Name: Sugumar Prabhakaran **Course**: APS502 - Winter 2021
Student ID: 994126815 **Due Date**: 16 Apr 2021, 5pm EST

FINANCIAL ENGINEERING I - COMPUTATIONAL PROJECT

PROBLEM 1

PART 1. Find lowest-cost bond portfolio that covers the stream of liabilities as per table below:

Year	**1**	**2**	**3**	**4**	**5**	**6**
Obligation	500	200	800	400	700	900

1.1.1 Mathematical Formulation

Notation:
- s_j: spot rate from present to year j, where $j = 1, \ldots, 6$
- $f_{j,k}$: forward rate between years j and k
- y_j: obligation (liability) in year j
- p_i: price of bond i, where $i = 1, \ldots, 13$
- c_{ij}: cash flow from bond i in year j

We need to calculate the following forward rates: $f_{1,2},\ f_{2,3},\ f_{3,4},\ f_{4,5},\ f_{5,6}$, using:

$$f_{j,k} = \left[\frac{(1+s_k)^k}{(1+s_j)^j}\right]^{\frac{1}{k-j}} - 1$$

Therefore: $f_{1,2} = 2.0\%,\ f_{2,3} = 3.0\%,\ f_{3,4} = 4.0\%,\ f_{4,5} = 5.0\%,\ f_{5,6} = 6.0\%$

Decision Variables:
- x_i: quantity of bond i held in the portfolio
- z_j: carry forward amount from years j to $j+1$

Objective Function:

general form: (total bond portfolio cost)

$$\min \sum_{i=1}^{13} (p_i \cdot x_i) = \min(p_1 x_1 + \cdots + p_{13} x_{13})$$

$$\min(108x_1 + 94x_2 + 99x_3 + 92.7x_4 + 96.6x_5 + 95.9x_6 + 92.9x_7 + 110x_8 + 104x_9 + 101x_{10} + 107x_{11} + 102x_{12} + 95.2x_{13})$$

Subject to (constraints):

general form: (yearly obligation constraints)

$$\sum_{i=1}^{13} (c_{ij} \cdot x_i) + (1 + f_{j-1,j}) z_{j-1} - z_j \geq y_j,\ \forall j = 1, \ldots, 6, \text{ and } z_0,\ z_6 = 0$$

$$10x_1 + 7x_2 + 8x_3 + 6x_4 + 7x_5 + 6x_6 + 5x_7 + 10x_8 + 8x_9 + 6x_{10} + 10x_{11} + 7x_{12} + 100x_{13} - z_1 \geq 500$$

$$10x_1 + 7x_2 + 8x_3 + 6x_4 + 7x_5 + 6x_6 + 5x_7 + 10x_8 + 8x_9 + 6x_{10} + 110x_{11} + 107x_{12} + (1.02)z_1 - z_2 \geq 200$$

$$10x_1 + 7x_2 + 8x_3 + 6x_4 + 7x_5 + 6x_6 + 5x_7 + 110x_8 + 108x_9 + 106x_{10} + (1.03)z_2 - z_3 \geq 800$$

$$10x_1 + 7x_2 + 8x_3 + 6x_4 + 7x_5 + 106x_6 + 105x_7 + (1.04)z_3 - z_4 \geq 400$$

$$10x_1 + 7x_2 + 8x_3 + 106x_4 + 107x_5 + (1.05)z_4 - z_5 \geq 700$$

$$100x_1 + 107x_2 + 108x_3 + (1.06)z_5 \geq 900$$

general form: (bond qty and carryover amount domain constraints)

$$x_i, z_j \geq 0, \qquad \forall i = 1, \ldots, 13 \text{ and } j = 1, \ldots, 6$$

$$x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7,\ x_8,\ x_9,\ x_{10},\ x_{11},\ x_{12},\ x_{13} \geq 0$$

$$z_1,\ z_2,\ z_3,\ z_4,\ z_5 \geq 0$$

1.1.2 Matrix Formulation

Objective function vector form: $f^{\mathbf{T}} \cdot \mathbf{x}$

$$f^{\mathbf{T}} = \begin{bmatrix} 108 & 94 & 99 & 92.7 & 96.6 & 95.9 & 92.9 & 110 & 104 & 101 & 107 & 102 & 95.2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$
$$\mathbf{x}^{\mathbf{T}} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} & x_{12} & x_{13} & z_1 & z_2 & z_3 & z_4 & z_5 \end{bmatrix}$$

Inequality constraints vector form: $\mathbf{A} \cdot \mathbf{x} \leq \mathbf{b}$

$$A = \begin{bmatrix} -10 & -7 & -8 & -6 & -7 & -6 & -5 & -10 & -8 & -6 & -10 & -7 & -100 & 1 & 0 & 0 & 0 & 0 \\ -10 & -7 & -8 & -6 & -7 & -6 & -5 & -10 & -8 & -6 & -110 & -107 & 0 & -1.02 & 1 & 0 & 0 & 0 \\ -10 & -7 & -8 & -6 & -7 & -6 & -5 & -110 & -108 & -106 & 0 & 0 & 0 & 0 & -1.03 & 1 & 0 & 0 \\ -10 & -7 & -8 & -6 & -7 & -106 & -105 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.04 & 1 & 0 \\ -10 & -7 & -8 & -106 & -107 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.05 & 1 \\ -100 & -107 & -108 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.06 \end{bmatrix}$$

$$\mathbf{b}^{\mathbf{T}} = \begin{bmatrix} -500 & -200 & -800 & -400 & -700 & -900 \end{bmatrix}$$

Domain constraints vector form: $\mathbf{lb} \leq \mathbf{x}$

$$\mathbf{lb} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

1.1.3 Matlab Solution

The full MATLAB code is located in the **Appendix**. Here are the results from ***linprog***:

Quantity of bond and carry over amounts:

$$\mathbf{x}^{\mathbf{T}} = \begin{bmatrix} 0 & 8.4112 & 0 & 0 & 5.9918 & 2.8224 & 0 & 0 & 6.3171 & 0 & 0.2883 & 0 & 3.2883 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Objective function value - (total bond portfolio cost): $f_{val} =$ **$2641.00.** Note that there is no carryover between years.

PART 2. Add a constraint that at most 50% of the bond portfolio's value is in B-rated bonds:

1.2.1 Mathematical Formulation

general form: (50% total portfolio value in B-rated bonds constraint)

$$\frac{(p_1x_1+p_2x_2+p_3x_3+p_4x_4+p_5x_5+p_6x_6)}{(p_1x_1+p_2x_2+p_3x_3+p_4x_4+p_5x_5+p_6x_6+p_7x_7+p_8x_8+p_9x_9+p_{10}x_{10}+p_{11}x_{11}+p_{12}x_{12}+p_{13}x_{13})} \leq 0.5$$

$$p_1x_1 + p_2x_2 + p_3x_3 + p_4x_4 + p_5x_5 + p_6x_6 - p_7x_7 - p_8x_8 - p_9x_9 - p_{10}x_{10} - p_{11}x_{11} - p_{12}x_{12} - p_{13}x_{13} \leq 0$$

$$108x_1 + 94x_2 + 99x_3 + 92.7x_4 + 96.6x_5 + 95.9x_6 - 92.9x_7 - 110x_8 - 104x_9 - 101x_{10} - 107x_{11} - 102x_{12} - 95.2x_{13} \leq 0$$

1.2.2 Matrix Formulation

Update Inequality constraint matrix and vector:

$$\mathbf{A} = \begin{bmatrix} -10 & -7 & -8 & -6 & -7 & -6 & -5 & -10 & -8 & -6 & -10 & -7 & -100 & 1 & 0 & 0 & 0 & 0 \\ -10 & -7 -8 & -6 & -7 & -6 & -5 & -10 & -8 & -6 & -110 & -107 & 0 & -1.02 & 1 & 0 & 0 & 0 & \\ -10 & -7 & -8 & -6 & -7 & -6 & -5 & -110 & -108 & -106 & 0 & 0 & 0 & 0 & -1.03 & 1 & 0 & 0 \\ -10 & -7 & -8 & -6 & -7 & -106 & -105 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.04 & 1 & 0 \\ -10 & -7 & -8 & -106 & -107 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.05 & 1 \\ -100 & -107 & -108 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.06 \\ 108 & 94 & 99 & 92.7 & 96.6 & 95.9 & -92.9 & -110 & -104 & -101 & -107 & -102 & -95.2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{b^T} = \begin{bmatrix} -500 & -200 & -800 & -400 & -700 & -900 & 0 \end{bmatrix}$$

1.2.3 Matlab Solution

The full MATLAB code is located in the **Appendix**. Here are the results from ***linprog***:

Quantity of bond and carry over amounts:

$$\mathbf{x^T} = \begin{bmatrix} 0 & 8.4112 & 0 & 0 & 5.5027 & 0 & 3.3566 & 0 & 6.3502 & 0 & 0.3183 & 0 & 3.3183 & 0 & 0 & 0 & \$49.84 & 0 \end{bmatrix}$$

Objective function value - (total bond portfolio cost): $f_{val} =$ **$2644.40.** Note there is a small carry over from year 4 to year 5.

PART 3. Repeat part 2 but with at most 25% of the bond portfolio's value in B-rated bonds:

1.3.1 Mathematical Formulation

general form: (25% total portfolio in B-rated bonds)

$$\frac{(p_1x_1+p_2x_2+p_3x_3+p_4x_4+p_5x_5+p_6x_6)}{(p_1x_1+p_2x_2+p_3x_3+p_4x_4+p_5x_5+p_6x_6+p_7x_7+p_8x_8+p_9x_9+p_{10}x_{10}+p_{11}x_{11}+p_{12}x_{12}+p_{13}x_{13})} \leq 0.25$$

$$3p_1x_1 + 3p_2x_2 + 3p_3x_3 + 3p_4x_4 + 3p_5x_5 + 3p_6x_6 - p_7x_7 - p_8x_8 - p_9x_9 - p_{10}x_{10} - p_{11}x_{11} - p_{12}x_{12} - p_{13}x_{13} \leq 0$$

$$324x_1 + 282x_2 + 297x_3 + 278.1x_4 + 289.8x_5 + 287.7x_6 - 92.9x_7 - 110x_8 - 104x_9 - 101x_{10} - 107x_{11} - 102x_{12} - 95.2x_{13} \leq 0$$

1.3.2 Matrix Formulation

Update Inequality constraint matrix and vector:

$$\mathbf{A} = \begin{bmatrix} -10 & -7 & -8 & -6 & -7 & -6 & -5 & -10 & -8 & -6 & -10 & -7 & -100 & 1 & 0 & 0 & 0 & 0 \\ -10 & -7-8 & -6 & -7 & -6 & -5 & -10 & -8 & -6 & -110 & -107 & 0 & -1.02 & 1 & 0 & 0 & 0 & \\ -10 & -7 & -8 & -6 & -7 & -6 & -5 & -110 & -108 & -106 & 0 & 0 & 0 & 0 & -1.03 & 1 & 0 & 0 \\ -10 & -7 & -8 & -6 & -7 & -106 & -105 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.04 & 1 & 0 \\ -10 & -7 & -8 & -106 & -107 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.05 & 1 \\ -100 & -107 & -108 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.06 \\ 324 & 282 & 297 & 278.1 & 289.8 & 287.7 & -92.9 & -110 & -104 & -101 & -107 & -102 & -95.2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{b^T} = \begin{bmatrix} -500 & -200 & -800 & -400 & -700 & -900 & 0 \end{bmatrix}$$

1.3.3 Matlab Solution

The full MATLAB code is located in the **Appendix**. Here are the results from ***linprog***:

Quantity of bond and carry over amounts:

$$\mathbf{x^T} = \begin{bmatrix} 0 \ 7.1271 \ 0 \ 0 \ 0 \ 0 \ 10.4068 \ 0 \ 6.4637 \ 0 \ 0.4215 \ 0 \ 3.4215 \ 0 \ 0 \ 0 \ \$742.60 \ \$129.62 \end{bmatrix}$$

Objective function value - (total bond portfolio cost): $f_{val} =$ **\$2679.80.** Note there is a carry over from year 4 to 5 and year 5 to 6.

1.4 Summary

The cost of the three portfolios are summarized in the table below. Portfolio 1 is the least expensive and portfolio 3 is the most expensive. However, portfolio 1 has a much higher quantity of B-rated (riskier) bonds.

	Description	Total Portfolio Cost
Portfolio 1	No additional constraints	$2641.00
Portfolio 2	At most 50% value in B-rated bonds	$2644.40
Portfolio 3	At most 25% value in B-rated bonds	$2679.80

PROBLEM 2

PART 1A. Use Yahoo Finance to get monthly adjusted closing prices of SPY, GOVT and EEMV between Jan 2014 and Jan 2021. Compute each assets: (1) expected returns, (2) standard deviations, and (3) covariances between assets

2.1A.1 Mathematical Formulation

Notation:
i: asset reference, where: SPY = 1, GOVT = 2 and EEMV =3
$\bar{r}_i$: arithmetic mean return of asset i
μ_i: geometric mean return of asset i
σ_i: standard deviation of asset i returns
σ_{ij}: covariance between asset i and asset j, where $i \neq j$

$(\sigma_i)^2 = \sigma_{ii}$: variance of asset i

2.1A.2 Matrix Representation and Determined Values

Values have been determined as per the MATLAB code **Appendix**.

Arithmetic mean return:

$$\bar{\mathbf{r}} = \begin{vmatrix} r_1 \\ r_2 \\ r_3 \end{vmatrix} = \begin{vmatrix} 0.0114 \\ 0.0023 \\ 0.0046 \end{vmatrix}$$

Geometric mean return:

$$\mu = \begin{vmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{vmatrix} = \begin{vmatrix} 0.01058 \\ 0.00224 \\ 0.00390 \end{vmatrix}$$

Standard deviation of returns:

$$\sigma = \begin{vmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \end{vmatrix} = \begin{vmatrix} 0.0412 \\ 0.0115 \\ 0.0380 \end{vmatrix}$$

Covariance Matrix:

$$\mathbf{H} = \begin{vmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{vmatrix} = \begin{vmatrix} 1.685e-03 & -1.458e-04 & 1.120e-03 \\ -1.458e-04 & 1.302e-04 & -3.230e-05 \\ 1.120e-03 & -3.230e-05 & 1.430e-03 \end{vmatrix}$$

PART 1B. Use mean-variance optimization to generate efficient frontier of the three assets. Create table with optimal weights (w_i) and portfolio variance (σ_P^2) for each expected return (R).

2.1B.1 Mathematical Formulation

Decision variable:

w_i: weight of asset i of overall portfolio

$$\mathbf{w^T} = \begin{vmatrix} w_1 & w_2 & w_3 \end{vmatrix}$$

Objective function:

general form: (minimize portfolio variance σ_P^2)

$$\min \frac{1}{2}\mathbf{w^T} \cdot \mathbf{H} \cdot \mathbf{w}$$

$$min \begin{vmatrix} w_1 & w_2 & w_3 \end{vmatrix} \cdot \begin{vmatrix} 1.685e-03 & -1.458e-04 & 1.120e-03 \\ -1.458e-04 & 1.302e-04 & -3.230e-05 \\ 1.120e-03 & -3.230e-05 & 1.430e-03 \end{vmatrix} \cdot \begin{vmatrix} w_1 \\ w_2 \\ w_3 \end{vmatrix}$$

Subject to:

general form: (obtain an expected return)

$$\sum_{i=1}^{3}(\mu_i \cdot w_i) = R$$

$$\mu_1 w_1 + \mu_2 w_2 + \mu_3 w_3 = R$$

$$0.01058 w_1 + 0.00224 w_2 + 0.00390 w_3 = R$$

general form: (portfolio weights must sum to 1)

$$\sum_{i=1}^{3}(w_i) = 1$$

$$w_1 + w_2 + w_3 = 1$$

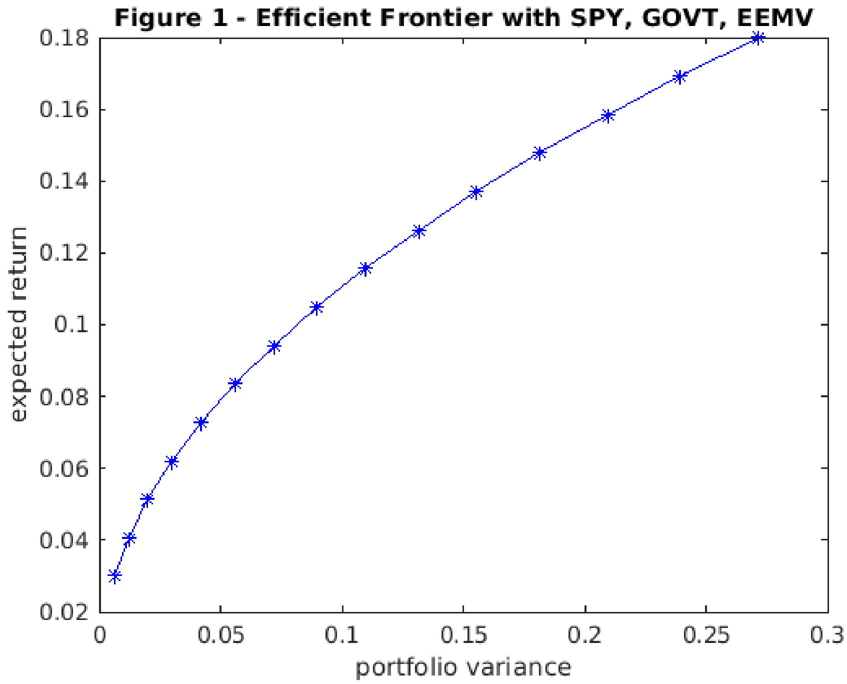
2.1B.2 Matrix Formulation

Equality constraints in Matrix form: $\mathbf{A_{eq}} \cdot \mathbf{w} = \mathbf{b_{eq}}$

$$\begin{vmatrix} 0.01058 & 0.00224 & 0.00390 \\ 1 & 1 & 1 \end{vmatrix} \cdot \begin{vmatrix} w_1 \\ w_2 \\ w_3 \end{vmatrix} = \begin{vmatrix} R \\ 1 \end{vmatrix}$$

2.1B.3 MATLAB Results

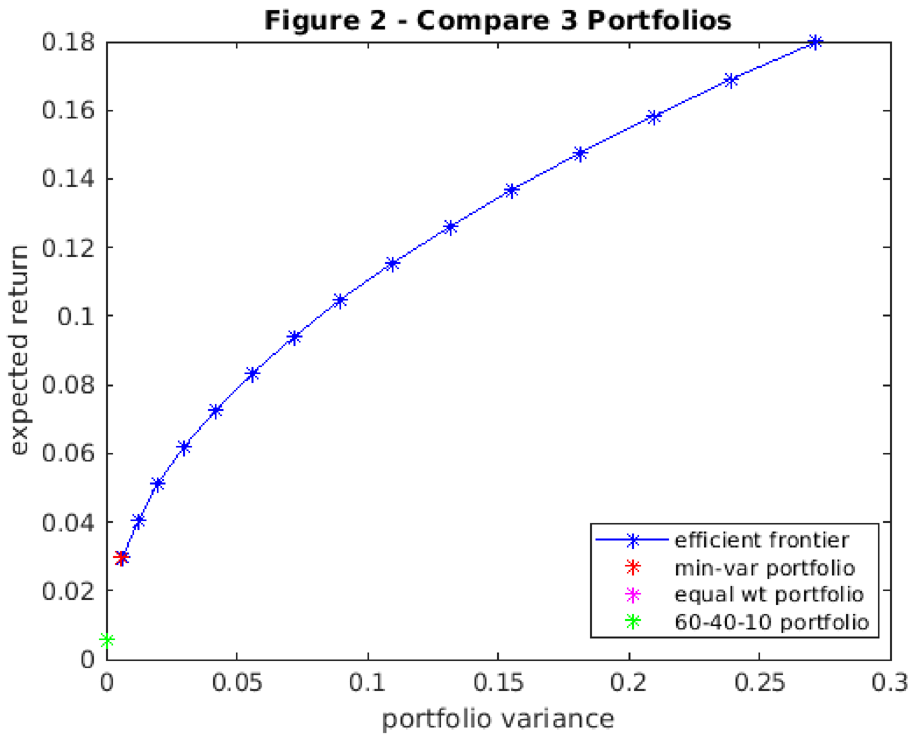
An efficient frontier was obtained in MATLAB as per the code in the Appendix. For the table of expected returns, portfolio variance and weights (w1, w2, w3), pleas refer to the appendix. Below, you will see in Figure 1, the efficient frontier for three assets: SPY, GOVT and EEMV:



PART 1C. Use returns from Feb 21 to determine exp_returns and variance for (1) equal weight portfolio and (2) 60% SPY- 30% GOVT -10% EEMV portfolio. Compare with the minimum variance portfolio obtained on the efficient frontier. The results of this are show in the table below:

Portfolio	**Expected Portfolio Rtn**	**Portfolio Variance**
Min Variance Portfolio	0.030	0.00617
Equal Weight	0.006	0.00055
60% SPY- 40% GOVT- 10% EEMV	0.012	0.00071

I plotted these portfolios on the efficient frontier curve for a graphical view below. The equal weight portfolio did not appear on the graph due to its really low return and variance. Overall, the best portfolio for return is the min-variance portfolio as it is on the efficient frontier.



PART 2A. Repeat PART 1 but with 8 assets now, including CME, BR, CBOE, ICE and ACN:

2.1A.1 Mathematical Formulation

Notation:
i: asset reference: SPY = 1, GOVT = 2, EEMV = 3, CME = 4, BR = 5, CBOE = 6, ICE = 7, and ACN = 8
All other notation remains the same

2.1A.2 Matrix Representation and Determined Values

Values have been determined as per the MATLAB code **Appendix**.

Arithmetic mean return:

$$\bar{\mathbf{r}} = \begin{vmatrix} \bar{r_1} \\ \bar{r_2} \\ \bar{r_3} \\ \bar{r_4} \\ \bar{r_5} \\ \bar{r_6} \\ \bar{r_7} \\ \bar{r_8} \end{vmatrix} = \begin{vmatrix} 0.0114 \\ 0.0023 \\ 0.0046 \\ 0.0164 \\ 0.0195 \\ 0.0106 \\ 0.0139 \\ 0.0167 \end{vmatrix}$$

Geometric mean return:

$$\mu = \begin{vmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \\ \mu_7 \\ \mu_8 \end{vmatrix} = \begin{vmatrix} 0.01058 \\ 0.00224 \\ 0.00390 \\ 0.01494 \\ 0.01781 \\ 0.00836 \\ 0.01254 \\ 0.01519 \end{vmatrix}$$

Covariance Matrix:

Standard deviation of returns:

$$\mathbf{H} = \begin{vmatrix} \sigma_{11} & \dots & \sigma_{18} \\ \sigma_{21} & \dots & \sigma_{28} \\ \sigma_{31} & \dots & \sigma_{38} \\ \sigma_{41} & \dots & \sigma_{48} \\ \sigma_{51} & \dots & \sigma_{58} \\ \sigma_{61} & \dots & \sigma_{68} \\ \sigma_{71} & \dots & \sigma_{78} \\ \sigma_{81} & \dots & \sigma_{88} \end{vmatrix} \qquad \sigma = \begin{vmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \\ \sigma_7 \\ \sigma_8 \end{vmatrix} = \begin{vmatrix} 0.0412 \\ 0.0115 \\ 0.0380 \\ 0.0553 \\ 0.0588 \\ 0.0669 \\ 0.0530 \\ 0.0558 \end{vmatrix}$$

$$\mathbf{H} = \begin{vmatrix} 1.685e-03 & -1.458e-04 & 1.120e-03 & 8.377e-04 & 1.443e-03 & 9.063e-04 & 1.177e-03 & 1.740e-03 \\ -1.458e-04 & 1.301e-04 & -3.229e-05 & -1.071e-04 & -7.785e-06 & -5.560e-05 & -1.272e-04 & -1.271e-04 \\ 1.120e-03 & -3.229e-05 & 1.428e-03 & 2.218e-04 & 9.239e-04 & 3.306e-04 & 3.924e-04 & 9.07e-04 \\ 8.377e-04 & -1.071e-04 & 2.218e-04 & 3.019e-03 & 1.079e-03 & 2.137e-03 & 1.811e-03 & 1.03e-03 \\ 1.443e-03 & -7.785e-06 & 9.239e-04 & 1.079e-03 & 3.415e-03 & 9.667e-04 & 1.202e-03 & 1.87e-03 \\ 9.063e-04 & -5.560e-05 & 3.306e-04 & 2.137e-03 & 9.667e-04 & 4.423e-03 & 1.605e-03 & 1.13e-03 \\ 1.177e-03 & -1.272e-04 & 3.924e-04 & 1.811e-03 & 1.202e-03 & 1.605e-03 & 2.776e-03 & 1.51e-03 \\ 1.740e-03 & -1.271e-04 & 9.075e-04 & 1.030e-03 & 1.873e-03 & 1.130e-03 & 1.512e-03 & 3.07e-03 \end{vmatrix}$$

<u>PART 2B</u>. Use mean-variance optimization to generate efficient frontier of the 8 assets. Create table with optimal weights (w_i) and portfolio variance (σ_P^2) for each expected return (R).

<u>2.2B.1 Mathematical Formulation</u>

<u>Decision variable</u>:

w_i: weight of asset i of overall portfolio

$$\mathbf{w^T} = \begin{vmatrix} w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 & w_8 \end{vmatrix}$$

<u>Objective function</u>:

general form: (minimize portfolio variance σ_P^2)

$$\min \frac{1}{2}\mathbf{w^T} \cdot \mathbf{H} \cdot \mathbf{w}$$

<u>Subject to:</u>

general form: (obtain an expected return)

$$\sum_{i=1}^{8}(\mu_i \cdot w_i) = R$$

$$\mu_1 w_1 + \mu_2 w_2 + \mu_3 w_3 + \mu_4 w_4 + \mu_5 w_5 + \mu_6 w_6 + \mu_7 w_7 + \mu_8 w_8 \ = R$$

$$0.01058w_1 + 0.00224w_2 + 0.00390w_3 + 0.01494w_4 + 0.01781w_5 + 0.00836w_6 + 0.01254w_7 + 0.01519w_8 = R$$

general form: (portfolio weights must sum to 1)

$$\sum_{i=1}^{8}(w_i) = 1$$
$$w_1 + w_2 + w_3 + w_4 + w_5 + w_6 + w_7 + w_8 = 1$$

2.1B.2 Matrix Formulation

Equality constraints in Matrix form: $\mathbf{A_{eq}} \cdot \mathbf{w} = \mathbf{b_{eq}}$

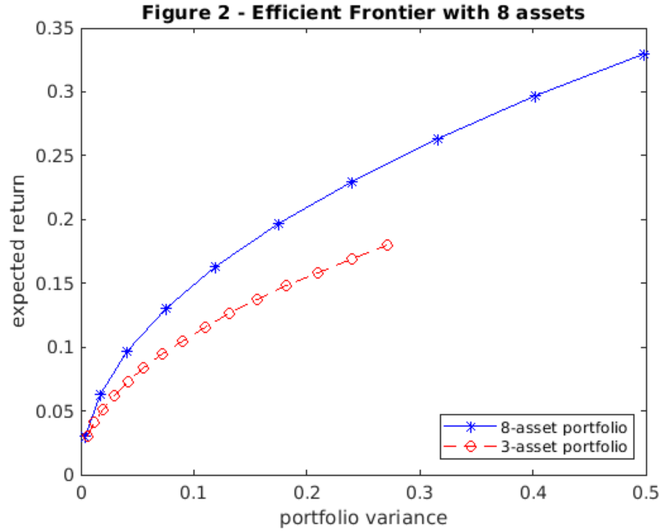
$$\mathbf{A_{eq}} = \begin{vmatrix} 0.01058 & 0.00224 & 0.00390 & 0.01494 & 0.01781 & 0.00836 & 0.01254 & 0.01519 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{vmatrix}$$

$$\mathbf{b_{eq}} = \begin{vmatrix} R \\ 1 \end{vmatrix}$$

2.1B.3 MATLAB Results

Finally, an efficient frontier diagram was produce for the 8-asset portfolio along with the original 3-asset portfolio for comparison. From this diagram, it's clear that the 8-asset portfolio has potential for higher returns and surprisingly, for the same variance as the 3-asset portfolio. I don't believe diversification has anything to do with this since we already contain several ETFs in the 3-asset portfolio. With more time, I would investigate this.

Figure 2 - Efficient Frontier with 8 assets



Appendix - MATLAB Code & Results Output

```
%--------------------------------------------------------------
% PROBLEM 1 - PART 1
%--------------------------------------------------------------
% As per matrix formulation in report, create vectors and matrices:

% obj function coefficients
f = [108; 94; 99; 92.7; 96.6; 95.9; 92.9; 110;
     104; 101; 107; 102; 95.2; 0; 0; 0; 0; 0];

% inequality constraint matrix A
A = [-10 -7 -8 -6 -7 -6 -5 -10 -8 -6 -10 -7 -100 1 0 0 0 0;
     -10 -7 -8 -6 -7 -6 -5 -10 -8 -6 -110 -107 0 -1.02 1 0 0 0;
     -10 -7 -8 -6 -7 -6 -5 -110 -108 -106 0 0 0 0 -1.03 1 0 0;
     -10 -7 -8 -6 -7 -106 -105 0 0 0 0 0 0 0 0 -1.04 1 0;
     -10 -7 -8 -106 -107 0 0 0 0 0 0 0 0 0 0 0 -1.05 1;
     -100 -107 -108 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1.06];

% inequality constraint vector b
b = [-500; -200; -800; -400; -700; -900];

% lower bound and upper bound vectors
lb = zeros(18,1);
ub = inf*ones(18,1);

% calculate objective function value fval and decision var x
[x, fval] = linprog(f,A,b,[],[],lb,ub)
```

```
Optimal solution found.
x = 18×1
            0
   8.4112e+00
            0
            0
   5.9918e+00
   2.8224e+00
            0
            0
   6.3171e+00
            0
      ⋮

fval =
   2.6410e+03
```

```
%--------------------------------------------------------------
% PROBLEM 1 - PART 2
%--------------------------------------------------------------
% updated constraint matrix A2 with <=50% B-rated bond row
A2 = [-10 -7 -8 -6 -7 -6 -5 -10 -8 -6 -10 -7 -100 1 0 0 0 0;
      -10 -7 -8 -6 -7 -6 -5 -10 -8 -6 -110 -107 0 -1.02 1 0 0 0;
      -10 -7 -8 -6 -7 -6 -5 -110 -108 -106 0 0 0 0 -1.03 1 0 0;
      -10 -7 -8 -6 -7 -106 -105 0 0 0 0 0 0 0 0 -1.04 1 0;
      -10 -7 -8 -106 -107 0 0 0 0 0 0 0 0 0 0 0 -1.05 1;
```

```
     -100 -107 -108 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1.06;
     108 94 99 92.7 96.6 95.9 -92.9 -110 -104 -101 -107 -102 -95.2 0 0 0 0 0];

% updated constraint vector b2
b2 = [-500; -200; -800; -400; -700; -900; 0];

% calculate new objective function value fval2 and dec. var x2
[x2, fval2] = linprog(f,A2,b2,[],[],lb,ub)
```

```
Optimal solution found.
x2 = 18×1
            0
   8.4112e+00
            0
            0
   5.5027e+00
            0
   3.3566e+00
            0
   6.3502e+00
            0
      ⋮

fval2 =
   2.6444e+03
```

```

%--------------------------------------------------------------
% PROBLEM 1 - PART 3
%--------------------------------------------------------------
% updated constraint matrix A3 with <=25% B-rated bond row
A2 = [-10 -7 -8 -6 -7 -6 -5 -10 -8 -6 -10 -7 -100 1 0 0 0 0;
     -10 -7 -8 -6 -7 -6 -5 -10 -8 -6 -110 -107 0 -1.02 1 0 0 0;
     -10 -7 -8 -6 -7 -6 -5 -110 -108 -106 0 0 0 0 -1.03 1 0 0;
     -10 -7 -8 -6 -7 -106 -105 0 0 0 0 0 0 0 0 -1.04 1 0;
     -10 -7 -8 -106 -107 0 0 0 0 0 0 0 0 0 0 -1.05 1;
     -100 -107 -108 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1.06;
     324 282 297 278.1 289.8 287.7 -92.9 -110 -104 -101 -107 -102 -95.2 0 0 0 0 0];

% updated constraint vector b3
b2 = [-500; -200; -800; -400; -700; -900; 0];

% calculate new objective function value fval3 and dec. var x3
[x3, fval3] = linprog(f,A2,b2,[],[],lb,ub)
```

```
Optimal solution found.
x3 = 18×1
            0
   7.1271e+00
            0
            0
            0
            0
   1.0407e+01
            0
   6.4637e+00
            0
      ⋮
```

```
fval3 =
   2.6798e+03
```

```
%-----------------------------------------------------------------
% PROBLEM 2 - PART 1A
%-----------------------------------------------------------------
% Load .csv data from yahoo finance
SPY1 = readmatrix('SPY.csv');    % Jan 14 to Jan 21 monthly data
SPY2 = readmatrix('SPY2.csv');   % Feb 14 to Feb 21 monthly data
GOVT1 = readmatrix('GOVT.csv');  % Jan 14 to Jan 21 monthly data
GOVT2 = readmatrix('GOVT2.csv'); % Feb 14 to Feb 21 monthly data
EEMV1 = readmatrix('EEMV.csv');  % Jan 14 to Jan 21 monthly data
EEMV2 = readmatrix('EEMV2.csv'); % Feb 14 to Feb 21 monthly data

% Return (r_it) = (Month 2 Close - Month 1 Close)/Month 1 Close
SPY_rtn = (SPY2(:, 6) - SPY1(:,6))./SPY1(:, 6);      % mo. return
GOV_rtn = (GOVT2(:, 6) - GOVT1(:,6))./GOVT1(:, 6);   % mo. return
EEM_rtn = (EEMV2(:, 6) - EEMV1(:,6))./EEMV1(:, 6);   % mo. return

% Calculate arithmetic means (rbar_i)
SPY_rbar = mean(SPY_rtn)
```

```
SPY_rbar =
   1.1425e-02
```

```
GOV_rbar = mean(GOV_rtn)
```

```
GOV_rbar =
   2.3076e-03
```

```
EEM_rbar = mean(EEM_rtn)
```

```
EEM_rbar =
   4.6158e-03
```

```
% Calculate geometric Means (mu_i)
SPY_rtn_t = 1;      % initialize
GOV_rtn_t = 1;      % intiialize
EEM_rtn_t = 1;      % initialize

% loop through to calculate total product of returns
for t = 1:85        %T = 85 data points
    SPY_rtn_t = (1 + SPY_rtn(t,:))*SPY_rtn_t;   % cumulative
    GOV_rtn_t = (1 + GOV_rtn(t,:))*GOV_rtn_t;   % cumulative
    EEM_rtn_t = (1 + EEM_rtn(t,:))*EEM_rtn_t;   % cumulative
end

% take the 'T'th root to get final geometric mean (mu_i)
SPY_mu = SPY_rtn_t^(1/85) - 1
```

```
SPY_mu =
   1.0581e-02
```

```
GOV_mu = GOV_rtn_t^(1/85) - 1
```

```
GOV_mu =
   2.2429e-03
```

```
EEM_mu = EEM_rtn_t^(1/85) - 1
```

```
EEM_mu =
   3.8973e-03
```

```

% Calculate standard deviations
SPY_std = std(SPY_rtn)
```

```
SPY_std =
   4.1295e-02
```

```
GOV_std = std(GOV_rtn)
```

```
GOV_std =
   1.1476e-02
```

```
EEM_std = std(EEM_rtn)
```

```
EEM_std =
   3.8025e-02
```

```

% Calculate covariances (SPY: i=1, GOV: i=2, EEM: i=3)
rtn_matrix = [SPY_rtn GOV_rtn EEM_rtn];     % i
format shortE
cov_matrix = cov(rtn_matrix,1)
```

```
cov_matrix = 3×3
   1.6852e-03  -1.4581e-04   1.1204e-03
  -1.4581e-04   1.3015e-04  -3.2298e-05
   1.1204e-03  -3.2298e-05   1.4289e-03
```

```

%-----------------------------------------------------------------
% PROBLEM 2 - PART 1B
%----------------------------------------------------------------
% Generating efficient frontier of three assets: SPY, GOVT, EEMV with
% shorting

R = linspace(0.03, 0.18, 15);    % expected return data points
H = cov_matrix;                  % covariance matrix
f = zeros(3,1);                  % [0 0 0]T

Aeq = [SPY_mu GOV_mu EEM_mu; ones(1,3)]  %Equality constraint matrix
```

```
Aeq = 2×3
   1.0581e-02   2.2429e-03   3.8973e-03
   1.0000e+00   1.0000e+00   1.0000e+00
```

```

% initiate vector, matrix to store values from loop:
p_var = zeros(size(R,1),1);     % portfolio variance (obj function)
w = zeros(3, size(R,1));        % asset weights (dec. variable)
```

```
% loop through i from 1 to 15
for i = 1:size(R,2)
    beq = [R(i); 1];                          % Equality constraint vector
    % quadratic program
    [w(:,i), p_var(i)] = quadprog(H, f, [], [], Aeq, beq,[],[]);
end
```

```
Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
```

```
Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
```

```
% Results table of: 'R', 'p_var', w_1, w_2, w_3:
T = array2table([R',p_var', w']);
T.Properties.VariableNames(1:5) = {'expected return', 'port. var', 'w_1', 'w_2', 'w_3'}
```

T = 15×5 table

	expected return	port. var	w_1	w_2	w_3
1	3.0000e-02	6.1676e-03	3.8958e+00	-3.8295e-02	-2.8575e+00
2	4.0714e-02	1.2114e-02	5.4068e+00	-4.0984e-01	-3.9970e+00
3	5.1429e-02	2.0060e-02	6.9179e+00	-7.8138e-01	-5.1365e+00
4	6.2143e-02	3.0004e-02	8.4289e+00	-1.1529e+00	-6.2760e+00
5	7.2857e-02	4.1948e-02	9.9399e+00	-1.5245e+00	-7.4154e+00
6	8.3571e-02	5.5890e-02	1.1451e+01	-1.8960e+00	-8.5549e+00
7	9.4286e-02	7.1832e-02	1.2962e+01	-2.2676e+00	-9.6944e+00
8	1.0500e-01	8.9773e-02	1.4473e+01	-2.6391e+00	-1.0834e+01
9	1.1571e-01	1.0971e-01	1.5984e+01	-3.0106e+00	-1.1973e+01
10	1.2643e-01	1.3165e-01	1.7495e+01	-3.3822e+00	-1.3113e+01
11	1.3714e-01	1.5559e-01	1.9006e+01	-3.7537e+00	-1.4252e+01
12	1.4786e-01	1.8153e-01	2.0517e+01	-4.1253e+00	-1.5392e+01
13	1.5857e-01	2.0946e-01	2.2028e+01	-4.4968e+00	-1.6531e+01
14	1.6929e-01	2.3940e-01	2.3539e+01	-4.8684e+00	-1.7671e+01
15	1.8000e-01	2.7133e-01	2.5050e+01	-5.2399e+00	-1.8810e+01

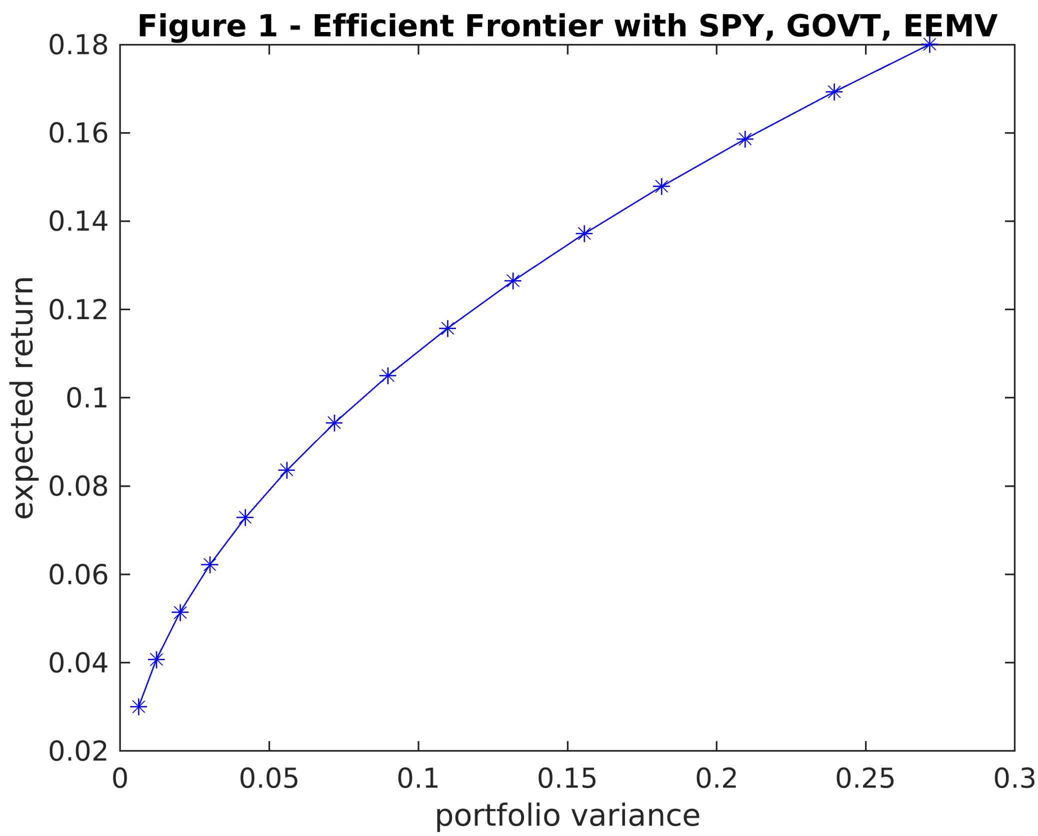
```
display(T)
```

T = 15×5 table

	expected return	port. var	w_1	w_2	w_3
1	3.0000e-02	6.1676e-03	3.8958e+00	-3.8295e-02	-2.8575e+00
2	4.0714e-02	1.2114e-02	5.4068e+00	-4.0984e-01	-3.9970e+00
3	5.1429e-02	2.0060e-02	6.9179e+00	-7.8138e-01	-5.1365e+00
4	6.2143e-02	3.0004e-02	8.4289e+00	-1.1529e+00	-6.2760e+00
5	7.2857e-02	4.1948e-02	9.9399e+00	-1.5245e+00	-7.4154e+00
6	8.3571e-02	5.5890e-02	1.1451e+01	-1.8960e+00	-8.5549e+00
7	9.4286e-02	7.1832e-02	1.2962e+01	-2.2676e+00	-9.6944e+00
8	1.0500e-01	8.9773e-02	1.4473e+01	-2.6391e+00	-1.0834e+01
9	1.1571e-01	1.0971e-01	1.5984e+01	-3.0106e+00	-1.1973e+01
10	1.2643e-01	1.3165e-01	1.7495e+01	-3.3822e+00	-1.3113e+01
11	1.3714e-01	1.5559e-01	1.9006e+01	-3.7537e+00	-1.4252e+01
12	1.4786e-01	1.8153e-01	2.0517e+01	-4.1253e+00	-1.5392e+01

	expected return	port. var	w_1	w_2	w_3
13	1.5857e-01	2.0946e-01	2.2028e+01	-4.4968e+00	-1.6531e+01
14	1.6929e-01	2.3940e-01	2.3539e+01	-4.8684e+00	-1.7671e+01
15	1.8000e-01	2.7133e-01	2.5050e+01	-5.2399e+00	-1.8810e+01

```
% Plot efficient frontier
figure('Name','Efficient Frontier - With Shorting');
plot(p_var, R, 'b-*');
title('Figure 1 - Efficient Frontier with SPY, GOVT, EEMV');
xlabel('portfolio variance');
ylabel('expected return');
```




```
%------------------------------------------------------------------
% PROBLEM 2 - PART 1C
%-----------------------------------------------------------------
% minimum variance portfolio: from Table T, occurs at:
% exp_rtn = 0.03, var = 0.006, w_1 = 3.9, w_2 = -0.04, w_3 = -2.9

% use feb_rtns for exp rtn calculations
feb_rtns = [SPY_rtn(85,:) GOV_rtn(85,:) EEM_rtn(85,:)];

% equal weight portfolio
w3 = [0.33; 0.33; 0.33];
```

```
var_3 = w3' * H * w3
```

```
var_3 =
   5.5853e-04
```

```
exp_rtn_3 = feb_rtns * w3
```

```
exp_rtn_3 =
   6.1120e-03
```

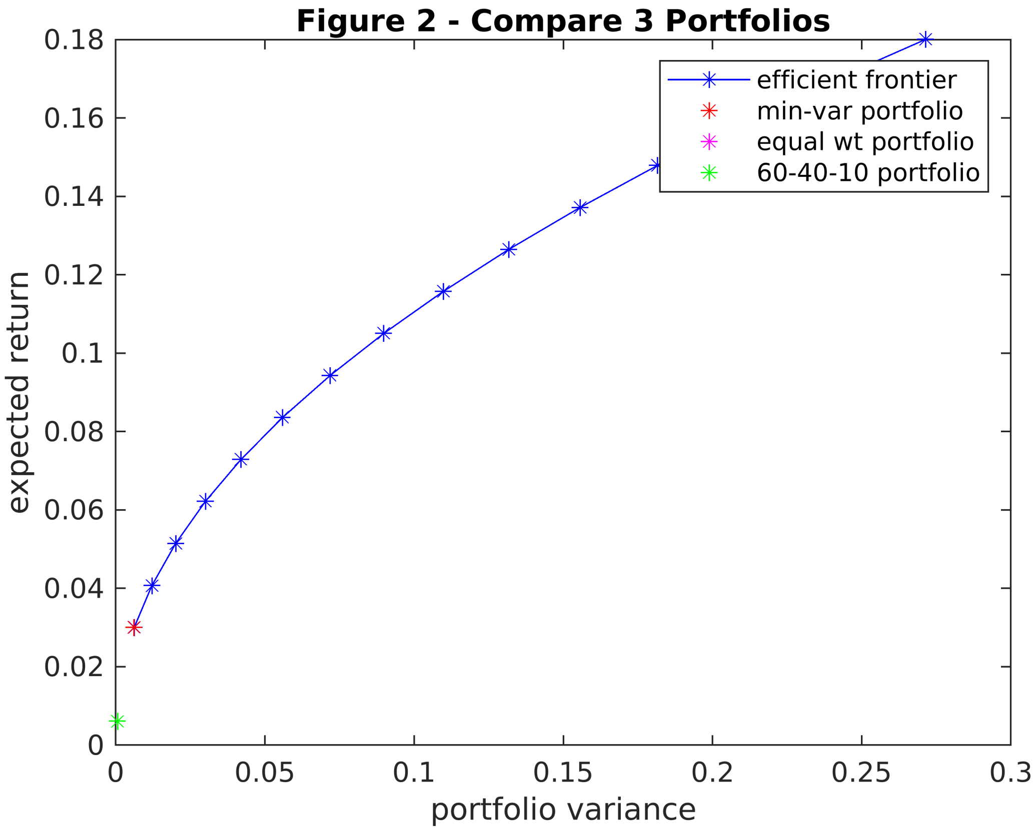
```
% 60% SPY, 30% GOVT, 10% EEMV portfolio
w4 = [0.6; 0.3; 0.1];
var_4 = w4' * H * w4
```

```
var_4 =
   7.1269e-04
```

```
exp_rtn_4 = feb_rtns * w4
```

```
exp_rtn_4 =
   1.1920e-02
```

```
% Plot efficient frontier
figure('Name','Efficient Frontier');
plot(p_var, R, 'b-*')
hold on
scatter(0.006, 0.03, 'r*')
scatter(var_3, exp_rtn_3, 'm*')
scatter(var_3, exp_rtn_3, 'g*')
legend('efficient frontier', 'min-var portfolio', 'equal wt portfolio', '60-40-10 portf
title('Figure 2 - Compare 3 Portfolios');
xlabel('portfolio variance');
ylabel('expected return');
```




```
%------------------------------------------------------------------
% PROBLEM 2 - PART 2A
%------------------------------------------------------------------
% Load .csv data from yahoo finance
CME1 = readmatrix('CME.csv');    % Jan 14 to Jan 21 monthly data
CME2 = readmatrix('CME2.csv');   % Feb 14 to Feb 21 monthly data
BR1 = readmatrix('BR.csv');      % Jan 14 to Jan 21 monthly data
BR2 = readmatrix('BR2.csv');     % Feb 14 to Feb 21 monthly data
CBOE1 = readmatrix('CBOE.csv');  % Jan 14 to Jan 21 monthly data
CBOE2 = readmatrix('CBOE2.csv'); % Feb 14 to Feb 21 monthly data
ICE1 = readmatrix('ICE.csv');    % Jan 14 to Jan 21 monthly data
ICE2 = readmatrix('ICE2.csv');   % Feb 14 to Feb 21 monthly data
ACN1 = readmatrix('ACN.csv');    % Jan 14 to Jan 21 monthly data
ACN2 = readmatrix('ACN2.csv');   % Feb 14 to Feb 21 monthly data

% Return (r_it) = (Month 2 Close - Month 1 Close)/Month 1 Close
CME_rtn = (CME2(:, 6) - CME1(:,6))./CME1(:, 6);      % mo. return
BR_rtn  = (BR2(:, 6) - BR1(:,6))./BR1(:, 6);         % mo. return
CBO_rtn = (CBOE2(:, 6) - CBOE1(:,6))./CBOE1(:, 6);   % mo. return
ICE_rtn = (ICE2(:, 6) - ICE1(:,6))./ICE1(:, 6);      % mo. return
ACN_rtn = (ACN2(:, 6) - ACN1(:,6))./ACN1(:, 6);      % mo. return

% Calculate arithmetic means (rbar_i)
CME_rbar = mean(CME_rtn)
```

```
CME_rbar =
```

```
   1.6435e-02
```

```
BR_rbar  = mean(BR_rtn)
```

```
BR_rbar =
   1.9472e-02
```

```
CBO_rbar = mean(CBO_rtn)
```

```
CBO_rbar =
   1.0627e-02
```

```
ICE_rbar = mean(ICE_rtn)
```

```
ICE_rbar =
   1.3913e-02
```

```
ACN_rbar = mean(ACN_rtn)
```

```
ACN_rbar =
   1.6718e-02
```

```
% Calculate geometric Means (mu_i)
CME_rtn_t = 1;      % initialize
BR_rtn_t = 1;      % intiialize
CBO_rtn_t = 1;      % initialize
ICE_rtn_t = 1;      % intiialize
ACN_rtn_t = 1;      % initialize

% loop through to calculate total product of returns
for t = 1:85        %T = 85 data points
    CME_rtn_t = (1 + CME_rtn(t,:))*CME_rtn_t;   % cumulative
    BR_rtn_t = (1 + BR_rtn(t,:))*BR_rtn_t;      % cumulative
    CBO_rtn_t = (1 + CBO_rtn(t,:))*CBO_rtn_t;   % cumulative
    ICE_rtn_t = (1 + ICE_rtn(t,:))*ICE_rtn_t;   % cumulative
    ACN_rtn_t = (1 + ACN_rtn(t,:))*ACN_rtn_t;   % cumulative
end

% take the 'T'th root to get final geometric mean (mu_i)
CME_mu = CME_rtn_t^(1/85) - 1
```

```
CME_mu =
   1.4943e-02
```

```
BR_mu = BR_rtn_t^(1/85) - 1
```

```
BR_mu =
   1.7811e-02
```

```
CBO_mu = CBO_rtn_t^(1/85) - 1
```

```
CBO_mu =
   8.3644e-03
```

```
ICE_mu = ICE_rtn_t^(1/85) - 1
```

```
ICE_mu =
```

```
   1.2545e-02
```

```
ACN_mu = ACN_rtn_t^(1/85) - 1
```

```
ACN_mu =
   1.5185e-02
```

```
% Calculate standard deviations
CME_std = std(CME_rtn)
```

```
CME_std =
   5.5274e-02
```

```
BR_std = std(BR_rtn)
```

```
BR_std =
   5.8791e-02
```

```
CBO_std = std(CBO_rtn)
```

```
CBO_std =
   6.6904e-02
```

```
ICE_std = std(ICE_rtn)
```

```
ICE_std =
   5.3000e-02
```

```
ACN_std = std(ACN_rtn)
```

```
ACN_std =
   5.5817e-02
```

```
% Calculate covariances
% (SPY=1, GOV=2, EEM=3,CME=4, BR=5, CBO=6, ICE=7, ACN=8)
rtn_matrix2 = [SPY_rtn GOV_rtn EEM_rtn CME_rtn BR_rtn CBO_rtn ICE_rtn ACN_rtn];
format shortE
cov_matrix2 = cov(rtn_matrix2,1)
```

```
cov_matrix2 = 8×8
   1.6852e-03  -1.4581e-04   1.1204e-03   8.3779e-04   1.4438e-03   9.0638e-04 ⋯
  -1.4581e-04   1.3015e-04  -3.2298e-05  -1.0718e-04  -7.7851e-06  -5.5607e-05
   1.1204e-03  -3.2298e-05   1.4289e-03   2.2182e-04   9.2395e-04   3.3062e-04
   8.3779e-04  -1.0718e-04   2.2182e-04   3.0192e-03   1.0796e-03   2.1379e-03
   1.4438e-03  -7.7851e-06   9.2395e-04   1.0796e-03   3.4157e-03   9.6670e-04
   9.0638e-04  -5.5607e-05   3.3062e-04   2.1379e-03   9.6670e-04   4.4234e-03
   1.1779e-03  -1.2725e-04   3.9242e-04   1.8114e-03   1.2028e-03   1.6056e-03
   1.7403e-03  -1.2717e-04   9.0753e-04   1.0301e-03   1.8730e-03   1.1309e-03
```

```
%-----------------------------------------------------------------
% PROBLEM 2 - PART 2B
%----------------------------------------------------------------
% Generating efficient frontier using all 8 assets:

R2 = linspace(0.03, 0.33, 10);      % expected return data points
H2 = cov_matrix2;                   % covariance matrix
```

```
f2 = zeros(8,1);                        % [0 0 0 0 0 0 0 0]T

Aeq2 = [SPY_mu GOV_mu EEM_mu CME_mu BR_mu CBO_mu ICE_mu ACN_mu;
       ones(1,8)];  %Equality constraint matrix

% initiate vector, matrix to store values from loop:
p_var2 = zeros(size(R2,1),1);  % portfolio variance (obj function)
w2 = zeros(8, size(R2,1));     % asset weights (dec. variable)

% loop through i from 1 to 15
for i = 1:size(R2,2)
    beq2 = [R2(i); 1];           % Equality constraint vector
    % quadratic program
    [w2(:,i), p_var2(i)] = quadprog(H2, f2, [], [], Aeq2, beq2,[],[]);
end
```

```
Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.
```

```
<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>

Minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
```

```
% Results table of: 'R', 'p_var', w_i
T2 = array2table([R2',p_var2', w2']);
T2.Properties.VariableNames(1:10) = {'exp return', 'port. var','w_1', 'w_2', 'w_3', 'w_
```

T2 = 10×10 table

	exp return	port. var	w_1	w_2	w_3	w_4	w_5	w_6
1	3.0000e-02	3.3734e-03	7.3605e-01	-3.3325e-02	-1.1559e+00	8.1014e-01	7.7997e-01	-3.6327e-01
2	6.3333e-02	1.6875e-02	1.4247e+00	-1.1665e+00	-2.5601e+00	1.7783e+00	1.8092e+00	-8.0269e-01
3	9.6667e-02	4.0752e-02	2.1134e+00	-2.2997e+00	-3.9643e+00	2.7465e+00	2.8384e+00	-1.2421e+00
4	1.3000e-01	7.5005e-02	2.8021e+00	-3.4328e+00	-5.3685e+00	3.7146e+00	3.8677e+00	-1.6815e+00
5	1.6333e-01	1.1963e-01	3.4908e+00	-4.5660e+00	-6.7727e+00	4.6828e+00	4.8969e+00	-2.1210e+00
6	1.9667e-01	1.7464e-01	4.1795e+00	-5.6992e+00	-8.1769e+00	5.6510e+00	5.9262e+00	-2.5604e+00
7	2.3000e-01	2.4002e-01	4.8682e+00	-6.8323e+00	-9.5810e+00	6.6191e+00	6.9554e+00	-2.9998e+00
8	2.6333e-01	3.1577e-01	5.5569e+00	-7.9655e+00	-1.0985e+01	7.5873e+00	7.9846e+00	-3.4392e+00
9	2.9667e-01	4.0191e-01	6.2456e+00	-9.0986e+00	-1.2389e+01	8.5554e+00	9.0139e+00	-3.8786e+00
10	3.3000e-01	4.9841e-01	6.9343e+00	-1.0232e+01	-1.3794e+01	9.5236e+00	1.0043e+01	-4.3181e+00

```
display(T2)
```

T2 = 10×10 table

	exp return	port. var	w_1	w_2	w_3	w_4	w_5	w_6
1	3.0000e-02	3.3734e-03	7.3605e-01	-3.3325e-02	-1.1559e+00	8.1014e-01	7.7997e-01	-3.6327e-01
2	6.3333e-02	1.6875e-02	1.4247e+00	-1.1665e+00	-2.5601e+00	1.7783e+00	1.8092e+00	-8.0269e-01
3	9.6667e-02	4.0752e-02	2.1134e+00	-2.2997e+00	-3.9643e+00	2.7465e+00	2.8384e+00	-1.2421e+00
4	1.3000e-01	7.5005e-02	2.8021e+00	-3.4328e+00	-5.3685e+00	3.7146e+00	3.8677e+00	-1.6815e+00
5	1.6333e-01	1.1963e-01	3.4908e+00	-4.5660e+00	-6.7727e+00	4.6828e+00	4.8969e+00	-2.1210e+00
6	1.9667e-01	1.7464e-01	4.1795e+00	-5.6992e+00	-8.1769e+00	5.6510e+00	5.9262e+00	-2.5604e+00
7	2.3000e-01	2.4002e-01	4.8682e+00	-6.8323e+00	-9.5810e+00	6.6191e+00	6.9554e+00	-2.9998e+00
8	2.6333e-01	3.1577e-01	5.5569e+00	-7.9655e+00	-1.0985e+01	7.5873e+00	7.9846e+00	-3.4392e+00
9	2.9667e-01	4.0191e-01	6.2456e+00	-9.0986e+00	-1.2389e+01	8.5554e+00	9.0139e+00	-3.8786e+00
10	3.3000e-01	4.9841e-01	6.9343e+00	-1.0232e+01	-1.3794e+01	9.5236e+00	1.0043e+01	-4.3181e+00

```
% Plot efficient frontier
figure('Name','Efficient Frontier - With Shorting');
plot(p_var2, R2, 'b-*');
hold on
plot(p_var, R, 'r--o');
legend('8-asset portfolio', '3-asset portfolio')
title('Figure 3 - Efficient Frontier with 8 assets');
xlabel('portfolio variance');
ylabel('expected return');
```

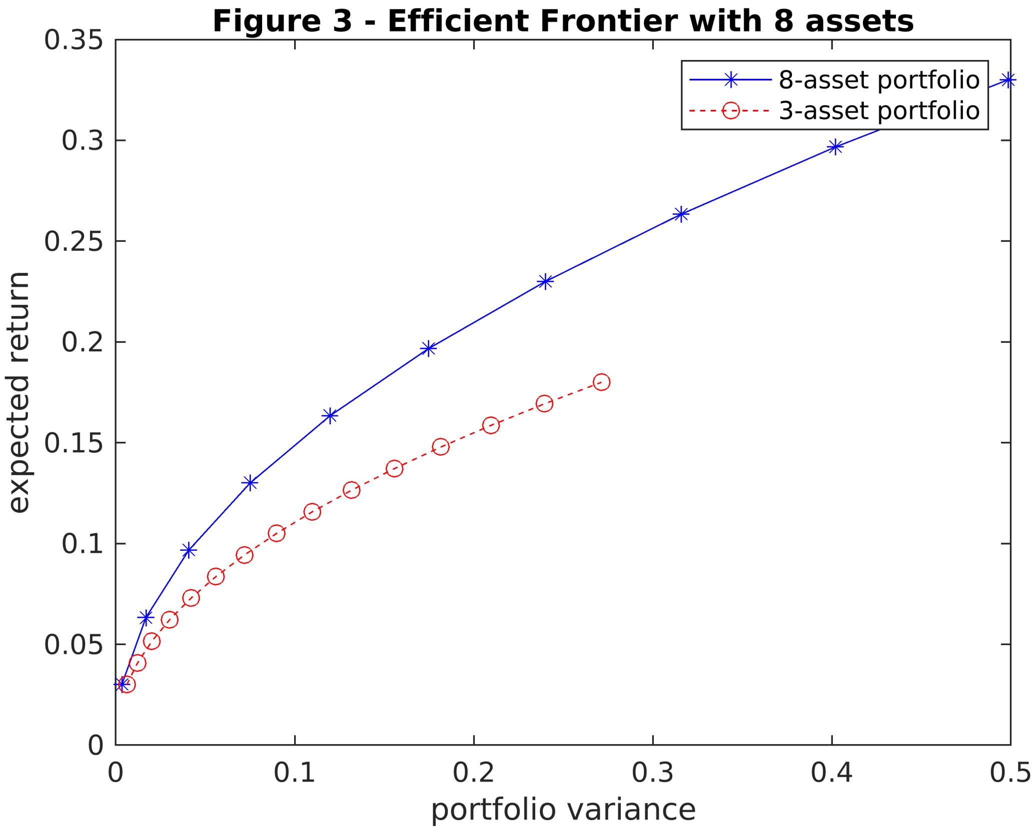


Figure 3 - Efficient Frontier with 8 assets
8-asset portfolio
3-asset portfolio
expected return
portfolio variance
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0
0.1
0.2
0.3
0.4
0.5