36106 Managerial Decision Modeling
Monte Carlo Simulation in Excel: Part IV

Kipp Martin
University of Chicago
Booth School of Business

November 29, 2017

Reading and Excel Files

Reading:

- Handout: Optimal Procurement with Spot Purchases

Files used in this lecture:

- navy.xlsx
- navy_key.xlsx
- navy_key_2.xlsx
- tailExamples.xlsx
- markowitzRiskOptimizerSim.xlsx
- markowitzRiskOptimizerSim_key.xlsx
- markowitzRiskOptimizerVaR_1.xlsx
- markowitzRiskOptimizerVaR_key1.xlsx
- markowitzRiskOptimizerVaR_2.xlsx
- markowitzRiskOptimizerVaR_key2.xlsx

Learning Objectives

1. Learn how to use RiskOptimizer

2. Understand the pitfalls of optimizing expected values. Implementing the decision that gives the optimal value of $E(f(X))$ may not a good idea.

3. Understand the importance of tail behavior.

4. Learn how to put constraints on tail behavior in RiskOpitmizer (how to cover your **okole**)!

Lecture Outline

@RISK Optimizer
- Optimal Procurement with Spot Purchases
- Markowitz Mean-Variance Model

Tail Management
- VaR
- CVaR

Summary

@RISK Optimizer

We are taking tools from earlier in the quarter and extending them to allow for stochastic parameters.

- Goal Seek $\Longleftrightarrow$ @RISK Goal Seek (Week Eight)
- Data Table $\Longleftrightarrow$ @RISK RiskSimtable (Week Nine)
- Solver $\Longleftrightarrow$ @RISK Optimizer (Now!)

@RISK Optimizer

@RISK Optimizer works very much like Solver.

We still have our ABC's.

- A – Adjustable cells
- B – Best cell which is the objective function cell
- C – Constraints

Only now, the **parameter cells (the black cells) can be distributions!**

Optimal Procurement with Spot Purchases

- The Navy Supply System Command is headquartered in Mechanicsberg, Pennsylvania.
- It is a procurement organization with an annual budget in the billions of dollars.
- The resupply is handled by COG (cognizant ordering group) managers who make buys in their separate areas, e.g., electronics, ordnance, etc.
- Because of the rapid obsolescence of some military inventory, a manager might hold back part of budget until late in the year when demand becomes better known.

Optimal Procurement with Spot Purchases

Objective: Build an optimization model to determine the optimal order quantity at the start of each year in order to minimize cost.

- Demand is not known when the order is placed.
- At the time the order is placed, demand must be estimated.
- We treat demand as a random variable.
- For now assume no budget limit.
- The COG manager can make spot purchases to satisfy any demand in excess of the initial purchase quantity.
- Stock outs are not allowed.

Optimal Procurement with Spot Purchases

Key Parameters:

p – purchase price (initial)

v – marginal salvage value at year end

s – purchase price in the spot market

X – demand random variable

$g(X)$ – the p.d.f. (probability density function) of demand

Q – the optimal initial purchase quantity

Assume the following relationship among costs.

$$v < p \leq s$$

If $p = s$, what is the optimal initial purchase quantity, i.e. $Q = ?$

Optimal Procurement with Spot Purchases

The Model: There are two cases.

Case 1: $X \leq Q$ – in this case the total cost is

$$C(Q, X) = p * Q - v * (Q - X)$$

Case 2: $X > Q$ – in this case the total cost is

$$C(Q, X) = p * Q + s * (X - Q)$$

Therefore the expected cost of $C(Q, X)$ is

$$E(C(Q, X)) = pQ - \int_0^Q v * (Q - X) g(X) dX + \int_Q^\infty s * (X - Q) g(X) dX$$

Optimal Procurement with Spot Purchases

Excel Implementation: Here are the parameters (see **navy.xlsx**).

	A	B
1	**No Budget Model**	
2		
3	initial purchase cost - p	$ 10.00
4	marginal salvage value - v	$ 7.00
5	spot market price - s	$ 12.00
6	mean	140
7	standard deviation	20
8		
9		
10	Q	
11	Demand Random Variable - X	
12	Initial Purchase Cost	
13	Salvage Value (Q > X)	
14	Spot Market Cost (X > Q)	
15		
16	Total Cost =	
17	Risk Mean	

Optimal Procurement with Spot Purchases

Critical Concept: We can use @RISK to calculate the **expected value of a tail.**

Let $g(X)$ denote the pdf of a random variable X and $f(X)$ an arbitrary function of X. Assume we want to calculate the weighted left tail

$$\int_0^Q f(X)g(X)dX.$$

In @RISK insert the function

```
=IF( X <= Q, f(X), 0)
```

Step 1: Insert an @RISK Output and run a simulation.

Step 2: Find the @RISK Mean of the simulation run.

Optimal Procurement with Spot Purchases

Expected Tail Calculation: Assume $f(X) = 7 * (Q - X)$.

X	f(X)	g(X)
43	\$14.00	0.1
44	\$7.00	0.2
45	\$0.00	0.2
46	-\$7.00	0.4
47	-\$14.00	0.1

If $Q = 45$ then

$$\int_0^Q f(X)g(X)dX = 7 * 2 * .1 + 7 * 1 * .2 + 7 * 0 * .2 = 2.8$$

See **tailExamples.xlsx** for the simulation.

Optimal Procurement with Spot Purchases

Excel Implementation: Put in the correct Excel formulas for Case 1 and Case 2.

Case 1: $X \leq Q$ – in this case the cost is (note the initial purchase cost is not included here)

```
=IF(Q >= X, salvage_value*(Q-X),0)
```

Case 2: $X > Q$ – in this case the cost is (note the initial purchase cost is not included here)

```
=IF(X > Q, spot_price*(X-Q), 0)
```

We put the initial purchase cost (which happens for all X) in cell **B12**.

```
=Q*price
```

Optimal Procurement with Spot Purchases

Our total cost function is therefore

```
C(Q, X) = p*Q - IF(Q >= X, salvage_value*(Q-X),0)
                 + IF(X > Q, spot_price*(X-Q), 0)
```

We evaluate $E(C(Q,X))$ using simulation.

We find the value of Q that maximizes $E(C(Q,X))$.

Optimal Procurement with Spot Purchases

Now put the model into @RISK Optimizer

Step 1: define the objective function

- The objective is to minimize the expected cost so select **Minimum** under **Optimization Goal**.
- Set the **Cell** that has the objective function. This is cell `B16`. This cell contains the random variable which is the cost function $C(Q)$.
- Specify the **Statistic**. In this case it is **Mean.**

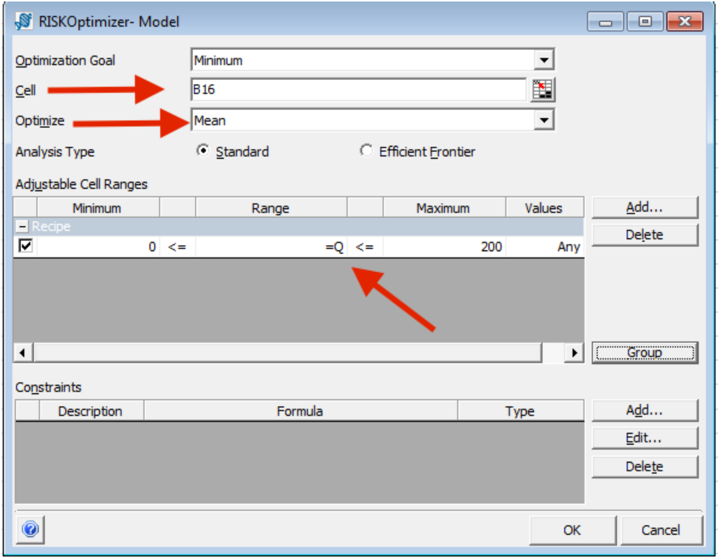
Optimal Procurement with Spot Purchases

Step 2: specify the adjustable cells.

- Click on **Add** under **Adjustable Cell Ranges**.
- Select the variable in the range named **Q**. This is the order quantity variable.
- Set the **minimum** value of the variable to 0 and the **maximum** value to 200. Why is 200 reasonable?

Optimal Procurement with Spot Purchases

See the definition of the objective function and adjustable cells.

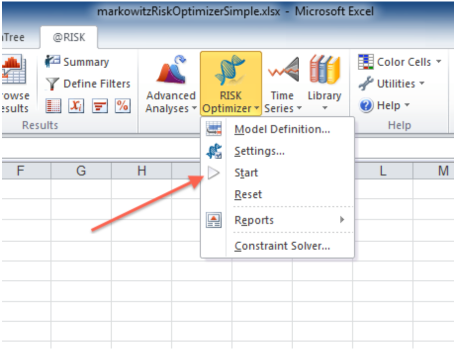


Optimal Procurement with Spot Purchases

Step 3: specify the constraints. There are none in this model.

Step 4: run the model.

Caution: when testing make sure number of iterations is 100; you can make this larger later.



Optimal Procurement with Spot Purchases

Here is the result. It finds an optimal order quantity of 134.94.

	A	B	C
1	**No Budget Model**		
2			
3	initial purchase cost - p	$ 10.00	
4	marginal salvage value - v	$ 7.00	
5	spot market price - s	$ 12.00	
6	mean	140	
7	standard deviation	20	
8			
9			
10	Q	134.9450478	
11	Demand Random Variable - X	150.2084438	
12	Initial Purchase Cost (p*Q)	$ 1,349.45	
13	Salvage Value (Q > X)	0	
14	Spot Market Cost (X > Q)	183.1607523	
15			
16	Total Cost =	$ 1,532.61	
17	Risk Mean	$ 1,438.68	
18			

Optimal Procurement with Spot Purchases

Here is part of the optimization log, showing information on the first 16 iterations.

@RISK (RISKOptimizer): Log of All Trials

Performed By: Kipp Martin

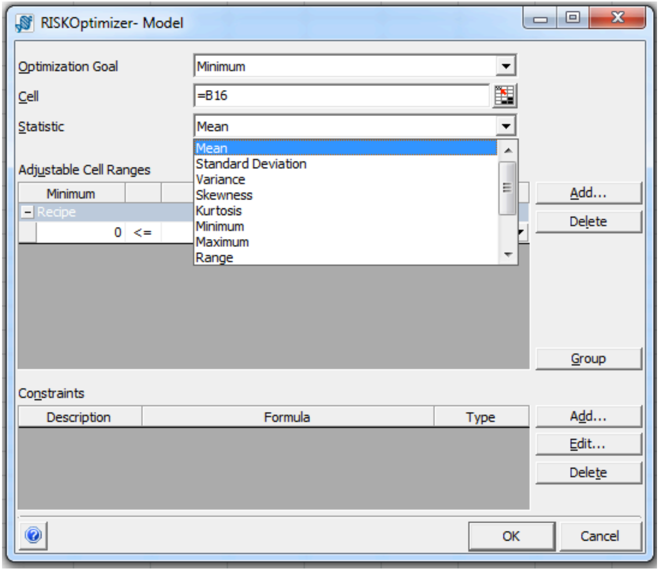
Date: Saturday, March 07, 2015 8:42:49 PM

Model: navy_key.xlsx

Trial	Elapsed Time	Iterations	Result	Goal Cell Statistics				Adjustable Cells
				Mean	Std. Dev.	Min.	Max.	Q
1	2.31481E-05	1000	$ 1,439.94	$ 1,439.94	$ 192.56	$ 957.93	$ 2,232.48	140
2	3.47222E-05	1000	$ 1,480.90	$ 1,480.90	$ 238.01	$ 837.93	$ 2,312.48	100
3	4.62963E-05	1000	$ 1,680.06	$ 1,680.06	$ 240.13	$ 922.16	$ 2,512.48	0
4	4.62963E-05	1000	$ 1,580.08	$ 1,580.08	$ 140.25	$ 1,137.93	$ 2,112.48	200
5	5.78704E-05	1000	$ 1,579.42	$ 1,579.42	$ 240.13	$ 821.53	$ 2,411.84	50.31621534
6	6.94444E-05	1000	$ 1,490.29	$ 1,490.29	$ 148.18	$ 1,044.89	$ 2,174.50	168.9870996
7	6.94444E-05	1000	$ 1,626.43	$ 1,626.43	$ 240.13	$ 868.53	$ 2,458.85	26.81389672
8	8.10185E-05	1000	$ 1,532.45	$ 1,532.45	$ 240.13	$ 774.56	$ 2,364.87	73.80128847
9	8.10185E-05	1000	$ 1,449.37	$ 1,449.37	$ 226.17	$ 895.51	$ 2,274.09	119.1957691
10	9.25926E-05	1000	$ 1,530.22	$ 1,530.22	$ 141.79	$ 1,087.55	$ 2,146.06	183.2066309
11	0.000104167	1000	$ 1,455.91	$ 1,455.91	$ 166.67	$ 999.21	$ 2,204.95	153.7623632
12	0.000104167	1000	$ 1,654.58	$ 1,654.58	$ 240.13	$ 896.68	$ 2,487.00	12.73710792
13	0.000115741	1000	$ 1,503.72	$ 1,503.72	$ 239.73	$ 802.63	$ 2,336.01	88.23571029
14	0.000115741	1000	$ 1,602.49	$ 1,602.49	$ 240.13	$ 844.59	$ 2,434.91	38.78435944
15	0.000127315	1000	$ 1,438.68	$ 1,438.68	$ 201.98	$ 943.29	$ 2,242.23	135.1219088
16	0.000138889	1000	$ 1,438.92	$ 1,438.92	$ 206.46	$ 935.97	$ 2,247.11	132.6828632

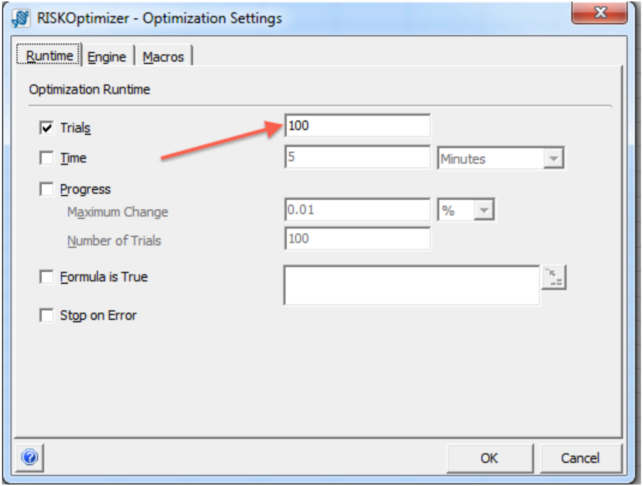
Optimal Procurement with Spot Purchases

Note the wide variety of options for the objective function.



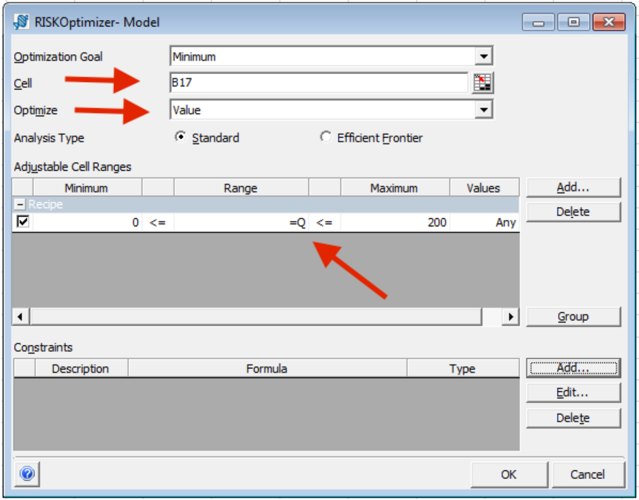
Optimal Procurement with Spot Purchases

In RiskOptimizer be conservative with the number of trials.



Optimal Procurement with Spot Purchases

An Alternate Approach: Minimize **RiskMean** by **Value.** (see navy_key_2.xlsx)



Optimal Procurement with Spot Purchases

Sanity Tests:

1. If $Q = 0$, what is the expected cost?

2. If we increase s what is the effect on the optimal Q?

Optimal Procurement with Spot Purchases

Model Variation: Assume budget of B dollars. This budget includes

1. the initial purchase cost $p * Q$
2. the spot market purchases

We can now have stock outs at fee of f dollars per stock out.

Optimal Procurement with Spot Purchases

The Model with a Budget Constraint: There are now **three** cases.

For notational convenience let

$$y = (B - p * Q)/s.$$

What does y represent?

Case 1: $X \leq Q$ – in this case the cost is (same as before)

$$p * Q - v * (Q - X)$$

Case 2: $Q < X \leq Q + y$ – in this case the cost is (same as before)

$$p * Q + s * (X - Q)$$

Case 3: $Q + y < X$ – in this case the cost is

$$p * Q + s * y + f * (X - Q - y)$$

Optimal Procurement with Spot Purchases

The Model with a Budget Constraint: The expected cost function is

$$
\begin{aligned}
E(c(Q,X)) \quad = \quad & pQ - \int_0^Q v * (Q-X) g(X) dX + \int_Q^{y+Q} s * (X-Q) g(X) dX \\
& + \int_{y+Q}^{\infty} (s * y + f * (X - Q - y))\, g(X) dX
\end{aligned}
$$

Optimal Procurement with Spot Purchases

Excel Implementation: Put in the correct formula for each case.

Case 1: $X \leq Q$ – in this case the cost is (note the initial purchase cost is not included here)

```
=IF(Q >= X,salvage_value*( Q-X), 0)
```

Case 2: $X > Q$ – in this case the cost is (note the initial purchase cost is not included here)

```
=IF(AND(X > Q, X <=Q+ B24), spot_price*(X-Q),0)
```

Case 3:

```
=IF(X > Q+B24, penalty*(X-B24-Q) + spot_price*B24,0)
```

Optimal Procurement with Spot Purchases

Run a simulation with new cost function.

Would you expect the optimal value of Q to be larger or smaller with the budget constraint.

Markowitz Mean-Variance Model

Here is where we are headed:

1. We first build a "simulation" version of the mean-variance Solver model in @RISK Optimizer.
2. We again flashback to the previous lecture and observe that the expected return constraint is very misleading.
3. We put constraints on the *tail behavior* of expected returns.

Markowitz Mean-Variance Model

1. A **deterministic optimization** model is a model with no random variables in the objective function or constraints.

2. A **simulation optimization** model is a model with at least one random variable in the objective function and/or constraints.

Markowitz Mean-Variance Model

First, in order to better familiarize ourselves with @RISK Optimizer we duplicate the Solver inputs for the mean-variance model and build the simulation equivalent.

The "deterministic" version of the mean-variance model is

$$
\begin{aligned}
\min XCX^\top & & & \\
\sum_{i=1}^{n} \mu_i X_i & \geq & R & \\
\sum_{i=1}^{n} X_i & = & 1 & \\
X_i & \leq & 1, & \quad i = 1, \ldots, n \\
X_i & \geq & 0, & \quad i = 1, \ldots, n
\end{aligned}
$$

There are no random variables in the above model, only statistics. We built the above model in Solver.

Markowitz Mean-Variance Model

Now build the **simulation** version of the mean-variance model.

We now work with random variables!

The portfolio return random variable is $r_X X + r_Y Y + r_Z Z$.

It is a weighted sum of the r_X, r_Y, and r_Z random variables

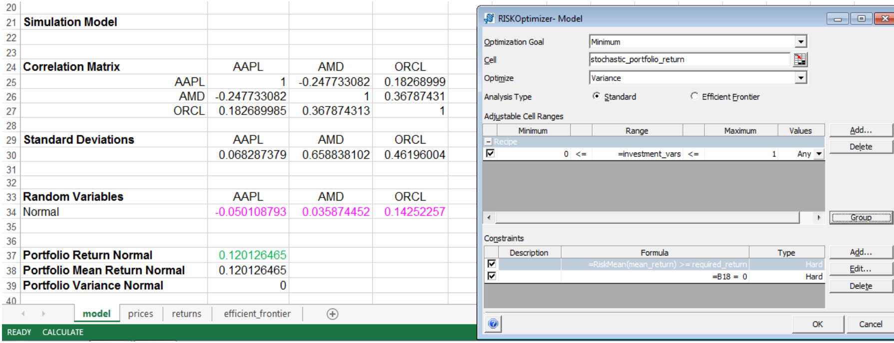
Therefore, the model to implement is

$$\begin{aligned} \min Var(r_X X + r_Y Y + r_Z Z) & & \\ X + Y + Z &=& 1 \\ Exp(r_X X + r_Y Y + r_Z Z) &\geq& .12 \\ X, Y, Z &\geq& 0 \end{aligned}$$

See `markowitzRiskOptimizerSim.xlsx`

Markowitz Mean-Variance Model

Building the model in `markowitzRiskOptimizerSim.xlsx`.



Markowitz Mean-Variance Model

Step 1: define the objective function

- The objective is to minimize risk as measured by variance so select **Minimum** under **Optimization Goal**.
- Set the **Cell** that has the objective function. This is cell B39 that has range name **stochastic_portfolio_return**. This cell contains the random variable which is the portfolio return.

  $$r_X X + r_Y Y + r_Z Z.$$

  It is the weighted sum of the return random variables of the three stocks.
- Specify the **Statistic**. In this case it is **Variance.**

Markowitz Mean-Variance Model

Step 2: specify the adjustable cells.

- Click on **Add** under **Adjustable Cell Ranges**.
- Select the variables in the range named **investment_vars**. These are the variables in `B9:D9` and they correspond to the investment levels in the three stocks.
- Set the **minimum** value of the variables to 0 and the **maximum** value to 1.0.

Markowitz Mean-Variance Model

Step 3: Specify the constraints.

$$
\begin{aligned}
X + Y + Z &= 1 \\
Exp(r_X X + r_Y Y + r_Z Z) &\geq .12
\end{aligned}
$$

The unity constraint is a *deterministic constraint* – there are no random variables – the **Statistic to Constrain** is the Value.

The expected return constraint is a *simulation constraint* – the **Statistic to Constrain** is the Mean.

Markowitz Mean-Variance Model

Critical Concept: When the **Statistic to Constrain** is set to **Value** then you **CANNOT** input something like

$$r_X X + r_Y Y + r_Z Z \geq .12$$

However you could input

$$\mathrm{RiskMean}(r_X X + r_Y Y + r_Z Z) \geq .12$$

and set the **Statistic to Constrain** to **Value**, or you could input

$$r_X X + r_Y Y + r_Z Z \geq .12$$

and set the **Statistic to Constrain** to **Mean**.

Markowitz Mean-Variance Model

Key Concept:

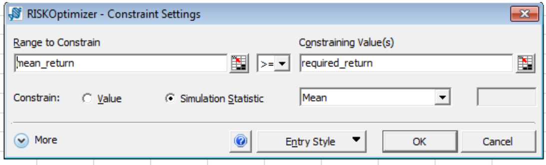
If a function has arguments that are all deterministic, then if we know the arguments, we know the function value.

However, if one of the arguments is a random variable, or depends on a random variable, then we cannot evaluate the function. We can only measure *statistics* of the function.

Markowitz Mean-Variance Model

Step 3: Here is how we add the return constraint. Note the **Statistic to Constrain.**

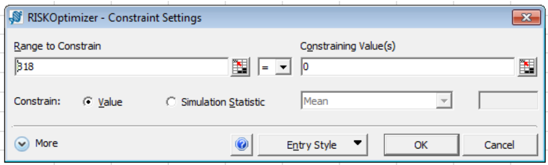
- Click on **Add** under **Constraints**.
- Duplicate what you see below.



Markowitz Mean-Variance Model

Step 3: Here is how we add the unity constraint. Note the **Statistic to Constrain.**

- Click on **Add** under **Constraints**.
- Duplicate what you see below.



Markowitz Mean-Variance Model

Model Results: Compare with Solver deterministic solution.

	Solver Solution	RISK Optimizer Simulation Solution
X	0.1169	0.0000
Y	0.0000	0.2100
Z	0.88311	0.7900
Variance	0.1677	0.1901

Why did we get different values for the variance?

Rerun the RISK Optimizer model **starting** with the Solver solution.
What happens?

Markowitz Mean-Variance Model

Model Results: Compare with Solver deterministic solution.

Key Point 1 – The Bad: RISK Optimizer implements **heuristic** methods when solving simulation models. It may not find the best solution.

Key Point 2 – The Good: RISK Optimizer is far more flexible and allows us to solve problems that Solver cannot.

Tail Management

Where are we? We built a deterministic and a stochastic model with the constraint that expected return was at least 12%.

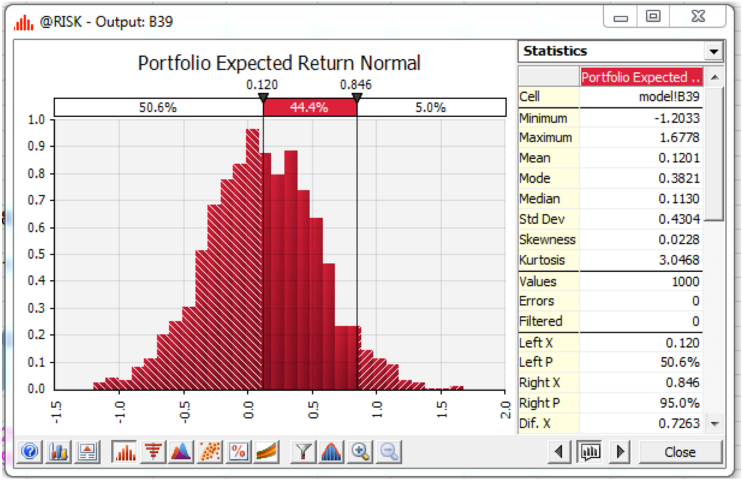
Where are going? We care about our Tail!

The next slides look at the percentage of returns below 12% and 0%, respectively.

The **tail** behavior is not a pretty **tale** to tell.

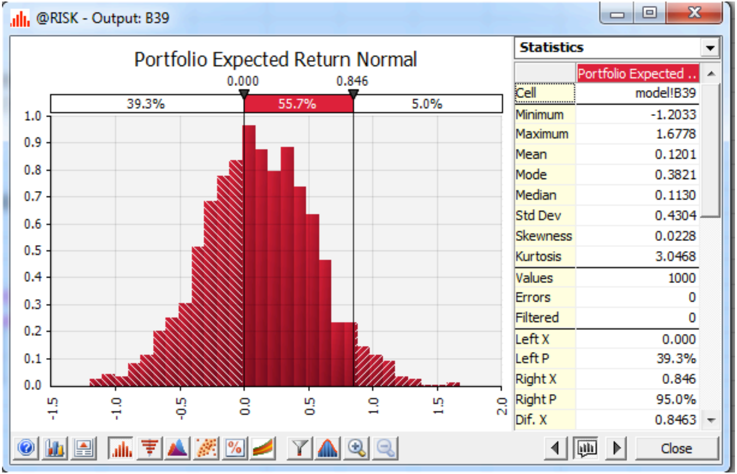
Tail Management

Over 50 percent of the time we do not meet our required return of 12%.



Tail Management

About 39% of the time we have **negative returns**!



Tail Management

Many decision makers care very much about the *left part of the tail of a distribution.*

My GM story.

The basic idea:

- Each allocation of stocks defining a portfolio has an associated distribution of returns.
- Different portfolios have different distributions and therefore different tails.
- We want to find a stock allocation that results in a distribution with a lower tail that we like.

Tail Management

We are going to study the following "tail management" techniques:

1. Value at Risk (**VaR**)
2. Conditional Value at Risk (**CVaR**)

@RISK Tools:

1. RiskPercentile – returns an X (target) for a given percentile (P)
2. RiskTarget – returns a percentile (P) for a given X (target)
3. Percentile(X for a Given P)
4. Target(P for a given X)

VaR

See the case study about VaR at Amazon and FedEx:

`http://www.palisade.com/cases/VCU.asp`

VaR

Value at Risk (VaR) was first popularized by JPMorgan Chase & Co. in the early 1990s (then, just JP Morgan).

The concept behind VaR is to constrain the left tail of the distribution of returns.

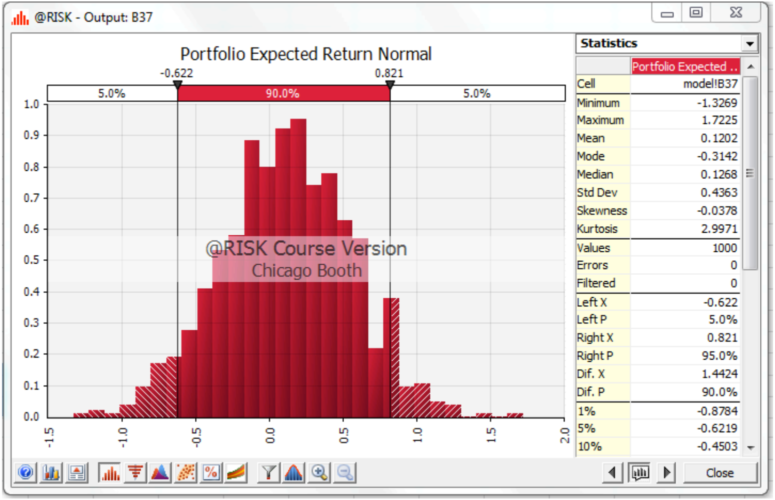
Assume the initial value of the portfolio is 1 dollar and there is a probability of 5 percent of incurring a loss of 30 cents or more. Then there is a **value at risk** of 30 cents at 5 percent.

Consider the solution of $X = 0$, $Y = 0.21$, and $Z = 0.79$ in the workbook `markowitzOptimizerSim.xlsx`.

What is the VaR at 5 percent? See next slide.

VaR

What is the VaR at 5 percent?



VaR

Objective: Improve our tail! Add the VaR constraint that there is a value at risk of $.30 at 5 percent.

In other words, the probability that the portfolio has a return of -.3 or less is 5 percent.

We keep our required return constraint, but lower the 12 percent required return to 2.5 percent.

There are two ways to do this.

VaR

Method 1: Put a **value** constraint on the **simulation result statistic** associated with the distribution of interest (returns).

Step 1: enter the following formula into any cell (in our case `B40`)

```
RiskPercentile(stochastic_portfolio_return,VaR_percentile,1)
```

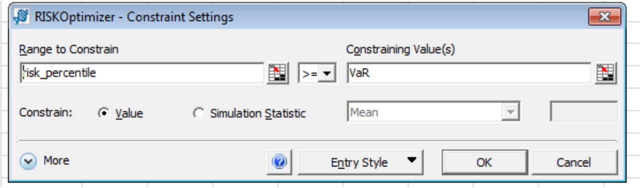
where `stochastic_portfolio_return` references the portfolio returns and `VaR_percentile` references a cell with the .05 percentile.

Step 2: make the **Statistic to Constrain** a **Value** that requires the RiskPercentile function to return greater than or equal to the target of -.3.

VaR

Method 1: Add a **Value statistic to constrain** that says:

```
RiskPercentile(stochastic_portfolio_return,VaR_percentile,1)
 >= VaR
```



VaR

Method 2: Use the distribution explicitly in the constraint and constrain a statistic associated with the distribution.

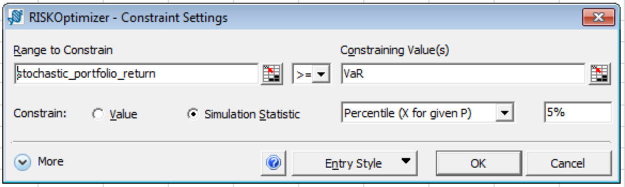
Step 1: add a constraint that says the portfolio returns are greater than or equal to the target of -.3.

Step 2: make the **Statistic to Constrain** a Percentile(X for a Given P) where the value of P is 0.05.

VaR

Method 2: Add a **Percentile (X for given P) constraint** that says:

```
stochastic_portfolio_return >= VaR
```



VaR

Let's implement in RISK Optimizer.

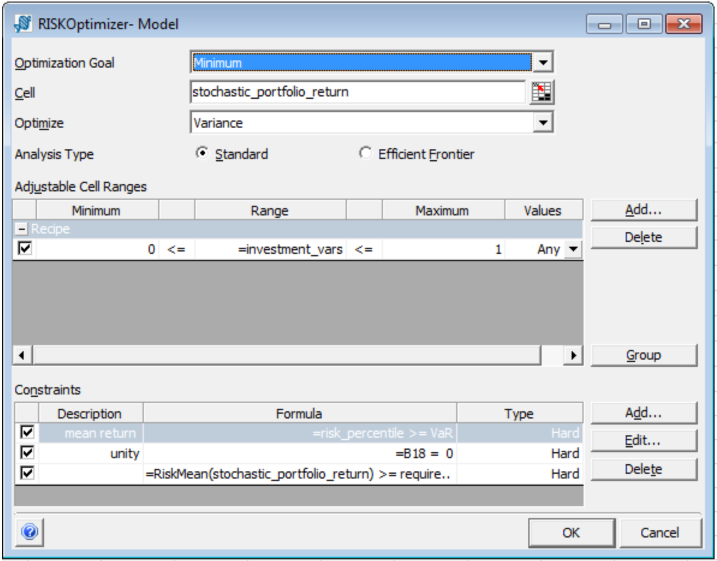
See **markowRiskOptimizerVar_1.xlsx** for Method 1.

See **markowRiskOptimizerVar_2.xlsx** for Method 2.

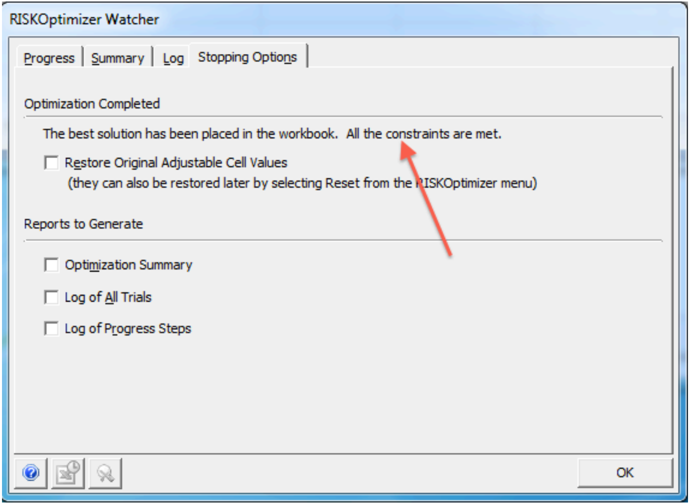
See the range **A39:B42**.

VaR

In both cases the model is



Make Sure You Have a Feasible Solution



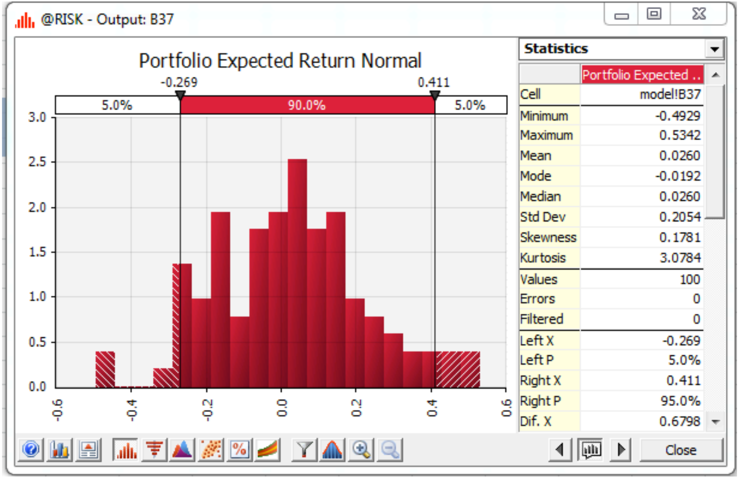
VaR

Model Solution:

	Solution $\mu = .025$	**Solution** $\mu = .12$
X	0.5292	0
Y	0.1333	0.2100
Z	0.3375	0.7900
VaR (0.05)	-0.27	-0.62

VaR

Model Solution: We now have



Note the improvement over -.622.

CVaR

There are problems with VaR.

Assume we have a VaR constraint that there is a value at risk of $.30 at 5 percent.

Every return in the bottom 5 percent tail *gets the same weight*.

All returns below -.3 count the same.

The Conditional Value at Risk (CVaR), or expected shortfall, does not give the same weight to each outcome.

For a given value of α, the CVaR is the average value of the worst $\alpha\%$ of possible outcomes.

CVaR

First, two important @RISK functions.

1. **RiskTruncate(min, max):** returns the sample results for the distribution that lie between **min** and max.

2. **RiskTruncateP(0, α):** returns the sample results for the distribution in the lower α percentile.

CVaR

CVaR Calculation: Let $f(X) = X$.

X	f(X)	g(X)
43	43	0.1
44	44	0.2
45	45	0.2
46	46	0.4
47	47	0.1

If $\alpha = 0.3$ then

$$CVaR = (1/3) * 43 + (2/3) * 44 = 43.67$$

We say the CVaR is 43.67 for the 30% tail.

See **tailExamples.xlsx** for the simulation.

CVaR

The CVaR is easy to calculate in @RISK. See
`http://kb.palisade.com/index.php?pg=kb.page&id=171`.

Use RiskTruncateP(0, α) to return the lower α percent of the tail.

For example,

```
=RiskMean(B12,RiskTruncateP(0,0.3))
```

will return the average of the lower 30% of the simulation outcomes where cell `B12` is the distribution output cell.

See the spreadsheet **CVaR** in **tailExamples.xlsx**.

CVaR

Practice Exam Question: Build the Markowitz model to minimize variance subject to

1. An expected return of at least 2.5%.
2. The unity constraint.
3. The CVaR of the worst 5% of returns is greater than or equal to -.5.

Summary

RISK Optimizer allows you to build models with **random variables.**

Also, at long last, you can now put IF statements in your model like this:

```
IF(A1 > 100, B1, C1)
```

where `A1` is adjustable. Risk Optimizer can handle IF statements!

Yes, *I know exactly what you are thinking*. Go ahead and go ballistic!

Professor Martin, are you $*&# kidding me!

You are a $*&# idiot!!!!

The toughest person in this room should beat your sorry Okole!

Why are you telling me this in Week 10?

Markowitz Mean-Variance Model

Model Results: Look at the "optimal" value for column 3. It should be .1677 **NOT** .1901.

RISK Optimizer failed when we **went stochastic**!

	Solver Solution	**RISK Optimizer Simulation Solution**
X	0.1169	0.0000
Y	0.0000	0.2100
Z	0.88311	0.7900
Variance	0.1677	0.1901

Summary

1. RISK Optimizer is **NOT** an optimization solver.

2. RISK Optimizer typically **WILL NOT** find the optimal solution.

3. RISK Optimizer implements heuristic algorithms.

4. RISK Optimizer is **SLOW**.

PAU!