Creating a Portfolio Optimization Framework that Protects Against Inflation Uncertainty

Anojan Palarajah, Bill Tang, Seyon Vasantharajan, Bryan Wan, Hamed Zakeri

Supervised by Roy H. Kwon

December 8, 2014

Contents

1 Abstract

This paper details the use of a portfolio optimization framework. The objective of this framework is to construct a portfolio that protects against inflation risk while obtaining competitive returns, thereby having a low inflation risk premium. It utilizes a Regime-Switching model for inflation in conjunction with Inflation Betas to create a single-factor model based on MVO. The resulting portfolio is tested out-of-sample against standard MVO, the S&P 500 and two mutual funds. Different parameters for testing include the start/end dates and the minimum return constraint for MVO. The cumulative returns are then plotted and compared. In addition the resulting measures of risk for both standard MVO and our model are compared.

2 Outline

An autoregressive equation is used to model the inflation rates in collaboration with a Regime Switching model. Two distinct equations are obtained representing the corresponding mean, variance and autoregressive coefficient associated with each regime. An n-period non-recombining binomial tree is then constructed with the Markov transition probabilities for the inflation rate. The terminal node values are computed using recursion (as the process is autoregressive). The expectation is then taken over all the terminal nodes with the transition probabilities being the probability measure. The resulting value is the expected inflation rate, n periods from now conditional on the current regime. A similar process is done to find the expected inflation rate variance, n periods from now conditional on the current regime. The next step is to find the nominal expected returns of all the assets in the asset pool. This is done using geometric returns from prior data. The covariances of the assets to each other are then computed. The expected returns are perturbed by an inflation factor. This factor is the inflation Beta of the asset, computed using regression, multiplied by the expected inflation rate generated.

The objective function follows the form of a single-factor model and has a noise term that is computed using the expected inflation variance solved for. The quadratic program formulated is then solved for and the resulting weights, for investment in each asset, are returned. Rebalancing over time is performed but with an additional transaction cost constraint that ensures the impracticality of dramatic changes in allocation cannot occur.

3 Motivation

Inflation rates have a large impact on the Stock Market. If inflation rates are high, they diminish the real returns of portfolios. This is because inflation causes a rise in the price of all goods and services leading to less consumer spending and consequently less revenue for companies. Meeting inflation targets is a delicate process that the Federal Reserve must attend to. The general target is around the 2% range, which is a healthy inflation rate. The number is chosen to keep unemployment rates low and to meet the demand of the growing economy and its corresponding GDP. If the rate ends up being too low, there is the risk of deflation, which leads to a depression in the economy. This is the time period where all assets in a stock market are highly correlated with each other as their value drops due to the depression. The objective is to try to minimize losses in comparison to other benchmark portfolios.

Inflation rates also affect the volatility of stock returns with higher inflation rates being correlated to higher volatility in the stock market. There is a real need for institutional and consumer investors to be able to protect their portfolio against inflation. For the past 20 years, Inflation has remained steadily low. Because of this, inflation rates are expected to rise in the near future so the desire to deal with this upending reality exists. Currently, the inflation risk premium being paid for by investors is too high and diminishes their return substantially. Investments such as TIPS (Treasury Inflation Protected Securities) are too costly and do not offer an appealing risk/return profile. Therefore there is a need to create an optimization framework that addresses this problem.

4 Inflation Background

4.1 Causes of Inflation

Before a common currency, exchanges of goods were based upon physical commodities such as raw materials and resources. Nowadays transactions are made based on the purchasing power of currencies, and inflation rates of currencies represent a dynamic in the purchasing power of money.

The two main influences on inflation rates are the conditions of the Money supply (amount of money available) and the Real Economy (supply and demand for products). While most commodities are limited, the amount of money supply for the USD has no physical limit. It is influenced by government decisions and cannot be directly hedged against.

The more complicated cause of inflation is affected by conditions of the economy. Inflation can increase in the face of demand-pull, when higher demand and lower supply makes buyers willing to pay more for products, and cost-push, where prices rise to accommodate increasing costs or lower demand to maintaining profits.

There are two types of inflation: expected inflation and unexpected inflation. Expected inflation gets priced into the market without shock while unexpected inflation creates a source of volatility in the markets. Examples of events that caused unexpected inflation include the 1973 Oil crisis, where many Arab countries declared an oil embargo. This caused the oil prices to rise and the inflation rates to follow.

4.2 Industries most Impacted by Inflation

The assets and sectors most correlated with inflation rate changes are those of physical resources such as oil, gas, commodities, raw materials and real estate. These resources directly affect the energy sector, as well as those depending on commodities such as transportation and retail. They vary positively with inflation and are good assets to include in a portfolio that hedges against inflation risk.

4.3 Controlling Inflation Risk

Rising Inflation rates indicates decreasing purchasing power. This is observable in commodities and raw materials such as oil, gas and resources. The profit margins of companies rely on low production costs. Inflation increases the cost of these resources, which, in order to maintain profit margins, must increase the prices of the products as well. The transportation industry, and cost of all its services directly depends on oil and gas prices. Therefore inflation changes can be observed in rising prices among resource dependant industries.

The effectiveness of an asset in hedging against inflation depends on it's inflation beta. An inflation Beta of 2 would imply that for every x% increase in the inflation rate, the asset would go up by 2x% on average.

5 Markov Regime Switching

In this framework, we will assume that the behaviour of the inflation rate (π_t), for time $t = 1, ..., T$, can be described using a first-order autoregression, where the parameters are dependant on $S_t = 1, ..., K$, which is an indicator variable specifying the regime at time t:

$$\pi_t = \mu_{S_t} + \phi_{S_t}\pi_{t-1} + \varepsilon_{S_t} \text{ where } \varepsilon_{S_t} \sim N(0, \sigma^2_{S_t}) \tag{1}$$

If inflation were to exhibit a single state of behaviour, the values of μ, ϕ and σ would remain constant through time since there would no dependency on the regime S_t. From the examination of past history, inflation seems to exhibit 2 distinguishable states of behaviour: normal inflation and hyperinflation. Given that the transition of regimes is not deterministic, a 2-state Markov Chain will be used to model the probabilistic nature of regime switches. By the definition of a Markov Chain, the value of the current regime is solely dependant on the previous one.

$$Pr(S_t = i \rightarrow S_t = j) = Pr(S_t = j | S_t = i) = \rho_{ij}$$

These transition probabilities can be re-written into a matrix form $\mathbf{P}$:

$$\begin{bmatrix} \rho_{11} & \rho_{12} & \cdots & \rho_{1K} \\ \rho_{21} & \rho_{22} & \cdots & \rho_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{K1} & \rho_{K2} & \cdots & \rho_{KK} \end{bmatrix}$$

In this 2-state case, $\mathbf{P}$ will only have a dimension of 2x2:

$$\begin{bmatrix} \rho_{11} & 1 - \rho_{11} \\ 1 - \rho_{22} & \rho_{22} \end{bmatrix}$$

6 Estimating Parameters using MLE

The parameters of this model can be estimated using either Maximum Likelihood Estimation (MLE) or Bayesian Inference. Given the convenience of an existing Matlab package which uses MLE to solve for the parameters of an autoregressive regime-switching model, MLE was selected as the method for estimating parameters. In order to perform MLE, this first step is to obtain the probability density function for π_t and the log likelihood function from Equation(1).

$$f(\pi_t | \pi_{t-1}, \Theta) = \frac{1}{\sqrt{2\pi}\sigma_{S_t}} \exp(-\frac{\pi_t - \phi_{S_t}\pi_{t-1} - \mu_{S_t}}{2\sigma^2_{S_t}}) \tag{2}$$

$$\ln L = \sum_{t=1}^{T} \ln f(\pi_t | \pi_{t-1}, \Theta) = \sum_{t=1}^{T} \ln[\frac{1}{\sqrt{2\pi}\sigma_{S_t}} \exp(-\frac{\pi_t - \phi_{S_t}\pi_{t-1} - \mu_{S_t}}{2\sigma^2_{S_t}})] \tag{3}$$

$$\Theta = [\mu_1, \mu_2, \phi_1, \phi_2, \sigma_1, \sigma_2]$$

If the regimes S_t can be observed, there would be no need for ρ_{11} and ρ_{22}. In that case, estimating only $\mu_1, \mu_2, \phi_1, \phi_2, \sigma_1$ and σ_2 using MLE by maximizing Equation(3) would sufficiently characterize

the whole model. However, as indicated earlier, this is clearly not the case as the inflation regimes are not observable.

Given that the regimes can not be observed, it is first necessary to consider the likelihood of π_t as a weighted average of the likelihood function in each regime S_t, where the weights are each regime's probability $Pr(S_t = j)$.

$$f(\pi_t|S_t = j, \pi_{t-1}, \Theta) = \frac{1}{\sqrt{2\pi}\sigma_j}\exp(-\frac{\pi_t - \phi_j\pi_{t-1} - \mu_j}{2\sigma_j^2}) \tag{4}$$

$$\ln L = \sum_{t=1}^{T} \ln \sum_{j=1}^{2} (f(\pi_t|S_t = j, \pi_{t-1}, \Theta)Pr(S_t = j)) \tag{5}$$

The probabilities $Pr(S_t = j)$ can be inferred using a technique known as Hamilton's filter which includes using the transition probabilities and iteratively updating $Pr(S_t = j)$ as more values of inflation are observed. Consequently, the list of parameters which define the distribution of π_t include Θ as well as the two state transition probabilities, ρ_{11} and ρ_{22}. The iterative algorithm of Hamilton's filter can be defined in 4 steps:

1. Start at $t = 0$ with an initial value for inflation π_0 and an initial guess for the probabilities of each regime $Pr(S_0 = j)$ for $j = 1, 2$

2. Increment t. Calculate the prior probabilities of each regime S_t, given ψ_{t-1} which represents the set of all the observed values of inflation up to time $t - 1$. The answer is found as a weighted average of the posterior probabilities of regimes S_{t-1}, where the weights are ρ_{ij}, the transition probabilities found in matrix $\mathbf{P}$.

$$Pr(S_t = j|\psi_{t-1}) = \sum_{i=1}^{2} \rho_{ij} Pr(S_{t-1} = i|\psi_{t-1}) \tag{6}$$

3. Obtain the posterior probability of each regime S_t, given the newly observed value of inflation at time t, by multiplying Equation(4) and Equation(6) together and then dividing by the sum of that product over the two possible regimes.

$$Pr(S_t = j|\psi_t) = \frac{f(\pi_t|S_t = j, \psi_{t-1}, \Theta)Pr(S_t = j|\psi_{t-1})}{\sum_{j=1}^{2} f(\pi_t|S_t = j, \psi_{t-1}, \Theta)Pr(S_t = j|\psi_{t-1})} \tag{7}$$

4. Repeat steps 2-3 until t = T which means that all the observations of inflation in the sample have been reached. This should provide a set of filtered probabilities for each regime from $t = 1$ to $t = T$.

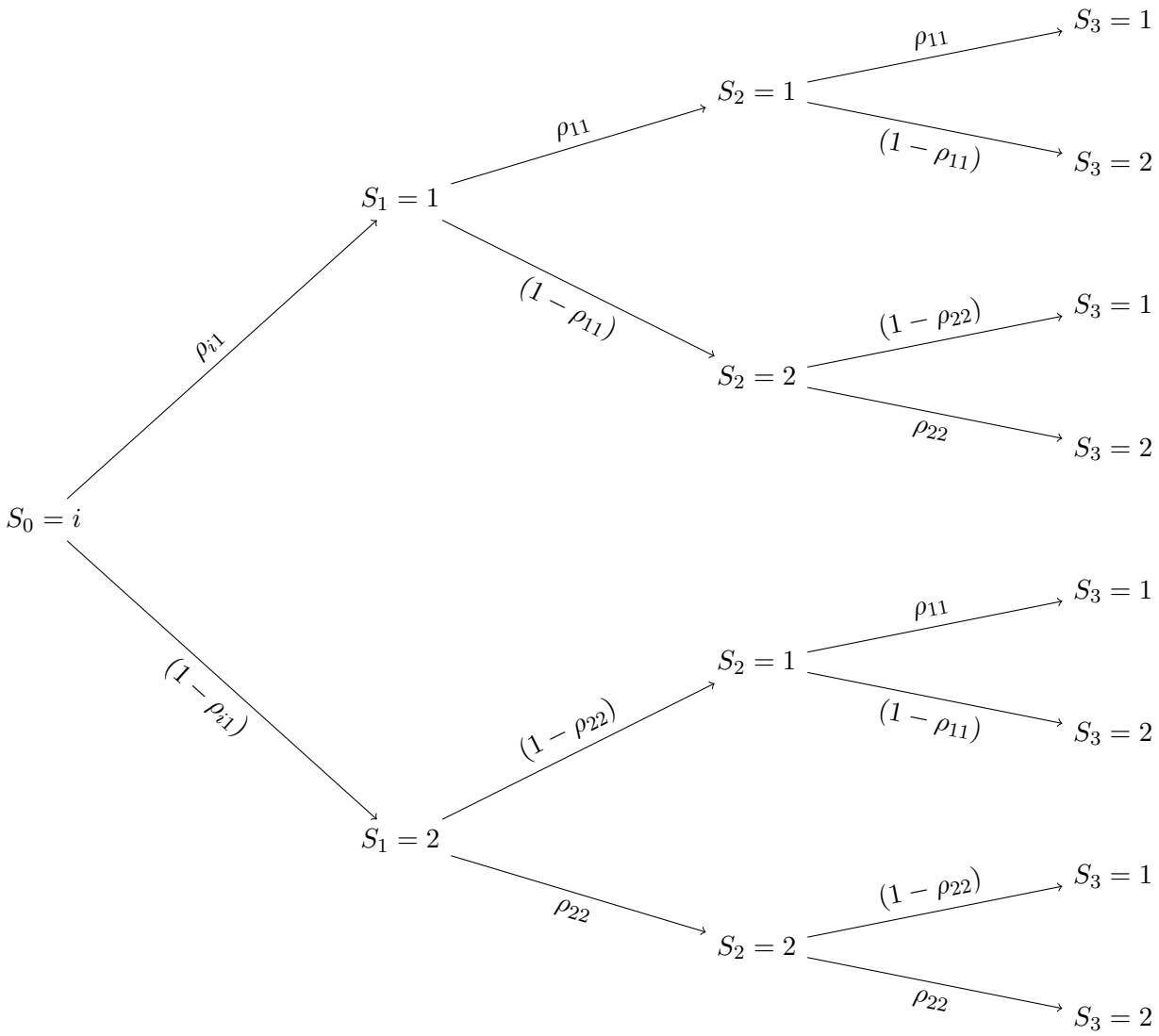
In short, the log likelihood function can be more specifically written as:

$$\ln L = \sum_{t=1}^{T} \ln \sum_{j=1}^{2} (f(\pi_t|S_t = j, \psi_{t-1}, \Theta)Pr(S_t = j|\psi_t)) \tag{8}$$

By inputting an adequately large time-series or observed inflation values and maximizing Equation(8) as a function of Θ and the transition probabilities, an optimal estimate for the model parameters can be obtained.

7 Simulating Markov Tree

From the output of the Matlab MLE package, the state of the regime at time T can be obtained. Then, taking this time to be the new time 0, and using the other Markov Regime-Switching parameters that are also obtained using the Matlab MLE package, we can simulate a Markov Tree for τ time periods aimed at modelling the future behaviour of inflation. The figure below illustrates a sample Markov Tree fro 3 time periods.



8 Expected Mean of Inflation

From the Markov tree, we want to compute $\mathbb{E}[\pi_\tau|S_0]$, which represents the expected future mean of inflation, given the initial regime. Also, assume that π_0 is observed, thus $\mathbb{E}[\pi_0|S_0] = \pi_0$. First, to obtain the mean of future inflation, we must take 2 sets of expected values.

1. Compute the expected values of π_τ at each of the 2^τ terminal nodes assuming that the simulated regimes are actually observed. Because our model of π_t is autoregressive, the expected value of inflation at every node in the Markov Tree will depend on the expected value of inflation at its preceding node. In other words, every expected value of π_τ will be path-dependant $\mathbb{E}[\pi_\tau|\psi_\tau]$, where ψ_τ represents the sequence of regimes up to time τ.

$$\begin{aligned}
\mathbb{E}[\pi_\tau|\psi_\tau] &= \mathbb{E}[\mu_{S_\tau} + \phi_{S_\tau}\pi_{\tau-1} + \varepsilon_{S_\tau}] \\
&= \mu_{S_\tau} + \phi_{S_\tau}\mathbb{E}[\pi_{\tau-1}|\psi_{\tau-1}] \\
&= \mu_{S_\tau} + \phi_{S_\tau}\mu_{S_{\tau-1}} + \phi_{S_\tau}\phi_{S_{\tau-1}}\mathbb{E}[\pi_{\tau-2}|\psi_{\tau-2}] \\
&= \mu_{S_\tau} + \phi_{S_\tau}\mu_{S_{\tau-1}} + \phi_{S_\tau}\phi_{S_{\tau-1}}\mu_{S_{\tau-2}} + \phi_{S_\tau}\phi_{S_{\tau-1}}\phi_{S_{\tau-2}}\mathbb{E}[\pi_{\tau-3}|\psi_{\tau-3}] \\
&= \mu_{S_\tau} + \sum_{t=0}^{\tau-1}[\mu_{S_t}\prod_{j=t+1}^{\tau}\phi_{S_j}]
\end{aligned} \tag{9}$$

2. Compute a weighted average of $\mathbb{E}[\pi_\tau|\psi_\tau]$,which are calculated from Equation(9), with respect to the paths of transition probabilities because the regimes are not actually observed. The probabilities of each path $Pr(\psi_\tau)$ can be determined recursively as the product of the transition probabilities for each regime switch along the path ψ_τ.

$$\begin{aligned}
\mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1} = 1] &= \rho_{11}\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 1] + (1-\rho_{11})\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 2] \\
\mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1} = 2] &= (1-\rho_{22})\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 1] + \rho_{22}\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 2]
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[\pi_\tau|\psi_{\tau-2}, S_{\tau-2} = 1] &= \rho_{11}\mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1} = 1] + (1-\rho_{11})\mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1} = 2] \\
\mathbb{E}[\pi_\tau|\psi_{\tau-2}, S_{\tau-2} = 2] &= (1-\rho_{22})\mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1} = 1] + \rho_{22}\mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1} = 2]
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[\pi_\tau|\psi_{\tau-2}, S_{\tau-2} = 1] = \rho_{11}(\rho_{11}\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 1] + (1-\rho_{11})\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 2]) + \\
(1-\rho_{11})((1-\rho_{22})\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 1] + \rho_{22}\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 2])
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[\pi_\tau|\psi_{\tau-2}, S_{\tau-2} = 1] = (1-\rho_{22})(\rho_{11}\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 1] + (1-\rho_{11})\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 2]) + \\
\rho_{22}((1-\rho_{22})\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 1] + \rho_{22}\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau = 2])
\end{aligned}$$

$$\vdots$$

$$\mathbb{E}[\pi_\tau|S_0] = \sum_{i=1}^{2^\tau} Pr(\psi_{\tau i})\mathbb{E}[\pi_\tau|\psi_{\tau i}] \tag{10}$$

$$\text{where } Pr(\psi_\tau) = \prod_{t=1}^{\tau}\rho_{S_{t-1}S_t} \text{ and } S_0 \text{ is known}$$

9 Expected Variance of Inflation

The expected future variance of inflation given only the initial regime, $\text{Var}[\pi_\tau|S_0]$, can be calculated using a similar approach. However, unlike the expected mean, the expected variance at the terminal nodes are independent from its previous nodes.

$$\begin{aligned}\text{Var}(\pi_\tau|\psi_\tau) &= \mathbb{E}[(\pi_\tau - \mathbb{E}[\pi_\tau|\psi_\tau])^2|\psi_\tau] \\ &= \mathbb{E}[(\pi_\tau - \mu_{S_\tau} - \phi_{S_\tau}\pi_{\tau-1})^2|\psi_\tau] \\ &= \mathbb{E}[\varepsilon^2_{S_\tau}|\psi_\tau] \\ &= \sigma^2_{S_\tau}\end{aligned} \tag{11}$$

Similar to expected mean, we now need to take a weighted average of these $\text{Var}(\pi_\tau|\psi_\tau)$ with respect to the transition probabilities in order to get $\text{Var}[\pi_\tau|S_0]$. Unlike expected mean though, $\text{Var}(\pi_\tau|\psi_{\tau-1})$ is not just $\text{Var}(\pi_\tau|\psi_\tau)$ weighted by the transition probabilities. There is a jump component given that the conditional mean is different across regimes.

$$\text{Var}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=1] = \rho_{11}\sigma_1^2 + (1-\rho_{11})\sigma_2^2 + \rho_{11}(1-\rho_{11})(\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau=2] - \mathbb{E}[\pi_\tau|\psi_\tau, S_\tau=1])^2$$
$$\text{Var}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=2] = (1-\rho_{22})\sigma_1^2 + \rho_{22}\sigma_2^2 + (1-\rho_{22})\rho_{22}(\mathbb{E}[\pi_\tau|\psi_\tau, S_\tau=2] - \mathbb{E}[\pi_\tau|\psi_\tau, S_\tau=1])^2$$

$$\begin{aligned}\text{Var}[\pi_\tau|\psi_{\tau-2}, S_{\tau-2}=1] = \rho_{11}\text{Var}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=1] + (1-\rho_{11})\text{Var}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=2] + \\ \rho_{11}(1-\rho_{11})(\mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=2] - \mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=1])^2\end{aligned}$$

$$\begin{aligned}\text{Var}[\pi_\tau|\psi_{\tau-2}, S_{\tau-2}=2] = (1-\rho_{22})\text{Var}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=1] + \rho_{22}\text{Var}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=2] + \\ (1-\rho_{22})\rho_{22}(\mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=2] - \mathbb{E}[\pi_\tau|\psi_{\tau-1}, S_{\tau-1}=1])^2\end{aligned}$$

After recursing back to time 0, we can obtain $\text{Var}(\pi_\tau|S_0)$.

10 Inflation Beta

We use inflation betas a la Wang and Bekaert (2010) as a measure of how an asset moves relative to inflation. Using historical data, linear regression is applied to find a beta corresponding to each asset available for investing during the testing horizon. Ideally, regime specific betas would have been calculated using historical periods corresponding to each of the regimes. However, due to a lack of older asset data, and the stagnation of inflation regimes over the periods for which assets were available, only betas for one of the regimes would be calculated. As such, we use the sample data for the period leading directly into the investing horizon to find betas that we feel would be most applicable to the period. Betas are calculated using the polyfit function in MATLAB. These inflation betas are then fed into the main optimization model as described later.

11 Markov Inflation Factor Model

Our model modifies the commonly accepted mean-variance framework to account for risk and return variation related to inflation. The model uses the Markov regime switching models to find the expected inflation and inflation variance, and then incorporates these when finding the adjusted returns and variances of each asset. The complete model is presented below, and the objective and each constraint are more thoroughly explained in the subsequent sections.

$$\text{minimize} \sum_{i=1}^{n}(\beta_i^2\sigma_f^2 + \sigma_{\epsilon i}^2)x_i^2 + \sum_{i=1}^{n}\sum_{j=1}^{n}\beta_i\beta_j\sigma_f^2 x_i x_j$$

$$\begin{aligned}\text{subject to } \sum_{i=1}^{n}(\mu_i x_i - t(\delta_i^+ + \delta_i^-) &\geq \mathrm{R} \\ x_i - \delta_i^+ + \delta_i^- &= x_i^{t-1} \qquad i = 1, 2, \ldots, n \\ \sum_{i=1}^{n} x_i &= 1 \\ x_i, \delta_i^+, \delta_i^- &\geq 0 \qquad i = 1, 2, \ldots, n\end{aligned}$$

11.1 Objective

$$\sum_{i=1}^{n}(\beta_i^2\sigma_f^2 + \sigma_{\epsilon i}^2)x_i^2 + \sum_{i=1}^{n}\sum_{j=1}^{n}\beta_i\beta_j\sigma_f^2 x_i x_j$$

The objective function is a modification on the standard variance equation for a single factor MVO. β_i represents the inflation beta of the asset as per the previous section; σ_f^2 represents the inflation variance, as calculated through the Markov tree equations; $\sigma_{\epsilon i}^2$ represents the random noise variance for asset i calculated through single factor model; x_i represents the proportion of total wealth allocated into asset i.

The key difference in this objective, as opposed to a standard SF model on inflation, is in the inflation and noise variances. Unlike standard factor models, these variances are calculated accounting for potential future states using Markov regime switching, instead of using just historical data. As a result, our objective, and accordingly our measure of risk, is no longer standard portfolio variance as would be the result of a regular single factor model. It is rather an inflation adjusted variance measure. See the results and analysis section for a comprehensive look at the impact of this measure.

11.2 Portfolio Return and Turnover Constraint

$$\sum_{i=1}^{n}(\mu_i x_i - t(\delta_i^+ + \delta_i^-)) \geq \mathrm{R}$$

This constraint forces the portfolio's expected return subject to some transaction cost to be greater than or equal to some minimum return specified by the investor. μ_i is the inflation adjusted expected return of asset i; δ_i^+ and δ_i^- represent the amount of turnover from the previous periods allocation to the new allocation into asset i; t is the transaction cost; R is the investor's minimum acceptable portfolio return.

The method used for adjusting an asset's return for inflation is to perturb the nominal return $\mu_{i_{\mathrm{nom}}}$ by the expected inflation scaled by the asset's inflation beta, i.e.

$$\mu_i = \mu_{i_{\mathrm{nom}}} + \beta_i \mu_f$$

By making this adjustment, assets that have positive inflation betas are favoured over the others as they are given an expected return "boost". Generally speaking, stocks with high inflation betas tend to be good hedges against inflation and as such, our model is more likely to choose stocks that would perform well within hyper inflationary periods.

The turnover variables stem from the transaction costs incurred by an absolute change in portfolio allocation:

$$\sum_{i=1}^{n}(\mu_i x_i - t|x_i - x_i^{t-1}|) \geq \mathrm{R}$$

The decision variables in our model are a result of linearizing the absolute value as in:

$$x_i - \delta_i^+ + \delta_i^- = x_i^{t-1} \qquad i = 1, 2, \ldots, n$$

where x_i^{t-1} represents the previous period's allocation into asset i

11.3 Feasibility Constraints

$$\sum_{i=1}^{n} x_i = 1, \qquad x_i, \delta_i^+, \delta_i^- \geq 0 \qquad i = 1, 2, \ldots, n$$

The sum of the individual asset allocations cannot exceed our total wealth; short selling is not allowed; turnover variables must be positive for optimization feasibility.

12 Data Generation

Data generation was done using Python to scrape the Yahoo Finance website for .csv formatted files. The tools used in the scraping program include: Native Python libraries (os, requests, csv), BeautifulSoup library to parse the Yahoo Finance website and Virtual Environment Setup (Virtualbox and Vagrant)

The program is split into 2 parts:

1. Scrape.py - Get data from Yahoo Finance Website

   Yahoo Finance returns a page based on the parameters given in the query URL via the http GET method. For example, the URL

   https://ca.finance.yahoo.com/q/hp?s=GE&a=00&b=2&c=1962&d=10&e=30&f=2014&g=m&z=66

   will display the first 66 monthly historical prices for GE (General Electric) between January 2, 1962 and Nov 30, 2014 on the Yahoo Finance Website.

   The python script saves the html of the entire page, and BeautifulSoup parses the date and price for each entry. The script then repeats with 132 in place of 66 to request the next 66 asset prices. When given a list of ticker symbols to scrape for, the program loops through each one and for each asset, performs the same iteration over the historical prices and saves the data in a csv file.

2. Delete.py - Filter and Format data for MATLAB

   After the first script has been completed, each asset will have a .csv file with all of the yahoo prices saved. This second script goes through and searches for dividend payments, duplicates in dates, and ensures that two consecutive asset prices do not occur within the same month since we want monthly prices. This creates a new folder containing quality controlled asset prices. The parameters given to the python scripts were for monthly opening stock prices between January 1960 and September 2014.

13 Benchmarks

We chose 4 different benchmarks to compare our portfolio against.

1. Standard Mean-Variance Optimization (MVO)

   Standard MVO is a cornerstone portfolio diversification framework aimed at minimizing the portfolio variance while maintaining a certain level of portfolio return. The model is still commonly used in practice, thus its performance under the same set of time and rebalancing periods should serve as a good benchmark for our inflation-hedging model.

2. VWELX (Vanguard Wellington Fund Investor Shares)

VWELX is a mutual fund incepted in July 1929, whose investment objective is to provide long-term capital appreciation and moderate current income. This moderate allocation fund has had an excellent track-record and is rated highly in the industry. Hence, its performance is also a decent benchmark.

3. LOMMX (CGM Mutual Fund)

   LOMMX is a mutual fund incepted in November 1929, whose investment objective is to achieve reasonable long-term capital appreciation while prudently protecting its capital from undue risks. It is an aggressive allocation fund with a long history and thus is another suitable benchmark.

4. S&P500

   The S&P500, which represents the set of the 500 largest market-cap US companies, is a classic benchmark against which most investment vehicles are compared against.

14 Risk Measures

Aside from comparing the return performances of our model with the benchmark returns, the risk measures as well as Sharpe Ratio, Modified Sharpe Ratio and Portfolio Beta will be compared.

$$\text{Sharpe Ratio} = \frac{\overline{r}_p - r_f}{\sigma_p}$$
$$\overline{r}_p = \sum_{i=1}^{n} \mu_i x_i$$
$$\sigma_p = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} \sigma_i \sigma_j x_i x_j}$$
$$r_f = \text{3-month treasury bill rate (secondary market)}$$

The Sharpe ratio is a highly regarded risk-adjusted measurement of return used in finance. It represents the amount of excess return received per extra unit of volatility (a higher Sharpe ratio is more favourable). The portfolio return is assumed to be normally distributed. The Sharpe ratio is expected to be higher for the standard MVO portfolio since MVO's objective function specifically minimizes the traditional measure of portfolio variance.

$$\text{Modified Sharpe Ratio} = \frac{\overline{r}_p - r_f}{\sigma_p}$$
$$\overline{r}_p = \sum_{i=1}^{n} \mu_i x_i$$
$$\sigma_p = \sqrt{\sum_{i=1}^{n} (\beta_i^2 \sigma_f^2 + \sigma_{\varepsilon i}^2) x_i^2 + \sum_{i=1}^{n}\sum_{j=1}^{n} \beta_i \beta_j \sigma_f^2 x_i x_j}$$
$$r_f = \text{3-month treasury bill rate (secondary market)}$$

The Modified Sharpe ratio is a created risk measurement unique to this paper. It is similar to the Sharpe ratio but instead uses the objective function of the Inflation-Hedged portfolio equation as its variance. The modification accounts for the risk posed by a high inflation rate. It is similar to the Sharpe ratio in the sense that a higher value is expected for the inflation-hedged portfolio (since the objective function minimizes the measure of variance). Both measures will be examined when assessing the risk-adjusted return of the portfolio and the value of the model over the time period.

$$\text{Weighted Portfolio Inflation Beta} = \sum_{i=1}^{n} \beta_{inf,i} x_i$$

As shown above, the Weighted Portfolio Inflation Beta(WPIB) is computed by summing the products of the asset allocation weights and their corresponding inflation betas. The WPIB provides insight into the difference between standard MVO and our model. For the most part, the WPIB is higher for our model representing an allocation skewed towards assets with higher inflation betas. Surprisingly, sometimes standard MVO boasts a higher WPIB than our model. This can be explained by looking at the form of our model.

The return constraint of our model incorporates the expected return of the assets. The usual expected return is perturbed by the asset inflation beta $\times$ (expected inflation rate) in our model. This means that assets with higher inflation betas will generally have higher expected returns. In order to meet the return constraint there must be a large allocation in these assets.

The objective function of our model shows a different picture. When minimizing the objective function, with the inflation betas as coefficients of the asset weights, allocation in assets with lower inflation betas are favoured. Note that this corresponds to the variance term but the intuition for the covariance term is slightly different. In this case, the objective function has a preference to reduce the covariance term by investing in assets with a smaller product of inflation betas. This means that some of the assets invested in, are likely to have negative inflation betas. This makes sense in the current iteration of the model, due to the balancing by the return constraint. However there is room for future improvement here. Changing the Beta terms to be 1/Beta is one of the many ideas discussed in the improvements section.

Note that the WPIB is not displayed in association with any of the graphs because it is not a true risk measure but rather, is used behind the scenes to better understand the chosen allocation.

These risk-measure results demonstrate the true nature of our model and its strategy. It attempts to hedge against inflation risk by actively investing in assets that move with inflation. In the case of inflation shocks or rising inflation rates, the portfolio will remain stable or even increase while other portfolios may decrease in value. This is only true if the minimum return constraint is high enough as setting it too low can lead to a higher WPIB for the standard MVO portfolio.

15 Results

15.1 1983-1993

Figure 1: Cumulative returns of model and benchmarks from 1983-1993; with rebalancing

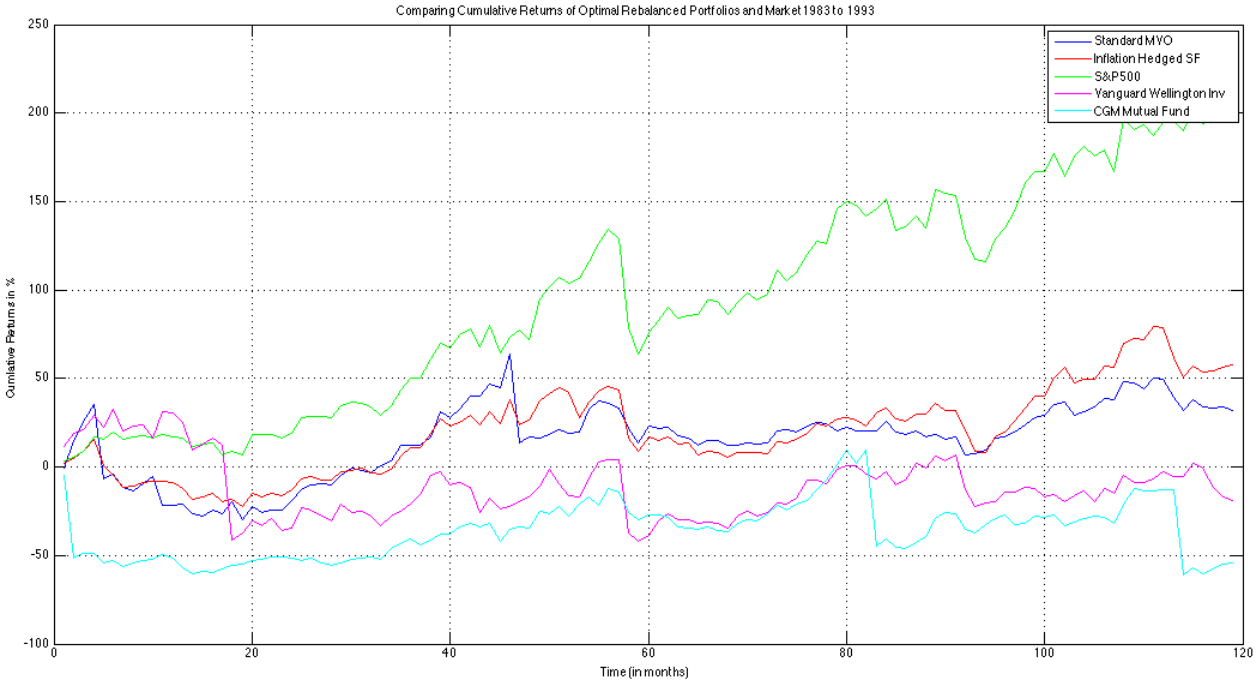


Table 1: Inflation Rates between 1986 and 1987

Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1986	3.9	3.1	2.3	1.6	1.5	1.8	1.6	1.6	1.8	1.5	1.3	1.1
1987	1.5	2.1	3.0	3.8	3.9	3.7	3.9	4.3	4.4	4.5	4.5	4.4

Table 2: Sharpe and Modified Sharpe Ratios from 1983-1993

	Modified Sharpe Ratio	Sharpe Ratio
Inflation-Hedged	0.3951	0.0756
Standard MVO	0.0007	4.1220

Used S&P500 data. Minimum Monthly Return of 0.005.

Regime Switching model constructed using data starting from 1914

Figure 1 displays the performance of various portfolios from 1986 to 1987 where US inflation rates rose from 1.1% to 4.5%. Since the Inflation-Hedged portfolio consists of high inflation beta assets, the portfolio price rises when the inflation rate jumps as opposed to the standard MVO portfolio which suffers a sharp drop in price, specifically when the inflation rate jumps from 1.5% to 3.8% in 4 months. The Inflation-Hedged portfolio also outperforms the standard MVO portfolio at the end of the time frame. This corresponds to the Gulf War, which caused a mild increase in inflation rates. Although crude oil prices doubled from \$20 to \$40 during this time, the lack of change in the inflation rate indicates the end of the strong correlation between oil prices and inflation rates. This is the main reason for excluding time series data for commodities like oil prices when constructing the regime switching model for inflation.

15.2 1990-1998

Figure 2: Cumulative Returns of model and benchmarks from 1990-1998; without rebalancing

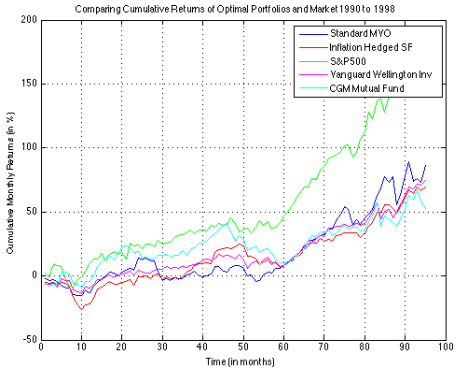
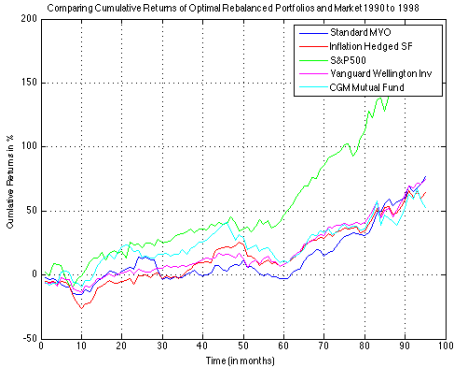


Figure 3: Cumulative Returns of model and benchmarks from 1990-1998; with rebalancing



Used S&P500 and NYSE data. Minimum Monthly Return of 0.005.
Regime Switching model constructed using data starting from 1914

Figures 2 and 3 show that the standard MVO portfolio and our Inflation-Hedged portfolio perform similarly from 1990 to 1998. This time period showed little change in inflation rate.

Table 3: Modified Sharpe Ratios from 1990-1998

	Period 1	Period 2
Inflation-Hedged	-0.0396	-0.8890
Standard MVO	0.0413	-0.0048

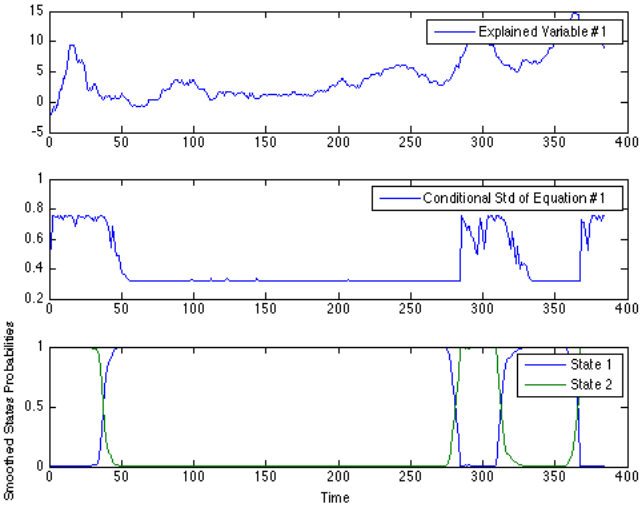
Table 4: Sharpe Ratios from 1990-1998

	Period 1	Period 2
Inflation-Hedged	-0.0149	-0.3907
Standard MVO	0.8709	-7.0746

Results in Tables 3 and 4 are contrary to normal expectations. When comparing to the standard MVO portfolio, the Inflation-Hedged portfolio has an inferior modified Sharpe ratio and a superior Sharpe ratio.

15.3 1983-1991

Figure 4: Regime Switching from 1950 to 1981



Used S&P500 and NYSE data. Minimum Monthly Return of 0.005.

Regime Switching model constructed using data starting from 1914

Regime 1 was the dominant regime until the 1970's. Then Regime 2 took over, which corresponds to the 1973 oil crisis when inflation rates rose to over 12% throughout the year of 1974, temporarily returning to normal, but shooting up again in 1979.

Figure 5: Cumulative Returns of model and benchmarks from 1983-1991; without rebalancing

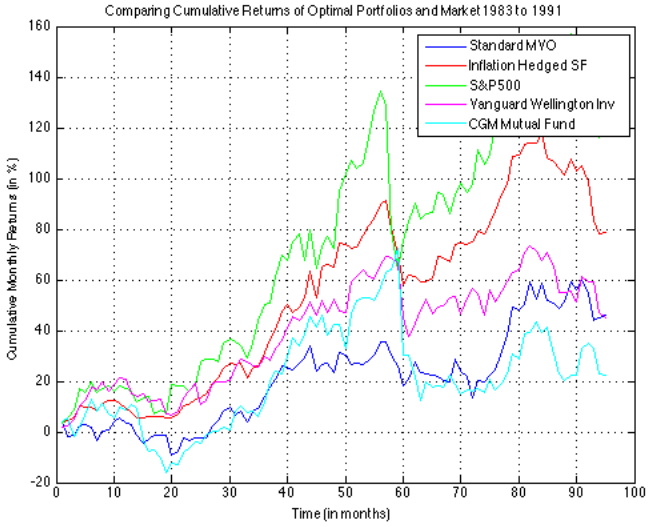
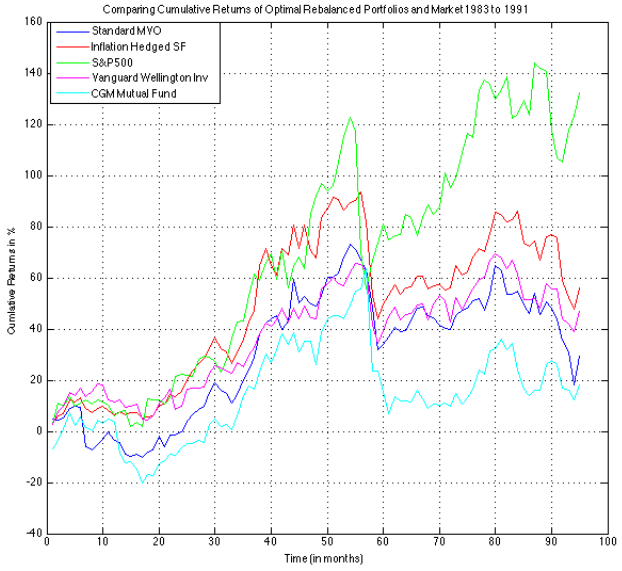


Figure 6: Cumulative returns of model and benchmarks from 1983-1991; with rebalancing



Looking at Figures 5 and 6, we can see that over the 8 years, the inflation-hedged portfolio performs better than the standard MVO portfolio. Note: The sharp drop in portfolio value at around 56 months corresponds to Black Monday when stock markets around the world crashed. It is not completely apparent as to why the inflation-hedged portfolio performs better but testing over many time intervals and desired return constraints has shown that the inflation-hedged portfolio follows a cumulative return graph very similar to that of standard MVO.

Interestingly, the non-rebalanced portfolio outperforms the rebalanced portfolio for both the inflation-hedged portfolio and standard MVO. This may also be a result of Black Monday where the returns used for re-optimization are overly pessimistic and do not accurately represent the expected return of an asset over the next time period.

Table 5: Modified Sharpe Ratios from 1983-1991

	Period 1	Period 2
Inflation-Hedged	-0.2790	0.9613
Standard MVO	-0.0901	0.0066

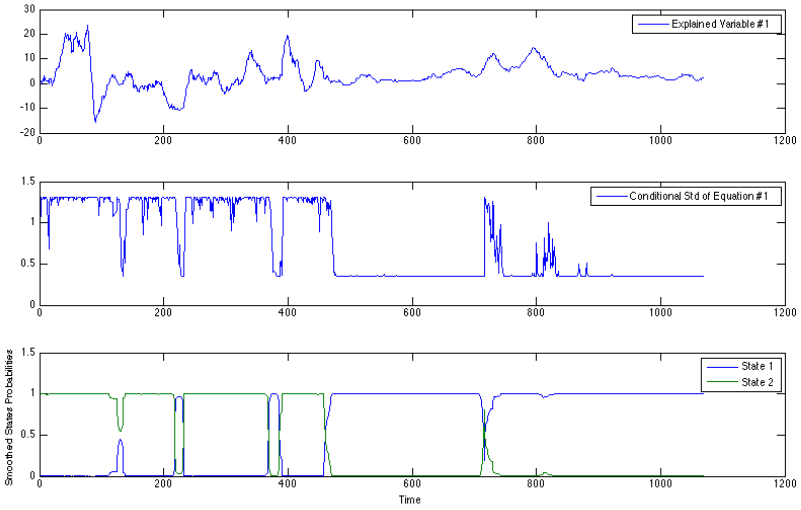
Table 6: Sharpe Ratios from 1983-1991

	Period 1	Period 2
Inflation-Hedged	-0.1314	0.3682
Standard MVO	-0.5037	0.3611

Tables 5 and 6 indicate that the modified Sharpe ratio is much better for the inflation-hedged portfolio than for standard MVO, as expected. This is for the time period 1987-1991. This period removes significant short-term return anomalies (resulting from the 1987 crash), which skews the sharpe ratio for the time period of 1983-1987, making this period unsuitable. However, the Sharpe ratios for both portfolios are very close in value with the inflation-hedged portfolio performing slightly better. This is promising because it shows that the Inflation-Hedged Portfolio has the potential to outperform in terms of the custom risk measure (the modified Sharpe ratio), while maintaining an acceptable level of the conventional risk measure (the Sharpe ratio).

15.4 2004-2012

Figure 7: Regime Switching from 1914 to 2002



As can be seen, the inflation regime stays at regime 1 for several decades. There is a slight jump in the 1970's in inflation rate but it remains in the same Regime. This is different from other Regime-Switching models because the regimes are constructed using data as early as 1914. Inflation rates are much more volatile during the early 1900's than a typical inflation shock occuring before 1950. This means that the conditions necessary for regime 2 are quite strict, and thus regime 2 does not occur for any significant time before 1950.

Figure 8: Cumulative Returns of model and benchmarks from 2004-2012; without rebalancing

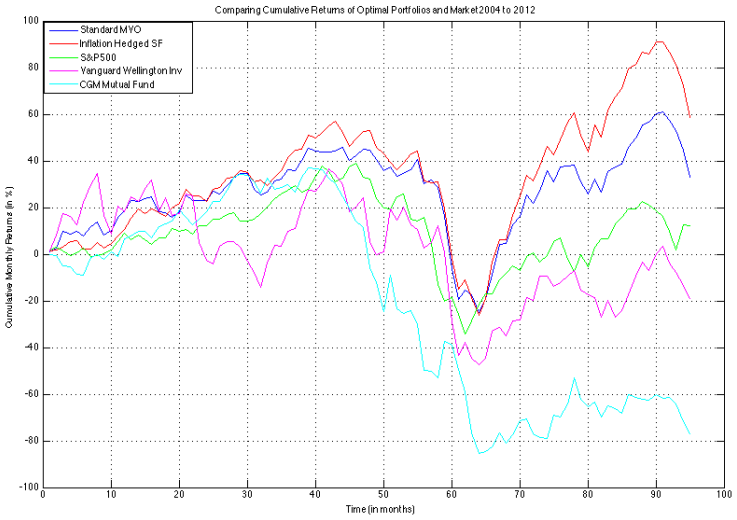
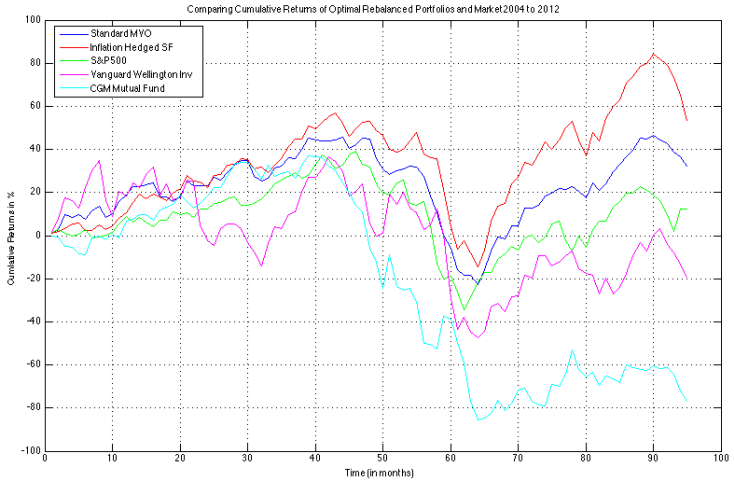


Figure 9: Cumulative Returns of model and benchmarks from 2004-2012; with rebalancing



Used S&P500 and NYSE data. Minimum Monthly Return of 0.005.
Regime Switching model constructed using data starting from 1914

Similar to the 1983-1991 graph, this graph shows that the inflation-hedged portfolio can produce aggressive returns and performs well during stable inflation periods. The financial crisis of 2008 caused troughs in all the portfolios with the lowest trough (and highest value)occuring in the inflation-hedged portfolio. The inflation rate during the crisis was slightly negative. It is expected that investing in assets with high inflation betas would cause a negative affect but the graph indicates this effect is either negligible or non-existent.

16 Improvements and Modifications

Moving forward, there are many modifications that can be made to the process and/or model which may lead to better results. The foremost thing is to collect more time-series data. Currently, the model has approximately 31 assets which have historical data a early as 1970. This is quite a small amount and the purchasing of older time series data for assets in the S&P500/NYSE should yield more conclusive results. It will also allow for closer analysis in regards to what assets are invested in. It can also allow for industry constraints to be included, mandating a minimum weight invested in companies belonging to particular industries. The older data is important because there have been no large inflation shocks in the last 20 years. During 1970's however, there were large inflation shocks in 1973 and 1979 corresponding to an oil crisis and the Iranian revolution respectively.

Another way to gauge the effectiveness of the model would be to apply it to different countries and their corresponding stock exchanges. Countries whose inflation rates have been volatile or suffered large shocks in recent years, such as Turkey, would be ideal for testing.

The addition of older time series data(ideally dating to before 1950) will allow for the calculation of regime-dependent inflation betas. The Markov tree would then be constructed not for the expected inflation rates, but for the expected asset return. The terminal values would be the nominal expected return plus the product of the inflation-beta associated with that node and the inflation rate for that node(which has to be calculated recursively). This would yield a much longer computation as it has to be computed for each asset but it should yield better results overall.

The current model for inflation rates is an AR(1) process that only uses the previous inflation rate. The addition of a macroeconomic variable, such as the monthly unemployment rate, as another autoregressive term may make the model more accurate in forecasting inflation. The unemployment rate in particular, holds information on inflation targeting by the Federal Reserve as the Phillips curve shows the connection between the two. It further allows for more complex regimes to be defined such as stagflation, where higher levels of both unemployment and inflation are experienced as opposed to the usual inverse relationship between the two.

The transition probabilities in our model are constant. A notable improvement is to use duration-dependent transition probabilities, meaning that they change over time. This makes sense because if you have been in a regime for a long time, it should become more likely to change regimes.

Another important addition would be to add a user specification for their tolerance to inflation risk. One way is to add a coefficient ranging from 0 to 2 in front of the term that perturbs an asset's nominal return. This would influence the assets' expected returns, as a larger coefficient would lead to a larger allocation in assets with higher inflation betas and the contrary for a smaller coefficient.

Modifying or completely changing the objective function is another consideration. In its current form, the covariance of the assets is expressed by using the inflation Betas as an intermediate term. Separating this to have a term representing the assets covariance without using inflation beta would be useful because it allows for a distinction between the two different risks. The program currently computes CAPM beta as well and including this as a second factor is another potential extension.

17 Alternative Model

$$\text{minimize} \sum_{c_1=1}^{m} \sum_{c_2=1}^{m} \sum_{i=1}^{n_{c_1}} \sum_{j=1}^{n_{c_2}} \sigma_{ic_1} \sigma_{j_c 2} x_{ic_1} x_{jc_2}$$

$$\begin{aligned}
\text{subject to } & \sum_{c=1}^{m} \sum_{i=1}^{n_c} \mu_{ic} x_{ic} \geq \text{R} \\
& \sum_{c=1}^{m} \sum_{i=1}^{n_c} \delta_{ic} x_{ic} = 1 \\
& \sum_{c=1}^{m} \sum_{i=1}^{n_c} ind_{ic}^{r} x_{ic} = indalloc^{r} \qquad \forall r \in \text{Tech, Gold, Oil, etc.} \\
& \sum_{r=1}^{rtotal} indalloc^{r} = 1 \\
& \sum_{r=1}^{rtotal} ind_{ic}^{r} = 1 \\
& x_{ic} \geq 0 \\
& \delta_{ic} = 0 \text{ or } 1 \\
& ind_{ic}^{r} = 0 \text{ or } 1 \qquad \forall r \in \text{Tech, Gold, Oil, etc.} \\
0 \leq & indalloc^{r} \leq 1 \qquad \forall r \in \text{Tech, Gold, Oil, etc.}
\end{aligned}$$

C_0 is the set of markets
m is the total number of markets
I_c is the set of available investments in market $c \in C_0$
n_{c1} is the number of assets in market $c1 \in C_0$
r is a set of industries used in the model

$indalloc^r$ is the corresponding desired allocation in each industry

An alternative framework that was considered was an MVO model that hedged against inflation indirectly by investing in industries that were historically shown to either be unaffected by inflation or to vary positively with it, e.g. gold and oil. Note that these industries are not necessarily the typical sector divisions but stocks can be classified into industries according to the user's choosing. The model also invests in various markets in an effort to hedge against domestic inflation.

The objective function is to minimize the standard portfolio variance which accounts for the covariance of all assets in all markets. The constraints are as follows:

1. Minimum expected return constraint (same for MVO). Just like the objective function there is an extra summation to represent the allocation over all markets/currencies.
2. Cardinality constraint (optional). It determines which of the assets can be allocated into.
3. Primary way in which the model would create an inflation-hedged portfolio. By specifying an exact allocation for each industry, the model can invest more weights into assets belonging to industries known to be good inflation hedges. This is assuming that those industries are assigned relatively higher indalloc values.
4. Allocated industry weights sum to 1.
5. Each asset only belongs to one industry.
6. No short selling.
7. Definition of the binary variables representing whether an asset is invested in and whether an asset belongs to a particular industry.

While this model has inherit characteristics that show that it accounts for inflation, it has some flaws. First of all, it is not necessarily reliable for the user to do his/her own analysis and assign weights to the industries. It is also not necessarily true that all stocks in an industry move the same way for inflation and there may be considerable variation. However, the ability for the user to specify a particular industry allocation breakdown may be a useful decision characteristic for portfolio managers.

The other problem with this model is that it implies that by investing in multiple markets/currencies, the resulting portfolio effectively hedges against inflation risk. While it may hedge against inflation risk in emerging countries such as Turkey it does not truly hedge against inflation risk in developed economies such as the United States. This is because the inflation risk present from investing in US markets is not high inflation rates, but uncertain inflation shocks where the inflation rate quickly rises from a low rate to a substantially higher one. These shocks however are generally cause by global phenomena such as an oil crisis and they affect inflation rates worldwide. Because of the high correlation between global inflation rates during these shocks, the methodology that investing in an internationally diversified portfolio is effective is partially discredited.

18 Literature Review

18.1 Existing Inflation-Hedging Optimization Frameworks

18.1.1 Inflation-hedging Portfolios in Different Regimes - (Briere, Signori)

Summary: This article utilizes an optimization framework known as the Mean/Shortfall Probability Investment Framework. The objective of this framework is set to maximize the inflation adjusted real returns while subject to the constraint that the probability of a shortfall remains lower than a threshold set by the investor. In other words, the probability of returns falling below a certain target must be controlled below a certain value. The results showed that the macroeconomic volatility significantly influences the investor?s optimal allocation. In a volatile regime, a ?safety-first? investor having a pure inflation target(real return of 0%) should be mainly invested in cash when his investment horizon is short, and should increase his allocation to inflation-linked bonds, equities, commodities and real estate when horizon increases. In a more stable economic environment, the same investor should be again mainly invested in cash when investment horizon is short and increase his investment in nominal bonds and to a lesser extent in commodities and equities when his horizon increases.

Relevance: This article provides another alternative optimization framework designed to hedge for inflation. The contrast in objective and constraints as well as the results obtained yield a broader base of ideas for formulating our own framework.

18.1.2 Dynamic Asset Allocation under Inflation - (Brennan, Xia)

Summary: The paper develops a simple framework with closed form expressions for an MVO problem with inflation and only nominal assets available for trade. In this paper?s model, short time horizons seem to be relatively insensitive to inflation variation, while long run investments seem to produce more pronounced effects.

Relevance: A mathematical background and framework for inflation adjusted optimization. Provides precedent and allows us further understanding to formulate our own model.

18.2 General Hedging Methods

18.2.1 Protecting a Portfolio Against Inflation Risk (Strongin, Petsch)

Summary: This paper defines Inflation Risk, emphasizes the importance of protecting portfolios against future inflation, and discusses current financial instruments that may be useful in hedging inflation. Its primary argument is that global diversification in equities is the superior way to protect against domestic inflation, while a diversified portfolio of commodities best protects against global inflation. It examines changes in monetary policy and how these changes impact asset returns through inflation. It also includes a quantitative assessment of the historical effectiveness of different inflation-hedging assets. A 12-month Return Regression is performed on US CPI inflation with T-Bills, the SP 500 and Gold being among the variables analyzed.

Relevance: This paper is one that is very relevant to the task at hand. It clearly defines the problem that is trying to be solved, through a section called "What is Inflation Risk?? It specifically mentions TIPS (Treasury Inflation Protected Securities), which is one of the many benchmarks that we intend to measure the success of our portfolio framework against. The paper makes the key point that TIPS has historically protected against global stagflations such as oil shocks, but this could have been better protected by commodity exposure. The inclusion of commodities in our portfolio now becomes a focal point of consideration in addition to fixed-income securities and equities.

18.2.2 Inflation - Hedging it and Trading it (Deutsche Bank)

Summary: This paper discusses Deutsche banks' strategy for hedging inflation especially in lieu of their view that inflation will be particularly volatile over the next 5 to 10 years. It discusses inflation-linked bonds including EUR sovereign linkers and US TIPS. It notes various risk measures such as the concepts of duration and convexity and how they affect the price of linkers. Linker Asset Swaps and the Leverage effect are explained in detail, where the investor buying a bond pays all the cash-flows from the bond and in exchange, receives LIBOR+x% until maturity, This type of asset is particularly useful if investing in the bonds of countries that are susceptible to default.

Relevance: Similar to the Vanguard paper, this one is useful because it comes directly from a financial institution, whose perspective may be different and more practical then one originating from purely academic source. It provides information about existing assets whose value is inherently linked to inflation. It also discusses Inflation indices such as CPI and what it's comprised of. It discusses the mechanics as well as the advantages and disadvantages of numerous complex financial instruments that can be used to hedge inflation. This serves as a good reference for existing assets that can serve as a benchmark for the portfolio framework we intend to create.

18.2.3 Common Stocks as a Hedge against Inflation - (Bodie)

Summary: The question of how an investor can hedge against inflation under MVO principles by using common stocks is addressed in this paper. The measure of hedging effectiveness was deemed

to be proportional to the reduction in variance of the real return of a bond in combination with the portfolio. It determines that inflation hedges depend on 2 parameters: 1)the ratio of variance of non-inflation component of real return on common stocks to the variance in their unanticipated inflation, and 2) the difference between the nominal return on the bond and the coefficient of anticipated inflation in the quation for real return on equity (explained in the paper). The research in the paper determine that in order to use the common stock as a hedge, one must short them.

Relevance: This paper gives us some usable metrics for measuring hedge effectiveness and some potential methods for hedging as well.

18.2.4 Common Stocks as a Hedge Against Inflation: Evidence from Century-long US data - (Kim, Ryoo)

Summary: Monthly stock data from 1900 onward was analyzed to investigate the relationship between common stocks and inflation. The paper concludes that US stocks have been effective hedges against inflation, and that prior studies had neglected certain elements, namely elasticity and asymmetric adjustment.

Relevance: Provides a better theoretical understanding of hedging methodologies through empirical evidence and analysis. It has some interesting contrasts from the other paper on common stocks and inflation, and as such both papers need to be analyzed further.

18.3 Modification Methods to Existing Frameworks

18.3.1 Examining Portfolio Optimization as a Regression Problem - (Brides)

Summary: This paper examines different types of mean-variance analysis as a series of related regression models. It treats the risk-free return as the dependent variable and the assets as the independent variables. The coefficients from regression are portfolio weights and the intention is that every portfolio along the efficient frontier will be a solution to the regression problem. Various models are constructed including ones with and without the existence of a risk-free asset. It uses F-tests to evaluate portfolios to test for structural changes in efficient portfolios and uses weighted least squares to deal with volatility clustering, which is the tendency of large absolute movements in asset prices to be followed by more large magnitude changes.

Relevance: The usefulness of this paper is in its highly mathematical nature and its creativity. It attempts to recast the mean-variance portfolio problem as a linear regression one and apply analytical tools that become available as a result of it becoming a regression problem. It teaches how to use the regression tool box and conduct regression inference. It also discusses the identification of historical outliers that are influential to final estimates, which can then be disregarded after identification. Overall, this paper serves as a fundamental reference in the case that we decide to build a regression model to deal with the asset allocation problem and hedge inflation.

18.3.2 Building a Statistical Linear Factor Model and a Global Minimum Variance Portfolio using estimated covariance matrices (Wilcox, Gebbie)

Summary: This paper compares various estimated covariance matrices through the use of a global minimum variance portfolio and a statistical factor based expected returns model. Topics discussed include covariance estimation techniques, the linear factor model used to estimate expected asset returns as well as construction of a minimum variance portfolio and a simulation. It intends to choose the best sample covariance estimator with the purpose of applying it to empirical data from emerging markets.

Relevance: Through a thorough discussion of the mathematics of many aspects of portfolio optimization, this paper provides a useful reference for statistically modeling various things. It covers covariance estimation methods and provides several new perspectives on methods that can be used. It builds a statistical linear factor model from scratch providing derivation and tests the predictive power of the model as well. The chapter on the construction of a minimum variance portfolio is particularly useful as it reviews standard MVO but compares the standard deviation resulting from applying four different covariance matrix estimation techniques. The usefulness of the paper to us derives from it being very useful as a mathematical review and introduction to new elements of the portfolio optimization problem.

18.4 The Relationship between Inflation, Monetary Policy & The Markets

18.4.1 The Effect of Inflation and Money Supply Announcements on Interest Rates - (Urich, Wachtel)

Summary: This paper investigates the effect of money supply and inflation announcement on the interest rate. The survey data used in this study are the expectation of money supply and producer and consumer price indexes. The results show that unanticipated announced changes in producer price index and money supply has an immediate effect on the markets money supply. On the other hand the consumer price index has no immediate effect and no evidence of long-term effect on the interest rate.

Relevance: Knowing how interest rate reacts to inflation changes is a crucial point when trying to develop and model a portfolio optimization frame work investing in assets where real value is affected by changes in inflation and interest rate. Therefore it is important to come up by a relation between inflation and interest rate and use it in the framework of our optimization model.

18.4.2 The Cause of Inflation ? (Mishkin)

Summary: The paper tries to explore the causes of inflation and the effects of monetary policy on the inflation process. In the United States of America the main cause of inflation is the goal of achieving high employment rate. There fore controlling the inflation rate might cost in terms of unemployment and decrease in the countries output.

Relevance: Knowing the different causes of inflation is the fundamental knowledge required to control inflation. Printing money, rising taxes and wages, changing the price of raw material and so on are some factors that affect inflation. Therefore by changing the magnitude of these factors, the Federal Reserve objective inflation rate can be obtained.

18.4.3 Does Philips curve conditionally help to forecast inflation? - (Dotsey et al.)

Summary: Many studies have been shown that in the past 20 years, the Philips curve has not been able to predict inflation any better than a univariate forecasting model. In this paper the Phillips curve method is put under the test using real life data and the results show that this method is unconditionally inferior to the univariate forecasting model and sometimes this difference is very significant.On the other hand, conditioning on different measures of the state of the economy can improve the Philip curve model significantly. In the end, improved Philips curve method forecast tend to be more accurate when the economy is weak and not so accurate when strong.

Relevance: Philips curve have been the foundation of inflation forecasting for many years. The models used today are mostly an improved version of Phillips curve. Understanding the Phillips curve model and its sub parts is essential to portfolio optimization model that we want to develop since we need to understand that how inflation rate is going to change in future and how it will affect the real value of the portfolio's assets.

18.4.4 Inflation Hedging for Long-Term Investors - (Attie, Roache)

Summary: This article provides a comparison of the inflation sensitivity of several traditional asset classes: cash, nominal bonds, equities and commodities.Through applying the vector error correction model, it was found that commodities provides decent short-term hedges to rises in inflation, but may not work over longer horizons. Cash and nominal bonds, though respond very poorly to the inflation shock, do recover in the long-term. Hence, tactical asset allocation could enhance the inflation hedging properties of the portfolio.

Relevance: This article only examined each asset class's individual response to shocks in inflation and did not analyze the responses in the context of portfolio allocation. As a result, no optimization framework was provided. Hence, while the article does provide some background about the inflation sensitivity property of each asset class, it may not be useful for other purposes.

18.5 Regime Switching, Markov Models & Inflation Betas

18.5.1 Inflation Regimes and Inflation Expectations - (Gagnon)

Summary: This paper examines the formation of expectations about future inflation over long time horizons. One of the key points it makes is that long-term inflation expectations depend on more than just the past 5 or 10 years, indeed on decades of past inflation data. It advocates the benefits of a regime-switching model as it can explain the peculiar time-series properties of inflation

movements. It analyzes various models of inflation including one that takes in the inflation target of the previous period.

Relevance: This paper is useful because it establishes a basis and reasoning for using 50+ years of data to construct our regime-switching model.

18.6 Markov Switching Models in Empirical Finance - (Guidolin)

Summary:
This paper argues that over long horizons, impactful shifts in economic policies, especially monetary policy, could influence the expected future value of inflation. Hence, it may not be sufficient to base future estimations exclusively on historical data.

Relevance:
Given that estimating the future value of inflation is a critical piece in our overall portfolio optimization strategy, factors worth considering such as monetary policy should be noted.

18.6.1 Time Series Analysis - (Hamilton)

Summary:
This textbook is designed to focus on the topic of time-series in economics. It also covers important relevant concepts including numerical methods, Bayesian analysis, Maximum Likelihood Estimation, etc.

Relevance:
Maximum Likelihood Estimation is a critical technique in the design of our framework. This book serves as a reference tool for that.

18.6.2 Regime Switching Stochastic Volatility and Its Empirical Analysis(Zhang))

Summary:
This papers models economic processes as stochastic volatility models with regime switching, meaning that the volatilities of the regime variables are also randomly distributed. Utilizing this framework, the paper analyzes several empirical datasets including the S and P 500, Federal rates, exchange rates, and demonstrates that the proposed stochastic volatility model can outperform certain benchmark models in terms of fit and predictive ability.

Relevance:
The models presented in this paper are beyond the scope our current existing knowledge. However, certain statistical techniques embedded in the model such as Maximum Likelihood Estimation are within the scope of our analysis, and hence can serve as reference tools.

18.6.3 Duration-Dependent Transitions in a Markov Model of U.S GNP Growth - (Durland)

Summary:
Hamilton's filter can be extended to allow state transitions to be duration dependant by imposing additional restrictions so that the transition probabilities become a function of not only the current state but also of the number of periods the regime variable has been in that state.

Relevance:
This paper provides an idea for expanding upon our regime-switching framework. Intuitively speaking, it seems sensible that the length of time inflation has been in a particular regime influences the likelihood of a possible regime switch.

18.6.4 A Markov Switching Model of Inflation: Looking at the Future during Uncertain Times(Barraez)

Summary:
This paper analyzes the dynamics of inflation in Venezuela during the last eighteen years, through a Markov-switching model driven by the New Keynesian Phillips curve, which describes the relationship between rates of unemployment and inflation. The parameter estimations were completed using the Expectation-Maximization algorithm and the regimes were found to be influence by the main economic events that occurred during each regime.

Relevance:
The problem that this paper tackles is very similar to ours, and hence provides possible ideas which may be worth exploring further.

18.6.5 How Do Regimes Affect Asset Allocation - (Ang)

Summary:
Within a global asset allocation framework focussed on equities, it is difficult to exploit the presence of regimes with different correlations and expected returns because the benefits of international diversification trump the costs of ignoring the regimes. However, for all-equity portfolios, a regime-switching strategy certainly out-performs static strategies out-of-sample.

Relevance:
We perform an out-of-sample regime-switching strategy for an all-equity portfolio in our framework, though we focus exclusively on the North American market. Within this paper, there are derivations for the formulas to determine the expection means and variances of the regime variable (inflation in our case) that are useful.

18.6.6 Inflation Risk and the Inflation Risk Premium- (Bekaert)

Summary:
This paper discusses several points surrounding the concept "inflation hedging". First, some estimates of "inflation betas" for standard bond and well-diversified equity indices for over 45 countries were provided. Second, it was concluded that standard securities are poor inflation hedges. Expanding the portfolio to include foreign bonds, real estate and gold only yields a marginal improvement. Third, inflation index linked bonds do provide noticeable benefits to all relevant parties unless the market they are traded in is highly deficient. Finally, current term structure research shows that the inflation risk premium is very sizable and varies substantially over time.

Relevance:
This paper provides several ideas surrounding "inflation hedging" which is the overall goal of our framework. The most pertinent idea is the consideration of inflation-indexed bonds (TIPS in US), which seems to be a valuable inflation hedging tool. Also, inflation Betas is a critical factor in defining our portfolio variance, which will serves as the objective function in our MVO model.

18.6.7 Inflation and Individual Equities - (Ang, Briere, Signori)

Summary:
The papers extensively explores the concept of inflation Betas, signifying how each assets covary with inflation rates. It was found that though the aggregate stock is a poor hedge against inflation, there are considerable amount of individual stocks with decent hedging ability. The paper shows that stock Betas vary substantially over time and out of sample estimates of stock Betas are not totally reliable. Hence, it is difficult to construct stock portfolios that are good inflation hedges out of sample using these inflation Betas.

Relevance:
Though imperfect, Inflation Beta is a well researched method for considering the influence of inflation on the behaviour of individual assets and it serves a critical function in our inflation hedging framework. This paper is especially valuable because it provides all the details on how to calculate and apply Inflation Betas in the context of portfolio optimization.

18.6.8 International Asset Allocation With Regime Shifts(Ang, Bekaert)

Summary:
This article discusses a regime-switching strategy based on correlations and volatilities, in the context of international diversification. The regime-switching strategy does yield benefits but its value pales in comparison to the benefit of international diversification, especially for an all-equity portfolio.

Relevance:
Given that we are focused exclusively on the domestic market, this paper validates the value of regime-switching.

18.6.9 Regime-Switching Models - (Hamilton))

Summary:
This paper goes through an example of modelling economic time series using first-order autoregression with a 2-state Markov Regime-Switching Model, where only the means of the economic variable is regime-dependant.

Relevance:
First-order autoregression and 2-state Markov Regime-Switching are also utilized in our model for parameter. We extend the framework explained in this paper by assuming that the volatilities and autoregressive coefficients are also regime-dependant.

18.7 Matlab Packages

18.7.1 MS Regress - The MATLAB Package for Markov RS Models - (Perlin)

Summary:
This paper explains the MATLAB MS Regress package which utilizes MLE for the estimation, simulation and forecasting of a general markov switching model.

Relevance:
This MATLAB is a critical solver which we utilized in our estimation of the model parameters for inflation.

18.7.2 An Implementation of Markov RS Model with Time Varying Transition Probabilities in MATLAB(Ding)

Summary:
This memo explains the extension on the MATLAB MS Regress Package by Marcelo Perlin where the transition probability matrix is now time-dependant as opposed to constant.

Relevance:
Incorporating a time-variant transition probability matrix is possible extension we could apply to our model.

19 Appendix

19.1 Execution Instructions

The MATLAB code can be downloaded from https://github.com/seyonv/MIE479

Steps for generating graphs:

1. Navigate to Capstone-MATLAB
2. Add to Path(Selected Folders and Subfolders) MS_REGRESS_FEX_1.08, Symbol_Files, Main-ProgramFiles
3. In CSV_Files, add nyse_SP to path
4. Open the file "a.m" and modify parameters as desired. The comments above the file explain the parameters and how to choose them
5. run "a.m"

Steps for viewing all MATLAB workspace variables:

1. Complete steps 1-3 in "Steps for generating graphs"
2. In "a.m", comment out all the current code, and uncomment the line "run create_inflation_hedged_portfolio.m'
3. In "create_inflation-hedged_portfolio.m", delete the function header and its corresponding end
4. Uncomment the section at the beginning
5. Define variables as desired and run "a.m"

19.2 Files Directory

For underlined file names, see detailed code in the following subsection.

- User Input Files
  - a.m
    File that the user runs. The user defines the input parameters which include time periods, desired return and ticker symbols to use.
  - create_inflation_hedged _portfolio.m
    Main program file where all operations are executed. It calls the appropriate data fetching functions, conducts MLE to generate the Regime-Switching model, simulates the Markov tree for expected inflation and conditional variance of inflation. It then fetches data again for out-of-sample data and tests the model against standard MVO, S&P 500 and the two mutual funds. Finally, the cumulative return results are plotted (graphs for rebalanced portfolio and non-rebalanced portfolio).

  - NYSE_symbols.csv
    Ticker list of all the stocks on the NYSE(it was used to help fetch csv files of individual stocks).

  - riskfreerate2.csv
    Monthly data on the 3-month treasury bill rate, which is assumed as the risk free rate when computing the sharpe and modified sharpe ratios

  - unemployment_rate.csv
    Monthly data on the US unemployment rate. Unemployment rate was not a factor considered in the current iteration of this project but may be added to our inflation Regime-Switching model as a linear term.

  - SP500tickernames.xlsx
    Ticker list of all the current S&P500 stocks.

- CSV_Files
  Folder for all the CSV files used for obtaining asset data. Note that in the current iteration, only nyse_SP is used.

  - inf_funds
    Folder for time series data for particular funds and ETFs whose goal is to provide inflation-protection. Many of these funds invest up to 80% of holdings into TIPS

  - inflation assets
    Folder for assets that have high inflation betas and correlate well with inflation. They are generally part of industries such as oil, gold and real estate.

  - nyse_SP
    Folder for csv files for NYSE stocks that date back to at least 1970 as well as all S&P500 stocks

  - S&P500 reformatted
    Folder for csv files associated S&P500 stocks

- Images for Reference
  Folder for various images used for reference. Images include rebalanced vs.non-rebalanced portfolios, our regime-switching simulation, matlab workspace associated with generating each of the figures, sample regime-switching model with 3 regimes and historical inflation rates.

- MS_Regress_FEX_1.08
  Folder for MATLAB package created by Marcelo Perlin. It contains example files on how to create different regime switching models. The MS_Regress_MSVAR function is used where MSVAR stands for Markov Switching Vector AutoRegressive. The function returns transition probabilities as well as the mean, autoregressive term and variance associated with each regime.

- Symbol_Files
  Folder for MATLAB files that act as a ticker list. Each MATLAB file stores the data from CSV files in the "CSV_Files" folder into two matrices. One contains the names of the stocks while the other contains the names of the corresponding stock CSV files(simply stockname.csv).

  - infsymbols.m
    Ticker list corresponding to assets that are positively correlated with inflation

  - secondSymbols.m
    A small subset, of 25 stocks, used for testing

  - thirdSymbols.m
    A very small subset, of 3 stocks, used for testing. Code is shown later to display the format of the files without taking up too much space.

  - Symbols_NYSE.m
    Symbols for stocks only on the NYSE

  - Symbols_SP.m
    Symbols for stocks only on the S&P500

  - Symbols_NYSE_SP.m
    Symbols for stocks on the NYSE and the S&P 500 (Duplicates removed)

  - Symbols.m
    Identical Copy to Symbols_SP

- MainProgramFiles

  - ManualComputation_InflationBeta.xlsx
    Excel file using regression to compute inflation betas. These results were compared to that computed by the MATLAB polyfit function used in the main code to validate that the Betas were compute correctly

- RegimeSwitching_MLE.m
  Function for computing the parameters that define the regime switching. The input parameters are the number of regimes to define and the inflation data. Assuming inflation is modelled using first-order autoregression, the function uses MLE to estimate the mean, variance and autoregressive term for each regime, along with the transition probabilities of going from one regime to another.

- Fetch_Functions
  Folder for all functions that fetch data from CSV files or from within existing MATLAB vectors. Note that all data used is organized from most recent to oldest so that the row number for a particular date is consistent among all CSV files.

  * all_inflation_data.m
    Function that retrieves all the historical inflation data (used for computing inflation betas, not regime-switching model). Data is returned in the form or infmonth, infyear and infprice so that it's easy to fetch inflation data between a particular time period by looking at infmonth and infyear.

  * all_riskfree_data.m
    Similar to "all_inflation_data" but for the 3-month treasury bill prices

  * all_stock_data.m
    Similar to all_inflation_data but also returns "fails_symbols" and "success_symbols" so that stocks csv files that don't work or don't exist can be examined

  * fetch_inflation_data2.m
    Function that takes in the return values of "all_inflation_data" and a desired time period and then returns the prices associated with that time period as well as the 2 indices of the infprice vector, where the data was fetched from.

  * fetch_regime_inflation _data.m
    Function that fetches inflation data from either the monthly, quarterly or annual spreadsheets. This data is used to compute the Regime-Switching model.

  * fetch_riskfree_data2.m
    Similar to "fetch_inflation_data2" but for 3-month treasury bill prices

  * SEC_fetch_stock_data.m
    Similar to "fetch_inflation_data2" but for all the stock prices. The function also returns the stock names that have data which go back as early as the input start date. The market price is returned as a separate vector.

- Inflation_data
  - inflation_data_monthly
    monthly inflation data (used for Regime-Switching model computation)
  - inflation_data_quarterly
    quarterly inflation data (used for Regime-Switching model computation)
  - inflation_data_annually
    annual inflation data (used for Regime-Switching model computation)
  - inflation_data_1200
    Same as "inflation_data_monthly" but it is organized differently.
- MarkovTreeFunctions
  - new_exp_inf2.m
    Main function used to compute the expected inflation rate
  - new_exp_infvar2.m
    Main function used to compute the expected variance for the inflation rate. The function first iterates through every terminal node and assigns each node either the variance of regime 1 or the variance of regime 2. Then it loops through the time periods(from n to 1) and computes the conditional variance for each node(representing each path). It calls the recursive function new_infvar_nodeval to do this and ultimately returns the expected inflation variance for the root node as well as the conditional variance associated with each path.
  - new_infvar_nodeval.m
    Recursive function that computes the expected inflation variance associated with each node. The base case is for period 1, which is dependant on the starting regime. For every period, it computes the expected conditional variance using the transition probabilities.
  - terminal_inflation.m
    Function that recursively computes the terminal node values with the first-order autoregressive equation. Input parameters are the regime-switching parameters (for each regime), the number of periods and the binary paths associated with each node.
  - terminal_nodes.m
    Function that goes through every terminal node(there are 2^T of them) and assigns a

value to it that represents the unique path associated with it. For example, a node representing the path, Initial Regime → Regime 2 → Regime 2 → Regime , would have the path 110 associated with it.

- Miscellaneous_functions
  - * divide_interval.m
    Function used for dividing the desired dates(in months) into equally spaced periods. It returns a series of start and end index matrices. The first division always represents the initial in-sample data. The rest of the divisions represent the extra data used for each iteration or rebalancing.

  - * regimecount.m
    Function that goes through the smoothing probabilities associated with each time period and returns the associated regime with each date(based on the probability of being in which regime is higher). This function is particularly important when used in context with the current date because knowledge of the current regime is necessary in order to generate the correct Markov trees.

  - * return_inflation _file.m
    Function that returns the appropriate inflation file depending on whether you want to use monthly, quarterly or annual data to generate the Regime-Switching model. Monthly is good because it accounts for all fluctuations in regime price, but there are computational limitations on how far you can recurse. 2^20(about a million) is about as far as one can realistically recurse and this returns the expected inflation rate 1 year and 8 months from now. Quarterly is good because it can not only account for inflation fluctuations(though not as well as monthly), but it can also allow predictions of expected inflation up to 5 years from now. Hence, quarterly has a computational benefit. Annual does not really account for many fluctuations in the inflation rate so it is advised not to use it.

- MVO_functions
  - * benchmark_MVO.m
    Function that computes the standard MVO portfolio(adjusted for transaction costs). It returns the asset allocation and objective function that quadprog returns.

  - * main_MVO.m
    Main file where our optimization model is applied and the asset weights are computed. The function takes in historical inflation, asset returns, transaction costs, inflation betas, desired return, the previous portfolio allocation and the expected inflation rate and expected inflation variance generated from the Markov tree. The function uses the inflation betas for each asset and modifies the nominal return of each asset by adding (inflation Beta) × (the expected inflation rate). This is the

expected asset return(μ) used in the final model. The expected inflation variance is used as the conventional variance of the market term and it is used to compute the noise term as well. The function returns the asset allocation, the objective function value, an adj vector which is a combination of many variables that allows for easy comparison, nominal return vector and the covariance vector(temp_Q). The temp_Q vector is used to compute the Sharpe ratio associated with our custom model.

* MVO_comparison.m
  Function that returns the cumulative returns (essentially portfolio value over time) of five portfolios: standard MVO, custom inflation-hedged portfolio, S&P500 and the two mutual funds. It takes in the asset allocations of the first two portfolios and computes the cumulative returns. It then plots cumulative returns and regular returns on separate graphs. These graphs are used to view a comparison of the portfolio values over time.

* MVO_params.m
  Function that simply returns the geometric mean returns and the expected return for each asset as well as the covariance matrix

* solve_beta3.m
  Function which takes the asset prices and inflation rate values as input parameters. It computes the corresponding asset returns and then uses MATLAB's polyfit function to compute the inflation Beta's associated with each asset over the time period of data inserted. The function also returns an R-squared value which represents how "accurate" the value of Beta is for each asset. This value is not really considered because it's not expected to be very good since the link between the inflation rate and the stock price of an asset is not very strong.

* solve_mvo_params.m
  Identical to MVO_params.m

- SharpeRatioCalculation
  * calculateSharpeRatio.m
    Function that computes the sharpe ratio and modified sharpe ratio associated for our model and the standard MVO. Input parameters are the expected return and covariance of each asset, the current period's risk free rate as well as the asset allocation and computed objective function(temp_Q) for our model and for standard MVO.

  * sharperatio2.m
    Function that reforms the actual Sharpe ratio equation and is called from "calculateSharpeRatio"

- Python Files
  Python is used for data scraping. You need to feed in a .csv file with ticker samples. The file name must be changed in both files in order for the scraping to work.
  - scrape.py
    File that gets the data from yahoo finance in the format of month, date, year & price
  - delete.py
    File that removes duplicates and ensures there is only 1 date for each month

19.3 MATLAB Code for Supporting Files

- a.m

```
% INFLATION_INTERVAL is 'monthly','quarterly' or 'annually'
% and is used to computer the regime history

% ------------------------------------------------------------------------

% SYMBOLFILE specifies which file to fetch symbols from. A list of the files
% available are given below

% -Symbols_SP.m has the full S&P500 Stocks. It is important to note that only some
% of these have aviable price history that dates back to 1974.
% -SecondSymbols has a small subset and is used for testing
% -ThirdSYmbols is an even smaller subset, having only 3 stocks.
% -Symbols_NYSE has a full list of all NYSE stocks. However at this time,
%  only a subset of these 3300+ stocks have local csv files avaiable for fetching
% -Symbols_NYSE2 has a list of the NYSE stocks that have local csv files
% -Symbols_NYSE_SP has a list of NYSE stocks AND S&P500 stocks. ALl of these
% are avaiable for fetching. It is the recommended file to use because it has
% the most data avaiable which is important for early years where most stocks
% dont have time series data
```

```
19 %
   ------------------------------------------------------------------------

20 % SYEAR/EYEAR is the start/end year of optimization for the regeims
21 %
   ------------------------------------------------------------------------

22 % NUM_OF_TIME_PERIODS is the number of divisions in the time
       interval
23 % note that the in sample period to get data is
24 % (syear-eyear)/num_of_time_periods.
25
26 % **ensure that syear-eyear is divisble by num_of_time_periods
27 %
   ------------------------------------------------------------------------

28 % NUM_OF_REOP is the number of times the portfolio is reoptimized.
29 % note that this is a specified amount and there is no submodel to
       determine
30 % whether or not to reoptimize because the transcation cost
       constraint
31 % prevents drastic changes in asset allocation
32 %
   ------------------------------------------------------------------------

33 % DESIRED_R IS THE minimum monthly return constraint. Converting
       this
34 % to an annual return is (1+desired_R)^12-1
35 %
   ------------------------------------------------------------------------

36 % DESIRED_TRANSACTION is the the user-specified fixed transaction
       cost
37
38
39 inflation_interval='monthly';
40 SymbolFile=''Symbols_NYSE_SP.m'';
41 startyear=1981;
42 endyear=1991;
43 num_of_time_divisions=5;
44 num_of_reoptimization_periods=2;
45 minimum_return=0.005;
46 transaction_cost=0.0001;
47
48
```

```
49  create_inflation_hedged_portfolio...
50          (inflation_interval,SymbolFile,startyear,endyear,
               num_of_time_divisions,...
51             num_of_reoptimization_periods, minimum_return,
                  transaction_cost);
52
53 %Commment all the lines above and Uncomment the line below if you
       want to
54 % see all the workspace variables
55
56 % run create_inflation_hedged_portfolio.m;
```

- create_inflation_hedged_portfolio.m

```
1
2
3
4
5  function [] = create_inflation_hedged_portfolio...
6                  (inflation_interval,SymbolFile,...
7                   syear,eyear,num_of_time_periods,num_of_reop, ...
8                   desired_R,desired_transaction)
9
10 tic
11
12 % Uncomment the lines below when making
       create_inflation_hedged_portfolio a non-function
13 % inflation_interval='monthly';
14 % SymbolFile='''Symbols_NYSE_SP.m''';
15 % syear=2002;
16 % eyear=2012;
17 % num_of_time_periods=5;
18 % num_of_reop=1;
19 % desired_R=0.005;
20 % desired_transaction=0.0001;
21
22
23
24
25 %
       ------------------------------------------------------------------------

26 % STEP 1: RETRIEVING HISTORICAL INFLATION RATES & ALL STOCK DATA
       AVAILABLE
27 % NOTE: THIS CAN BE PREPROCESSED
28
```

```
%can be monthly, quarterly or annual
begcol=1;
[inf_file endcol] = return_inflation_file(inflation_interval);

diffyear=eyear-syear;
eachyear=diffyear/num_of_time_periods;
T_reb= (diffyear-eachyear)*12/(num_of_reop+1);

[MLEinf_data, inf_avg] =...
    fetch_regime_inflation_data(1914,syear,begcol,endcol,inf_file);

%THis gets a list of the Stock ticker symbols used as the pool for
% asset allocation along with their corresponding csv files
run(eval(SymbolFile));

%THis fetches a list of ALL the stock/inflation/riskfree data for
    all dates
% Note that this is used for the optimization model and NOT for
    creating
% the regime switching model

[month day year price fail_symbols success_symbols]=...
                                                    all_stock_data(
                                                       SP500_symb_csv,
                                                       SP500_symb);

[infmonth infyear infprice ] = all_inflation_data('
    inflation_rate_1200.csv');
 [rfmonth rfyear rfprice] = all_riskfree_data('riskfreerate2.csv');
%
    ------------------------------------------------------------------------


% STEP 2: SOLVE FOR THE MLE PARAMETERS OF THE REGIME-SWITCHING
    MODEL
% AS WELL AS DETERMING THE REGIME THAT EACH PERIOD BELONGS TO
% (WHICH PRODUCES THE REGIME STATE GRAPH)

k=2; %DECLARE THE NUMBER OF REGIMES

[Spec_Out p11 p22 p12 p21 var1 var2 var3 ar1 ar2 ar3 c1 c2 c3]= ...
                                        RegimeSwitching_MLE(k,MLEinf_data
                                            );

timelength=length(MLEinf_data);

```

```
65 [whichregime, countregime] =regimecount(k,Spec_Out.smoothProb,
       timelength);
66
67 %Parameter 1 is the number of monthly time periods, 2 is the
       current inflation rate
68 curr_regime=whichregime(timelength);
69 % curr_regime=2;
70 curr_inf_rate=MLEinf_data(timelength);
71 disp('CHECKPOINT 1');
72
73 %
       ------------------------------------------------------------------------

74 % TEMP CODE FOR TESTING NEW INFLATION EXPECTED VARIANCE
75
76 markov_periods=10; %this is chosen number of periods for markov
       tree(in months)
77
78 [expected_inf, tnodes,tnodeval]=new_exp_inf2(markov_periods,
       curr_inf_rate,curr_regime...
79                         ,c1,c2,ar1,ar2,p11,p12,p21,p22);
80 [expected_inf_var] = ...
81      new_exp_infvar2(markov_periods,1,tnodeval,var1,var2,p11,p12,p21
           ,p22);
82
83 %
       ------------------------------------------------------------------------

84 % STEP 4 : RETRIEVE THE ASSET AND MARKET PRICES FOR THE DESIRED
       TIME PERIODS
85
86 lcase_month='Jan'
87 ucase_month=upper(lcase_month);
88
89 [tcurrprices currpricenames2 market_prices num_assets2
       catch_assets2 totalmonths] = ...
90          SEC_fetch_stock_data...
91          (lcase_month,eyear,lcase_month,syear,month,day,year,price,
               success_symbols);
92
93 %Use the fetch inflation data to compute the inflation rates and
       the riskfree rates
94 [inf_prices] = fetch_inflation_data2...
95      (ucase_month,eyear,ucase_month,syear,infmonth,infyear,infprice)
           ;
```

```
96
97  [riskfree_prices] = fetch_inflation_data2...
98      (ucase_month,eyear,ucase_month,syear,infmonth,infyear,rfprice);
99
100 disp('CHECKPOINT 3');
101
102
103 [beg_indices end_indices]=divide_interval(num_of_time_periods,
        totalmonths);
104
105 asset_prices=cell2mat(tcurrprices);
106
107 %Parameters are first number of time divisions, then total months
108
109
110 currassetprices=asset_prices(beg_indices(1):end_indices(1)+1,:);
111 currmarketprices=market_prices(beg_indices(1):end_indices(1)+1);
112 currinfprices=inf_prices(beg_indices(1):end_indices(1)+1);
113 currriskfreeprices=riskfree_prices(beg_indices(1):end_indices(1)+1)
        ;
114
115
116
117 % asset_prices=cell2mat(currprices2);
118 asset_prices_with_market=[currmarketprices currassetprices];
119 asset_prices_with_inf=[currinfprices currassetprices];
120
121 %
        ------------------------------------------------------------------------

122 % STEP 5: SOLVE FOR THE MVO PARAMETERS, THE MUS, Q'S AND R'S FOR
        ALL ASSETS AND
123 % THE MARKET OVER THE SPECIFIED TIME PERIODS. ALSO SOLVE FOR THE
        CAPM BETAS OR
124 % EACH ASSET
125
126 [asset_mu,asset_Q,asset_r]= solve_mvo_params(currassetprices,1,size
        (currassetprices,1));
127
128 %Solve for only the market
129 [Market_mu,Market_Q,Market_r]= solve_mvo_params(currmarketprices,1,
        size(currmarketprices,1));
130
131
132 % Calculate relevant parameters for testing horizon
```

```
133
134
135
136 disp('CHECKPOINT 4');
137
138 % desired_R=0.002;
139 % desired_transaction=0.0001
140
141 [Inf_Beta R2_inf]    =solve_beta3(asset_prices_with_inf,1);
142 i_Inf_Beta=Inf_Beta;
143
144
145 currinfprices2=currinfprices(1:end-1);
146 xalloc=zeros(num_assets2,1);
147
148 [modelMVO_x(1,:) modelMVO_var MVO_adjret_diagQ nom_ret temp_Q] =
        ...
149     main_MVO(currinfprices2,asset_r,expected_inf/100,
            expected_inf_var,...
150                                     Inf_Beta',desired_R,
                                           desired_transaction,xalloc);
151
152 i_MVO_adjret_diagQ_InfBeta=MVO_adjret_diagQ;
153 i_nom_ret=nom_ret;
154 i_temp_Q=temp_Q;
155
156 model_portfolio_beta(1,1) = modelMVO_x(1,:)*Inf_Beta';
157 % [modelMVO_x modelMVO_var temp_Q adj_ret nom_ret] = main_MVO(
        currinfprices,asset_r,1.8/100,0.04,...
158 %                                                       Inf_Beta
        ',0.000002,0.0005,0.05,xalloc);
159
160 %
    ------------------------------------------------------------------------------

161 futureassetprices=asset_prices(beg_indices(2):end_indices(
        num_of_time_periods),:);
162 futuremarketprices=market_prices(beg_indices(2):end_indices(
        num_of_time_periods));
163 futureinfprices=inf_prices(beg_indices(2):end_indices(
        num_of_time_periods));
164 futureriskfreeprices=riskfree_prices(beg_indices(2):end_indices(
        num_of_time_periods));
165
166 [future_asset_mu,future_asset_Q,future_asset_r]= ...
```

```
167      solve_mvo_params(futureassetprices,1,size(futureassetprices,1))
             ;
168
169  %Solve for only the market
170  [future_Market_mu,future_Market_Q,future_Market_r]=
         solve_mvo_params(futuremarketprices,1,size(futuremarketprices,1)
         );
171
172
173
174
175  [future_MK1_r] = future_asset_r(:,end);
176  [future_MK2_r] = future_asset_r(:,end-1);
177
178  [benchMVO_x(1,:) benchMVO] = ...
179  benchmark_MVO(asset_mu', asset_Q, desired_R,desired_transaction,
         xalloc);
180
181  MVO_portfolio_beta(1,1) = benchMVO_x(1,:)*Inf_Beta';
182
183  [cumul_benchMVO cumul_modelMVO cumul_SP cumul_MF1 cumul_MF2] = ...
184      MVO_comparison(benchMVO_x(1,:)', modelMVO_x(1,:)',
             future_asset_r,future_Market_r,...
185           future_MK1_r,future_MK2_r,syear+eachyear,eyear);
186
187  obj_vals = [modelMVO_var benchMVO_x(1,:)*temp_Q*benchMVO_x(1,:)'
188               modelMVO_x(1,:)*asset_Q*modelMVO_x(1,:)' benchMVO];
189
190
191  %
         ------------------------------------------------------------------------

192  %Using our model version of the Sharpe ratio, Preface M
193
194  [P_SRATIOS(:,1) S_SRATIOS(:,1)]=calculateSharpeRatio(asset_mu,
         asset_Q, temp_Q, ...
195       modelMVO_x(1,:), benchMVO_x(1,:),modelMVO_var,
             currriskfreeprices);
196
197
198  if (num_of_reop>0)
199
200      MVO_returns_reb(1:T_reb)= ...
201           future_asset_r(1:T_reb,:)*benchMVO_x(1,:)';
202      inf_returns_reb(1:T_reb) = ...
```

```
203         future_asset_r(1:T_reb,:)*modelMVO_x(1,:)';
204
205
206
207       for p = 1:num_of_reop
208
209     %   RECALCULATE RELEVANT PARAMETERS AND REOPTIMIZE
210
211         markov_periods =  10;
212
213         beg_indices(1) = beg_indices(1) + T_reb;
214         end_indices(1) = end_indices(1) + T_reb;
215
216
217
218         currassetprices=asset_prices(beg_indices(1):end_indices(1)
                ,:);
219         currmarketprices=market_prices(beg_indices(1):end_indices
                (1));
220         currinfprices=inf_prices(beg_indices(1):end_indices(1));
221         currriskfreeprices=riskfree_prices(beg_indices(1):
                end_indices(1)+1);
222
223
224         [expected_inf, tnodes,tnodeval]=...
225             new_exp_inf2(markov_periods,futureinfprices(T_reb*p+1)
                    ,...
226                 curr_regime,c1,c2,ar1,ar2,p11,p12,p21,p22);
227
228         [expected_inf_var] = ...
229             new_exp_infvar2(markov_periods,curr_regime,tnodeval,
                    ...
230                             var1,var2,p11,p12,p21,p22);
231
232         asset_prices_with_inf=[currinfprices currassetprices];
233
234         [Inf_Beta R2_inf]    =solve_beta3(asset_prices_with_inf,1);
235
236         currinfprices2=currinfprices(1:end-1);
237
238         [asset_mu,asset_Q,asset_r]= solve_mvo_params(
                currassetprices,1,size(currassetprices,1));
239
240         [modelMVO_x(p+1,:) modelMVO_var MVO_adjret_diagQ nom_ret] =
                ...
```

```
241             main_MVO(currinfprices2,asset_r,expected_inf/100,
                    expected_inf_var,...
242                         Inf_Beta',desired_R,desired_transaction,
                                modelMVO_x(p,:)');
243
244         model_portfolio_beta(p+1,1) = modelMVO_x(p+1,:)*Inf_Beta';
245
246         [benchMVO_x(p+1,:) benchMVO] = ...
247          benchmark_MVO(asset_mu', asset_Q, desired_R,
                desired_transaction, benchMVO_x(p,:)');
248         MVO_portfolio_beta(p+1,1) = benchMVO_x(p+1,:)*Inf_Beta';
249
250         MVO_returns_reb(T_reb*p+1:min((p+1)*T_reb,size((
                future_asset_r),1)))= ...
251             future_asset_r((T_reb*p+1):...
252                     min((p+1)*T_reb,size((future_asset_r),1)),:)...
253                 *benchMVO_x(p+1,:)';
254
255         inf_returns_reb(T_reb*p+1:min((p+1)*T_reb,size((
                future_asset_r),1))) = ...
256             future_asset_r(T_reb*p+1:min((p+1)*T_reb,size((
                    future_asset_r),1)),:)...
257              *modelMVO_x(p+1,:)';
258
259         [P_SRATIOS(:,p+1) S_SRATIOS(:,p+1)]=...
260             calculateSharpeRatio(asset_mu, asset_Q, temp_Q, ...
261                 modelMVO_x(p+1,:), benchMVO_x(p+1,:), modelMVO_var,
                        currriskfreeprices);
262
263     end
264
265       cumul_MVO_reb(1) = MVO_returns_reb(1);
266       cumul_inf_reb(1) = inf_returns_reb(1);
267
268
269
270      T = length(MVO_returns_reb);
271
272       for i = 2:T
273           cumul_MVO_reb(i) = (1+cumul_MVO_reb(i-1))*(1+
                MVO_returns_reb(i)) - 1;
274           cumul_inf_reb(i) = (1+cumul_inf_reb(i-1))*(1+
                inf_returns_reb(i)) - 1;
275       end
276
```

```
277             figure
278             plot(1:T,cumul_MVO_reb*100, '-b');
279             hold all
280             plot(1:T,cumul_inf_reb*100, '-r');
281             hold all
282             plot(1:length(cumul_SP),cumul_SP*100, '-g');
283             hold all
284             plot(1:T,cumul_MF1*100, '-m');
285             hold all
286             plot(1:T,cumul_MF2*100, '-c');
287
288         h = {'Standard MVO', 'Inflation Hedged SF', 'S&P500',...
289                 'Vanguard Wellington Inv','CGM Mutual Fund'};
290             h = legend(h);
291
292             grid on;
293
294             title(['Comparing Cumulative Returns of Optimal Rebalanced
                    Portfolios and Market'...
295                     ,num2str(syear+eachyear),' to ',num2str(eyear)]);
296
297             xlabel('Time (in months)')
298             ylabel('Cumlative Returns in %')
299 end
300 portfolio_betas = [model_portfolio_beta MVO_portfolio_beta];
301 clear model_portfolio_beta
302 clear MVO_portfolio_beta
303 toc
304 %
```

```
305 % STEP 10. AT THE END OF THE LOOP, PLOT THE PORTFOLIO'S PERFORMANCE
        OVER TIME AND COMPARE IT TO
306 %    HOW THE STANDARD S&P500 INDEX DID (PLOT BOTH ON SAME GRAPH)
307 rmpath('m_Files');
308 rmpath('data_Files');
309
310 %Comment/uncomment the end below if you want to see/not see all the
        workspace variables
311 end
```

- thirdSymbols.m

```
1 SP500_symb= {'AA'
2 'ABC'
3 'ABT'
```

```
'LOMMX'
'VWELX'
'^GSPC'
};
SP500_symb=SP500_symb';

SP500_symb_csv={'AA.csv'
    'ABC.csv'
    'ABT.csv'
    'LOMMX.csv'
        'VWELX.csv'
        '^GSPC.csv'
 };
SP500_symb_csv=SP500_symb_csv';
```

- RegimeSwitching_MLE.m

```
function [Spec_Out p11 p22 p12 p21 var1 var2 var3 ar1 ar2 ar3 c1 c2
    c3] = ...
                 RegimeSwitching_MLE(k,inf_data)

                disp('Computing Parameters of Regime Switching
                   Model');
                addpath('../../MS_Regress_FEX_1.08/m_Files'); % add
                    'm_Files' folder to the search path
                addpath('../../MS_Regress_FEX_1.08/data_Files');

                logRet=inf_data;  % load some Data.

                % Defining dependent variables in system (solving
                   for two dependent variables)
                dep=logRet(:,1);
                nLag=1;                                 % Number of
                   lags in system
                k=k;
                                    % Number of States
                doIntercept=1;                          % add intercept
                    to eqsuations
                %%
                advOpt.diagCovMat=0
                          %if this value is 1, then only the
                   elements on the diagonal(the variances) are
                   estimated
                %%
                advOpt.distrib='Normal';                % The
                   Distribution assumption (only 'Normal' for MS
```

```
   VAR models) )this is the default avlue
advOpt.std_method=1;                % Defining the
   method for calculation of standard errors. See
   pdf file for more details

Spec_Out=MS_VAR_Fit(dep,nLag,k,doIntercept,advOpt);
disp('COMPLETED MS_VAR_FIT function');

% make the transition probabilites easier to call
% where pij is probability of going to regime j
   from regime i
p11=Spec_Out.Coeff.p(1,1);
p22=Spec_Out.Coeff.p(2,2);
p12=Spec_Out.Coeff.p(2,1);
p21=Spec_Out.Coeff.p(1,2);

% The variances associated with each regime
var1=cell2mat(Spec_Out.Coeff.covMat(1,1));
var2=cell2mat(Spec_Out.Coeff.covMat(1,2));

%autoregressive terms for each regime
%Have to use two lines to properly index values
ar1=Spec_Out.Coeff.S_Param(1,1);
ar1=ar1{1}(2,1);

ar2=Spec_Out.Coeff.S_Param(1,1);
ar2=ar2{1}(2,2);

%intercepts for each regime (These are the mean
   inflation rates for the regime)
c1=Spec_Out.Coeff.S_Param(1,1);
c1=c1{1}(1,1);

c2=Spec_Out.Coeff.S_Param(1,1);
c2=c2{1}(1,2);

%Different return values depending on whether there
    are 2 or 3 regimes
if (k==2)
        t=num2cell(zeros(1,8));
        [p13,p23,p33,p32,p31,var3,ar3,c3]=deal(t
            {:});

elseif (k==3)
        p13=1-p11-p12;
```

```
                        p23=1-p22-p21;
                        p33=Spec_Out.Coeff.p(3,3);
                        p32=Spec_Out.Coeff.p(2,3);
                        p31=Spec_Out.Coeff.p(1,3);
                        var3=cell2mat(Spec_Out.Coeff.covMat(1,3));
                        ar3=Spec_Out.Coeff.S_Param(1,1);
                        ar3=ar2{1}(2,3);
                        c3=Spec_Out.Coeff.S_Param(1,1);
                        c3=c3{1}(1,3);
                end

end
```

- all_inflation_data.m

```
% This function returns the inflation rates used to compute
    inflation betas

function [infmonth infyear infprice] = all_inflation_data(filename)

        [infmonth infyear infprice] = textread(filename, '%s %d %f'
            ,'delimiter',',');

end
```

- all_riskfree_data.m

```
function [rfmonth rfyear rfprice] = all_riskfree_data(filename)

        %[infmonth, infyear, infprice] = deal(cell(1,1));


        [rfmonth rfyear rfprice] = ...
                                        textread(filename, '%s %d %
                                            f','delimiter',',');

end
```

- all_stock_data.m

```
function [month day year price fail_symbols success_symbols] = ...
        all_stock_data(SP500_symb_csv,SP500_symb)




        [month,day,year,price,fail_symbols,success_symbols]=deal(
            cell(1,1));
```

```
8
9           fc=0;
10          sc=1;
11          for i=1:length(SP500_symb_csv)
12                  try
13                          curr_csv=char(SP500_symb_csv(i));
14                          curr_symb=char(SP500_symb(i));
15                          success_symbols{sc}=curr_symb;
16
17                          [month{sc} day{sc} year{sc} price{sc}] =
                                ...
18                           textread(curr_csv ,'%s %d %d %f','
                                delimiter',',: ');
19
20                          sc=sc+1;
21
22
23                  catch
24                          curr_symb=char(SP500_symb(i));
25                          fc=fc+1;
26                          fail_symbols{fc}=curr_symb;
27                  end
28          end
29          %BY THE END OF THIS LOOP WE HAVE DAILY TIME SERIES DATA
30          %NOW ITERATE THROUGH MONTH VECTOR AND DELETE EVERY ELEMENT
31          % WHICH HAS THE SAME MONTH
32
33          % price_names=[];
34
35          % for (i=1:length(success_symbols))
36          %       price_names=[price_names; success_symbols(i)];
37          % end
38
39          % success_symbols=price_names;
40
41
42
43
44
45
46
47  end
```

- fetch_inflation_data2.m

```
1  %NOTE FORMAT OF INPUT MONTH IS SUPPOSED TO
```

```
function [inftimeprices beg_ind end_ind] = fetch_inflation_data2(
    smonth,syear,emonth,eyear,...
```

```
        x=strmatch(smonth,infmonth);

        for i=1:length(x)
                if infyear(x(i))==syear
                        beg_ind=x(i);
                end
        end
        x=strmatch(emonth,infmonth);
        for i=1:length(x)
                if infyear(x(i))==eyear
                        end_ind=x(i);
                end
        end

        inftimeprices=flipud(data(beg_ind:end_ind));


end
```

- fetch_regime_inflation_data.m

```
%returns inflation data used to compute the RS model. It may be
    either
%monthly, quarterly or annually depending on the filename passed
function [inf_data inf_data_avg] =  ...
                fetch_regime_inflation_data(s_year,e_year,s_month,
                    e_month,filename)

        s_year=s_year-1914;
        e_year=e_year-1914;
        inf_data=csvread(filename,s_year,s_month,[s_year,s_month,
            e_year,e_month]);

        inf_data=inf_data';
        inf_data=inf_data(:);

```

```
13          %inf_data_avg=csvread(filename,s_year,13,[s_year,13,e_year
                ,13]);
14          inf_data_avg=csvread...
15          (filename,s_year,e_month+1,[s_year,e_month+1,e_year,e_month
                +1]);
16
17
18  end
```

- fetch_riskfree_data2.m

```
1  %NOTE FORMAT OF INPUT MONTH IS SUPPOSED TO
2  function [riskfreeprices] = fetch_riskfree_data2(smonth,syear,
       emonth,eyear,...
3
4
5           x=strmatch(smonth,infmonth);
6
7           for i=1:length(x)
8                   if infyear(x(i))==syear
9                           beg_ind=x(i);
10                  end
11          end
12          x=strmatch(emonth,infmonth);
13          for i=1:length(x)
14                  if infyear(x(i))==eyear
15                          end_ind=x(i);
16                  end
17          end
18
19          inftimeprices=flipud(data(beg_ind:end_ind));
20
21
22  end
```

- SEC_fetch_stock_data.m

```
1  %INPUT PARAMETERS
2  % month, day, year, price represent all the time series data
3  % succes_symbols represents the assets that succesfully
4  % FOR NOW ASSUME ROW 3000 IS THE SAME DATE FOR EVERY ASSET
```

```
5  function [prices price_names2 marketprice num_assets catch_assets
       ...
6                   interval_size] = ...
7            SEC_fetch_stock_data(smonth,syear,emonth,eyear,month,day,
                 year,data,success_symbols)
8
9
10 %------CODE TO FIND THE STARTING AND ENDING INDICES ----------
11          x=strmatch(smonth,month{1});
12          for i =1:length(x)
13                  if year{1}(x(i))==syear
14                          beg_ind=x(i);
15                  end
16          end
17
18          x=strmatch(emonth,month{1});
19          % disp(emonth);
20          % disp(month{1});
21          % disp(x);
22          % disp(eyear);
23          % disp(year);
24          disp(eyear);
25          for i =1:length(x)
26                  disp(year{1}(x(i)));
27                  if year{1}(x(i))==eyear
28                          end_ind=x(i);
29                  end
30          end
31          interval_size=end_ind-beg_ind;
32 %--------------AFTER FINDING THE INDICES ITERATE THROUGH ASSETS TO
       GET PRICE DATA -----
33          count=1;
34          %beg_ind=beg_ind-1;
35          %end_ind=end_ind-1;
36          catch_assets=1;
37          %The last element
38
39          for i = 1:(length(success_symbols)-1)
40                  %symb_matrix{i}=SP500_symb_csv{i};
41                  try
42                          % disp('entered try block')
43                          prices{count}=flipud(data{i}(beg_ind:
                                end_ind));
44                          price_names{count}=success_symbols(i);
45                          count=count+1;
```

```
                        %disp(SP500_symb_csv(i))
                catch
                        catch_assets=catch_assets+1;
                end
        end
        num_assets=count-1; % add 1 for the market index

        %retrieve time series price information for the marketprice
        try
                marketprice=data{end}(beg_ind:end_ind);
                marketname=success_symbols(end);
        catch
        end

        marketprice=flipud(marketprice);


        price_names2=[];

        for (i=1:length(price_names))
                price_names2=[price_names2; price_names{i}];
                % disp(price_names2)
        end



end
```

- new_exp_inf2.m

```
% This function generates the markov tree and comptues the expected
    inflation raet

%curr_regime is 0 or 1
function [inf_rate,tnodes,tnodeval3] = new_exp_inf2(n,y0,
    curr_regime,c1,c2,ar1,ar2,p11,p12,p21,p22)

        % Compute an associated binary value for each termianl node
            which represents
        % the unique pathe taken to get to that node
        tnodes=terminal_nodes(n);

        % compute that actual terminal inflation values conditional
            on the path taken
        tnodeval=terminal_inflation(n,tnodes,c1,c2,ar1,ar2,y0);
        tnodeval2{n}=tnodeval;
```

```
13          maxelements=2^n;
14
15          % iterate over every time period and compute the values for
                each node
16          for i=n:-1:1
17                  tnodes2=terminal_nodes(i);
18                  a=terminal_inflation(i,tnodes2,c1,c2,ar1,ar2,y0);
19                  % disp(i);
20                  % disp(a);
21                  tnodeval3(i,:)=[a zeros(1,maxelements-2^i)];
22          end
23
24
25          % transition probability matrix
26          p= [p11 p12
27              p21 p22];
28
29          inf_rate=0;
30          for i=1:2^n
31                  currprod=tnodeval(i);
32                  cnode=tnodes{i};
33                  %for the n-1 transition probabilities
34                  for j=n:-1:2
35                          ind1=str2num(cnode(j))+1;
36                          ind2=str2num(cnode(j-1))+1;
37                          currprod=currprod*p(ind2,ind1);
38                  end
39                  %for the initial transition probability
40                  ind3=str2num(cnode(1))+1;
41                  currprod=currprod*p(curr_regime,ind3);
42
43                  inf_rate=inf_rate+currprod;
44
45          end
46
47
48  end
```

- new_exp_infvar2.m

```
1  % THis function is used to compute the expected variance of the
       inflation rate
2  function [inf_var inf_var_node_val ] =...
3           new_exp_infvar2(n,start_regime,tnodeval,var1,var2,p11,p12,
                p21,p22)
4
```

```
5
6
7           p= [p11 p12
8                   p21 p22];
9           % For every terminal node, assign either variance of regime
                1 or variance of regime 2
10          % as its value
11          for i=1:2^n
12                  if (mod(i,2)==0)
13                          initialvarval(i)=var1;
14                  elseif (mod(i,2)==1)
15                          initialvarval(i)=var2;
16                  end
17          end
18
19
20          infvar_node_val=initialvarval;
21
22          % Iterate over every time period and call the recursive
                function new_infvar_nodeval
23          % until you reach the initial time period
24          for i=n:-1:1
25                  infvar_node_val=...
26                          new_infvar_nodeval(i,p,infvar_node_val,
                               tnodeval(i,:),start_regime);
27          end
28          inf_var=infvar_node_val(1);
29
30  end
```

- new_infvar_nodeval.m

```
1  % This is the recursive function that commputes the expected
       inflation variance
2  % associated with each node
3  function [nodeval] = new_infvar_nodeval(n,p,varval,muinfval,
       start_regime)
4
5           n=n-1;
6
7           % Determines the two hcild nodes of a node and uses those
                values to get
8           % the variances used in the expectation formula
9           for i=1:2^n
10                          var2(i)=varval(2*i-1);
11                          var1(i)=varval(2*i);
```

```
12                          mudiff(i)=(muinfval(2*i-1)-muinfval(2*i))
                                ^2;
13          end
14
15          % This is the base case and you use either p(1,1) and p
                (1,2) or
16          % p(2,1) and p(2,2) depending on the starting regime
17          if n==0
18                  if (start_regime==1)
19                          prob=p(1,1);
20
21
22                  elseif (start_regime==2)
23                          prob=p(2,1);
24                  end
25                  nodeval(i)=prob*var1(i)+(1-prob)*var2(i)+prob*(1-
                        prob)*mudiff(i);
26
27          else
28                  % This is the recursive case but the node value can
                        still be computed
29                  % at that point.
30                  for i=1:2^n
31                          if mod(i,2) == 0
32                                  prob=p(1,1);
33                          else
34                                  prob=p(2,1);
35                          end
36                          nodeval(i)=prob*var1(i)+(1-prob)*var2(i)+
                                prob*(1-prob)*mudiff(i);
37                  end
38                  % end
39          end
40  end
```

- terminal_inflation.m

```
1  % This function applies the closed form equation to compute the
       terminal
2  % value associated with each node in the markov tree for expected
       inflation
3  % This equation is simply the result of recursion(due to the
       autoregressive term)
4  % and is unique for each node.
5  function [tnodeval] = terminal_inflation(n,tnodes,c1,c2,ar1,ar2,y0)
6
```

```
7           %{
8           a0=c1;
9           a1=c2;
10          b0=ar1;
11          b1=ar2;
12      beta1 and beta2 represent the betas for regime 1 and 2
13          %}
14          a=[c1 c2];
15          b=[ar1 ar2];
16
17          %iterates over each terminal node
18          tnodeval = [];
19          for i=1:2^n
20                  cnode=tnodes{i};
21                  cnodeval = 0;
22                  %For a given terminal node, iterates over each time
                        step
23                  for j=n:-1:1
24                          %period=n-j+1;
25                           if (j==n)
26                          %       disp('Value of a() is');
27                          %       a(str2num(cnode(n))+1)
28                                  cnodeval = cnodeval + a(str2num(
                                        cnode(n))+1);
29                          else
30                                  ind_j=str2num(cnode(j))+1;
31                                  currprod=a(ind_j);
32
33                                 % over number of terms for a given
                                        time step
34                                  for k = j+1:n;
35                                          % disp('Value of a() is');
36                                          % a(str2num(cnode(n))+1)
37                                          ind_k=str2num(cnode(k))+1;
38                                          currprod = currprod*b(ind_k
                                                );
39                                  end
40                                  cnodeval=cnodeval+currprod;
41                          end
42                          if (j==1)
43                                  currprod=1;
44                                  for k= 1:n
45                                          ind_k=str2num(cnode(k))+1;
46                                          currprod=currprod*b(ind_k);
47                                  end
```

```
                                        cnodeval=cnodeval+currprod*y0;
                                end
                end
                tnodeval(i)=cnodeval;
        end

end
```

- terminal_nodes.m

```
%This function returns a vector of all the terminal nodes
function [S_bin] = terminal_nodes(n)
        for i=0:2^n-1
                S_bin{i+1}=dec2bin(i,n);
        end
end
```

- divide_interval.m

```
% This function takes in the total months and total intervals
% and returns matrices of equally spaced interval(except the last one
% may be slightly longer in the case of numintervals not being a factor
% of total months.
function [beg_ind end_ind] = divide_interval(numintervals, totalmonths)

        beg_ind=[];
        end_ind=[];
        indice_size=floor(totalmonths/numintervals);

        for i=1:numintervals-1
                beg_ind=[beg_ind; indice_size*(i-1)+1];
                end_ind=[end_ind; indice_size*i];
        end
        %if mod(totalmonths,numintervals)>0

                beg_ind=[beg_ind; (indice_size*(numintervals-1)+1)];
                end_ind=[end_ind; (indice_size*numintervals+mod(totalmonths,numintervals))];
        % end


end
```

- regimecount.m

```
%input parameters are # of states, smoothingprobabilities,and no. of time periods
%THe function returns a matrix of the resulting regime for each period
%and counts the total type of each regime for the historical inflation data

%The regime associated with each time is ultimately the greater of the two
% probabilities of that time period belonging to that regime
function [whichregime, countregime] = ...
                regimecount(k,smoothProb,timelength)



        %NOte that the size of Spec_Out.filtProb is (# of time periods) x (# of states)
        %IMPORTANT: NOTE THAT IT Is likely to change .filtProb to .smoothProb
        for i=1:length(smoothProb(1,:))
                r_prob(:,i)=smoothProb(:,i);
        end


        %initializes countregime matrix of size k(no. of states)
        countregime=zeros(1,k);

        for i=1:timelength
                if (k==3)
                        if (r_prob(i,1)>=r_prob(i,2) && r_prob(i,1)>=r_prob(i,3))
                                whichregime(i)=1;
                                countregime(1)=countregime(1)+1;
                        elseif (r_prob(i,2)>=r_prob(i,1) && r_prob(i,2)>=r_prob(i,3))
                                whichregime(i)=2;
                                countregime(2)=countregime(2)+1;
                        else
                                whichregime(i)=3;
                                countregime(3)=countregime(3)+1;
                        end
                elseif (k==2)
                        if (r_prob(i,1)>=r_prob(i,2))
                                whichregime(i)=1;
```

```
                countregime(1)=countregime(1)+1;
            else
                whichregime(i)=2;
                countregime(2)=countregime(2)+1;
            end
        end
    end

end
```

- return_inflation_file.m

```
% This function returns the apprtopriate inflation fil
% for monthly, quarterly or annual data

function [inf_file endcol] = return_inflation_file(
    inflation_interval)
    if (strcmp(inflation_interval,'monthly'))
        inf_file='inflation_data_monthly.csv';
        endcol=12;
    elseif (strcmp(inflation_interval,'quarterly'))
        inf_file='inflation_data_quarterly.csv';
        endcol=4;
    elseif (strcmp(inflation_interval,'annually'))
        inf_file='inflation_data_annual.csv';
        endcol=1;
    end
end
```

- benchmark_MVO.m

```
%% STANDARD MARKOWITZ MVO MODEL VIA QUADPROG
% Input the number of assets, the range of desired returns, the
    mean vector
% for the assets, and the covariance matrix. The function will
    return the
% optimal portfolio weights in MVO_x, and the corresponding
    variance
% function value in MVO_var.
%
% [optimal weights, corresponding optimal objective function]
%   = sMVO(# of assets, range of desired returns, expected return
    vector,
%           covariance matrix)

% Set quadprog parameters
```

```
function [ MVO_x MVO_var] = benchmark_MVO( mu, Q, return_range,
    transaction_cost, previous_portfolio)


        n= length(mu);

        c = [zeros(3*n,1);];
        Aeq = [ones(1,n) zeros(1,2*n)
                    eye(n) -eye(n) eye(n)];
        beq = [1; previous_portfolio];
        A=[-mu' transaction_cost*ones(1,2*n)];
        lb=zeros(3*n,1);
        ub=9999*ones(3*n,1);
        R = return_range;

        tempQ=Q;
        Q = [Q zeros(n,2*n); zeros(2*n, 3*n)];

        % Set quadprog options
        options = optimset('Algorithm', 'interior-point-convex', '
            TolFun', 1/10^10, 'MaxIter', 1000, 'TolCon', 1/(10^10));


        %Solve MVO and store SD values for plotting

        %Modify this so that it works for variable length R (length
            >1)
        for i = 1:length(R);

            b=[-R(i); ];

            [MVO_x(i,:), MVO_var(i,1)] = quadprog(Q, c, A, b,Aeq,
                beq, lb, ub, [], options);

            %MVO_std = MVO_var.^.5;
            MVO_x=MVO_x(i,1:n);
        end

        adjvector = [MVO_x' mu diag(tempQ)];
end
```

- main_MVO.m

```
function [ MVO_x MVO_var adjvector  nominal_return temp_Q] =
    main_MVO( historical_inflation, historical_asset_returns,...
        expected_inflation, inflation_variance, infbeta, R,
```

```
        transaction_cost, previous_portfolio)
    % expected_inflation, inflation_variance, infbeta, R,
        transaction_cost, max_transactions, previous_portfolio)
%{
historical_inflation: time series data for inflation values
    over the
relevant period

**IN DECIMAL FORM**

historical_asset_returns: time series data for relevant assets
    over
relevant periods

expected_inflation: inflation calculated through markov RS
    model

inflation_variance: inflation variance calculated through
    markov RS

beta: vector of n beta values corresponding to each asset for
    the current
regime

%}

r_M = historical_inflation';
r_it = historical_asset_returns;
mu_M = expected_inflation;
del_M = inflation_variance;

[T, n] = size(r_it);
c = [zeros(3*n,1);];
Aeq = [ones(1,n) zeros(1,2*n)
       eye(n) -eye(n) eye(n)];
beq = [1; previous_portfolio];
lb=zeros(3*n,1);

ub=9999*ones(3*n,1);

nominal_return = prod(1+r_it).^(1/T) - 1;
nominal_return = nominal_return';
% disp(size(nominal_return));
% disp(size(infbeta));
% disp(size(mu_M));
```

```
41      adjusted_return = nominal_return + infbeta*mu_M;
42      alpha = nominal_return(1:end)-infbeta*mu_M;
43      % disp(nominal_return(1:end))
44      % disp(infbeta:mu_M)
45      % disp(size(r_it(:,1)));
46      % disp(size(alpha(1)));
47      % disp(size(infbeta(1)));
48      for i=1:n
49          epsi(:,i)=r_it(:,i)-(alpha(i)+infbeta(i)*r_M(:,1));
50      end
51      % infbeta=infbeta.^(-1);
52      del_epsi=diag(cov(epsi));
53      for i = 1:n;
54          for j = 1:n;
55              if i==j
56                  Q_sf(i,i)=infbeta(i)^2*del_M+del_epsi(i);
57              else
58                  Q_sf(i,j)=infbeta(i)*infbeta(j)*del_M;
59              end
60          end
61      end
62
63      temp_Q = Q_sf;
64
65      Q_sf = [Q_sf zeros(n,2*n); zeros(2*n, 3*n)];
66
67      options = optimset('Algorithm', 'interior-point-convex', '
            TolFun', 1/10^10, 'MaxIter', 1000, 'TolCon', 1/(10^10));
68
69      % Solve Single Factor model and store SD values
70      disp('--------------------')
71      disp(size(adjusted_return'));
72      disp(size(zeros(1,2*n)));
73      disp(size(zeros(1,n)));
74      disp(size(ones(1,2*n)));
75
76      for i = 1:length(R);
77          A=[-(adjusted_return') transaction_cost*ones(1,2*n);];
78              %zeros(1,n) transaction_cost * ones(1,2*n)];
79          b=[-R(i); ]%max_transactions];
80
81
82          [MVO_x(i,:), fval(i,1)] = quadprog(Q_sf, c, A,b, Aeq,beq
                ,...
83                                              lb,ub,[],options);
```

```
84          MVO_var(i,1)=(MVO_x(i,:)*Q_sf*MVO_x(i,:)');
85          MVO_x=MVO_x(i,1:n);
86
87      end
88
89      adjvector = [MVO_x' adjusted_return diag(temp_Q) infbeta];
90
91  end
```

- MVO_comparison.m

```
1  % This function takes in the asset allocatiosn of standard MVO and
       our custom inflation
2  % hedged model. It also tak
3
4  %% MVO_COMPARISON
5  %{
6       Take in the two MVO weightings, one for standard MVO and the
           other for
7       inflation hedged MVO, then show their projected returns over
           the sample
8       period, alongside the market returns for that period.
9
10      This function can be modified accodingly if more benchmark
           models are
11      added.
12
13      MVO_x: vector of weights corresponding to standard MVO w/
           transactions
14      inf_x: vector of weights corresponding to our model
15      asset_returns: relevant for the relevant assets over the sample
16      period
17      market_returns: market returns
18 %}
19 function [cumul_MVO cumul_inf cumul_SP cumul_MF1 cumul_MF2] =
       MVO_comparison( MVO_x, inf_x, projected_returns,...
20      market_returns,MF_returns1,MF_returns2,startyear,endyear)
21
22
23      [T n] = size(projected_returns);
24
25
26      MVO_returns = projected_returns*MVO_x;
27      inf_returns = projected_returns*inf_x;
28
29      cumul_MVO(1) = MVO_returns(1);
```

```
30      cumul_inf(1) = inf_returns(1);
31      cumul_SP(1) = market_returns(1);
32      cumul_MF1(1) = MF_returns1(1);
33      cumul_MF2(1) = MF_returns2(1);
34
35
36      for i = 2:T
37          cumul_MVO(i) = (1+cumul_MVO(i-1))*(1+MVO_returns(i))  - 1;
38          cumul_inf(i) = (1+cumul_inf(i-1))*(1+inf_returns(i)) - 1;
39          cumul_SP(i) =  (1+cumul_SP(i-1))*(1+market_returns(i)) - 1;
40          cumul_MF1(i) =  (1+cumul_MF1(i-1))*(1+MF_returns1(i)) - 1;
41          cumul_MF2(i) =  (1+cumul_MF2(i-1))*(1+MF_returns2(i)) - 1;
42      end
43
44      portfolio_MVO=1 * (1 + cumul_MVO);
45      portfolio_inf=1 * (1 + cumul_inf);
46      portfolio_SP=1 * (1 + cumul_SP);
47
48      figure
49      plot(1:T,cumul_MVO*100, '-b');
50      hold all
51      plot(1:T,cumul_inf*100, '-r');
52      hold all
53      plot(1:T,cumul_SP*100, '-g');
54      hold all
55      plot(1:T,cumul_MF1*100, '-m');
56      hold all
57      plot(1:T,cumul_MF2*100, '-c');
58
59  %  plot(1:T,currinfprices,'-m');
60
61      h = {'Standard MVO', 'Inflation Hedged SF', 'S&P500',...
62           'Vanguard Wellington Inv','CGM Mutual Fund'};
63      h = legend(h);
64
65      grid on;
66
67      title(['Comparing Cumulative Returns of Optimal Portfolios and
           Market'...
68           ,num2str(startyear),' to ',num2str(endyear)])
69      xlabel('Time (in months)')
70      ylabel('Cumulative Monthly Returns (in %)')
71
72
73      figure
```

```
74      plot(1:T,MVO_returns, '-b');
75      hold all
76      plot(1:T,inf_returns, '-r');
77      hold all
78      plot(1:T,market_returns,'-g');
79      hold all
80    %  plot(1:T,currinfprices, '-m');
81
82      h = {'Standard MVO', 'Inflation Hedged SF', 'S&P500','inflation
            rate'};
83
84      title(['Comparing Return Values of Optimal Portfolios and
           Market'...
85          ,num2str(startyear),' to ',num2str(endyear)])
86      xlabel('Time (in months)')
87      ylabel('Monthly Returns')
88
89
90
91      h = legend(h);
92
93      grid on;
94
95      hold all
96
97
98  end
```

- MVO_params.m

```
1  %This function is passed in the pricedata for ONE asset
2  function [ mu, Q, r_it ] = mvo_params(pricedata, end_pred)
3  %% DETERMINE EXPECTED RETURNS AND COVARIANCES FOR ASSETS
4  % Input a matrix of time series data for the desired assets, as
       well as
5  % the number of days in the estimation horizon starting from day 1,
       and the
6  % function will return the expected returns and covariances
       calculated from
7  % the time series data from the estimation horizon.
8  %
9  % [expected returns, covariance matrix] = param_data(time series,
10 %                                                   estimation
       horizon)
11
12 r_it = (data(2:end_pred,:)./data(1:end_pred-1,:)) - 1;
```

```
[T, n] = size(r_it); %time periods, assets
mu = prod(1+r_it).^(1/T) - 1;
Q = cov(r_it);

end
```

- nominal_parameters.m

```
function [ nominal_return, nominal_Q] = nominal_parameters(
    asset_data, market_data)
% take in the relevant time series data for assets and market
% and return the nominal return and covariances for assets

% Model: Single factor CAPM



        r_it = (asset_data(2:end,:)./asset_data(1:end-1,:)) - 1;
        r_M = (market_data(2:end,:)./market_data(1:end-1,:)) - 1;

        [T, n_assets] = size(r_it);

        nominal_return = prod(1+r_it).^(1/T) - 1;
        mu_M = prod(1+r_M).^(1/T) - 1;

        del_M=sum((r_M(:,1) - mu_M(1)).^2)/T; % Factor variance
        beta=(sum(r_it(:,1:end).*repmat(r_M(:,1),1,n_assets))/T-
            mean(r_it(:,1:end)*...
             mean(r_M(:,1)))/del_M);
        alpha = nominal_return(1:end)-beta*mu_M;

        %Noise vector
        for i=1:n_assets
            epsi(:,i)=r_it(:,i)-(alpha(i)+beta(i)*r_M(:,1));
        end
        del_epsi=diag(cov(epsi));

        % Single factor covariance
        for i = 1:n_assets;
            for j = 1:n_assets;
                if i==j
                    nominal_Q(i,i)=beta(i)^2*del_M+del_epsi(i);
                else
                    nominal_Q(i,j)=beta(i)*beta(j)*del_M;
                end
            end
```

```
        end
end
```

- solve_beta3.m

```
% This function is used to compute the inflation Beta, but it can also
% compute the CAPM beta if desired. The second parameter passed in can either be
% inflation rate date or market data
function [solved_beta R_squared] = solve_beta3(tprice,inf_or_market)

    %MARKET - CAPM
        if (inf_or_market==2)
                r_it=tprice(2:end,:)./tprice(1:end-1,:)-1;
    %MARKET - INFLATION
        elseif (inf_or_market==1)
                r_it=tprice(2:end,2:end)./tprice(1:end-1,2:end)-1;
                [T n] = size(r_it)
        disp(size(tprice(:,1)))
                r_it=[tprice(1:T,1) r_it];
        end

        [T, n]=size(r_it);                                  %number of time period T and asset n
        mu=prod(1+r_it).^(1/T)-1;                           % Geometric mean for factor and assets
        del_M=sum((r_it(:,1)-mu(1)).^2)/T;                  % variance of factor under geometric mean

        %Calculate the Beta coefficient

    if inf_or_market==1
        for i = 2:n
        p = polyfit(r_it(:,1)./100,r_it(:,i),1);
        solved_beta(i - 1) = p(1);

        yfit = polyval(p,r_it(:,1)./100);
        y = r_it(:,i);
        yresid = y - yfit;
        SSresid = sum(yresid.^2);
        SStotal = (length(y)-1) * var(y);
        R_squared(i - 1) = 1 - SSresid/SStotal;
    end

```

```
    else
    for i = 2:n
        p = polyfit(r_it(:,1),r_it(:,i),1);
        solved_beta(i - 1) = p(1);

        yfit = polyval(p,r_it(:,1));
        y = r_it(:,i);
        yresid = y - yfit;
        SSresid = sum(yresid.^2);
        SStotal = (length(y)-1) * var(y);
        R_squared(i - 1) = 1 - SSresid/SStotal;
    end


end
```

- solve_mvo_params.m

```
%This function takes in a matrix of asset prices,

function [mu, Q, r_it] = solve_mvo_params(asset_prices,beg_pred,
    end_pred)

        data=asset_prices;
        r_it = (data((beg_pred+1):end_pred,:)./data(beg_pred:
            end_pred-1,:)) - 1;
        [T, n] = size(r_it);
        mu = prod(1+r_it).^(1/T) - 1;
        Q = cov(r_it);

end
```

- calculateSharpeRatio.m

```
% This function computes the sharpe ratio and modified sharpe ratio
    computed
% for each portfolio
function[P_SRATIOS S_SRATIOS] = ...
                calculateSharpeRatio(asset_mu, asset_Q, temp_Q,
                    modelMVO_x, benchMVO_x, ...
                                                        modelMVO_var
                                                            ,
                                                        currriskfreeprices
                                                        )

        %Using our model version of the Sharpe ratio, Preface M

```

```
9           [P_premodel_sratios] = ...
10                  (asset_mu*modelMVO_x'-currriskfreeprices(end)/100)
                        /(modelMVO_var^0.5);
11
12          [P_preMVO_sratios P_preMVO_smu P_preMVO_s_sigmap]=...
13              sharperatio2(asset_mu,temp_Q,benchMVO_x,
                    currriskfreeprices(end)/100);
14
15          %
                ------------------------------------------------------------------------

16          %Using regular version of the Sharpe ratio, Preface S
17          [S_premodel_sratios S_premodel_smu S_premodel_s_sigmap]=...
18              sharperatio2(asset_mu,asset_Q,modelMVO_x,
                    currriskfreeprices(end)/100);
19
20          [S_preMVO_sratios S_preMVO_smu S_postMVO_sigmap]=...
21              sharperatio2(asset_mu,asset_Q,benchMVO_x,
                    currriskfreeprices(end)/100);
22
23          P_SRATIOS = [P_premodel_sratios; P_preMVO_sratios];
24          S_SRATIOS = [S_premodel_sratios; S_preMVO_sratios];
25  end
26
27  % [P_SRATIOS(:,1) S_SRATIOS(:,1)]=calculateSharpeRatio(asset_mu,
        asset_Q, temp_Q, ...
28  %       modelMVO_x(1,:), benchMVO_x(1,:),modelMVO_var,
        currriskfreeprices);
```

- sharperatio2.m

```
1   %Takes in the asset prices for the desired periods
2   function [sratio mu_p sigma_p] = ...
3           sharperatio2(asset_mu, asset_cov, xalloc, riskfreeprice)
4
5
6           mu_p=asset_mu*xalloc';
7           sigma_p=(xalloc*asset_cov*xalloc')^0.5;
8
9           sratio=(mu_p-riskfreeprice)/sigma_p;
10
11  end
12
13  % [P_SRATIOS(:,1) S_SRATIOS(:,1)]=calculateSharpeRatio(asset_mu,
        asset_Q, ...
14  %               modelMVO_x, benchMVO_x,modelMVO_var,
```

```
currriskfreeprices);
```