This Week

Monday

- Modeling and Solving Nonlinear Optimization Problems

Wednesday

- Lab Exercise: Portfolio Optimization

Topics

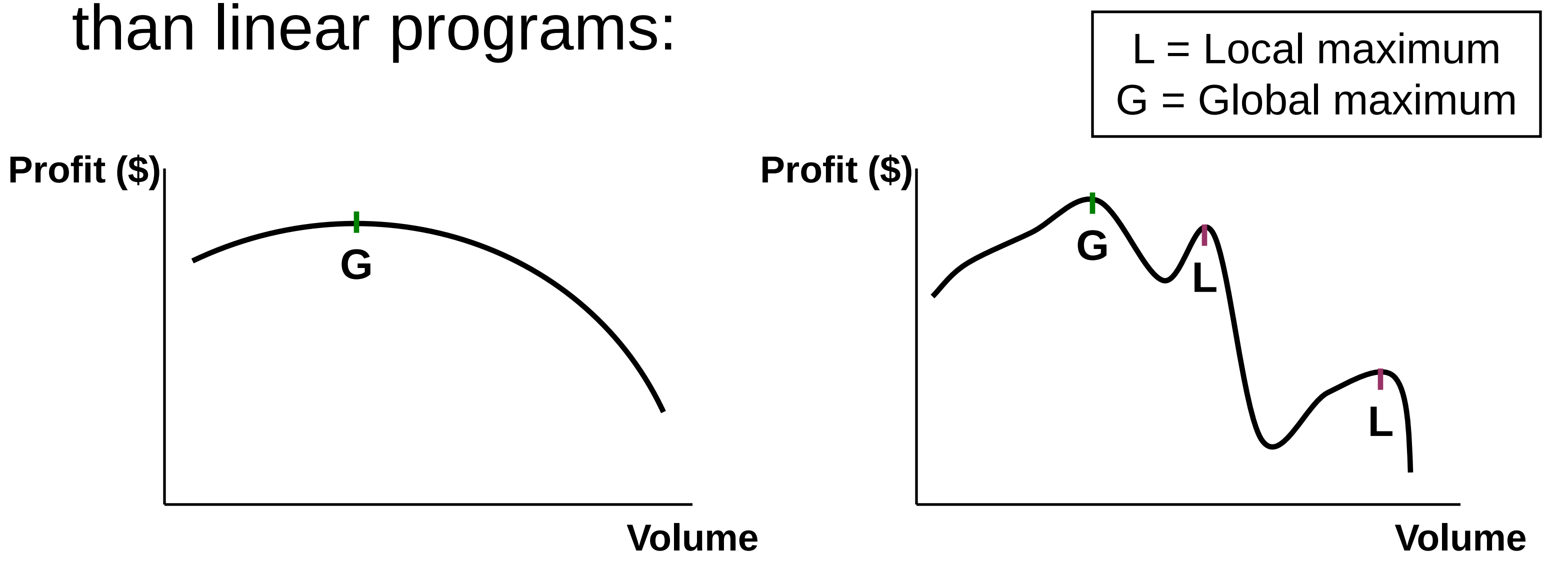
- **Why Some Problems are Harder to Solve than Others**
- **ILPs from Week 6 Lab Exercise**
- **Portfolio Optimization**

A Problem Solving Framework

1. *Define the Problem*
2. *Collect and Organize Data*
3. *Characterize Uncertainty and Data Relationships*
4. *Build an Evaluation Model*
5. *Formulate a Solution Approach*
6. *Evaluate Potential Solutions*
7. *Recommend a Course of Action*

"Hard" Optimization Problems

- While many types of business problems can be characterized using linear programming models, there are many others that have ***nonlinear properties*** and must be solved using different methods.
- Depending on the underlying structure of the model, these problems can be ***significantly harder to solve*** than linear programs:



Some Classes of "Hard" Optimization Problems

- Integer Linear Programming (ILP) Models
  - Variables are restricted to integer values instead of a continuous range of values.
  - e.g. $x_1 \in \{0,1,2, \ldots\}$ instead of $x_1 \geq 0$.

- Nonlinear Programming (NLP) Models
  - Objective function and/or constraints have nonlinear terms.
  - e.g., Minimize $\dfrac{100{,}000}{Q} + 10Q$ or $4\, x_1 x_2 \geq 100$.

LPs are frequently used to approximate and/or to solve sub-problems of ILPs and NLPs.

Some Practical Applications

- ILP Applications
  - ***Box Packing***
  - ***Workforce scheduling***
  - Vehicle routing
  - Facility location selection
- NLP Applications
  - ***Inventory optimization***
  - ***Workshop planning with uncertain attendance***
  - ***Portfolio optimization***
  - Price-setting in elastic demand environments
  - Determining IRR for cash flow models over time

Box Packing

Recall from lab: Pack items into the ***minimum number of boxes without exceeding the box capacities***:

Total Boxes Used:	6									
Packing Assignment Matrix			Boxes							
	Weight (lbs)	Item	A	B	C	D	E	F	G	Item Packed
Items to be Packed	1.2	1	0	0	1	0	0	0	0	1
	2.3	2	0	0	0	1	0	0	0	1
	3.8	3	0	0	0	0	0	1	0	1
	3.3	4	0	0	0	0	0	1	0	1
	1.1	5	0	0	1	0	0	0	0	1
	3.8	6	0	0	0	0	0	0	1	1
	2.3	7	0	0	1	0	0	0	0	1
	0.2	8	0	0	0	0	0	0	1	1
	0.8	9	0	0	1	0	0	0	0	1
	3.2	10	1	0	0	0	0	0	0	1
	3.1	11	0	0	0	0	0	0	1	1
	3.5	12	0	0	0	1	0	0	0	1
	0.6	13	0	0	0	0	0	1	0	1
	2.3	14	0	0	0	0	0	0	1	1
	0.9	15	0	0	0	0	0	1	0	1
	2.7	16	0	0	0	0	1	0	0	1
	3.6	17	0	0	0	1	0	0	0	1
	3.2	18	0	0	0	0	1	0	0	1
	4.3	19	0	0	1	0	0	0	0	1
	3.2	20	1	0	0	0	0	0	0	1
	Box Used		1	0	1	1	1	1	1	
	Current Weight of Box		6.400	0.000	9.700	9.400	5.900	8.600	9.400	
	Maximum Weight for Box		10.0	10.0	10.0	10.0	10.0	10.0	10.0	
	% of Items in Box		10%	0%	25%	15%	10%	20%	20%	100% =% of Items Packed

Problem Formulation

Let the ***decision variable*** $\boldsymbol{x_{ij}}$ indicate whether item *i* is placed in box *j*. Let the ***decision variable*** $\boldsymbol{y_j}$ indicate whether box *j* is in use. The problem then becomes:

$$
\begin{array}{lll}
\text{Minimize} & \displaystyle\sum_{j=A}^{J} y_j & \text{(Total Boxed Used)} \\
\text{subject to} & : & \\
& \displaystyle\sum_{j=A}^{J} x_{ij} = 1 & \text{for all items } i \text{ (Item Assignment)} \\
& \displaystyle\sum_{i=1}^{20} \frac{x_{ij}}{20} \le y_j & \text{for all boxes } j \text{ (Box Usage)} \\
& \displaystyle\sum_{i=1}^{20} w_i x_{ij} \le C_j & \text{for all boxes } j \text{ (Box Capacity)} \\
& 1 \ge x_{ij} \ge 0 \text{ and INTEGER} & \text{for all } i, j \\
& 1 \ge y_j \ge 0 \text{ and INTEGER} & \text{for all } j
\end{array}
$$

Solving IPs With Excel Solver

Solver can be used to solve ILPs and NLPs in the same way it is used to solve LPs, provided the data are organized so that you have one cell representing each "piece" of the formulated problem:

Total Boxes Used: 6

Packing Assignment Matrix			Boxes							
	Weight (lbs)	Item	A	B	C	D	E	F	G	Item Packed
Items to be Packed	1.2	1	0	0	1	0	0	0	0	1
	2.3	2	0	0	0	1	0	0	0	1
	3.8	3	0	0	0	0	0	1	0	1
	3.3	4	0	0	0	0	0	1	0	1
	1.1	5	0	0	1	0	0	0	0	1
	3.8	6	0	0	0	0	0	0	1	1
	2.3	7	0	0	1	0	0	0	0	1
	0.2	8	0	0	0	0	0	0	1	1
	0.8	9	0	0	1	0	0	0	0	1
	3.2	10	1	0	0	0	0	0	0	1
	3.1	11	0	0	0	0	0	0	1	1
	3.5	12	0	0	0	1	0	0	0	1
	0.6	13	0	0	0	0	0	1	0	1
	2.3	14	0	0	0	0	0	0	1	1
	0.9	15	0	0	0	0	0	1	0	1
	2.7	16	0	0	0	0	1	0	0	1
	3.6	17	0	0	0	1	0	0	0	1
	3.2	18	0	0	0	0	1	0	0	1
	4.3	19	0	0	1	0	0	0	0	1
	3.2	20	1	0	0	0	0	0	0	1
	Box Used		1	0	1	1	1	1	1	
	Current Weight of Box		6.400	0.000	9.700	9.400	5.900	8.600	9.400	
	Maximum Weight for Box		10.0	10.0	10.0	10.0	10.0	10.0	10.0	
	% of Items in Box		10%	0%	25%	15%	10%	20%	20%	100% =% of Items Packed

Minimize $\sum_{j=A}^{J} y_j$ (Total Boxed Used)

subject to :

$$\sum_{j=A}^{J} x_{ij} = 1 \quad \text{for all items } i \text{ (Item Assignment)}$$

$$\sum_{i=1}^{20} \frac{x_{ij}}{20} \le y_j \quad \text{for all boxes } j \text{ (Box Usage)}$$

$$\sum_{i=1}^{20} w_i x_{ij} \le C_j \quad \text{for all boxes } j \text{ (Box Capacity)}$$

$$1 \ge x_{ij} \ge 0 \text{ and INTEGER} \quad \text{for all } i, j$$

$$1 \ge y_j \ge 0 \text{ and INTEGER} \quad \text{for all } j$$

An Optimal Integer Solution

Total Boxes Used: **5**

Packing Assignment Matrix			Boxes							
	Weight (lbs)	Item	A	B	C	D	E	F	G	Item Packed
Items to be Packed	1.2	1	0	0	0	0	1	0	0	1
	2.3	2	0	0	0	0	1	0	0	1
	3.8	3	1	0	0	0	0	0	0	1
	3.3	4	0	1	0	0	0	0	0	1
	1.1	5	1	0	0	0	0	0	0	1
	3.8	6	0	1	0	0	0	0	0	1
	2.3	7	0	0	0	0	1	0	0	1
	0.2	8	0	1	0	0	0	0	0	1
	0.8	9	1	0	0	0	0	0	0	1
	3.2	10	0	0	1	0	0	0	0	1
	3.1	11	0	0	0	1	0	0	0	1
	3.5	12	0	0	0	1	0	0	0	1
	0.6	13	0	0	0	0	1	0	0	1
	2.3	14	0	0	0	0	1	0	0	1
	0.9	15	0	0	0	0	1	0	0	1
	2.7	16	0	1	0	0	0	0	0	1
	3.6	17	0	0	1	0	0	0	0	1
	3.2	18		0	1	0	0	0	0	1
	4.3	19	1	0	0	0	0	0	0	1
	3.2	20	0	0	0	1	0	0	0	1
	Box Used		1	1	1	1	1	0	0	
	Current Weight of Box		10.000	10.000	10.000	9.800	9.600	0.000	0.000	
	Maximum Weight for Box		10.0	10.0	10.0	10.0	10.0	10.0	10.0	
	% of Items in Box		20%	20%	15%	15%	30%	0%	0%	100% =% of Items Packed

This solution was constructed *manually*. After running simplex-based branch & bound for several minutes on my desktop PC, Solver was unable to find a 5-box solution. This is NOT uncommon, even for relatively small ILP instances (like this one).

An Optimal Relaxed Solution

(y_js integer only)

Total Boxes Used: 5

Packing Assignment Matrix			Boxes							
	Weight (lbs)	Item	A	B	C	D	E	F	G	Item Packed
Items to be Packed	1.2	1	0	0	0	1	0	0	0	1
	2.3	2	0	0	1	0	0	0	0	1
	3.8	3	0	0	0	0.7632	0.2368	0	0	1
	3.3	4	0.9091	0	0.0909	0	0	0	0	1
	1.1	5	0.0909	0.9091	0	0	0	0	0	1
	3.8	6	0	0	0	1	0	0	0	1
	2.3	7	0	1	0	0	0	0	0	1
	0.2	8	0	1	0	0	0	0	0	1
	0.8	9	0	0	1	0	0	0	0	1
	3.2	10	0	1	0	0	0	0	0	1
	3.1	11	0	0	1	0	0	0	0	1
	3.5	12	0	0	1	0	0	0	0	1
	0.6	13	0	1	0	0	0	0	0	1
	2.3	14	0	0	0	0	1	0	0	1
	0.9	15	0	0	0	1	0	0	0	1
	2.7	16	0	1	0	0	0	0	0	1
	3.6	17	0	0	0	0	1	0	0	1
	3.2	18	1	0	0	0	0	0	0	1
	4.3	19	0.721	0	0	0.279	0	0	0	1
	3.2	20	0	0	0	0	1	0	0	1
	Box Used		1	1	1	1	1	0	0	
	Current Weight of Box		9.400	10.000	10.000	10.000	10.000	0.000	0.000	
	Maximum Weight for Box		10.0	10.0	10.0	10.0	10.0	10.0	10.0	
	% of Items in Box		14%	30%	20%	20%	16%	0%	0%	100% =% of Items Packed

After removing the integer constraints on the x_{ij}s and leaving the y_js as the only binary variables, Solver was able to find a 5-box solution to the relaxed problem quickly (although it still did not recognize this solution as optimal).

Workforce Scheduling

- Recall from lab**:** Construct a schedule that ***minimizes the total number of cashiers while adhering to the shift rules and meeting the minimum cashier requirements of each 4-hour time slot.***

Shift Coverage

Monday (M)			Tuesday (T)			Wednesday (W)			Thursday (R)			Friday (F)			Saturday (S)			Sunday (N)		
9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm
6	5	8	6	5	8	6	5	8	6	5	8	3	8	4	10	14	7	4	12	6

- The key to the problem is recognizing that there are 14 possible shift choices and, hence, 14 decision variables.

  7 “Morning” Schedules: 9am-5pm M-F, T-S, W-N…

  7 “Evening” Schedules: 1pm-9pm M-F, T-S, W-N…

Problem Formulation

x_{ij} = The number of employees whose 5-day work schedule begins on day i (i = M, T, , ..., N) and covers shift time j, where j = 1 denotes 9am - 5pm and j = 2 denotes 1pm - 9pm.

$$\text{Minimize} \quad \sum_{\mathrm{i=M}}^{\mathrm{N}} \sum_{\mathrm{j=1}}^{2} x_{ij} \qquad \text{(Total Employees)}$$

subject to :

$$\sum_{\mathrm{k}=i-4}^{\mathrm{i}} x_{k1} \geq N_{it} \;\; \forall \mathrm{i} = \mathrm{M}, \ldots \mathrm{N}, \mathrm{t} = 9\mathrm{am} \quad \text{(Morning Staffing Needs)}$$

$$\sum_{\mathrm{k}=i-4}^{\mathrm{i}} (x_{k1} + x_{k2}) \geq N_{it} \;\; \forall \mathrm{i} = \mathrm{M}, \ldots \mathrm{N}, \mathrm{t} = 1\mathrm{pm} \quad \text{(Midday Staffing Needs)}$$

$$\sum_{\mathrm{k}=i-4}^{\mathrm{i}} x_{k2} \geq N_{it} \;\; \forall \mathrm{i} = \mathrm{M}, \ldots \mathrm{N}, \mathrm{t} = 5\mathrm{pm} \quad \text{(Evening Staffing Needs)}$$

$$x_{\mathrm{ij}} \geq 0 \;\; \text{and INTEGER}, \forall \mathrm{i} = \mathrm{M}, \ldots \mathrm{N}, \mathrm{j} = 1,2$$

Optimal Integer Solution

Shift Coverage

	Monday (M)			Tuesday (T)			Wednesday (W)			Thursday (R)			Friday (F)			Saturday (S)			Sunday (N)		
	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm	9am-1pm	1pm-5pm	5pm-9pm
Total Needed:	6	5	8	6	5	8	6	5	8	6	5	8	3	8	4	10	14	7	4	12	6
Total Scheduled:	6	14	8	8	16	8	8	16	8	6	14	8	6	14	8	10	19	9	6	12	6

Number of Cashiers for each Shift Schedule

Morning M-F	**0**
Morning T-S	**4**
Morning W-N	**0**
Morning R-M	**2**
Morning F-T	**0**
Morning S-W	**4**
Morning N-R	**0**
Evening M-F	**0**
Evening T-S	**2**
Evening W-N	**1**
Evening R-M	**1**
Evening F-T	**1**
Evening S-W	**2**
Evening N-R	**4**
Total:	**21**

In contrast to Box Packing, Solver found this solution to the Shift Scheduling problem and recognized it as optimal almost instantly.

Note: This solution is *not* unique. There are multiple optimal solutions.

Portfolio Optimization

"October. This is one of the peculiarly dangerous months to speculate in stocks. The others are July, January, September, April, November, May, March, June, December, August and February."

— Mark Twain

Investment Risk and Return

- Risk - The ***potential for loss*** due to variability.
- For individual securities (stocks, bonds, etc.), risk is measured by the ***variance*** or ***standard deviation*** of future returns.
- Let $\boldsymbol{R_j}$ **= *rate of return of security j*** over a specified time period.
- R_j is a random variable. We denote its mean and variance by $\mu_j = E[R_j]$ and $\sigma_j^2 = Var[R_j]$.

Security Selection

- Given a choice of a single security *j* from a set of securities, most rational investors would look for a security *j* with a large μ_j and a small σ_j (relative to μ_j).

- When investing in a ***portfolio*** of securities, usually we can do better than simply choosing securities based on their individual merits. That is, by carefully choosing securities that are not all strongly correlated, we can ***reduce risk through diversification***.

Portfolio Rate of Return

- Suppose that among N securities $j = 1,\ldots,N$, we invest fraction w_j of our wealth in security j, so that:

$$w_1 + w_2 + \ldots + w_N = 1$$

- Then our ***portfolio rate of return, R_P***, is defined as:

$$R_P = \sum_{j=1}^{N} w_j R_j$$

- R_P is a random variable with mean:

$$\boxed{\mu_P = E[R_P] = E[\sum_{j=1}^{N} w_j R_j] = \sum_{j=1}^{N} w_j E[R_j] = \sum_{j=1}^{N} w_j \mu_j}$$

Portfolio Variance

- The ***variance of R_P*** is given by:

$$\sigma_P^{\ 2} = Var[R_P] = Var[\sum_{j=1}^{N} w_j R_j] = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j Cov(R_i, R_j)$$

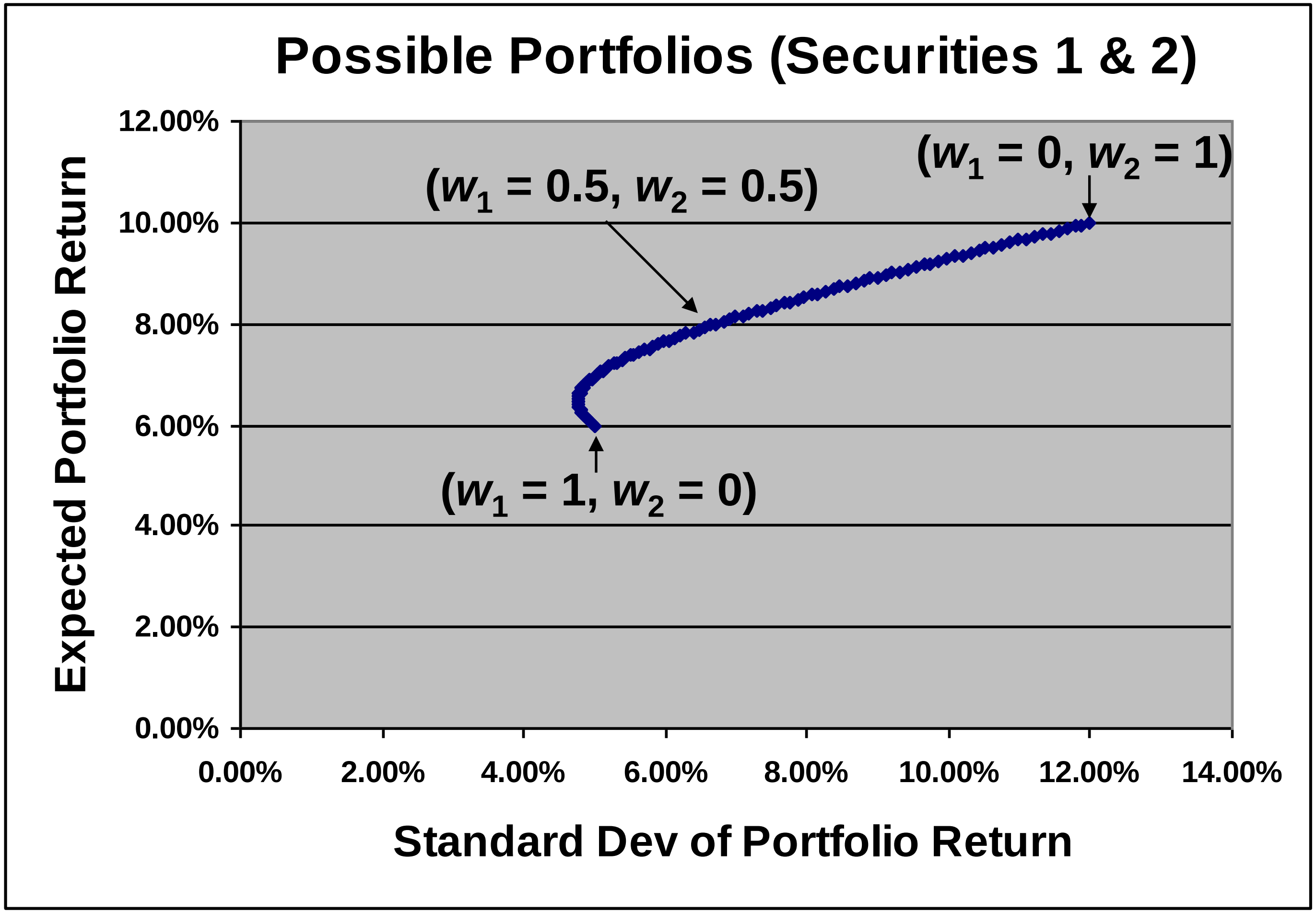
- Since $Cov(R_i, R_j) = \rho_{ij}\,\sigma_i\,\sigma_j$, where ρ_{ij} denotes the ***correlation*** between R_i with R_j, we have:

$$\boxed{\sigma_P^{\ 2} = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \rho_{ij} \sigma_i \sigma_j}$$

- Recall that $-1 \leq \rho_{ij} \leq 1$ for all i and j.

Example: Two Securities

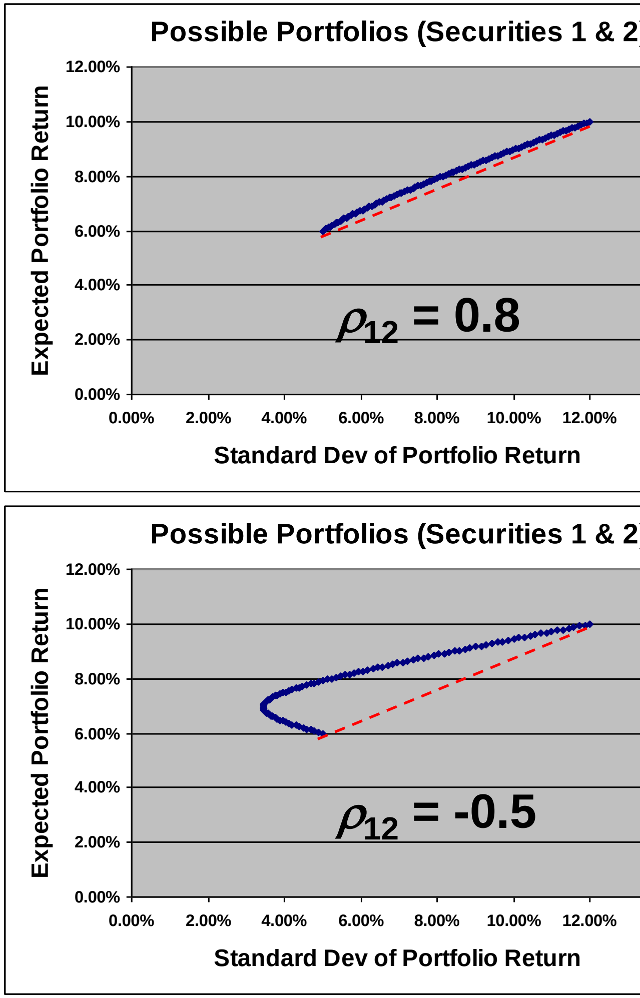
$(\sigma_1, \mu_1) = (5\%, 6\%)$ $\quad$ $(\sigma_2, \mu_2) = (12\%, 10\%)$ $\quad$ $\rho_{12} = 0.1$



Diversification

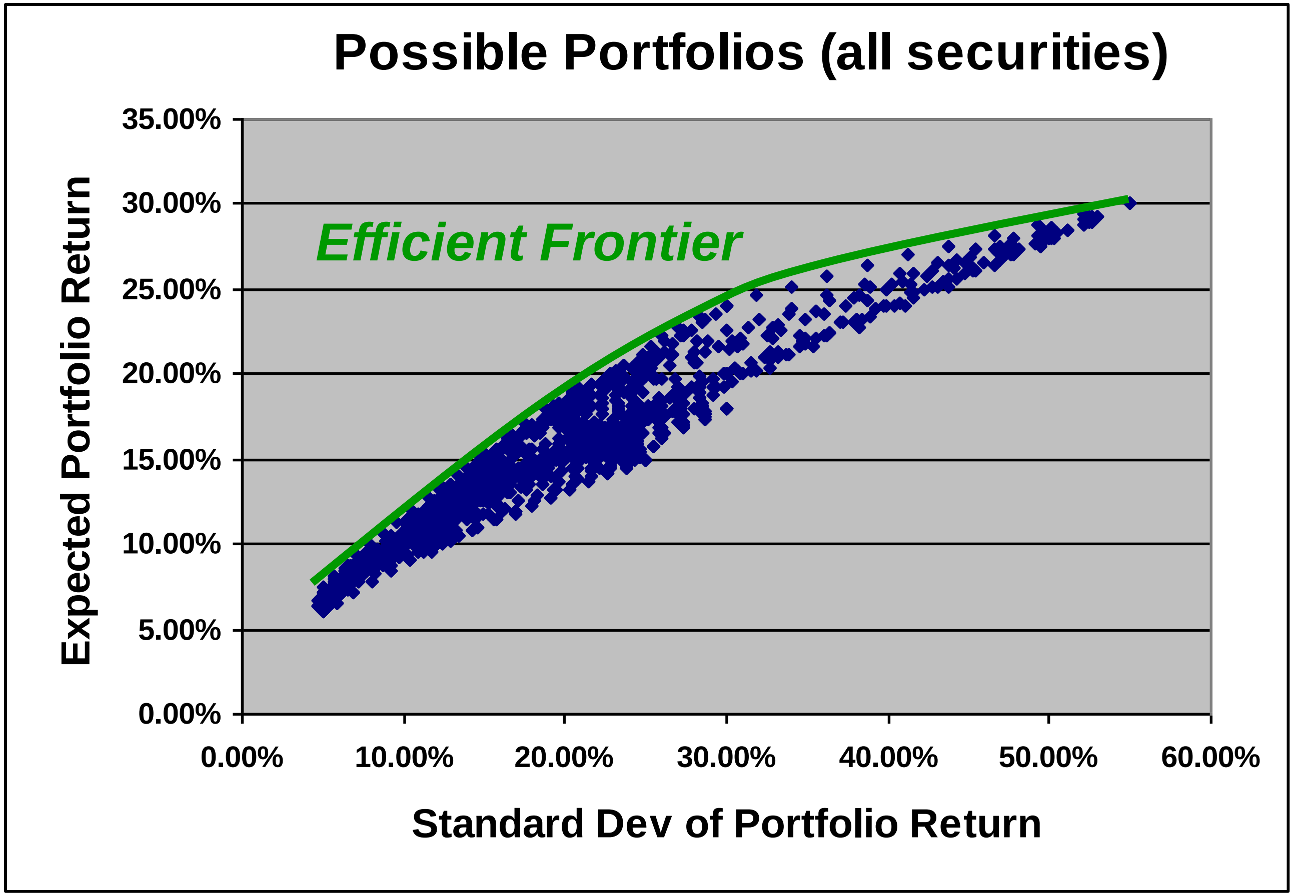
- When $\rho_{12} < 1$, the portfolio standard deviation σ_P satisfies:

$$\sigma_P < w_1\sigma_1 + w_2\sigma_2 \ (\text{- - -}).$$



Efficient Frontier

- A portfolio *P* is ***efficient*** if there is ***no other portfolio with a higher return AND a lower standard deviation***.
- The ***efficient frontier*** is the set of efficient portfolios.



Portfolio Optimization

- Given a target average rate of return, *TARR*, ***find the lowest-risk portfolio P that satisfies $\mu_P \geq TARR$***:

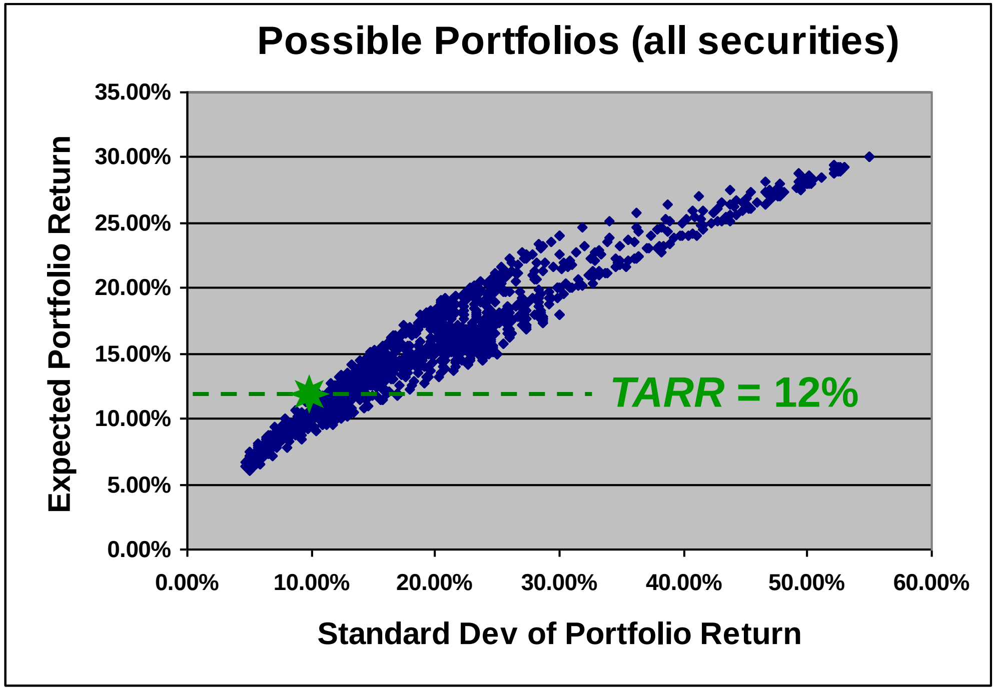
$$\text{Minimize} \quad \sqrt{\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \rho_{ij} \sigma_i \sigma_j} \qquad (\text{i.e., } \sigma_P)$$

subject to:

$$\sum_{j=1}^{N} w_j \mu_j \geq TARR$$

$$\sum_{j=1}^{N} w_j = 1$$

$$w_j \geq 0 \quad \text{for all } j = 1, \ldots N.$$



Portfolio Optimization

- OR, given a target maximum risk level, *TRISK*, ***find the highest-return portfolio P that satisfies $\sigma_P \leq TRISK$***:

$$\text{Maximize} \quad \sum_{j=1}^{N} w_j \mu_j \qquad (\text{i.e., } \mu_P)$$

subject to:

$$\sqrt{\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \rho_{ij} \sigma_i \sigma_j} \leq TRISK$$

$$\sum_{j=1}^{N} w_j = 1$$

$$w_j \geq 0 \quad \text{for all } j = 1, \ldots N.$$