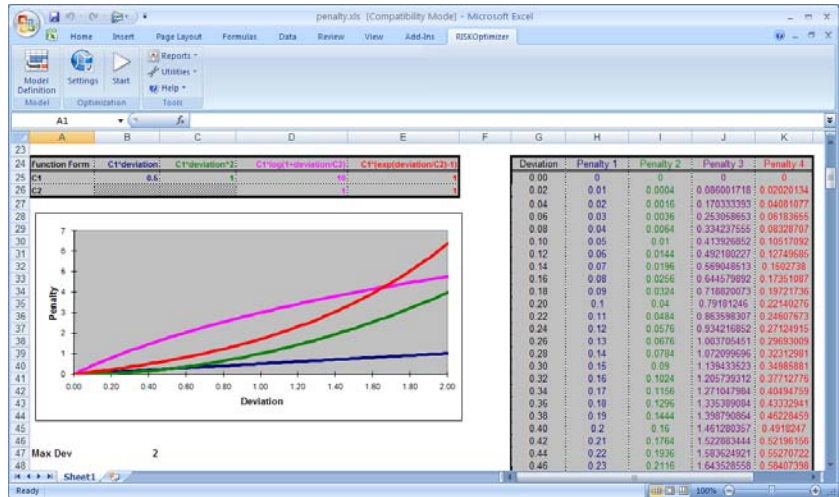
OK Cancel

More information about penalty functions is as follows:

- Entering a Penalty Function.** RISKOptimizer has a default penalty function which is displayed when you first enter a soft constraint. Any valid Excel formula, however, may be entered to calculate the amount of penalty to apply when the soft constraint is not met. An entered penalty function should include the keyword *deviation* which represents the absolute amount by which the constraint has gone beyond its limit. At the end of each simulation for a trial solution RISKOptimizer checks if the soft constraint has been met; if not, it places the amount of deviation in the entered penalty formula and then calculates the amount of penalty to apply to the target cell statistic.

The penalty amount is either added or subtracted from the calculated statistic for the target cell in order to make it less "optimal". For example, if *Maximum* is selected in the *Optimization Goal* field in the RISKOptimizer Model Dialog, the penalty is subtracted from the calculated statistic for the target cell.

- Viewing the Effects of an Entered Penalty Function.** RISKOptimizer includes an Excel worksheet PENALTY.XLS which can be used to evaluate the effects of different penalty functions on specific soft constraints and target cell results.



PENALTY.XLS allows you to select a soft constraint from your model whose effects you wish to analyze. You can then change the penalty function to see how the function will map a specific value for the unmet soft constraint into a specific penalized target value. For example, if your soft constraint is $A10 < 100$, you could use PENALTY.XLS to see what the target value would be if a value of 105 was calculated for cell A10.

- **Viewing the Penalties Applied.** When a penalty is applied to the target cell due to an unmet soft constraint, the amount of penalty applied can be viewed in the RISKOptimizer Watcher. In addition, penalty values are shown in Optimization Log worksheets, created optionally after optimization.

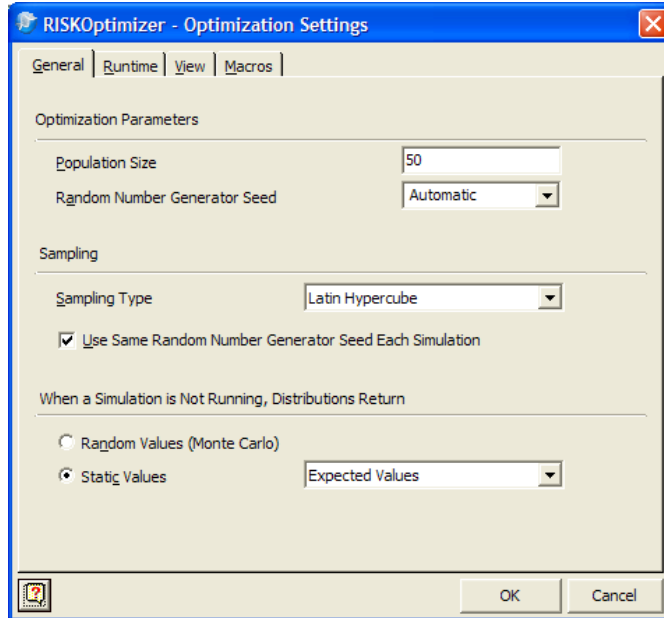
NOTE: If you place a solution in your worksheet using the **Update Adjustable Cell Values** options in the **Stop** dialog, the calculated target cell result shown in the spreadsheet will not include any penalties applied due to unmet soft constraints. Check the **RISKOptimizer optimization summary worksheet** to see the penalized target cell result and the amount of penalty imposed due to each unmet soft constraint.

- **Implementing Soft Constraints in Worksheet Formulas.** Penalty functions can be implemented directly in the formulas in your worksheet. If soft constraints are implemented directly in the worksheet they should not be entered in the main RISKOptimizer dialog. For more information on implementing penalty functions in your worksheet, see the section *Soft Constraints* in [Chapter 9: RISKOptimizer Extras](#).

Optimization Settings Command – General Tab

Defines the general settings for an optimization

The Optimization Settings dialog General tab displays settings for population size, display update, and random number generator seed.



Optimization Parameter Options on the General tab include:

- **Population Size.** The population size tells RISKOptimizer how many organisms (or complete sets of variables) should be stored in memory at any given time. Although there is still much debate and research regarding the optimal population size to use on different problems, generally we recommend using 30-100 organisms in your population, depending on the size of your problem (bigger population for larger problems). The common view is that a larger population takes longer to settle on a solution, but is more likely to find a global answer because of its more diverse gene pool.

- **Random Number Generator Seed.** The Random Number Generator Seed option allows you to set the starting seed value for the random number generator used in RISKOptimizer. When the same seed value is used, an optimization will generate the exact same answers for the same model as long as the model has not been modified and the **Use Same Random Number Generator Seed Each Sim** option is selected. The **Use Same Random Number Generator Seed Each Sim** option should be checked to insure that unnecessary randomness from simulation to simulation is not introduced into optimization results. The seed value must be an integer in the range 1 to 2147483647.

Sampling Options on the General tab include:

- **Sampling Type.** Simulation Sampling Type options set the type of sampling used during each simulation run during an optimization. Sampling types vary in how they draw samples from across the range of a distribution. Latin Hypercube sampling will accurately recreate the probability distributions specified by distribution functions in fewer iterations when compared with Monte Carlo sampling. We recommend using Latin Hypercube, the default sampling type setting, unless your modeling situation specifically calls for Monte Carlo sampling.
- **Use Same Random Number Seed for Each Simulation.** This option specifies that RISKOptimizer will use a repeatable random number stream each simulation it runs, insuring that distributions return the same samples each trial simulation in the optimization.

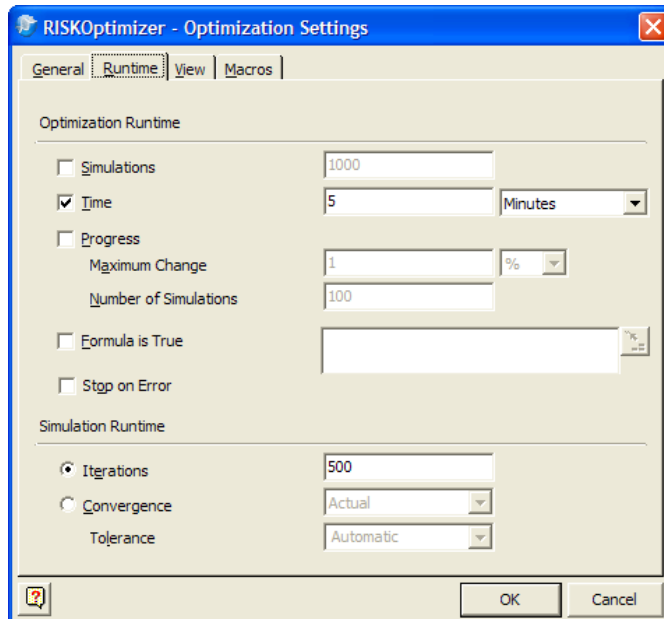
When a Simulation is Not Running, Distributions Return options control what is displayed when <F9> is pressed and a standard Excel recalculation is performed. Options include:

- **Random Values (Monte Carlo).** In this mode, distribution functions return a random Monte Carlo sample during a regular recalculation. This setting allows your worksheet values to appear as they would during execution of a simulation with new samples drawn for distribution functions each recalculation.
- **Static Values.** In this mode, distribution functions return Static values entered in a RiskStatic property function during a regular recalculation. If a static value is not defined for a distribution function, it will return:
 - **Expected Value**, or a distribution's expected or mean value. For discrete distributions, the setting "Corrected" Expected Value will use the discrete value in the distribution closest to the true expected value as the swap value.
 - **True" Expected Value.** This setting causes the same values to be swapped as the option "Corrected" Expected Value, except in the case of discrete distribution types such as DISCRETE, POISSON and similar distributions. For these distributions the true expected value will be used as the swap value even if the expected value could not occur for the entered distribution, i.e., it is not one of the discrete points in the distribution.
 - **Mode**, or a distribution's mode value.
 - **Percentile**, or the entered percentile value for each distribution.

Optimization Settings Command – Runtime Tab

Defines the runtime settings for an optimization

The Optimization Settings dialog Runtime tab displays RISKOptimizer settings that determine the runtime of the optimization. These stopping conditions specify how and when RISKOptimizer will stop during an optimization. Once you select the Start Optimization command, RISKOptimizer will continuously run, searching for better solutions and running simulations until the selected stopping criteria are met. You can turn on any number of these conditions, or none at all if you want RISKOptimizer to search indefinitely (until you stop it). When multiple conditions are checked, RISKOptimizer stops as soon as one of the chosen conditions is met. You may also override these selections and stop RISKOptimizer at any time manually using the stop button in the RISKOptimizer Watcher or Progress windows.



Optimization Runtime options on the Runtime tab include:

- **Simulations** - This option, when set, stops RISKOptimizer when the given number of simulations have been executed. A simulation is run for each trial solution generated by RISKOptimizer.

The Simulations setting is particularly useful when comparing RISKOptimizer's efficiency when trying different modeling methods. By changing the way you model a problem, or by choosing a different solving method, you may increase RISKOptimizer's efficiency. Having a model run a specified number of simulations will indicate how efficiently RISKOptimizer is converging on a solution, regardless of any differences in the number of variables chosen, the speed of the computer hardware being used, or the screen re-drawing time. The RISKOptimizer optimization summary worksheet is also useful in comparing results between runs. For more information on Optimization Summary worksheets, see the RISKOptimizer Watcher – Stopping Options section in this chapter.

- **Time** - This option, when set, stops RISKOptimizer from simulating scenarios after the given number of hours, minutes or seconds has elapsed. This entry can be any positive real number (600, 5.2, etc.).
- **Progress** - This option, when set, stops RISKOptimizer from simulating scenarios when the improvement in the target cell is less than the specified amount (change criterion). You can specify, as an integer, the number of simulations over which to check the improvement. A percentage value - such as 1% - can be entered as the maximum change value in the *Maximum Change* field.

Suppose that we are trying to maximize the mean of the target cell, and after 500 simulations, the best answer found so far is 354.8. If the "Progress" option is the only stopping condition selected, RISKOptimizer will pause at simulation #600 and will only continue if it is able to find an answer of at least 354.9 during those last 100 simulations. In other words, RISKOptimizer's answers have not improved at least 0.1 over the last 100 simulations, so it assumes there is little more improvement to be found, and stops the search. For more complex problems, you may want to boost the number of simulations that RISKOptimizer runs through (500) before deciding whether there is still sufficient improvement to go on.

This is the most popular stopping condition, because it gives the user an effective way to stop RISKOptimizer after the improvement rate is slowing down, and RISKOptimizer is not seeming to find any better solutions. If you are viewing the graphs of the best results on the Progress tab of the RISKOptimizer Watcher, you will see the graphs plateau or flatten out for a while before this condition is met and RISKOptimizer stops. “Progress” is really just an automatic way to do what you could do yourself with the graph -- let it run until the improvement levels off.

- **Formula is True.** This stopping condition causes the optimization to stop whenever the entered (or referenced) Excel formula evaluates to TRUE during the optimization.
- **Stop on Error.** This stopping condition causes the optimization to stop whenever an Error value is calculated for the target cell.

NOTE: You can also select no stopping conditions, and RISKOptimizer will run forever until you press the stop button on the RISKOptimizer Watcher window.

Simulation Runtime Options

Simulation Runtime options specify how RISKOptimizer will determine when to stop each simulation it runs. You can run each simulation for a fixed number of iterations, or, alternatively, let RISKOptimizer determine when to stop each simulation.

Simulation Runtime options on the Runtime tab include:

- **Iterations** - This option allows you to run each simulation for a fixed number of iterations. In this case, RISKOptimizer will run the specified number of iterations with each simulation that is performed for each trial solution generated by RISKOptimizer (unless stopped prematurely when an iteration constraint is not met).
- **Convergence** - Convergence can be used as a simulation stopping condition using **Actual** or **Projected** convergence:
 - **Actual** convergence instructs RISKOptimizer to stop each simulation when the distributions generated for both 1) the target cell of the optimization and 2) cells referenced in simulation constraints are stable and the statistics of interest converge. The amount of variation allowed in a statistic when it is marked as "converged" is set by the *Tolerance* option.

- **Projected Convergence** - This option instructs RISKOptimizer to stop each simulation when it can project internally that the distributions generated for both 1) the target cell of the optimization and 2) cells referenced in simulation constraints are stable. RISKOptimizer projects convergence based on the results of prior simulations that have been run during the optimization.

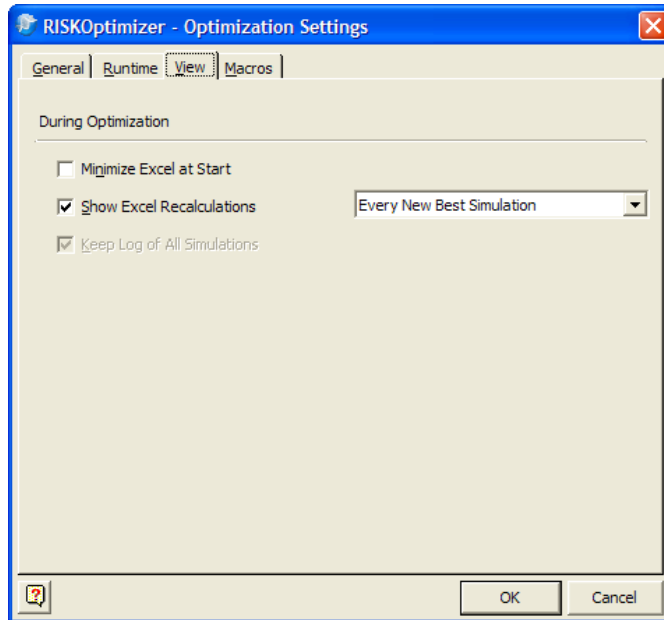
Allowing RISKOptimizer to control when to stop is recommended to insure that enough iterations are run (but not too many!) so that output statistics returned to the optimizer are stable. You may, however, wish to limit the iterations run by specifying a fixed number of iterations to speed your optimizations. This would be done when models are very large or each Excel recalculation of the model takes a long time.

The **Tolerance** option takes a value from 1 to 100 (or *auto*). This sets the amount of change allowed when the statistic of interest is said to be converged. A low setting requires that there be very little change in the statistic for it to be labeled as “converged”; in comparison, high settings close to 100 allow much greater variation in converged statistics. The trade off of low tolerance settings vs. high is the amount of time required to run an optimization. Many additional iterations may be required per simulation to achieve convergence with a low tolerance setting, often with no marked improvement in overall optimization results. Selecting *Auto* specifies that RISKOptimizer will set the convergence tolerance for you.

Optimization Settings Command – View Tab

Defines the view settings for an optimization

The Optimization Settings dialog View tab displays RISKOptimizer settings that determine what will be shown during an optimization.



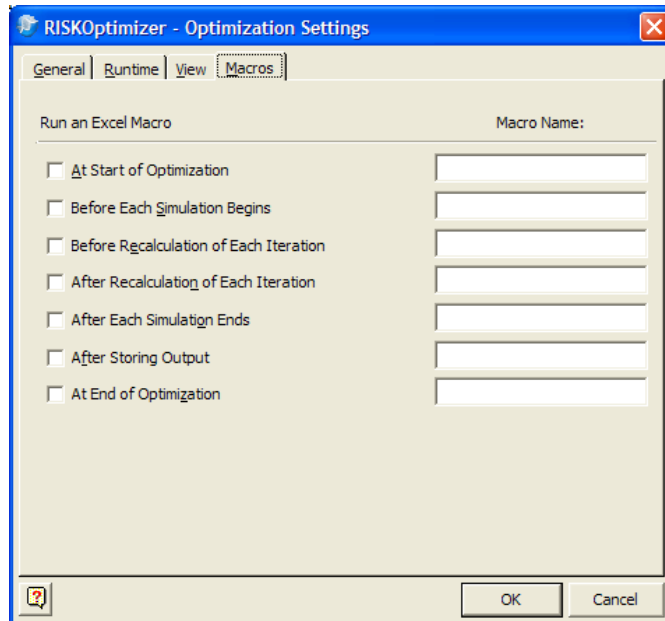
Options on the View tab include:

- **Minimize Excel at Start.** This option selects to minimize Excel when an optimization starts.
- **Show Excel Recalculations.** This specifies to update Excel either with **Every New Best Simulation**, or at the end of **Every Simulation**. Note that in some situations the screen will be updated independently of these settings, for example when optimization has been paused.
- **Keep Log of Simulations.** This option specifies that RISKOptimizer keeps a running log of each new simulation performed. This log can be viewed in the RISKOptimizer Watcher Window.

Optimization Settings Command – Macros Tab

Defines macros to be run during an optimization

VBA macros can be run at different times during an optimization and during the simulation run for each trial solution. This allows the development of custom calculations that will be invoked during an optimization.



Macros may be executed at the following times during an optimization:

- **At the Start of the Optimization** - macro runs after the Run icon is clicked; prior to the first trial solution being generated.
- **Before Each Simulation Begins** - macro runs prior to each simulation that is executed (a simulation for each trial solution generated by the optimizer).
- **Before Recalculation of Each Iteration** - macro runs after sampling but before recalculation of each iteration of each simulation that is executed. Macro is run after samples are drawn from probability distribution functions for the iteration but before subsequent recalculation of the model.

- **After Recalculation of Each Iteration** - macro runs after each iteration of each simulation that is executed. Macro is run after samples are drawn from probability distribution functions for the iteration and the recalculation of the model using those samples; but prior to the value for the target cell being collected.
- **After Each Simulation Ends** - macro runs after each simulation that is executed, but prior to the statistic that is being optimized for the target cell's distribution is stored.
- **After Storing Output** - macro runs after each simulation that is executed and after the statistic that is being optimized for the target cell's distribution is stored.
- **At the End of the Optimization** - macro runs when the optimization is completed.

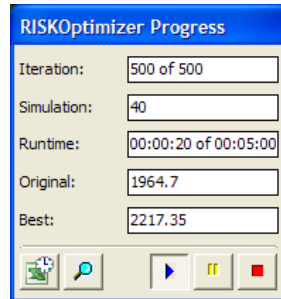
This feature allows calculations which only can be performed through the use of a macro to be made during an optimization. Examples of such macro-performed calculations are iterative "looping" calculations and calculations which require new data from external sources.

The **Macro Name** defines the macro to be run.

Start Optimization Command

Starts an optimization

Selecting the Start Optimization command or clicking the Start Optimization icon starts an optimization of the active model and workbook. As soon as RISKOptimizer is running, you will see the following RISKOptimizer **Progress window**.



The Progress window displays:

- **Iteration** or the number of iterations run in the current simulation.
- **Simulation** or the total number of simulations that have been executed and **#Valid** indicates the number of those simulations for which all constraints were met.
- **Runtime** or the elapsed time in the run
- **Original** or the original value for the statistic for the target cell as calculated from an initial simulation run using the existing values in the worksheet for the adjustable cells.
- **Best** or the current best value for the statistic for the target cell that is being minimized or maximized.

During an optimization the status bar in Excel also displays the current progress of the analysis.

Best= 2096.61 (Simulation #9) Original= 1731.73 Simulations= 10 Time= 0:00:06

Options on the RISKOptimizer Toolbar of the Progress window include:

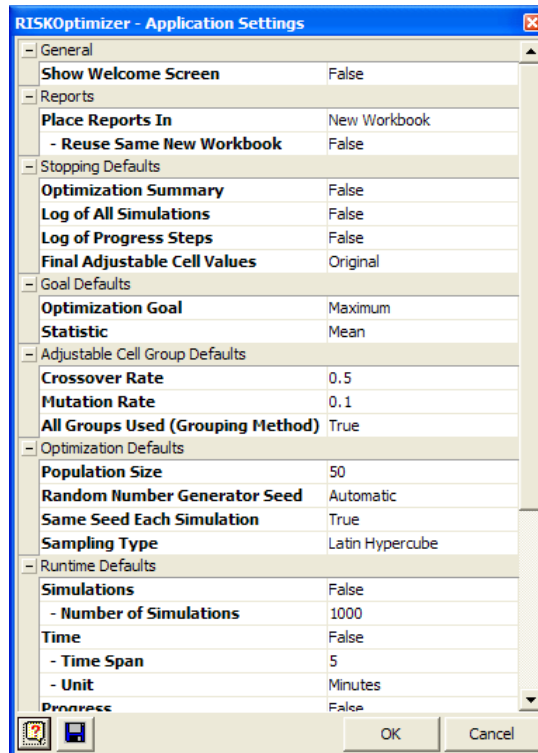
- **Display Excel Updating Options.** Selects to update the Excel display **Every Simulation**, on **Every New Best Solution** or **Never**. Note that in some situations the screen will be updated independently of these settings, for example when optimization has been paused.
- **Display RISKOptimizer Watcher.** Displays the full RISKOptimizer Watcher window.
- **Run.** Clicking the Run icon causes RISKOptimizer to begin searching for a solution based on the current description in the RISKOptimizer Model Dialog. If you pause RISKOptimizer you will still be able to click the Run icon to continue the search for better solutions.
- **Pause.** If you would like to pause the RISKOptimizer process, just click the Pause icon, and you temporarily “freeze” the RISKOptimizer process. While paused, you may wish to open and explore the RISKOptimizer Watcher and change parameters, look at the whole population, view a status report, or copy a graph.
- **Stop.** Stops the optimization.

Utilities Commands

Application Settings Command

Displays the Application Settings dialog where program defaults can be set

A wide variety of RISKOptimizer settings can be set at default values that will be used each time RISKOptimizer runs. These include Stopping Defaults, Default Crossover and Mutation Rates and others.

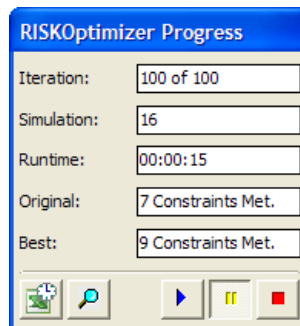


Constraint Solver Command

Runs the Constraint Solver

The Constraint Solver enhances RISKOptimizer's ability to handle model constraints. When RISKOptimizer runs an optimization, it is assumed that the original adjustable cell values meet all the hard constraints, i.e. that the original solution is valid. If that is not the case, the algorithm may run very many simulations before finding a first valid solution. However, if a model contains multiple constraints, then it may not be obvious what adjustable cell values will meet all of them.

If a RISKOptimizer model contains multiple hard constraints, and optimizations fail with all solutions invalid, you will be notified and the Constraint Solver can be run. The Constraint Solver runs an optimization in a special mode, in which the objective is to find a solution meeting all the hard constraints. The optimization progress is shown to the user in the same way as in regular optimizations. The **Progress Window** shows the number of constraints that are met in the original and best solutions.



A button in the Progress Window allows the user to switch to the RISKOptimizer Watcher. In the Constraint Solver mode the details of optimization progress are available like in regular mode optimizations, in **Progress**, **Summary**, **Log**, **Population** and **Diversity** tabs. In the Constraint Solver mode the Watcher contains an additional tab, entitled **Constraint Solver**. This tab shows the status of each hard constraint (**Met** or **Not Met**) for the Best, Original, and Last solution.

RISKOptimizer Watcher

Progress | Summary | Log | Population | Diversity | **Constraint Solver** | Stopping Options

Hard Constraints

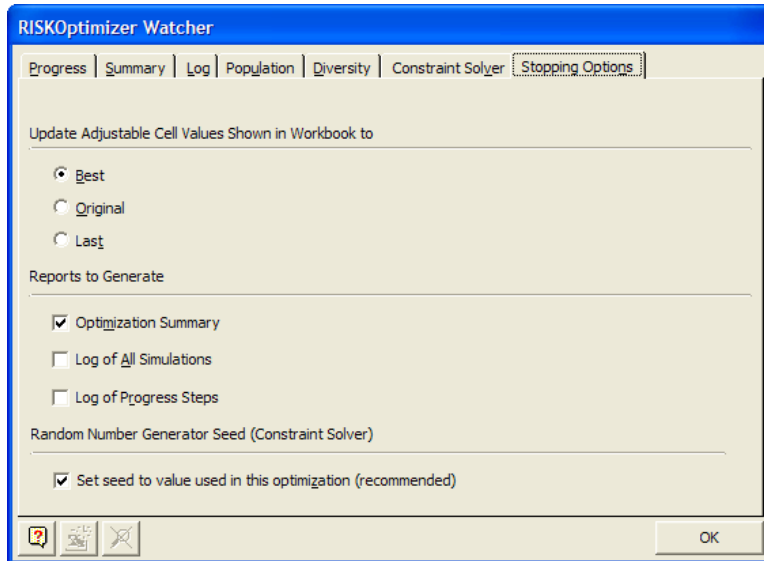
Best	Original	Last	Description	Formula
MET	MET	MET	Alma in Range	=RiskMean(\$K\$21) >= \$M\$21
MET	NOT MET	MET	Auburn in Range	=RiskMean(\$K\$22) >= \$M\$22
MET	NOT MET	MET	Antonito in Range	=RiskMean(\$K\$23) >= \$M\$23
MET	MET	MET	Appleton in Range	=RiskMean(\$K\$24) >= \$M\$24
MET	MET	MET	Barrow in Range	=RiskMean(\$K\$25) >= \$M\$25
NOT MET	NOT MET	NOT MET	Byers in Range	=RiskMean(\$K\$26) >= \$M\$26
MET	MET	MET	Carthage in Range	=RiskMean(\$K\$27) >= \$M\$27
MET	MET	MET	Cedar in Range	=RiskMean(\$K\$28) >= \$M\$29
MET	MET	MET	Dobbs in Range	=RiskMean(\$K\$29) >= \$M\$29
MET	MET	NOT MET	Dover in Range	=RiskMean(\$K\$30) >= \$M\$30

Number of Constraints=10 Best=9 Constraints Met. (Simulation #12) Original=7 Constraints Met.
Completed Simulations=36 Time=00:00:22

A Constraint Solver optimization stops automatically when a solution meeting all the hard constraints is found; it can also be stopped by clicking a button in the progress window or in the RISKOptimizer Watcher. After a Constraint Solver run, in the RISKOptimizer Watcher Stopping Options tab you can choose to keep the Best, Original, or Last solution, like in regular-mode optimizations.

Note there is no need to set up the Constraint Solver before a run. It uses the settings specified in the model, only changing the optimization objective: the new objective is to find a solution meeting all the hard constraints.

In the **Stopping Options** tab there is an additional recommended option to **Set Seed to Value Used in this Optimization**. This option is provided because if the random number generator seed is not fixed, then constraints that were met during a Constraint Solver run might not be met during a regular-mode run, even if adjustable cell values are the same (since simulation results depend on the seed). The option is grayed out if the seed was fixed in the Optimization Settings dialog before a Constraint Solver run.



RISKOptimizer Watcher

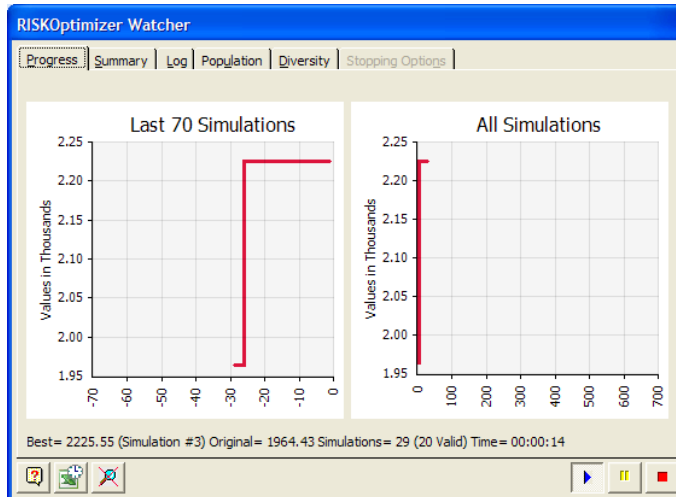
The magnifying glass icon on the RISKOptimizer Progress window toolbar displays the RISKOptimizer Watcher. RISKOptimizer Watcher is responsible for regulating and reporting on all RISKOptimizer activity.

From RISKOptimizer Watcher, you can change parameters and analyze the progress of the optimization. You can also see real-time information about the problem and information on RISKOptimizer's progress in the status bar across the bottom of RISKOptimizer Watcher.

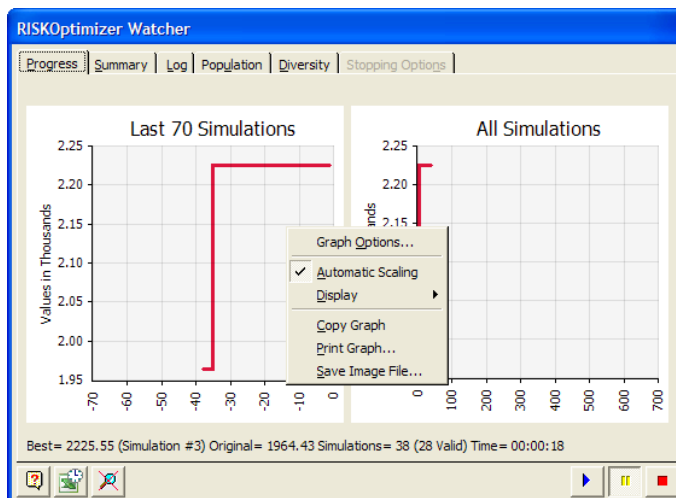
RISKOptimizer Watcher – Progress Tab

Displays progress graphs for target cell value

The **Progress Tab** in the RISKOptimizer Watcher graphically shows how results are changing, by simulation, for the selected target cell.

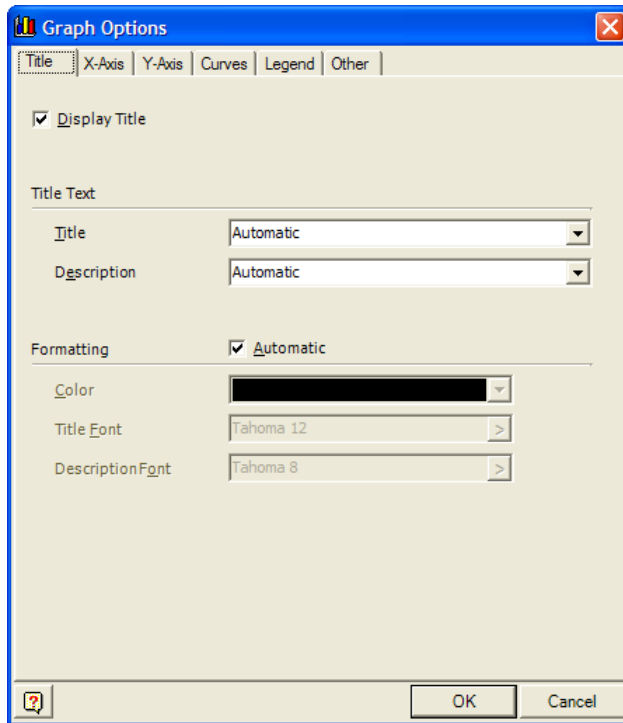


Progress graphs show the simulation count on the X-axis and target cell value on the Y-axis. Progress graphs can be rescaled by clicking on the axis limits and dragging the axis to the new scale value. Alternatively, right-clicking on the Progress graph can display the **Graph Options** dialog where further customization of the graphs is allowed.



Graph Options Dialog

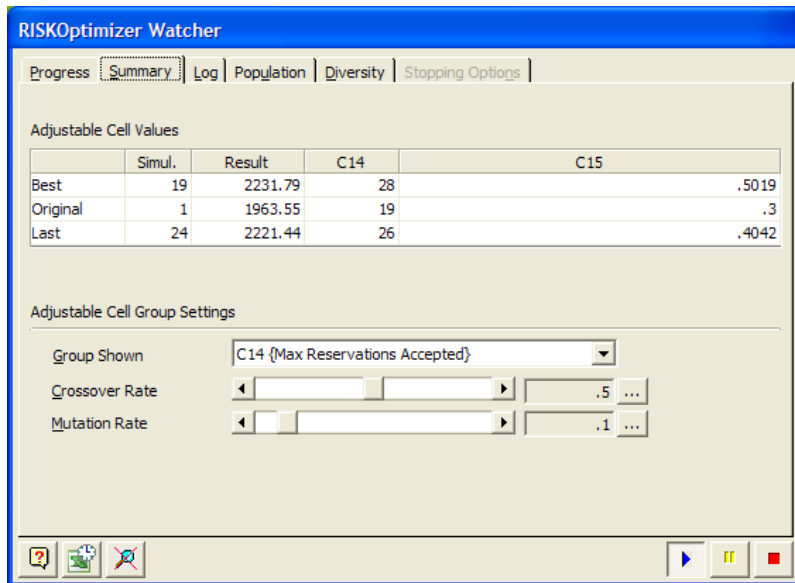
The Graph Options dialog displays settings that control the titles, legends, scaling and fonts used on the displayed graph.



RISKOptimizer Watcher – Summary Tab

Displays details for adjustable cell values

The **Summary Tab** in the RISKOptimizer Watcher displays a summary table of adjustable cell values tested during the optimization, along with tools for adjusting the crossover and mutation rate for each Adjustable Cell Group in the model.



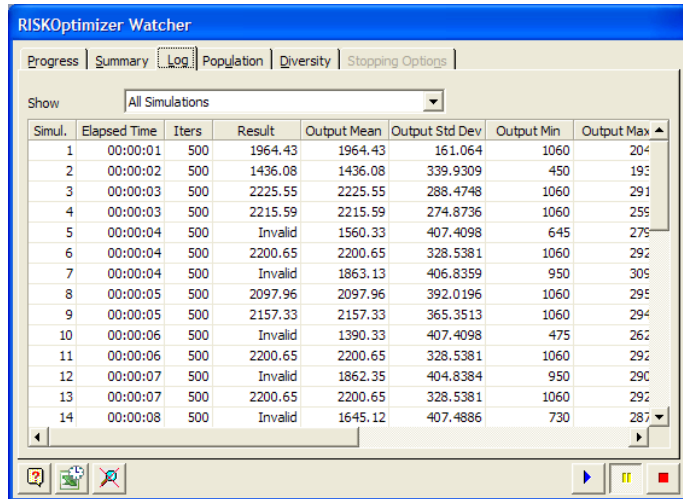
The **Adjustable Cell Group Settings** allows you to change the Crossover and Mutation rates of the genetic algorithm as the problem is in progress. Any changes made here will override the original setting of these parameters and will take place immediately, affecting the population (or group of adjustable cells) that was selected in the **Group Shown** field.

We almost always recommend using the default crossover of 0.5. For mutation, in many models you may turn it up as high as about 0.4 if you want to find the best solution and are willing to wait longer for it. Setting the mutation value to 1 (the maximum) will result in completely random guessing, as RISKOptimizer performs mutation after it performs crossover. This means that after the two selected parents are crossed over to create an offspring solution, 100% of that solution's "genes" will mutate to random numbers, effectively rendering the crossover meaningless (see "crossover rate, what it does" and "mutation rate, what it does" in the index for more information).

RISKOptimizer Watcher – Log Tab

Displays a log of each simulation run during the optimization

The **Log Tab** in the RISKOptimizer Watcher displays a summary table of each simulation run during the optimization. The log includes the results for the target cell, each adjustable cell and entered constraints. A log is only available if the option **Keep a Log of All Simulations** is selected in the Optimization Settings dialog **View** tab.



The screenshot shows the RISKOptimizer Watcher application window with the 'Log' tab selected. The 'Show' dropdown is set to 'All Simulations'. The table below displays the log data for 14 simulations.

Simul.	Elapsed Time	Iters	Result	Output Mean	Output Std Dev	Output Min	Output Max
1	00:00:01	500	1964.43	1964.43	161.064	1060	204
2	00:00:02	500	1436.08	1436.08	339.9309	450	193
3	00:00:03	500	2225.55	2225.55	288.4748	1060	291
4	00:00:03	500	2215.59	2215.59	274.8736	1060	259
5	00:00:04	500	Invalid	1560.33	407.4098	645	275
6	00:00:04	500	2200.65	2200.65	328.5381	1060	292
7	00:00:04	500	Invalid	1863.13	406.8359	950	305
8	00:00:05	500	2097.96	2097.96	392.0196	1060	295
9	00:00:05	500	2157.33	2157.33	365.3513	1060	294
10	00:00:06	500	Invalid	1390.33	407.4098	475	262
11	00:00:06	500	2200.65	2200.65	328.5381	1060	292
12	00:00:07	500	Invalid	1862.35	404.8384	950	290
13	00:00:07	500	2200.65	2200.65	328.5381	1060	292
14	00:00:08	500	Invalid	1645.12	407.4886	730	287

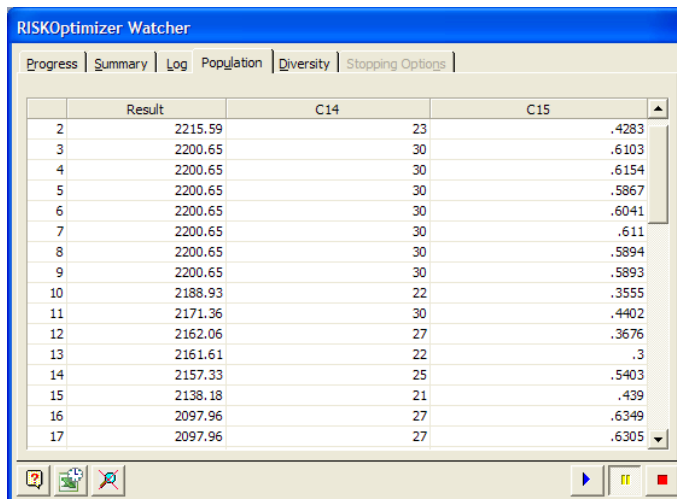
The Show options select to show a log of **All Simulations** or only those simulations where there was a **Progress Step** (i.e. where the optimization result improved). The log includes:

- 1) **Elapsed Time**, or the start time of the simulation
- 2) **Iters**, or the number of iterations run
- 3) **Result**, or the value of the target cell's statistic that you are trying to maximize or minimize, including penalties for soft constraints
- 4) **Output Mean, Output StdDev, Output Min and Output Max**, or the statistics for the probability distribution for the target cell that was calculated
- 5) **Input columns**, or the values used for your adjustable cells
- 6) **Constraint columns** showing whether your constraints were met

RISKOptimizer Watcher – Population Tab

Lists all the variables of each organism (each possible solution) in the current population

The population table is a grid which lists all the variables of each organism (each possible solution) in the current population. These organisms (“Org *n*”) are ranked in order from worst to best. Since this table lists all organisms in the population, the “population size” setting in the RISKOptimizer Settings dialog determines how many organisms will be listed here (default 50). In addition, the first column of the chart shows the resulting value of the target cell for each organism.



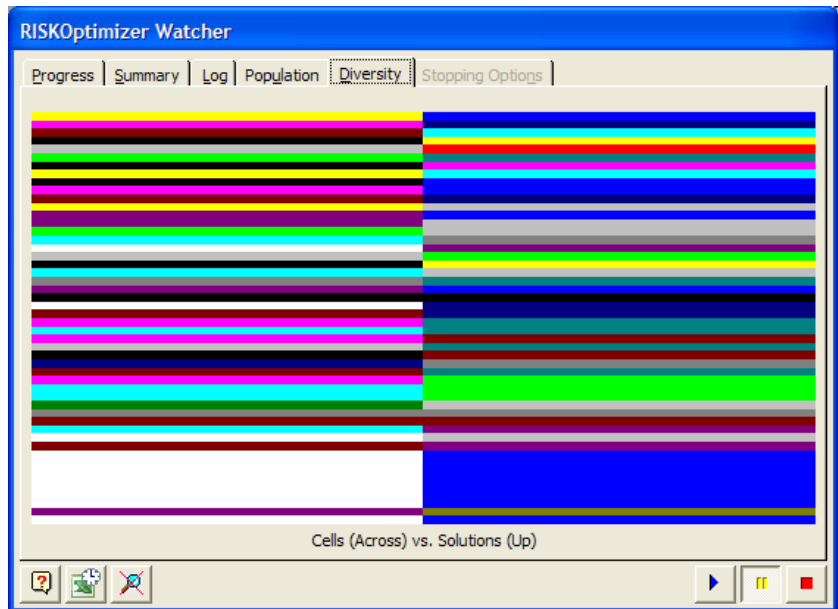
The screenshot shows the RISKOptimizer Watcher application window with the Population tab selected. The window has a blue title bar and a menu bar with options: Progress, Summary, Log, Population, Diversity, and Stopping Options. The main area contains a table with 5 columns: an index column, a Result column, and three columns labeled C14 and C15. The table lists 17 organisms, ranked from worst to best. The bottom of the window features a toolbar with icons for file operations and a status bar.

	Result	C14	C15
2	2215.59	23	.4283
3	2200.65	30	.6103
4	2200.65	30	.6154
5	2200.65	30	.5867
6	2200.65	30	.6041
7	2200.65	30	.611
8	2200.65	30	.5894
9	2200.65	30	.5893
10	2188.93	22	.3555
11	2171.36	30	.4402
12	2162.06	27	.3676
13	2161.61	22	.3
14	2157.33	25	.5403
15	2138.18	21	.439
16	2097.96	27	.6349
17	2097.96	27	.6305

RISKOptimizer Watcher – Diversity Tab

Displays a color plot of all variables in the current population

The plot on the Diversity tab assigns colors to adjustable cell values, based on how much the value of a given cell differs across the population of organisms (solutions) that are stored in memory at a given point. (Using the genetic optimization terminology, this is an indication of the diversity that exists in the gene pool.) Each vertical bar in the plot corresponds to one adjustable cell. Horizontal stripes within each bar represent the values of that adjustable cell in different organisms (solutions). The colors of the stripes are assigned by dividing the range between the minimum and maximum value for a given adjustable cell into 16 equal-length intervals; each of the intervals is represented by a different color. For example, in the picture the fact that the vertical bar representing the second adjustable cell is single-color means that the cell has the same value in each solution in memory.

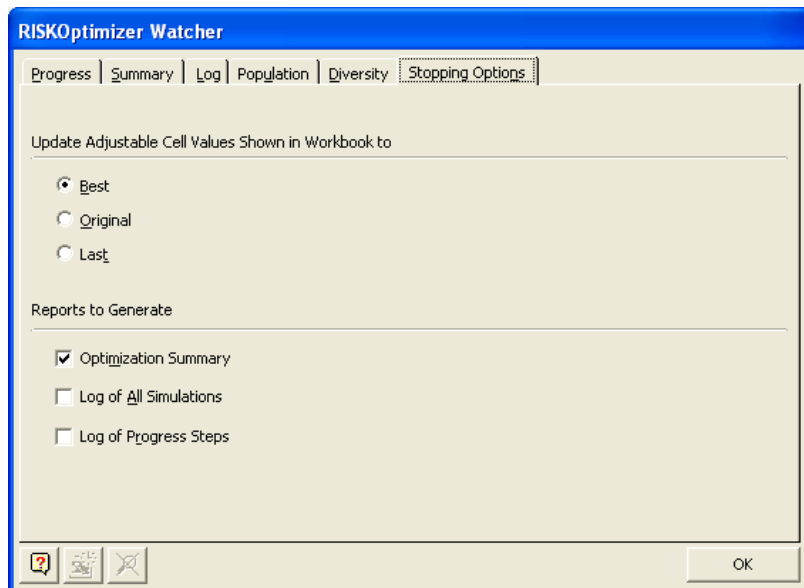


RISKOptimizer Watcher – Stopping Options Tab

Displays stopping options for the optimization

When you click the **Stop** button, the RISKOptimizer Watcher dialog **Stopping Options** tab is displayed. This includes the options available for updating your worksheet with the best calculated values for adjustable cells, restoring original values, and generating an optimization summary report.

Clicking OK destroys RISKOptimizer's population of solutions and places the selected values in your spreadsheet. If you wish to save any information about the RISKOptimizer session, including the population values, the time and number of trials run, be sure to select to create an optimization summary report.



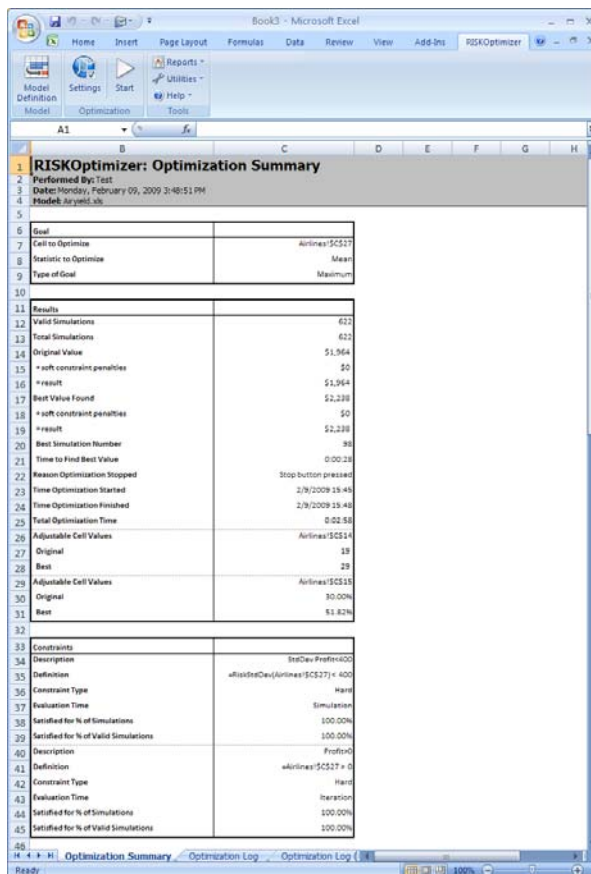
This dialog will also appear if one of the user specified stopping conditions has been met (number of requested trials have been evaluated, minutes requested have elapsed, etc.). The stop alert offers three choices for updating the adjustable cell values in your spreadsheet: **Best, Original and Last.**

- **Best .** This accepts RISKOptimizer's results and ends RISKOptimizer's search for better solutions. When you choose this option, the values of the best scenario RISKOptimizer has found in its search are placed into the adjustable cells of your spreadsheet.

- **Original.** This restores the adjustable cells to their original values before RISKOptimizer was run, and ends RISKOptimizer's search for better solutions.
- **Last.** This causes RISKOptimizer to place the last calculated values in the optimization in the worksheet. The Last Calculated Values option is particularly useful when debugging models.

The Reports to Generate options can generate optimization summary worksheets that can be used for reporting on the results of a run and comparing the results between runs. Report options include:

- **Optimization Summary.** This summary report contains information such as date and time of the run, the optimization settings used, the value calculated for the target cell and the value for each of the adjustable cells.



RISKOptimizer: Optimization Summary		
Performed By: Test		
Date: Monday, February 09, 2009 3:40:51 PM		
Model: Airline.xls		
Goal		
Cell to Optimize	Airline!\$C\$17	
Statistic to Optimize	Mean	
Type of Goal	Maximum	
Results		
Valid Simulations	622	
Total Simulations	622	
Original Value	\$1,964	
+ soft constraint penalties	\$0	
+ result	\$1,964	
Best Value Found	\$2,238	
+ soft constraint penalties	\$0	
+ result	\$2,238	
Best Simulation Number	98	
Time to Find Best Value	0:00:28	
Reason Optimization Stopped	Stop button pressed	
Time Optimization Started	2/9/2009 3:40:51 PM	
Time Optimization Finished	2/9/2009 3:41:19 PM	
Total Optimization Time	0:00:28	
Adjustable Cell Values		
Original	Airline!\$C\$14	
Best	38	
Original	28	
Adjustable Cell Values	Airline!\$C\$18	
Original	30.00%	
Best	61.82%	
Constraints		
Description	StdDev Profit <= 400	
Definition	=STDEV(Airline!\$C\$17) <= 400	
Constraint Type	Hard	
Evaluation Time	Simulation	
Satisfied for % of Simulations	100.00%	
Satisfied for % of Valid Simulations	100.00%	
Description	Profit >= 0	
Definition	=Airline!\$C\$17 >= 0	
Constraint Type	Hard	
Evaluation Time	Iteration	
Satisfied for % of Simulations	100.00%	
Satisfied for % of Valid Simulations	100.00%	

This report is useful for comparing the results of successive optimizations.

- **Log of All Simulations.** This report logs the results of all trial simulations performed.

Simulation	Elapsed Time	Iterations	Result	Goal Cell Statistics				Adjustable Cell		Hard Constraints	
				Mean	Std. Dev.	Min.	Max.	C14	C15	StdDev Profit<=400	Profit>=0
1	0:00:00	500	\$1,964	\$1,964	\$156	\$1,230	\$2,045	19	30.00%	Met	Met
2	0:00:02	500	\$1,790	\$1,790	\$316	\$1,055	\$2,375	30	24.40%	Met	Met
3	0:00:02	500	\$1,910	\$1,910	\$193	\$1,230	\$2,265	26	21.62%	Met	Met
4	0:00:02	500	\$2,037	\$2,037	\$302	\$1,205	\$2,570	20	47.60%	Met	Met
5	0:00:03	500	\$2,185	\$2,185	\$187	\$1,290	\$2,550	25	34.45%	Met	Met
6	0:00:03	500	\$2,016	\$2,016	\$382	\$1,120	\$2,875	22	59.54%	Met	Met
7	0:00:03	500	\$1,553	\$1,553	\$394	\$645	\$2,595	27	88.06%	Met	Met
8	0:00:03	500	\$1,907	\$1,907	\$339	\$1,035	\$2,595	19	56.14%	Met	Met
9	0:00:04	500	\$1,856	\$1,856	\$392	\$950	\$2,900	29	74.97%	Met	Met
10	0:00:04	500	\$1,298	\$1,298	\$394	\$390	\$2,340	30	99.43%	Met	Met
11	0:00:04	500	\$1,553	\$1,553	\$394	\$645	\$2,595	24	86.57%	Met	Met
12	0:00:05	500	\$1,587	\$1,587	\$302	\$840	\$2,065	29	13.75%	Met	Met
13	0:00:05	500	\$1,705	\$1,705	\$391	\$780	\$2,765	19	71.64%	Met	Met
14	0:00:05	500	\$2,143	\$2,143	\$228	\$1,290	\$2,435	21	36.87%	Met	Met
15	0:00:05	500	\$1,842	\$1,842	\$367	\$950	\$2,510	19	64.95%	Met	Met
16	0:00:06	500	\$1,875	\$1,875	\$213	\$840	\$2,155	21	17.16%	Met	Met
17	0:00:06	500	\$2,153	\$2,153	\$187	\$1,290	\$2,375	23	30.00%	Met	Met
18	0:00:06	500	\$2,101	\$2,101	\$193	\$1,290	\$2,615	28	36.32%	Met	Met
19	0:00:07	500	\$1,842	\$1,842	\$367	\$950	\$2,510	19	64.03%	Met	Met
20	0:00:07	500	\$1,641	\$1,641	\$230	\$645	\$2,045	25	12.39%	Met	Met

- **Log of Progress Steps.** This report logs the results of all trial simulations that improved the result for the target cell.

Simulation	Elapsed Time	Iterations	Result	Goal Cell Statistics				Adjustable Cell		Hard Constraints	
				Mean	Std. Dev.	Min.	Max.	C14	C15	StdDev Profit<=400	Profit>=0
1	0:00:00	500	\$1,964	\$1,964	\$156	\$1,230	\$2,045	19	30.00%	Met	Met
4	0:00:02	500	\$2,037	\$2,037	\$302	\$1,205	\$2,570	20	47.60%	Met	Met
5	0:00:03	500	\$2,185	\$2,185	\$187	\$1,290	\$2,550	25	34.45%	Met	Met
26	0:00:08	500	\$2,203	\$2,203	\$318	\$1,290	\$2,830	25	51.82%	Met	Met
34	0:00:11	500	\$2,237	\$2,237	\$244	\$1,290	\$2,800	27	49.08%	Met	Met
98	0:00:28	500	\$2,238	\$2,238	\$246	\$1,290	\$2,895	29	51.82%	Met	Met

Chapter 6: Optimization

Introduction	143
Optimization Methods.....	143
About Hill Climbing Algorithms.....	145
Excel Solver	148
Types of Problems	149
Linear Problems	149
Non-linear Problems.....	149
Table-based problems	151
Combinatorial problems	152

Introduction

RISKOptimizer combines optimization and simulation to allow you to optimize problems that have uncertain elements. The following three chapters of this manual provide background information on the analytical techniques used in RISKOptimizer, including 1) optimization, 2) genetic algorithms and 3) simulation.

Optimization Methods

Traditional Excel-based optimization problems analyzed using Solver or Evolver (optimization add-ins to Excel) are comprised of:

- An output or “target” cell that you wish to minimize or maximize
- A set of input or “adjustable cells” whose values you control
- A set of constraints that need to be met, often specified using expressions such as $COSTS < 100$ or $A11 \geq 0$.

During an optimization in Solver or Evolver, the adjustable cells are changed across allowable ranges you specify. For each possible set of adjustable cell values the model is recalculated and a new value for the target cell is generated. When the optimization is complete, an optimal solution (or combination of adjustable cell values) is found. This solution is the combination of adjustable cell values which yields the best (i.e., minimum or maximum) value for the target cell while satisfying the constraints you’ve entered.

Some optimization problems are much harder than others to solve. For tough problems, such as an Excel model for finding the shortest route between 1000 cities, it is not feasible to examine every possible solution. Such an approach would require years of calculations on the fastest computers.

To solve such problems, it is necessary to search through a subset of all possible solutions. By examining these solutions, we can get an idea of how to find better solutions. This is accomplished with an *algorithm*. An algorithm is simply a step-by-step description of how to approach a problem. All computer programs, for example, are built by combining numerous algorithms.

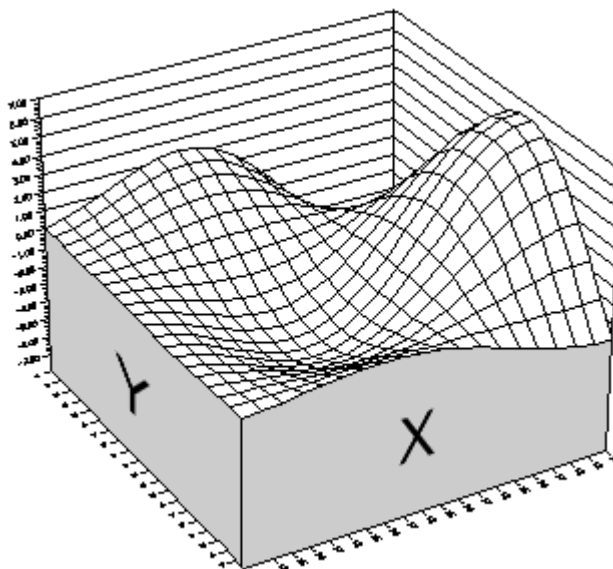
Let us start by exploring how most problem-solving algorithms represent a problem. Most problems can be divided into three basic components: inputs, a function of some kind, and a resulting output.

	Looking for:	Given this:	To get the best:
Problem Components	Inputs	Function	Output
Optimization in Excel	Variables	Model	Goal

Let us assume that our optimization problem involves two variables, X and Y. When placed in an equation, these two variables produce a result =Z. Our problem is to find the value for X and Y that produces the largest Z value. We can think of Z as a “rating”, which indicates how good any particular X,Y pairing is.

	Looking for:	Given this:	To get the best:
In this example	X and Y	Equation	Z

A plot of every single set of Xs,Ys, and the resulting Zs would produce a three-dimensional surface graph such as the one shown below.



A “landscape” of possible scenarios or solutions.

Each intersection of an X and Y value produces a Z height. The peaks and valleys of this “landscape” represent good and bad solutions

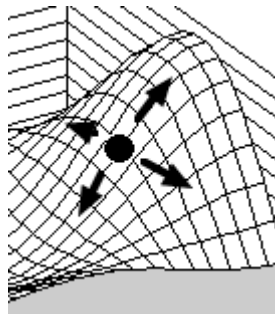
respectively. Searching for the maximum or highest point on this function by examining each solution would take far too much time, even with a powerful computer and the fastest program.* Remember that we are giving Excel just the function itself, not a graph of the function, and that we could just as easily be dealing with a 200-dimensional problem as with this two-dimensional problem. Thus, we need a method that will let us do fewer calculations and still find the maximum productivity.

About Hill Climbing Algorithms

Let us look at a simple algorithm called hill-climbing. Hill-climbing is an algorithm that works like this:

- 1) *Start at a random point on the landscape (take a random guess).*
- 2) *Walk a small distance in some arbitrary direction.*
- 3) *If you have walked to a new point that is higher, stay and repeat step 2. If your new point is lower, go back to your original point and try again.*

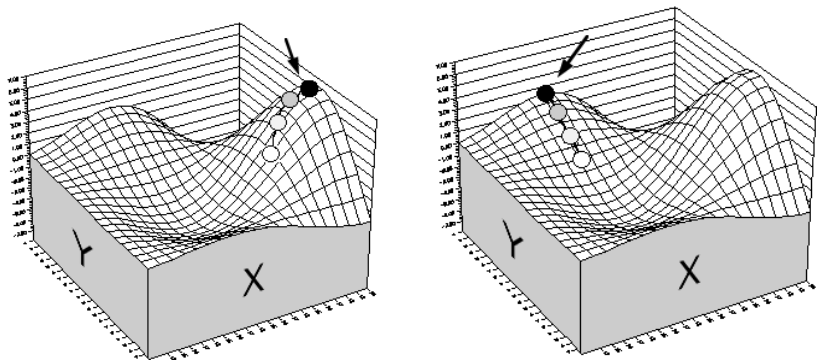
Hill-climbing tries only one solution or scenario at a time. We will use a black dot (•) to represent one possible solution (a set of X, Y and Z values). If we place the dot at the random starting point, we hope that our hill-climbing method will bring the dot to the highest point on the graph.



From the diagram above we can clearly see that we want the dot to go up the high hill to the right. However, we only know that because we have already seen the entire landscape. As the algorithm runs, it sees the landscape immediately around it, but not the entire landscape; *it sees the trees but not the forest.*

* In our diagram, the function is shown as a smooth landscape. In the rare cases where we deal with simple, smooth (differentiable) functions, it is possible to use calculus to find minima and maxima. However, most realistic problems are not described by such smooth functions.

In most real-world problems, the landscape is not so smooth, and would require years to calculate, so we only calculate the current scenario and the immediately surrounding scenarios. Imagine that the dot is a blindfolded man standing amidst smooth, rolling hills. If the man employed the hill-climbing algorithm, this man would put one foot in each direction, and only move when he felt higher ground. This man would successfully step his way upwards, and eventually would come to rest on the hilltop where the ground all around him was lower than the ground he was on. This seems simple enough. However, we get into a very serious problem if the man starts out in another place... he climbs up the wrong hill! (see the diagram below).



Even with a smooth function, hill climbing can fail if you start from a slightly different position (right).

Hill-climbing only finds the nearest hilltop, or *local maximum*. Thus, if your problem has a very rough and hilly solution landscape, as most realistic models do, hill-climbing is not likely to find the highest hill, or even one of the highest.

Hill-climbing has another problem; how do we actually find the terrain around our current location? If the landscape is described by a smooth function, it may be possible to use differentiation (a calculus technique) to find out which direction has the steepest slope. If the landscape is discontinuous or not differentiable (as is more likely in real-world problems), we need to calculate the “fitness” of surrounding scenarios.

For example, let's say a bank hires one security guard from 9:00am to 5:00pm to guard the bank, but the bank must give the officer two (2) half-hour breaks. We must try to find the optimum break times, given general rules about performance/fatigue ratios, and considering the different levels of customer activity throughout the day. We may start by trying out different combinations of duty breaks and evaluate them. If we currently use a schedule where the breaks are timed at 11:00am and 3:00pm, we might calculate the productivity of the surrounding scenarios:

Direction	Break 1 (x)	Break 2 (y)	"Score" (z)
Current Solution	11:00am	3:00pm	= 46.5
West Scenario	10:45am	3:00pm	= 44.67
East Scenario	11:15am	3:00pm	= 40.08
North Scenario	11:00am	3:15pm	= 49.227
South Scenario	11:00am	2:45pm	= 43.97

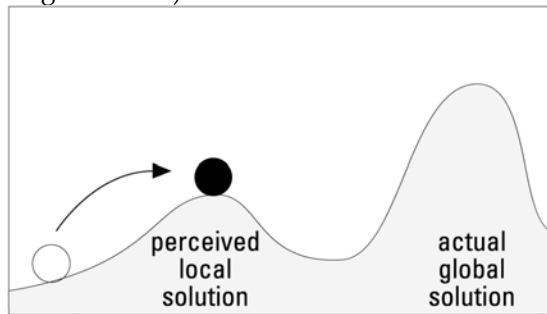
If we had three adjustable cells (breaks) instead of two, we would need to look at eight different directions. In fact, if we had just fifty variables, (quite realistic for a medium-sized problem), we would need to calculate productivity for 2^{50} , or over one quadrillion scenarios, just for one guard!!

There are modifications that can be made to hill-climbing to improve its ability to find global maxima (the highest hills on the entire landscape). Hill-climbing is most useful for dealing with unimodal (one-peak) problems, and that is why some analysis programs use the technique. Nevertheless, it is very limited for complex and/or large problems.

Excel Solver

Excel includes an optimization utility called **Solver**. Solver can solve two kinds of problems: linear problems and simple non-linear problems. It solves linear problems using a linear programming routine. This classic mathematical technique is often called the Simplex method, and it will always find perfect answers to small, purely linear problems.

Like most other *baby solvers*, the Microsoft Solver also solves non-linear problems, using a *hill climbing* routine (specifically, the GRG2 routine). A hill climbing routine starts with the current variable values and slowly adjusts them until the output of the model does not improve anymore. This means that problems with more than one possible solution may be impossible for Solver to solve well, because Solver ends up at a *local* solution and cannot jump over to the *global* solution (see figure below).



Landscape of possible solutions.

In addition, Solver requires that the function represented by your model be continuous. This means the output should change smoothly as the inputs are adjusted. If your model uses lookup tables, acquires noisy, real-time data from another program, contains random elements, or involves if-then rules, your model will be jumpy and discontinuous. Solver would not be able to solve such a problem.

Solver also puts a limit on the number of variables and the number of constraints in your problem (200) above which you must turn to a more powerful technique.

Types of Problems

Several different types of problems are typically optimized.

Linear Problems

In linear problems, all the outputs are simple linear functions of the inputs, as in $y=mx+b$. When problems only use simple arithmetic operations such as addition, subtraction, and Excel functions such as TREND() and FORECAST() it indicates there are purely linear relationships between the variables.

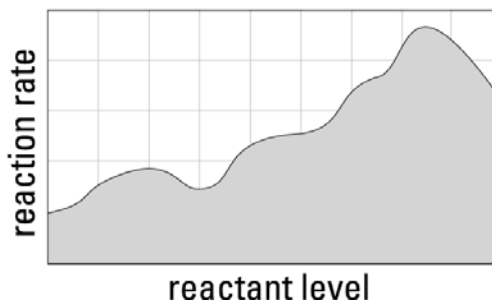
Linear problems have been fairly easy to solve since the advent of computers and the invention by George Dantzig of the Simplex Method. A simple linear problem can be solved most quickly and accurately with a linear programming utility. The Solver utility included with Excel becomes a linear programming tool when you set the “Assume Linear Model” checkbox.* Solver then uses a linear programming routine to quickly find the perfect solution. If your problem can be expressed in purely linear terms, you should use linear programming. Unfortunately, most real-world problems cannot be described linearly.

Non-linear Problems

If the cost to manufacture and ship out 5,000 widgets was \$5,000, would it cost \$1 to manufacture and ship 1 widget? Probably not. The assembly line in the widget factory would still consume energy, the paperwork would still need to be filled out and processed through the various departments, the materials would still be bought in bulk, the trucks would require the same amount of gas to deliver the widgets, and the truck driver would still get paid a full day’s salary no matter how full the load was. Most real-world problems do not involve variables with simple linear relationships. These problems involve multiplication, division, exponents, and built-in Excel functions such as SQRT() and GROWTH(). Whenever the variables share a disproportional relationship to one another, the problem becomes non-linear.

* For more specifics on Microsoft’s Solver utility, see the Excel User’s Guide.

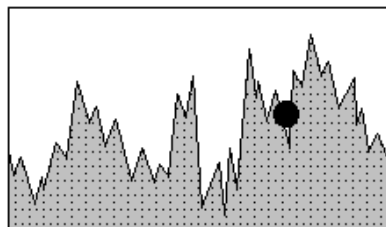
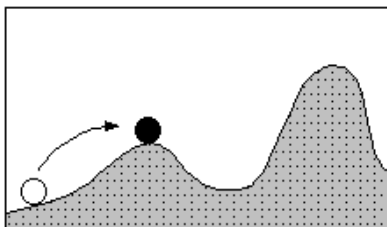
A perfect example of a non-linear problem is the management of a manufacturing process at a chemical plant. Imagine that we want to mix some chemical reactants together and get a chemical product as the result. The rate of this reaction may vary non-linearly with the amount of reactants available; at some point the catalyst becomes saturated and excess reactant just gets in the way. The following diagram shows this relationship:



If we simply need to find the minimum level of reactants that will give us the highest rate of reaction, we can just start anywhere on the graph and climb along the curve until we reach the top. This method of finding an answer is called hill climbing.

Hill climbing will always find the best answer if a) the function being explored is smooth, and b) the initial variable values place you on the side of the highest mountain. If either condition is not met, hill climbing can end up in a local solution, rather than the global solution.

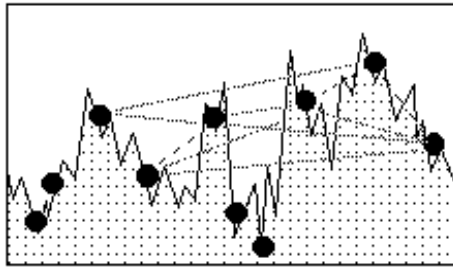
Highly non-linear problems, the kind often seen in practice, have many possible solutions across a complicated landscape. If a problem has many variables, and/or if the formulas involved are very noisy or curvy, the best answer will probably not be found with hill climbing, even after trying hundreds of times with different starting points. Most likely, a sub-optimal, and extremely local solution will be found (see figure below).



Hill climbing finds the local, but not global maximum.

Noisy data: Hill climbing not effective, even with multiple tries.

RISKOptimizer and Evolver (the source of the genetic algorithm-based optimizer used in RISKOptimizer) do not use hill climbing. Rather, they use a stochastic, directed search technique, dubbed a genetic algorithm. This lets RISKOptimizer jump around in the solution space of a problem, examining many combinations of input values without getting stuck in local optima. In addition, RISKOptimizer lets good scenarios “communicate” with each other to gain valuable information as to what the overall solution landscape looks like, and then uses that information to better guess which scenarios are likely to be successful.



RISKOptimizer generates many possible scenarios, then refines the search based on the feedback it receives.

Table-based problems

Many problems require the use of lookup tables and databases. For example, in choosing the quantities of different materials to buy, you might need to look up the prices charged for different quantities.

Tables and databases make problems discontinuous (non-smooth). That makes it difficult for hill-climbing routines like Solver to find optimal solutions. RISKOptimizer, however, does not require continuity in the functions it evaluates, and it can find good solutions for table-based problems, even problems that use many large, interrelated tables.

If your problem involves looking up values from a database, or a table of data in Excel, where the index of the table item is a variable or a function of a variable, you need to use Evolver or RISKOptimizer. If you only look up a single, constant item in a table (the same record is retrieved from the table regardless of the input variables' values), then you are really only dealing with a constant, and you can probably use Solver effectively.

Combinatorial problems

There is a large class of problems that are very different from the numerical problems examined so far. Problems where the outputs involve changing the order of existing input variables, or grouping subsets of the inputs are called combinatorial problems. These problems are usually very hard to solve, because they often require exponential time; that is, the amount of time needed to solve a problem with 4 variables might be $4 \times 3 \times 2 \times 1$, and doubling the number of variables to 8 raises the solving time to $8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$, or a factor of 1,680. The number of variables doubles, but the number of possible solutions that must be checked increases 1,680 times. For example, choosing the starting lineup for a baseball team is a combinatorial problem. For 9 players, you can choose one out of the 9 as the first batter. You can then choose one out of the remaining 8 as the second batter, one of the remaining 7 will be the third, and so on. There are thus $9 \times 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$ (9 factorial) ways to choose a lineup of 9 players. This is about **362,880** different orderings. Now if you double the number of players, there are 18 factorial possible lineups, or **6,402,373,705,000,000** possible lineups!

RISKOptimizer and Evolver's genetic algorithm intelligently searches through the possible permutations. This is much more practical than searching through *all* possibilities, and it is much more efficient than examining purely random permutations; sub-orders from good scenarios can be retained and used to create even better scenarios.

Chapter 7: Genetic Algorithms

Introduction	155
History.....	155
A Biological Example	158
A Digital Example	159

Introduction

RISKOptimizer uses genetic algorithms to search for optimal answers for simulation models. The genetic algorithms used are adapted from Evolver, an optimization add-in to Excel from Palisade Corporation. This chapter provides background information on genetic algorithms to give insights on how they are used for optimizing simulation models.

History

The first genetic algorithms were developed in the early 1970s by John Holland at the University of Michigan. Holland was impressed by the ease in which biological systems could perform tasks which eluded even the most powerful super-computers; animals can flawlessly recognize objects, understand and translate sounds, and generally navigate through a dynamic environment almost instantaneously.

For decades, scientists have promised to replicate these capabilities in machines, but we are beginning to recognize just how difficult this task is. Most scientists agree that any complex biological system that exhibits these qualities has evolved to get that way.

Evolution, so the theory goes, has produced systems with amazing capabilities through relatively simple, self-replicating building blocks that follow a few simple rules:

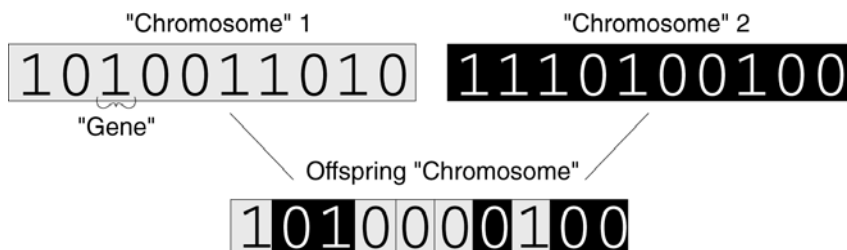
- 1) Evolution takes place at the level of the chromosome. The organism doesn't evolve, but only serves as the vessel in which the genes are carried and passed along. It is the chromosomes which are dynamically changing with each re-arrangement of genes.
- 2) Nature tends to make more copies of chromosomes which produce a more "fit" organism. If an organism survives long enough, and is healthy, its genes are more likely to be passed along to a new generation of organisms through reproduction. This principle is often referred to as "survival of the fittest". Remember that "fittest" is a relative term; an organism only needs to be fit in comparison to others in the current population to be "successful".
- 3) Diversity must be maintained in the population. Seemingly random mutations occur frequently in nature that ensure variation in the organisms. These genetic mutations often result in a useful, or even vital feature for a species' survival. With a wider spectrum of possible combinations, a population is also less susceptible to a

Evolution Theory

common weakness that could destroy them all (virus, etc.) or other problems associated with inbreeding.

Once we break down evolution into these fundamental building blocks, it becomes easier to apply these techniques to the computational world, and truly begin to move towards more fluid, more naturally behaving machines.

Holland began applying these properties of evolution to simple strings of numbers that represented chromosomes. He first encoded his problem into binary strings (rows of “1s” and “0s”) to represent the chromosomes, and then had the computer generate many of these “bit” strings to form a whole population of them. A fitness function was programmed that could evaluate and rank each bit string, and those strings which were deemed most “fit” would exchange data with others through a “crossover” routine to create “offspring” bit strings. Holland even subjected his digital chromosomes to a “mutation” operator, which injected randomness into the resulting “offspring” chromosomes to retain diversity in the population. This fitness function replaced the role of death in the biological world; determining which strings were good enough to continue breeding and which would no longer be kept in memory.



The program kept a given number of these “chromosomes” in memory, and this entire “population” of strings continued to evolve until they maximized the fitness function. The result was then decoded back to its original values to reveal the solution. John Holland remains an active pioneer in this field, and is now joined by hundreds of scientists and scholars who have devoted the majority of their time toward this promising alternative to traditional linear programming, mathematical, and statistical techniques.

Holland’s original genetic algorithm was quite simple, yet remarkably robust, finding optimal solutions to a wide variety of problems. Many custom programs today solve very large and complex real-world problems using only slightly modified versions of this original genetic algorithm.

**Modern
Adaptations of
Genetic
Algorithms**

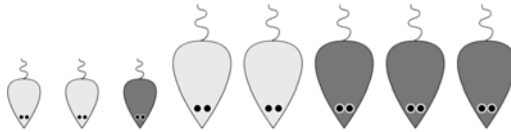
As interest swelled in academic circles, as serious computational power began moving its way into mainstream desktop machines, standards like Microsoft Windows and Excel made design and maintenance of complex models easier. The use of real numbers rather than bit string representations eliminated the difficult task of encoding and decoding chromosomes.

The popularity of the genetic algorithm is now growing exponentially, with seminars, books, magazine articles, and knowledgeable consultants popping up everywhere. The International Conference of Genetic Algorithms is already focusing on practical applications, a sign of maturity that eludes other “artificial intelligence” technologies. Many Fortune 500 companies employ genetic algorithms regularly to solve real-world problems, from brokerage firms to power plants, phone companies, restaurant chains, automobile manufacturers and television networks. In fact, there is a good chance that you have already indirectly used a genetic algorithm before!

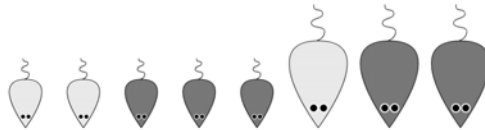
A Biological Example

Let us look at a simple example of evolution in the biological world (on a small scale). By “evolution” here we mean any change in the distribution or frequency of genes in a population. Of course, the interesting thing about evolution is that it tends to lead to populations that are constantly adapting to their environments.

Imagine that we are looking at a population of mice. These mice exhibit two sizes, small and large, and they exhibit two colors, light or dark. Our population consists of the following eight mice:



One day, cats move into the neighborhood and start eating mice. It turns out that darker mice and smaller mice are harder for the cats to find. Thus, different mice have different odds of avoiding the cats long enough to reproduce. This affects the nature of the next generation of mice. Assuming the old mice die soon after reproducing, the next generation of mice looks like this:



Notice that large mice, light mice, and especially large, light mice, are having trouble surviving long enough to reproduce. This continues in the next generation.



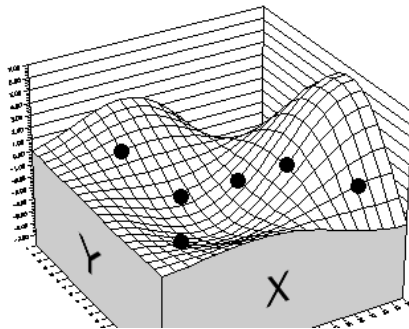
Now the population consists mostly of small, dark mice, because these mice are better suited for survival in this environment than other kinds of mice. Similarly, as the cats begin to go hungry with less mice to eat, perhaps those cats who prefer a lunch of grass will be better adapted, and pass along their grass-loving gene to a new

generation of cats. This is the central concept of “survival of the fittest”. More precisely, it could be phrased “survival until reproduction”. In evolutionary terms, being the healthiest bachelor in the population is worthless, since you must reproduce in order for your genes to influence future generations.

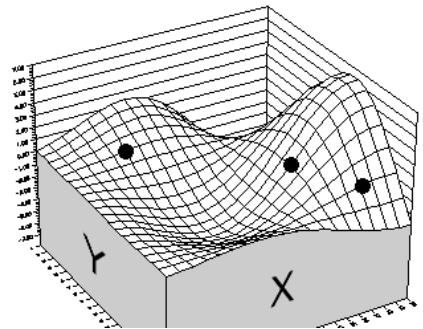
A Digital Example

Imagine a problem with two variables, X and Y , that produce a result Z . If we calculated and plotted the resulting Z for every possible X and Y values, we would see a solution “landscape” emerge (discussed also in [Chapter 6: Optimization](#)). Since we are trying to find the maximum “ Z ”, the peaks of the function are “good” solutions, and the valleys are “bad” ones.

When we use a genetic algorithm to maximize our function, we start by creating several possible solutions or scenarios at random (the black dots), rather than just one starting point. We then calculate the function’s output for each scenario and plot each scenario as one dot. Next we rank all of the scenarios by altitude, from best to worst. We keep the scenarios from the top half, and throw out the others.



First, create a whole “population” of possible solutions. Some will be better (higher) than others.

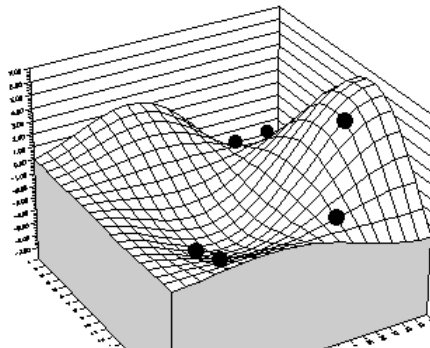


Next we rank them all and keep the solutions which yield better results.

Each of the three remaining scenarios duplicates itself, bringing the number of scenarios back up to six. Now comes the interesting part: Each of the six scenarios is made up of two adjustable values (plotted as an X and a Y coordinate). The scenarios pair off with each other at random. Now each scenario exchanges the first of its two adjustable values with the corresponding value from its partner. For example:

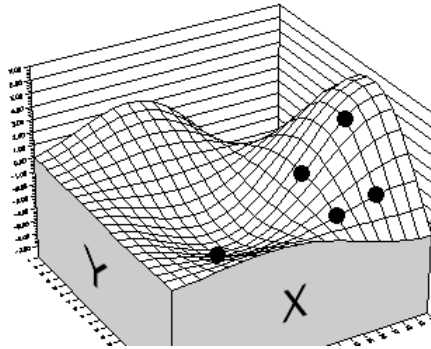
	Before	After
Scenario 1	3.4, 5.0	2.6, 5.0
Scenario 2	2.6, 3.2	3.4, 3.2

This operation is called crossing over, or *crossover*. When our six scenarios randomly mate and perform crossover, we may get a new set of scenarios such as this:



In the above example, we assume that the original three scenarios, a, b, and c, paired up with the duplicates, A, B, C, to form the pairs aB, bC, bA. These pairs then switched values for the first adjustable cell, which is equivalent in our diagram to exchanging the x and y coordinates between pairs of dots. The population of scenarios has just lived through a generation, with its cycle of “death” and “birth”.

Notice that some of the new scenarios result in lower output (lower altitude) than any we saw in the original generation. However, one scenario has moved high up on the tallest hill, indicating progress. If we let the population evolve for another generation, we may see a scene like the following:



You can see how the average performance of the population of scenarios increases over the last generation. In this example, there is not much room left for improvement. This is because there are only two genes per organism, only six organisms, and no way for new genes to be created. This means there is a limited **gene pool**. The gene pool is the sum of all the genes of all organisms in the population.

Genetic algorithms can be made much more powerful by replicating more of the inherent strength of evolution in the biological world; increasing the number of genes per organism, increasing the number of organisms in a population, and allowing for occasional, random mutations. In addition, we can choose the scenarios that will live and reproduce more like they occur naturally: with a random element that has a slight bias towards those that perform better, instead of simply choosing the best performers to breed (even the biggest and strongest lion may get hit with lightning)!

All of these techniques stimulate genetic refinement, and help to maintain diversity in the gene pool, keeping all kinds of genes available in case they turn out to be useful in different combinations. RISKOptimizer automatically implements all of these techniques.

Chapter 8: Simulation and Risk Analysis

Introduction	165
What is Risk?	165
Characteristics of Risk.....	166
The Need for Risk Analysis	167
Assessing and Quantifying Risk.....	168
Describing Risk with a Probability Distribution	169
Modeling Uncertainty in RISKOptimizer.....	171
Variables.....	171
Certain or Uncertain.....	171
Independent or Dependent	172
Analyzing a Model with Simulation.....	173
Simulation	173
How Simulation Works	174

Introduction

RISKOptimizer uses simulation to handle the uncertainty present in the Excel models it optimizes. Both the methods for 1) modeling the uncertainty present in a spreadsheet and 2) running a simulation on the model are taken from @RISK, a simulation and risk analysis add-in to Excel from Palisade Corporation. This chapter provides background information on risk and simulation to give insights on how simulation models are set up in RISKOptimizer.

What is Risk?

Everyone knows that "risk" affects the gambler about to roll the dice, the wildcatter about to drill an oil well, or the tightrope walker taking that first big step. But these simple illustrations aside, the concept of risk comes about due to our recognition of future uncertainty — our inability to know what the future will bring in response to a given action today. Risk implies that a given action has more than one possible outcome.

In this simple sense, every action is "risky", from crossing the street to building a dam. The term is usually reserved, however, for situations where the range of possible outcomes to a given action is in some way significant. Common actions like crossing the street usually aren't risky while building a dam can involve significant risk. Somewhere in between, actions pass from being non-risky to risky. This distinction, although vague, is important — if you judge that a situation is risky, risk becomes one criterion for deciding what course of action you should pursue. At that point, some form of Risk Analysis becomes viable.

Characteristics of Risk

Risk derives from our inability to see into the future, and indicates a degree of uncertainty that is significant enough to make us notice it. This somewhat vague definition takes more shape by mentioning several important characteristics of risk.

First, risk can be either objective or subjective. Flipping a coin is an objective risk because the odds are well known. Even though the outcome is uncertain, an objective risk can be described precisely based on theory, experiment, or common sense. Everyone agrees with the description of an objective risk. Describing the odds for rain next Thursday is not so clear cut, and represents a subjective risk. Given the same information, theory, computers, etc., weatherman A may think the odds of rain are 30% while weatherman B may think the odds are 65%. Neither is wrong. Describing a subjective risk is open-ended in the sense that you could always refine your assessment with new information, further study, or by giving weight to the opinion of others. Most risks are subjective, and this has important implications for anyone analyzing risk or making decisions based on a Risk Analysis.

Second, deciding that something is risky requires personal judgment, even for objective risks. For example, imagine flipping a coin where you win \$1 for a heads and lose \$1 for a tails. The range between \$1 and -\$1 would not be overly significant to most people. If the stakes were \$100,000 and -\$100,000 respectively, most people would find the situation to be quite risky. There would be a wealthy few, however, who would not find this range of outcomes to be significant.

Third, risky actions and therefore risk are things that we often can choose or avoid. Individuals differ in the amount of risk they willingly accept. For example, two individuals of equal net worth may react quite differently to the \$100,000 coin flip bet described above — one may accept it while the other refuses it. Their personal preference for risk differs.

The Need for Risk Analysis

The first step in Risk Analysis and modeling is recognizing a need for it. Is there significant risk involved in the situation you are interested in? Here are a few examples that might help you evaluate your own situations for the presence of significant risk:

- ◆ ***Risks for New Product Development and Marketing*** — Will the R&D department solve the technical problems involved? Will a competitor get to market first, or with a better product? How much impact will the proposed advertising campaign have on sales levels? Will production costs be as forecast? Will the proposed sales price have to be changed to reflect unanticipated demand levels for the product?
- ◆ ***Risks for Securities Analysis and Asset Management*** — How will a tentative purchase affect portfolio value? Will a new management team affect market price? Will an acquired firm add earnings as forecast? How will a market correction impact a given industry sector?
- ◆ ***Risks for Operations Management and Planning*** — Will a given inventory level suffice for unpredictable demand levels? Will labor costs rise significantly with upcoming union contract negotiations? How will pending environmental legislation impact production costs? How will political and market events affect overseas suppliers in terms of exchange rates, trade barriers, and delivery schedules?
- ◆ ***Risks for Design and Construction of a Structure (building, bridge, dam,...)*** — Will the cost of construction materials and labor be as forecast? Will a labor strike affect the construction schedule? Will the levels of stress placed on the structure by peak load crowds and nature be as forecast? Will the structure ever be stressed to the point of failure?
- ◆ ***Risks for Investment in Exploration for Oil and Minerals*** — Will anything be found? If a deposit is found, will it be uneconomical, or a bonanza? Will the costs of developing the deposit be as forecast? Will some political event like an embargo, tax reform, or new environmental regulations drastically alter the economic viability of the project?
- ◆ ***Risks for Policy Planning*** — If the policy is subject to legislative approval, will it be approved? Will the level of compliance with any policy directives be complete or partial? Will the costs of implementation be as forecast? Will the level of benefits be as projected?

Assessing and Quantifying Risk

Realizing that you have a risky situation is only the first step. How do you quantify the risk you have identified for a given uncertain situation? "Quantifying risk" means determining all the possible values a risky variable could take and determining the relative likelihood of each value. Suppose your uncertain situation is the outcome from the flip of a coin. You could repeat the flip a large number of times until you had established the fact that half of the times it comes up tails and half of the times heads. Alternatively, you could mathematically calculate this result from a basic understanding of probability and statistics.

In most real life situations, you can't perform an "experiment" to calculate your risk the way you can for the flip of a coin. How could you calculate the probable learning curve associated with introducing new equipment? You may be able to reflect on past experiences, but once you have introduced the equipment, the uncertainty is gone. There is no mathematical formula that you can solve to get the risk associated with the possible outcomes. You have to estimate the risk using the best information you have available.

If you can calculate the risks of your situation the way you would for a coin flip, the risk is objective. This means that everyone would agree that you quantified the risk correctly. Most risk quantification, however, involves your best judgment. There may not be complete information available about the situation, the situation may not be repeatable like a coin flip, or it just may be too complex to come up with an unequivocal answer. Such risk quantification is subjective, which means that someone might disagree with your evaluation.

Your subjective assessments of risk are likely to change when you get more information on the situation. If you have subjectively derived a risk assessment, you must always ask yourself whether additional information is available that would help you make a better assessment. If it is available, how hard and how expensive would it be to obtain? How much would it cause you to change the assessment you already have made? How much would these changes affect the final results of any model you are analyzing?

Describing Risk with a Probability Distribution

If you have quantified risk — determined outcomes and probabilities of occurrence — you can summarize this risk using a probability distribution. A probability distribution is a device for presenting the quantified risk for a variable. RISKOptimizer and @RISK (the simulation add-in to Excel used by RISKOptimizer) use probability distributions to describe uncertain values in your Excel worksheets and to present results. There are many forms and types of probability distributions, each of which describes a range of possible values and their likelihood of occurrence. Most people have heard of a normal distribution — the traditional "bell curve". But there are a wide variety of distribution types ranging from uniform and triangular distributions to more complex forms such as gamma and Weibull.

All distribution types use a set of arguments to specify a range of actual values and distribution of probabilities. The normal distribution, for example, uses a mean and standard deviation as its arguments. The mean defines the value around which the bell curve will be centered and the standard deviation defines the range of values around the mean. Over thirty types of distributions are available to you in RISKOptimizer for describing distributions for uncertain values in your Excel worksheets.

Modeling Uncertainty in RISKOptimizer

You are the "expert" at understanding the problems and situations that you would like to analyze. If you have a problem that is subject to risk, then RISKOptimizer and Excel can help you construct a complete and logical model.

A major strength of RISKOptimizer is that it allows you to work in a familiar and standard model building environment — Microsoft Excel. RISKOptimizer works with your Excel model, allowing you to model uncertainty, but still preserves the familiar spreadsheet capabilities. You presumably know how to build spreadsheet models in Excel — RISKOptimizer now gives you the ability to easily modify these models to include uncertainty.

Variables

Variables are the basic elements in your Excel worksheets that you have identified as being important ingredients to your analysis. If you are modeling a financial situation, your variables might be things like Sales, Costs, Revenues or Profits whereas if you are modeling a geologic situation, your variables might be things like Depth to Deposit, Thickness of Coal Seam or Porosity. Each situation has its own variables, identified by you. In a typical worksheet, a variable labels a worksheet row or column, for example:

Certain or Uncertain

You may know the values your variables will take in the time frame of your model — they are certain or what statisticians call "deterministic". Conversely, you may not know the values they will take — they are uncertain, or "stochastic". If your variables are uncertain you will need to describe the nature of their uncertainty. This is done with probability distributions, which give both the range of values that the variable could take (minimum to maximum), and the likelihood of occurrence of each value within the range. In RISKOptimizer, uncertain variables and cell values are entered as probability distribution functions, for example:

RiskNormal(100,10)

RiskUniform(20,30)

RiskExpon(A1+A2)

RiskTriang(A3/2.01,A4,A5)

These "distribution" functions can be placed in your worksheet cells and formulas just like any other Excel function.

Independent or Dependent

In addition to being certain or uncertain, variables in a model can be either "independent" or "dependent". An independent variable is totally unaffected by any other variable within your model. For example, if you had a financial model evaluating the profitability of an agricultural crop, you might include an uncertain variable called Amount of Rainfall. It is reasonable to assume that other variables in your model such as Crop Price and Fertilizer Cost would have no effect on the amount of rain — Amount of Rainfall is an independent variable.

A dependent variable, in contrast, is determined in full or in part by one or more other variables in your model. For example, a variable called Crop Yield in the above model should be expected to depend on the independent variable Amount of Rainfall. If there's too little or too much rain, then the crop yield is low. If there's an amount of rain that is about normal, then the crop yield would be anywhere from below average to well above average. Maybe there are other variables that affect Crop Yield such as Temperature, Loss to Insects, etc.

When identifying the uncertain values in your Excel worksheet, you have to decide whether your variables are independent or dependent. The DEPC and INDEPC functions in RISKOptimizer are used to identify independent and dependent variables. It is extremely important to correctly recognize dependency relationships between variables or your model might generate nonsensical results. For example, if you ignored the relationship between Amount of Rainfall and Crop Yield, RISKOptimizer might choose a low value for the rainfall at the same time it picked a high value for the crop yield — clearly something nature wouldn't allow.

Analyzing a Model with Simulation

Once you have placed uncertain values in your worksheet cells you have an Excel worksheet that RISKOptimizer can analyze.

Simulation

RISKOptimizer uses simulation, sometimes called Monte Carlo simulation, to account for uncertainty during an optimization. Simulation in this sense refers to a method whereby the distribution of possible outcomes is generated by letting a computer recalculate your worksheet over and over again, each time using different randomly selected sets of values for the probability distributions in your cell values and formulas. In effect, the computer is trying all valid combinations of the values of input variables to simulate all possible outcomes. This is just as if you ran hundreds or thousands of "what-if" analyses on your worksheet, all in one sitting.

What is meant by saying that simulation "tries all valid combinations of the values of input variables"? Suppose you have a model with only two input variables. If there is no uncertainty in these variables, you can identify a single possible value for each variable. These two single values can be combined by your worksheet formulas to calculate the results of interest — also a certain or deterministic value. For example, if the certain input variables are:

Revenues = 100
Costs = 90

then the result

Profits = 10

would be calculated by Excel from

Profits = 100 - 90

There is only one combination of the input variable values, because there is only one value possible for each variable.

Now consider a situation where there is uncertainty in both input variables. For example,

Revenues = 100 or 120
Costs = 90 or 80

gives two values for each input variable. In a simulation, RISKOptimizer would consider all possible combinations of these variable values to calculate possible values for the result, Profits.

There are four combinations:

Profits = Revenues - Costs

10 = 100 - 90

20 = 100 - 80

30 = 120 - 90

40 = 120 - 80

Profits also is an uncertain variable because it is calculated from uncertain variables.

How Simulation Works

In RISKOptimizer , simulation uses two distinct operations:

- ◆ **Selecting sets of values for the probability distribution functions contained in the cells and formulas of your worksheet**
- ◆ **Recalculating the Excel worksheet using the new values**

The selection of values from probability distributions is called sampling and each calculation of the worksheet is called an iteration. RISKOptimizer generates output distributions by consolidating single-valued results from all the iterations.

Chapter 9:

RISKOptimizer Extras

Adding Constraints	177
Range Constraints	178
Hard Constraints - customized	179
Iteration vs. Simulation Constraints.....	180
Soft Constraints	180
Penalty Functions	181
Entering a Penalty Function	181
Viewing the Effects of an Entered Penalty Function	182
Viewing the Penalties Applied.....	182
Entering Soft Constraints In Your Worksheet	183
More Examples of Penalty Functions	184
Using Penalty Functions	184
Multiple Goal Problems	185
Improving Speed.....	187
How RISKOptimizer's Optimization is Implemented	189
Selection	189
Crossover.....	189
Mutation.....	190
Replacement	190
Constraints.....	191

Adding Constraints

Realistic problems often have a number of constraints that must be met while we search for optimal answers. For example, in the tutorial which seeks the transformer design with the lowest cost, one of the constraints is that the transformer must remain cool, radiating no more than 0.16 watts/cm².

A scenario which meets all the constraints in a model is said to be a viable or “valid” solution. Sometimes it is difficult to find viable solutions for a model, much less to find the optimal viable solution. This may be because the problem is very complex, and only has a few viable solutions, or because the problem is over-specified (there are too many constraints, or some constraints conflict with others), and there are no viable solutions.

There are three basic kinds of constraints: *range* constraints, or min-max ranges placed on adjustable cells, *hard* constraints, which must always be met, and *soft* constraints which we would like to be met as much as possible, but which we may be willing to compromise for a big improvement in fitness.

Range Constraints

The simplest hard constraints are the ones that are placed on the variables themselves. By setting a certain range on each variable, we can limit the overall number of possible solutions RISKOptimizer will search through, resulting in a more efficient search. Enter Min and Max values in the Model window's Adjustable Cell Ranges section to tell RISKOptimizer the range of values that are acceptable for each variable.

RISKOptimizer - Model

Optimization Goal: Maximum
 Cell: =G2
 Statistic: Mean

Adjustable Cell Ranges

Minimum	Range	Maximum	Values
0	=B4:E4	5000	Integer

Constraints

Description	Formula	Type
	=G\$13:G\$15 <= \$I\$13:\$I\$15	Hard

Buttons: Add..., Delete, Group, OK, Cancel

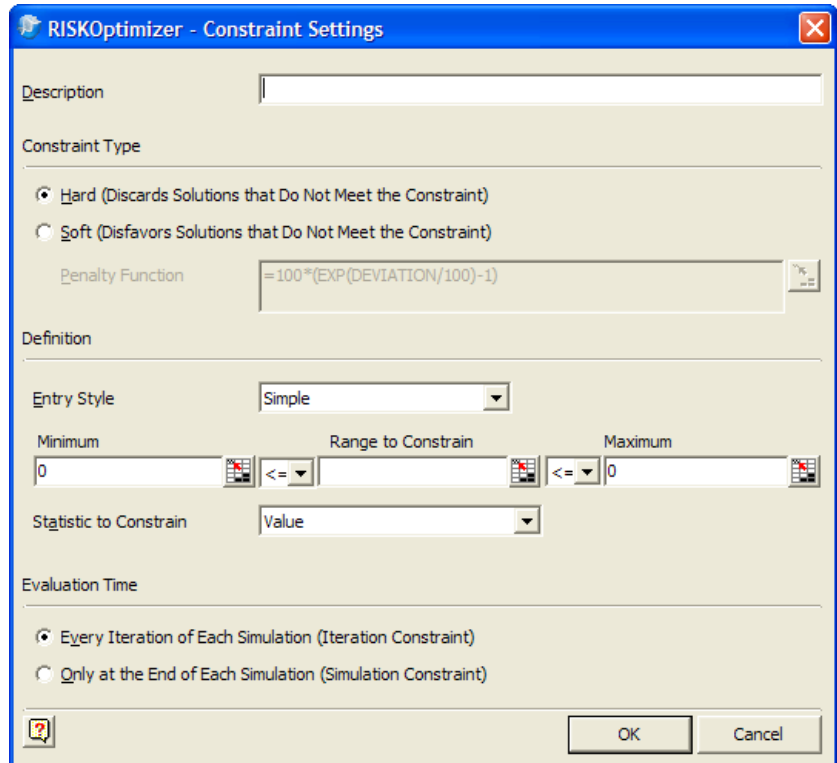
RISKOptimizer will only try values between 0 and 100,000 for the specified cells.

A second type of hard constraint placed on the variables is built in to each of RISKOptimizer's solving methods (recipe, order, grouping, etc.). For example, when we adjust variables using the budget solving method, that means RISKOptimizer is hard constrained to try only sets of values that add up the same amount. Like the Ranges setting, this hard constraint also reduces the number of possible scenarios that must be searched.

The integer option in the Model dialog box is also a hard constraint, telling RISKOptimizer to try only integer values (1, 2, 3 etc.) instead of real numbers (1.34, 2.034, etc.) when adjusting the variable values.

Hard Constraints - customized

Any constraint that falls outside the RISKOptimizer variable constraints can be entered using the Constraint Settings dialog.



The image shows the 'RISKOptimizer - Constraint Settings' dialog box. It has a blue title bar with the RISKOptimizer logo and a close button. The dialog is divided into several sections: 'Description' with a text input field; 'Constraint Type' with two radio buttons, 'Hard (Discards Solutions that Do Not Meet the Constraint)' (selected) and 'Soft (Disfavors Solutions that Do Not Meet the Constraint)'; 'Penalty Function' with a text input field containing the formula '=100*(EXP(DEVIATION/100)-1)' and a help icon; 'Definition' with a dropdown for 'Entry Style' set to 'Simple', and three input fields for 'Minimum' (0), 'Range to Constrain' (with a '<=' dropdown), and 'Maximum' (0), each with a help icon; a dropdown for 'Statistic to Constrain' set to 'Value'; and 'Evaluation Time' with two radio buttons, 'Every Iteration of Each Simulation (Iteration Constraint)' (selected) and 'Only at the End of Each Simulation (Simulation Constraint)'. At the bottom are a help icon, an 'OK' button, and a 'Cancel' button.

NOTE: Like evolution in nature, a genetic algorithm's problem-solving power lies primarily in its ability to freely explore many combinations of likely solutions, and naturally lean towards the best ones. If we forbid RISKOptimizer to even look at solutions that do not meet our demands, the genetic algorithm optimization process can be crippled.

It is always easier for RISKOptimizer to find solutions that meet the hard constraints if the initial scenario in the worksheet does itself meet the constraints. That lets RISKOptimizer know a starting point in the space of valid solutions. If you do not know of a scenario which meets the constraints, run RISKOptimizer with any initial scenario and it will do its best to find scenarios which meet the constraints.

Iteration vs. Simulation Constraints

Hard constraints in *RISKOptimize* may be evaluated **1) each iteration of a simulation run for a trial solution (an “iteration” constraint), or 2) at the end of the simulation run for a trial solution (a “simulation” constraint).**

- ♦ An **iteration constraint** is a constraint that is evaluated each iteration of a simulation run for a given trial solution. If an iteration results in values which violate the hard constraint, the simulation is stopped (and the trial solution rejected) and the next trial solution and its associated simulation begins.
- ♦ A **simulation constraint** is specified in terms of a simulation statistic for a spreadsheet cell; for example the *Mean of A11>1000*. In this case, the constraint is evaluated at the end of a simulation. A simulation constraint, as opposed to an iteration constraint, will never cause a simulation to be stopped prior to completion.

Soft Constraints

Forcing a program to find only solutions that meet all constraints can result in no viable solutions being found. Often, it is more useful to have an approximately viable solution, where maybe a few solutions fall short of meeting the constraints.

An alternative to the use of “hard constraints” that must be met is to reconfigure the problem with “soft constraints”; constraints that *RISKOptimizer* will ***tend to meet***. These soft constraints are often more realistic, and allow *RISKOptimizer* to try many more options. In the case of a highly constrained problem (where there are not very many possible solutions that would meet all your requirements), *RISKOptimizer*’s genetic algorithm will be more likely to find the best solution if it is allowed to get feedback on some solutions that are *close* to satisfying the constraints.

When constraints are design goals, such as “produce twice as many forks as knives”, it is often not so important to meet them exactly: especially if getting a perfectly balanced production schedule required a day-long optimization process. In this case, a good solution to the problem, that *almost* meets the constraint (production is 40% forks, 23% knives, 37% spoons), is usually better than waiting all day to find out that maybe there is no solution, because *all* the constraints could not possibly be met.

Penalty Functions

Soft constraints can easily be implemented in Excel through the use of *penalty functions*. Instead of telling RISKOptimizer that it absolutely cannot use certain values when looking for solutions, we allow those “invalid” values to be explored, but we will penalize such solutions accordingly. For example, your problem may involve finding the most efficient way to distribute goods with the constraint that you use only three trucks. A more accurate model would include a penalty function that allowed you to use more trucks, but added the tremendous cost to the bottom line. Penalty functions can be specified in the Constraint Settings dialog or entered directly in your model by adding formulas to represent the penalty functions.

Entering a Penalty Function

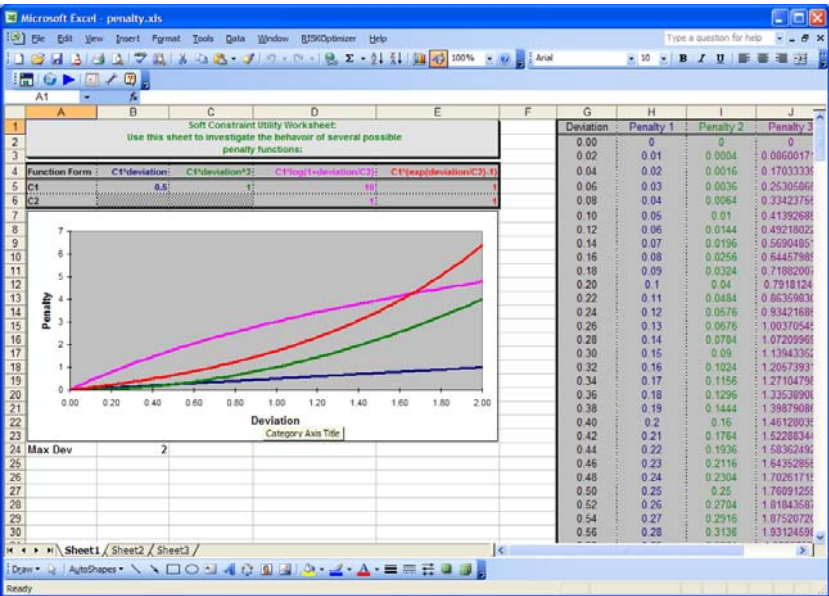
The screenshot shows the "RISKOptimizer - Constraint Settings" dialog box. The "Description" field is "StdDev Profit<400". Under "Constraint Type", the "Soft" option is selected. The "Penalty Function" field contains the formula "100*(EXP(deviation/100)-1)". In the "Definition" section, "Entry Style" is "Simple", "Range to Constrain" is "=C27" with a "<=" operator and a value of "400", and "Statistic to Constrain" is "Standard Deviation".

RISKOptimizer has a default penalty function which is displayed when you first enter a soft constraint. Any valid Excel formula, however, may be entered to calculate the amount of penalty to apply when the soft constraint is not met. An entered penalty function should include the keyword *deviation* which represents the absolute amount by which the constraint has gone beyond its limit. At the end of a trial solution's simulation RISKOptimizer checks if the soft constraint has been met; if not, it places the amount of deviation in the entered penalty formula and then calculates the amount of penalty to apply to the simulation statistic for the target cell that is being minimized or maximized.

The penalty amount is either added or subtracted from the calculated statistic for the target cell in order to make it less "optimal." For example, if *Maximum* is selected in the *Find the* field in the RISKOptimizer Model Dialog, the penalty is subtracted from the calculated statistic for the target cell.

Viewing the Effects of an Entered Penalty Function

RISKOptimizer includes an Excel worksheet PENALTY.XLS which can be used to evaluate the effects of different penalty functions on specific soft constraints and target cell results.



PENALTY.XLS allows you to select a soft constraint from your model whose effects you wish to analyze. You can then change the penalty function to see how the function will map a specific value for the unmet soft constraint into a specific penalized statistic for the target cell. For example, if your soft constraint is $A10 < 100$, you could use PENALTY.XLS to see what the target value would be if a value of 105 was calculated for cell A10.

Viewing the Penalties Applied

When a penalty is applied to the target cell due to an unmet soft constraint, the amount of penalty applied can be viewed in the RISKOptimizer Watcher. In addition, penalty values are shown in Optimization Log worksheets, created optionally after optimization.

Entering Soft Constraints In Your Worksheet

Penalty functions may also be entered directly in your worksheet. A Boolean penalty function will assign a set penalty on any scenario which does not meet the specified constraint. For example, if you wanted the value in cell B1(supply) to be at least as great as the value in cell A1(demand), you could create this penalty function in another cell: $=IF(A1>B1, -1000, 0)$. If the result of this cell were added to the statistic for the target cell, then every time RISKOptimizer tried a solution which violated that constraint (i.e. the supply did not meet the demand), the statistic for the target cell being maximized would show a value 1,000 lower than the real result. Any solution which violated this constraint would produce a low value for the statistic for the target cell, and eventually RISKOptimizer would “breed out” these organisms.

You can also use a scaling penalty function, which more accurately penalizes the solution relative to how badly it violates the constraint. This is often more practical in the real world, because a solution where supply did not quite meet demand would be better than a solution where supply didn’t even come close to the demand. A simple scaling penalty function computes the absolute difference between the constraint’s goal value and it’s actual value. For example, in the same problem where A1(demand) should not exceed B1(supply), we could assign the following penalty function: $=IF(A1>B1, (A1-B1)^2, 0)$. This kind of penalty function measures how close a constraint is to being met, and exaggerates that difference by squaring it. Now our penalty changes based on how badly a solution violates the constraint.

More Examples of Penalty Functions

For example, suppose you have created a manufacturing model where one of the constraints is that the amount of wood used should be equal to the amount of plastic used. This constraint is met when “AmountWood” = “AmountPlastic”. We want to find solutions that include the same amount of both materials, so we create a penalty function to discourage solutions that stray from our goal. The formula “=ABS(AmountWood-AmountPlastic)” computes the absolute (non-negative) difference between the amount of wood and the amount of plastic being used. By using the ABS() function, we arrive at the same penalty value if AmountWood is 20 greater than AmountPlastic, or if AmountPlastic is 20 less than AmountWood. Now when we optimize the model, our goal is to minimize the mean of the simulation results for this absolute difference.

Suppose instead we impose the following constraint: The amount of wood must be twice the amount of plastic. The penalty function would then be:

=ABS(AmountWood-AmountPlastic*2)

A different possible constraint is that the amount of wood should be **no less than** twice the amount of plastic. While the previous example produced a penalty if there was too much wood, in this case we only care if there is not enough wood; if AmountWood is ten times AmountPlastic, we want no penalty to be applied. The appropriate penalty function would then be:

=IF(AmountWood<AmountPlastic*2,
ABS(AmountPlastic*2-AmountWood),0)

If AmountWood is at least twice as great as AmountPlastic, the penalty function returns 0. Otherwise, it gives a measure of how much less than twice AmountPlastic the AmountWood value is.

Using Penalty Functions

After you have created penalty functions to describe the soft constraints in your model, you can combine them with your normal target cell formula to obtain a constrained target cell formula. In the example illustrated below, if cell C8 computes the total cost of a project, and cells E3:E6 contain five penalty functions, then you can create a formula in cell C10 such as =SUM(C8, E3:E6).

	C10		=SUM(C8,E3:E6)			
	A	B	C	D	E	
1		Cost of Project				
2			\$4,500		constraints	
3			\$300		25	
4			\$46,500		30	
5			\$1,200		12	
6			\$24,300		80	
7			\$76,800			
8		total	\$153,600			
9						
10			\$153,747			
11						

Create a cell that adds the constraints to your total, and minimize the mean of the simulation results for this cell.

This adds the penalties in column E to the cost in C8 to obtain a constrained or penalized cost function in C10. Note that if this were a maximization problem, you would subtract, rather than add, the penalties to the original target cell. Now when you use RISKOptimizer, you simply select this constrained cell, C10, as the target cell to be whose simulation statistic will be optimized.

When RISKOptimizer tries to optimize a constrained statistic for the target cell, the penalty functions will tend to force the search towards scenarios that meet the constraints. Eventually RISKOptimizer will end up with solutions that are good answers and that meet or nearly meet all constraints (the penalty functions will have values near 0).

Multiple Goal Problems

You may only specify one cell in the target cell field of RISKOptimizer, but you can still solve for multiple goals by creating a function that combines the two goals into one goal. For example, as a polymer scientist, you may be trying to create a substance that is flexible, but also strong. Your model computes the resulting strength, flexibility and weight that would result from a given mix of chemical combinations. The amounts of each chemical to use are the adjustable variables of the problem.

Since you want to maximize the Strength of the substance (in cell S3) but also maximize its Flexibility (in cell F3), you would create a new cell with the formula: =(S3+F3). This would be your new target cell, for the higher this number went, the better the overall solution.

If the flexibility was more important than the strength, we could change the formula in the target cell to read $= (S3 + (F3 * 2))$. This way, scenarios which increased the flexibility by a certain amount would look better (produce a higher fitness “score”) than scenarios which increased the strength by the same amount.

If you wanted to maximize the Strength of a substance (in cell S5) but also minimize its Weight (in cell W5), you would create a new cell with the following formula: $= (S5^2) - (W5^2)$. This formula would produce a higher number when the structure was both strong-and-light, a lower number when the structure was weak-and-heavy, and equally average numbers for weak-but-light and strong-but-heavy scenarios. You would therefore use this new cell as your target, and maximize its mean to satisfy both goals.

Improving Speed

When you use RISKOptimizer to solve a problem, you are using both the RISKOptimizer library of compiled routines to control the process and Excel's spreadsheet evaluation function to examine different scenarios. A large percentage of the time used by RISKOptimizer is actually used by Excel as it recalculates your spreadsheet. There are a number of things that can be done to speed up RISKOptimizer optimization and Excel's recalculation process.

- ◆ The speed of RISKOptimizer is directly related to the speed of your computer processor. A Pentium/200 will be roughly twice as fast as the Pentium/100. This means that RISKOptimizer will be able to evaluate twice as many trials in the same amount of time.
- ◆ Experiment with different simulation stopping conditions. Initial tests of a model should be done with a low fixed number of iterations per simulation. Once you are certain that your model and constraints are performing as desired, have RISKOptimizer determine how many iterations to run each simulation by selecting **Stop on Actual Convergence** or **Stop on Projected Convergence**. The setting Stop on Projected Convergence results in faster optimizations than Stop on Actual Convergence.
- ◆ Increase the **Convergence Tolerance** setting if you are using the simulation stopping condition Stop on Actual Convergence or Stop on Projected Convergence. This will keep RISKOptimizer from running unnecessary iterations without significantly changing simulation statistics. Setting Convergence Tolerance too high, however, will result in unstable simulation results.
- ◆ Try to avoid re-drawing in your window. Drawing graphics and numbers on the screen takes time, sometimes more than half the time spent optimizing! If you have charts or graphs on the sheet, they will slow down the re-calculate time significantly. You can tell Excel not to spend time drawing while RISKOptimizer is solving a problem by turning off the *Update Display* option in the RISKOptimizer Model Dialog or by minimizing the Excel sheet. You can see how much faster your problem is working by watching the status bar.
- ◆ Once RISKOptimizer has more or less converged on a solution, and there has been no improvement on the best solution in a while (e.g. last 1000 trials), you may want to increase the

mutation rate to allow RISKOptimizer to broaden its search for solutions, rather than continuing to refine solutions in the current population using primarily crossover. You can increase mutation rate through the RISKOptimizer Watcher using the Population Settings command.

- ◆ Set more tightly the ranges that the adjustable cells must fall between; this will create a smaller area in which RISKOptimizer must search for solutions, and should therefore speed up the process. Make sure that your ranges allow enough freedom for RISKOptimizer to explore all realistic solutions.

How RISKOptimizer's Optimization is Implemented

In this section we describe more specifically how RISKOptimizer's optimization algorithms are implemented.

NOTE: *You do not need to know this material in order to use RISKOptimizer.*

The majority of RISKOptimizer's genetic algorithm technology such as the *recipe* and *order* solving methods are based on academic work in the genetic algorithm field over the last ten years. However, most of the descendant solving methods included with RISKOptimizer, and the multiple groups of adjustable cells, backtracking, strategy, and probability features are unique to RISKOptimizer.

RISKOptimizer uses a steady-state approach. This means that only one organism is replaced at a time, rather than an entire "generation" being replaced. This steady state technique has been shown to work as well or better than the generational replacement method. To find out the equivalent number of "generations" RISKOptimizer has run, take the number of individual trials it has explored and divide that by the size of the population.

Selection

When a new organism is to be created, two parents are chosen from the current population. Organisms that have high fitness scores are more likely to be chosen as parents.

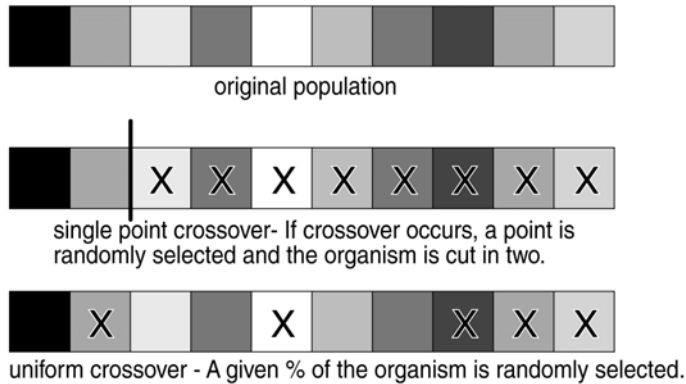
In RISKOptimizer, parents are chosen with a rank-based mechanism. Instead of some genetic algorithm systems, where a parent's chance to be selected for reproduction is directly proportional to its fitness, a ranking approach offers a smoother selection probability curve. This prevents good organisms from completely dominating the evolution from an early point.

Crossover

Since each solving method adjusts the variables in different ways, RISKOptimizer employs a different crossover routine optimized for that type of problem.

The basic recipe solving method performs crossover using a uniform crossover routine. This means that instead of chopping the list of variables in a given scenario at some point and dealing with each of the two blocks (called "single-point" or "double-point" crossover), two groups are formed by randomly selecting items to be in one group or another. Traditional x -point crossovers may bias the search with the irrelevant position of the variables, whereas the uniform

crossover method is considered better at preserving schema, and can generate any schema from the two parents.



The order solving method performs crossover using a similar algorithm to the order crossover operator described in L. Davis' *Handbook of Genetic Algorithms*.^{*} This selects items randomly from one parent, finds their place in the other parent, and copies the remaining items into the second parent in the same order as they appear in the first parent. This preserves some of the sub-orderings in the original parents while creating some new sub-orderings.

Mutation

Like crossover, mutation methods are customized for each of the different solving methods. The basic recipe solving method performs mutation by looking at each variable individually. A random number between 0 and 1 is generated for each of the variables in the organism, and if a variable gets a number that is less than or equal to the mutation rate (for example, 0.06), then that variable is mutated. The amount and nature of the mutation is automatically determined by a proprietary algorithm. Mutating a variable involves replacing it with a randomly generated value (within its valid min-max range).

To preserve all the original values, the order solving method performs mutation by swapping the positions of some variables in the organism. The number of swaps performed is increased or decreased proportionately to the increase and decrease of the mutation rate setting (from 0 to 1).

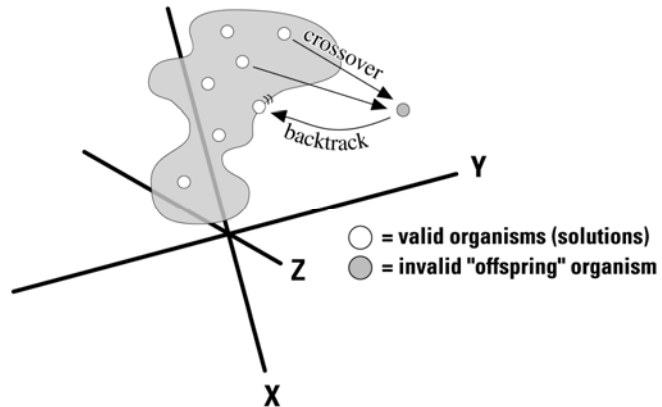
Replacement

Since RISKOptimizer uses a rank-ordered rather than generational replacement method, the worst-performing organisms are always replaced with the new organism that is created by selection, crossover, and mutation, regardless of its fitness "score".

^{*} Davis, Lawrence (1991). *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold.

Constraints

Hard constraints are implemented with Palisade's proprietary "backtracking" technology. If a new offspring violates some externally imposed constraints, RISKOptimizer backtracks towards one of the parents of the child, changing the child until it falls within the valid solution space.



Appendix A: Automating RISKOptimizer

VBA

RISKOptimizer comes with a complete macro language for building custom applications which use RISKOptimizer's capabilities.

RISKOptimizer's custom functions can be used in Visual Basic for Applications (VBA) for setting up and running optimizations and displaying the results from optimizations. For more information on this programming interface, see the RISKOptimizer Developer Kit help document, available via the RISKOptimizer Help menu.

Appendix B:

Troubleshooting / Q&A

Troubleshooting / Q&A

This section answers some commonly asked questions regarding RISKOptimizer and keeps you up to date on common questions, problems and suggestions. After reading through this section, you may call Palisade customer support at the numbers listed in the beginning chapter of this manual.

Q: Why am I having trouble getting a valid answer from RISKOptimizer?

A: Make sure that the RISKOptimizer dialog is set up correctly. Most of the problems are associated with the setting of the variables. Each group of adjustable cells should be exclusive, in that no single cell or range of cells is being treated with more than one solving method.

Q: Can RISKOptimizer deal with concepts or categories instead of just numbers?

A: RISKOptimizer can indirectly deal with any kind of data, since numbers are just symbols. Use a lookup table in Excel to translate between integers and strings of text. RISKOptimizer (like all computer programs) ultimately can only deal with numbers, but your interface may use those numbers to represent and display any strings.

Q: Even though I'm filling in the dialogs the same way, and letting RISKOptimizer run the same amount of time, why does RISKOptimizer sometimes find different solutions?

A: As is the case with natural selection in the biological world, the RISKOptimizer genetic algorithm will not always follow the same path when searching for solutions (unless you use a fixed random number generator seed). Ironically it is this "unpredictability" that allows RISKOptimizer to solve more types of problems, and often find better solutions than traditional techniques. RISKOptimizer's genetic algorithm engine is not just executing a series of pre-programmed commands, or plugging values through a mathematical formula, but it is efficiently experimenting with many random hypothetical scenarios simultaneously, and then refining the search through many "survival-of-the-fittest" operators which also contain random elements.

Q: Why is the best solution found not changing?

A: You may have specified the wrong target cell in the RISKOptimizer Model Dialog. RISKOptimizer is looking at this blank cell and the value does not change because there is no formula. To fix this, display the RISKOptimizer Model Dialog and select a proper target cell; i.e. one that accurately reflects how good/bad each possible solution is. A proper target cell has a formula which depends, directly or indirectly, on the variables RISKOptimizer is adjusting (adjustable cells).

Q: Some of the cells in my spreadsheet model contain “####” symbols.

A: If the cell is too small to display all of its contents, it will display several #### signs. Increase the size of the cell.

Q: RISKOptimizer is working OK, but is there any simple way to get better results?

A: Consider loosening the constraints in the problem, including variable ranges. Change some of your hard constraints to soft constraints via penalty functions (see Adding Constraints in Chapter 8: RISKOptimizer Extras). Too many restrictions on what RISKOptimizer can try may be preventing RISKOptimizer from exploring an area of possibilities that may yield better results. Remember, the longer you let RISKOptimizer explore the possibilities, the more likely it is to find the optimal solution. For more ideas on how to fine-tune RISKOptimizer, see Chapter 8: RISKOptimizer Extras.

The more scenarios RISKOptimizer can run through, the better. Speed up the RISKOptimizer process by turning off the “Every Recalculation” option for display update.

Appendix C: Additional Resources

Additional Learning Resources

The following list represents a select sampling of genetic algorithm and artificial-life-related materials. A star (*) indicates a Palisade favorite.

Books

- Bolles, R.C., & Beecher, M.D. (Eds.). (1988). *Evolution and Learning*. Lawrence Erlbaum.
- Beer, R.D. (1990). *Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology*. Academic Press.
- Davis, Lawrence (1987). *Genetic Algorithms and Simulated Annealing*. Palo Alto, CA: Morgan Kaufman.
- * Davis, Lawrence (1991). *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold.
- Darwin, Charles (1985). *On The Origin of Species*. London: Penguin Classics. (originally 1859)
- * Dawkins, Richard. (1976). *The Selfish Gene*. Oxford University Press.
- Eldredge, N. (1989). *Macroevolutionary Dynamics: Species, Niches, and Adaptive Peaks*. McGraw-Hill.
- Fogel, L., Owens, J., and Walsh, J. (1966). *Artificial Intelligence through Simulated Evolution*. New York: John Wiley and Sons.
- Goldberg, David (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley Publishing.
- Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: University of Michigan Press.
- Koza, John (1992). *Genetic Programming*. Cambridge, MA: MIT Press.
- * Langton, C.L. (1989). *Artificial Life*. MIT Press. [ALife I]
- Levy, Steven (1992). *Artificial Life*. New York: Pantheon.
- Meyer, J.-A., & S.W. Wilson (Eds.). (1991). *Proceedings of the First International Conference on Simulation of Adaptive Behavior: From Animals to Animats*. MIT Press/Bradford Books.
- * Proceedings of the Sixth International Conference (ICGA) on Genetic Algorithms (1995). San Mateo, CA: Morgan Kaufman Publishing. (Also available; the first five ICGA proceedings).
- Proceedings of the Workshop on Artificial Life (1990). Christopher G. Langton, Senior Editor. Reading, MA: Addison-Wesley Publishing.

- Rawlins, Gregory (1991). Foundations of Genetic Algorithms. San Mateo, CA: Morgan Kaufman Publishing.
- Richards, R.J. (1987). Darwin and the Emergence of Evolutionary Theories of Mind and Behavior. U. Chicago Press.
- Williams, G.C. (1966). Adaptation and Natural Selection. Princeton U. Press.

Articles

- * Antonoff, Michael (October, 1991). Software by Natural Selection. Popular Science, p. 70-74.
- Arifovic, Jasmina (January, 1994). Genetic Algorithm Learning and the Cobweb Model. In Journal of Economic Dynamics & Control v18 p.3
- * Begley, S (May 8, 1995). "Software au Naturel" In Newsweek p. 70
- Celko, Joe (April, 1993). Genetic Algorithms and Database Indexing. In Dr. Dobb's Journal p.30
- Ditlea, Steve (November, 1994). Imitation of Life. In Upside Magazine p.48
- Gordon, Michael (June, 1991). User-based Document Clustering by Redescribing Subject Descriptions with a Genetic Algorithm. In Journal of the American Society for Information Science v42 p.311
- Hedberg, Sara (September, 1994). Emerging Genetic Algorithms. In AI Expert, p. 25-29.
- Hinton, G.E., & Nowlan, S.J. (1987). How Learning Can Guide Evolution. In Complex Systems 1: p.495-502.
- * Kennedy, Scott (June, 1995). Genetic Algorithms: Digital Darwinism. In Hitchhiker's Guide to Artificial Intelligence Miller Freeman Publishers
- Kennedy, Scott (December, 1993). Five Ways to a Better GA. In AI Expert, p. 35-38
- Lane, A (June, 1995). The GA Edge in Analyzing Data. In AI Expert p.11
- Lee, Y.C. (Ed.). (1988). Evolution, learning, and cognition. In World Scientific.
- Levitin, G and Rubinovitz, J (August, 1993). Genetic Algorithm for Linear and Cyclic Assignment Problem. In Computers & Operations Research v20 p.575
- Marler, P., & H.S. Terrace. (Eds.). (1984). The Biology of Learning. Springer-Verlag.
- Mendelsohn, L. (December, 1994) Evolver Review In Technical Analysis of Stocks and Commodities. p.33
- Maynard Smith, J. (1987). When Learning Guides Evolution. In Nature 329: p.761-762.

- Murray, Dan (June, 1994). Tuning Neural Networks with Genetic Algorithms. In AI Expert p.27
- Wayner, Peter (January, 1991). Genetic Algorithms: Programming Takes a Valuable Tip from Nature. In Byte Magazine v16 p.361

Magazines & Newsletters

- Advanced Technology for Developers (monthly newsletter). Jane Klimasauskas, Ed., High-Tech Communications, 103 Buckskin Court, Sewickley, PA 15143 (412) 741-7699
- AI Expert (monthly magazine). Larry O'Brien, Ed., 600 Harrison St., San Francisco, CA 94107 (415) 905-2234. *Although AI Expert ceased publishing in the spring of 1995, its back issues contain many useful articles. Miller-Freeman, San Francisco.
- Applied Intelligent Systems (bimonthly newsletter). New Science Associates, Inc. 167 Old Post Rd., Southport, CT 06490 (203) 259-1661
- Intelligence (monthly newsletter). Edward Rosenfeld, Ed., PO Box 20008, New York, NY 10025-1510 (212) 222-1123
- PC AI Magazine (monthly magazine). Joseph Schmuller, Ed., 3310 West Bell Rd., Suite 119, Phoenix, AZ 85023 (602) 971-1869
- Release 1.0 (monthly newsletter). Esther Dyson, Ed., 375 Park Avenue, New York, NY 10152 (212) 758-3434
- Sixth Generation Systems (monthly newsletter). Derek Stubbs, Ed., PO Box 155, Vicksburg, MI, 49097 (616) 649-3592

Introduction to Simulation

If you are new to Simulation or if you would just like some more background information on the technique, the following books and articles might be helpful:

- * Baird, Bruce F. Managerial Decisions Under Uncertainty: John Wiley & Sons, Inc. 1989.
- * Clemen, Robert T. Making Hard Decisions: Duxbury Press, 1990.
- Hertz, D.B. "Risk Analysis in Capital Investment": HBR Classic, Harvard Business Review, September/October 1979, pp. 169-182.
- Hertz, D.B. and Thomas, H. Risk Analysis and Its Applications: John Wiley and Sons, New York, NY, 1983.
- Megill, R.E. (Editor). Evaluating and Managing Risk: PennWell Books, Tulsa, OK, 1984.
- Megill, R.E. An Introduction to Risk Analysis, 2nd Ed.: PennWell Books, Tulsa, OK, 1985.
- Morgan, M. Granger and Henrion, Max, with a chapter by Mitchell Small, Uncertainty: Cambridge University Press, 1990.
- Newendorp, P.D. Decision Analysis for Petroleum Exploration: Petroleum Publishing Company, Tulsa, Okla., 1975.
- Raiffa, H. Decision Analysis: Addison-Wesley, Reading, Mass., 1968.

Technical References to Simulation and Monte Carlo Techniques

If you would like a more in depth examination of simulation, sampling techniques and statistical theory, the following books may be useful:

- Iman, R. L., Conover, W.J. "A Distribution-Free Approach To Inducing Rank Correlation Among Input Variables": Commun. Statist.-Simula. Computa.(1982) 11(3), 311-334
- * Law, A.M. and Kelton, W.D. Simulation Modeling and Analysis: McGraw-Hill, New York, NY, 1991,1982.
- Rubinstein, R.Y. Simulation and the Monte Carlo Method: John Wiley and Sons, New York, NY, 1981.

Technical References to Latin Hypercube Sampling Techniques

If you are interested in the relatively new technique of Latin Hypercube sampling, the following sources might be helpful:

- Iman, R.L., Davenport, J.M., and Zeigler, D.K. "Latin Hypercube Sampling (A Program Users Guide)": Technical Report SAND79-1473, Sandia Laboratories, Albuquerque (1980).
- Iman, R.L. and Conover, W.J. "Risk Methodology for Geologic Disposal of Radioactive Waste: A Distribution - Free Approach to Inducing Correlations Among Input Variables for Simulation Studies": Technical Report NUREG CR 0390, Sandia Laboratories, Albuquerque (1980).
- McKay, M.D, Conover, W.J., and Beckman, R.J. "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code": Technometrics (1979) 211, 239-245.
- Startzman, R. A. and Wattenbarger, R.A. "An Improved Computation Procedure for Risk Analysis Problems With Unusual Probability Functions": SPE Hydrocarbon Economics and Evaluation Symposium Proceedings, Dallas (1985).

Examples and Case Studies Using Simulation

If you would like to examine case studies showing the use of Simulation in real life situations, see the following:

Hertz, D.B. and Thomas, H. Practical Risk Analysis - An Approach Through Case Histories: John Wiley and Sons, New York, NY, 1984.

* Murtha, James A. Decisions Involving Uncertainty, An @RISK Tutorial for the Petroleum Industry: James A. Murtha, Houston, Texas, 1993

- Newendorp, P.D. Decision Analysis for Petroleum Exploration: Petroleum Publishing Company, Tulsa, Okla., 1975.

- Pouliquen, L.Y. Risk Analysis in Project Appraisal: World Bank Staff Occasional Papers Number Eleven. John Hopkins Press, Baltimore, MD, 1970.

* Trippi, Robert R. and Truban, Efraim, Neural Networks: In Finance and Investing: Probus Publishing Co., 1993

Glossary

For additional information on any term, refer to the RISKOptimizer index in the following chapter.

Algorithm	A mathematically based step-by-step method of solving a certain kind of problem. All computer programs are built by combining many algorithms.
Adjustable Cell	A spreadsheet cell whose value can be adjusted by RISKOptimizer to try to optimize the value of the target cell. An adjustable cell is a variable value and should always contain a simple number, rather than an equation.
Baby Solver	<i>slang</i> Simple software programs that find the inputs which produce a desired output using a combination of linear programming techniques, or basic hill-climbing algorithms. Baby solvers often take guesses, then refine their answer to arrive at a “local” solution rather than a “global” solution.
Cell	The cell is the basic unit of a spreadsheet in which data is stored. There are up to 256 columns and 16,000 rows, for a total of more than 4 million cells, in each Excel worksheet.
Constraints	Constraints are conditions which should be met (soft constraints) or must be met (hard constraints) for a scenario to be considered valid.
Continuous Distribution	A probability distribution where any value between the minimum and maximum is possible (has finite probability). <i>See discrete distribution</i>
Crossover	In a genetically based context, crossing over is an exchange of equivalent genetic material between homologous chromatids during meiosis. In RISKOptimizer, the term crossover is used to express the computational equivalent to crossing over, where an exchange between variables yields new combinations of scenarios.
Cumulative Distribution	A cumulative distribution, or a cumulative distribution function, is the set of points, each of which equals the integral of a probability distribution starting at the minimum value and ending at the associated value of the random variable. <i>See cumulative frequency distribution, probability distribution</i>

**Cumulative
Frequency
Distribution**

A cumulative frequency distribution is the term for the output and the input cumulative distributions of RISKOptimizer. A cumulative distribution is constructed by cumulating the frequency (progressively adding bar heights) across the range of a frequency distribution. A cumulative distribution can be an "upwardly sloping" curve, where the distribution describes the probability of a value less than or equal to any variable value. Alternatively, the cumulative curve may be a "downwardly sloping" curve, where the distribution describes the probability of a value greater than or equal to any variable value.

See cumulative distribution

**Dependent
Variable**

A dependent variable is one that depends in some way on the values of other variables in the model under consideration. In one form, the value of an uncertain dependent variable can be calculated from an equation as a function of other uncertain model variables.

Alternatively, the dependent variable may be drawn from a distribution based on the random number which is correlated with a random number used to draw a sample of an independent variable.

See independent variable

Deterministic

The term deterministic indicates that there is no uncertainty associated with a given value or variable.

Dialog

The window on a computer screen that requests the user to provide information. Also called dialog box. RISKOptimizer contains two major dialogs; the RISKOptimizer Model Dialog, and the Adjustable Cells Dialog.

**Discrete
Distribution**

A probability distribution where only a finite number of discrete values are possible between the minimum and maximum.

See continuous distribution

Field

The basic unit of data entry. Depending on its field type, a field can contain text, pictures, or numbers. Most fields in the RISKOptimizer dialogs ask the user to input the location of spreadsheet cells, or options regarding how RISKOptimizer should behave.

**Fitness
Function**

This is a formula which can calculate how good or bad any proposed solution is to a given problem. The term is often used in the genetic algorithm field as an analogy to "fitness" in biological selection. Designing an accurate fitness function is critical when using a genetic algorithm to solve a problem. A simulation result for this fitness function becomes the goal or target value to be optimized.

Functions	In Excel, a function is a pre-defined formula that takes a value, performs an operation, and returns a value. Excel contains hundreds of built-in formulas (like "SUM") that save time, space, and are faster. For example, instead of typing A1+ A2+ A3+ A4+ A5+ A6, you can type SUM(A1:A6) and get the same result.
Frequency Distribution	Frequency distribution is the proper term for the output probability distributions and the input histogram distributions (HISTOGRM) of RISKOptimizer. A frequency distribution is constructed from data by arranging values into classes and representing the frequency of occurrence in any class by the height of the bar. The frequency of occurrence corresponds to probability.
Genetic Algorithm	A procedure for improving results of some operation by repeatedly trying several possible solutions and reproducing and mixing the components of the better solutions. The process is inspired by, and crudely similar to, the process of evolution in the biological world, where the fittest survive to reproduce.
Generation	In the field of genetic algorithms, each completely new population of "offspring" solutions is a new "generation". Some genetic algorithm routines mate all members of a population at once, creating a whole new "generation" of offspring organisms that replaces the previous population. RISKOptimizer evaluates and replaces one organism at a time (rank-ordered) and thus does not use the term "generation" in its documentation. This steady state technique works as well as generational replacement.
Genotype	In biology, this is the genetic constitution of an individual. The term usually refers to the sum total of the individual's genes. In the study of GAs, genotype is used to describe the artificial "chromosome" that is evaluated as a possible solution to the problem.
Global Maximum	The largest possible value for a given function. Complex functions or models may have many local maxima but only one global maximum.
Group of Adjustable cells	Each set of variables, along with the way they will be treated, is one group of adjustable cells. RISKOptimizer will list all groups of adjustable cells in the variables section of the RISKOptimizer Model dialog. This architecture allows complex problems to be built and described as several groups of adjustable cells.
Hard Constraints	A constraint that must always be met. For example, the ranges for variables in a recipe problem are hard constraints; a variable set to range between 10 and 20 can never have a value less than 10 or greater than 20. See also <i>soft constraints</i> .

Higher Moments	Higher moments are statistics of a probability distribution. The term generally refers to the skewness and kurtosis, the third and fourth moments respectively. The first and second moments are the mean and the standard deviation respectively. <i>See skewness, kurtosis, mean, standard deviation</i>
Hill-Climbing Algorithm	An optimization procedure that starts from a given scenario and repeatedly moves the scenario in small steps in the direction that will most improve it. Hill-climbing algorithms are fast and simple, but have two drawbacks. First, much work may be needed to find the direction of most improvement. Second, the algorithms usually climb the nearest hill, or local maximum. This prevents the algorithm from finding the global maximum in a difficult problem.
Independent Variable	An independent variable is one that does not depend in any way on the values of any other variable in the model under consideration. The value of an uncertain independent variable is determined by drawing a sample from the appropriate probability distribution. This sample is drawn without regard to any other random sample drawn for any other variable in the model. <i>See dependent variable</i>
Iteration	An iteration is one recalculation of the user's model during a simulation. A simulation consists of many recalculations or iterations. During each iteration, all uncertain variables are sampled once according to their probability distributions, and the model is recalculated using these sampled values. <i>Also known as a simulation trial</i>
Kurtosis	Kurtosis is a measure of the shape of a distribution. Kurtosis indicates how flat or peaked the distribution is. The higher the kurtosis value, the more peaked the distribution. <i>See skewness</i>
Latin Hypercube	Latin Hypercube sampling is a relatively new stratified sampling technique used in simulation modeling. Stratified sampling techniques, as opposed to Monte Carlo type techniques, tend to force convergence of a sampled distribution in fewer samples. <i>See Monte Carlo</i>
Local Maximum	The largest possible value for a given function within a given range of values. A local maximum exists at a set of values for variables in a function if slightly changing any or all of the variables' values produces a smaller result from the function. (Compare with global maximum).

Mean	The mean of a set of values is the sum of all the values in the set divided by the total number of values in the set. <i>Synonym: expected value</i>
Model	For the purposes of this manual, a model is a numeric representation, in Excel, of a real-world situation.
Monte Carlo	Monte Carlo refers to the traditional method of sampling random variables in simulation modeling. Samples are chosen completely randomly across the range of the distribution, thus necessitating large numbers of samples for convergence for highly skewed or long-tailed distributions. <i>See Latin Hypercube</i>
Most Likely Value	The most likely value or mode is the value that occurs most often in a set of values. In a histogram and a result distribution, it is the center value in the class or bar with the highest probability.
Mutation	In the biological world, gene mutation is the source of variation needed for effective natural selection. Likewise, a genetic algorithm uses mutation techniques to maintain diversity in a population of possible scenarios.
Optimization	The process of finding values for variables so that the output of a function can be maximized (made as large as possible) or minimized (made as small as possible). Optimization by equation solving is easy for smoothly changing functions with few variables, but extremely difficult for many real-world problems. Tough problems generally need a search mechanism. RISKOptimizer uses an optimizing search mechanism based upon a genetic algorithm.
Organism	A block of memory in a population that stores a set of variable values (scenario).
Penalty Function	A spreadsheet equation that RISKOptimizer can use to penalize scenarios that fail to meet some criteria. Penalty functions are used to help minimize side effects from scenarios or to achieve multiple goals. Unlike a hard constraint, a penalty function does allow invalid solutions to be explored; it just makes those solutions look bad so the population will evolve away from those solutions. Boolean penalties are either on or off, penalizing all invalid solutions by the same amount. Scaling penalties are more fluid, assigning a penalty in proportion to how badly a constraint is violated.
Percentile	A percentile is an increment of the values in a data set. Percentiles divide the data into 100 equal parts, each containing one percent of the total values. The 60th percentile, for example, is the value in the data set for which 60% of the values are below it and 40% are above.

Phenotypes	In biology, this is an observable trait of an individual which arises from interactions between genes, and between genes and the environment. In the study of GAs, phenotype is used to describe the individual variables or “genes” that make up one complete solution or “chromosome”. (see Genotype)
Population	The entire set of scenarios that RISKOptimizer keeps in memory from which new scenarios are generated. RISKOptimizer keeps one population of possible solutions for each group of adjustable cells in a system.
Probability	Probability is a measure of how likely a value or event is to occur. It can be measured from simulation data as frequency by calculating the number of occurrences of the value or event divided by the total number of occurrences. This calculation returns a value between 0 and 1 which then can be converted to percentage by multiplying by 100. <i>See frequency distribution, probability distribution</i>
Probability Distribution	A probability distribution or probability density function is the proper statistical term for a frequency distribution constructed from an infinitely large set of values where the class size is infinitesimally small. <i>See frequency distribution</i>
Random Number Generator	A random number generator is an algorithm for choosing random numbers, typically in the range of 0 to 1. These random numbers are equivalent to samples drawn from a uniform distribution with a minimum of 0 and a maximum of 1. Such random numbers are the basis for other routines that convert them into samples drawn from specific distribution types. <i>See random sample, seed</i>
Random Sample	A random sample is a value that has been chosen from a probability distribution describing a random variable. Such a sample is drawn randomly according to a sampling "algorithm". The frequency distribution constructed from a large number of random samples drawn by such an algorithm will closely approximate the probability distribution for which the algorithm was designed.

Ranges

In RISKOptimizer:

The user sets the range, or the highest and lowest value that RISKOptimizer is allowed to try when adjusting a certain variable. Although this is not necessary to solve a problem, setting these ranges limits the possibilities and hence narrows RISKOptimizer's search.

In Excel:

A block of contiguous cells in a worksheet that is defined by the upper left cell and the lower right cell (e.g. A5:C9 describes a range of 15 cells).

Scenario

A set of values for the variables in a spreadsheet model. Each scenario most often represents one possible solution.

Simulation

Simulation is a technique whereby a model, such as a Excel worksheet, is calculated many times with different input values with the intent of getting a complete representation of all possible scenarios that might occur in an uncertain situation.

Skewness

Skewness is a measure of the shape of a distribution. Skewness indicates the degree of asymmetry in a distribution. Skewed distributions have more values to one side of the peak or most likely value — one tail is much longer than the other. A skewness of 0 indicates a symmetric distribution, while a negative skewness means the distribution is skewed to the left. Positive skewness indicates a skew to the right. *See kurtosis*

Solution

Any given system contains many input variables producing an output. In RISKOptimizer, a "solution" will more often refer to one of the possible combinations of variables rather than *the* best combination.

Soft Constraints

When constraints do not necessarily have to be met, they can be made soft instead of hard. This is done by specifying a penalty function in RISKOptimizer or adding a penalty function to the target cell's fitness function.

It is often better for constraints to be soft if possible. This is because:

1. RISKOptimizer can usually solve softly-constrained problems faster, and
2. a soft-constraint model often will find a great solution that almost meets the soft constraints, which can be more valuable than a not-so-great solution that does meet hard constraints.

Solving Method

RISKOptimizer includes six of these methods, each using a customized algorithm to solve a specific type of problem. For each set of variables selected in a problem, the user must assign the solving method to be used on those variables. The six solving methods are: grouping, order, recipe, budget, project, and schedule.

**Standard
Deviation**

The standard deviation is a measure of how widely dispersed the values are in a distribution. Equals the square root of the variance.
See variance

Stochastic

Stochastic is a synonym for uncertain, risky.
See risk, deterministic

Status Bar

The status bar appears at the bottom of the Excel window, and displays RISKOptimizer's current activity.

**Survival of the
Fittest**

The idea that organisms better suited to an environment are more likely to live long enough to reproduce and spread their genes through the population's next generation.

Target Cell

The spreadsheet cell whose value we want to minimize or maximize. This cell is set in the RISKOptimizer Model dialog (select RISKOptimizer Model Definition command or the Model icon).

Trials

The process of RISKOptimizer generating a value for each variable in the problem, then recalculating the scenario for evaluation.

Index

A

Add - Adding Constraints	106
adjustable cells	42, 89
advertising selection example	69
algorithm, defined	143
alphabetize example	81
Application Settings command	127

B

backtracking	191
budget allocation example	63
budget solving method	
description	96
example	63, 69, 75

C

chemical equilibrium example	65
class scheduler example	67
combinatorial problems	152
Constraint Solver command	128
constraints	177–85
implementation	191
continuous models	148
crossover rate	134, 160
how it is implemented	189
what it does	101

D

databases	151
deviation keyword	110

<hr/>	
<i>E</i>	
Excel Solver	148
<hr/>	
<i>F</i>	
fitness function	35, 86
<hr/>	
<i>G</i>	
gene pool	161
generations	
why they aren't used	189
genetic operator	103, 104
global solution	
vs. local solution	148
Glossary	208
graphs	54, 132
GRG routines	148
grouping solving method	
<i>description</i>	94
example	73
<hr/>	
<i>H</i>	
hard constraints	46, 107
hill climbing	145
an example	150
described	149–50
Solver's use	148
<hr/>	
<i>I</i>	
integers	90
<i>iteration constraint</i>	30, 31, 51, 107, 119, 180
<hr/>	
<i>J</i>	
job shop example	71
<hr/>	
<i>L</i>	
landscape of solutions	144

Learning RISKOptimizer	14
linear problems	149
local solution	
vs. global solution	148
<i>logging simulation data</i>	52

M

minutes	118
Model dialog	41, 85
modeling uncertainty	4
multiple goal problems	185
mutation rate	134
how it is implemented	190
what it does	102

N

non-linear problems	149–50
---------------------	--------

O

Operators	103, 104
optimization	
methods	143
what is it?	19
Optimization Goal	41, 86
Optimization Runtime options	118
optimization stopping conditions	50
order solving method	
description	94
example	71, 79

P

Palisade Corporation	9
penalty functions	
examples	184
explained	181
using	184
Percentile	26, 87, 213
portfolio balancing example	73
portfolio mix example	75
probability distributions	18, 28
problems	
combinatorial	152

linear	149
non-linear	149–50
table-based	151
Progress window	125
project solving method	
description	97

R

Readme file	14
recipe solving method	
description	93
example	65, 81
Removing RISKOptimizer from your computer	11
replacement method	190
RISKOptimizer	
Tutorial	14
what is it?	17
<i>RISKOptimizer Watcher</i>	54, 131

S

salesman problem example	79
schedule solving method	
description	98
example	67
selection routine	189
Simplex Method	149
<i>simulation constraint</i>	30, 31, 51, 107, 108, 119, 120, 180
simulation optimization process	26
Simulation Runtime options	119
<i>simulation stopping conditions</i>	51
soft constraints	46, 107, 109, 180
Solver	148
solving methods	
as constraints	178
budget	96
example	63, 69, 75
grouping	94
example	73
order	94
example	71, 79
project	97
recipe	93
example	65, 81
schedule	98

example	67
speed, improving	187
status bar	125, 131, 216
Stop on Actual Convergence	119
Stop on Projected Convergence	120
stopping conditions	118

T

table-based problems	151
Target	87
target cell	29, 41, 86, 216
technical specifications	189
traveling salesman example	79
tutorial	14

U

Use Same Random Number Generator Seed Each Sim	114
--	-----

V

Values	90
--------	----

W

Watcher	54, 131
Watcher – Diversity Tab	137
Watcher – Log Tab	135
Watcher – Population Tab	136
Watcher – Progress Tab	132
Watcher – Stopping Options Tab	138
Watcher – Summary Tab	134