AMS-512 Capital Markets and Portfolio Theory

Ex Post and *ex Ante* Optimization

Robert J. Frey, Research Professor
Stony Brook University, Applied Mathematics and Statistics

frey@ams.sunysb.edu
http://www.ams.sunysb.edu/~frey

Estimation of Statistics for Portfolio Optimization

Mean-variance optimization requires us to have values for the vector of means and matrix of covariances for the assets in our investment universe. Even when conditions have remained stable over time, we can only estimate these values statistically. It may seem that we can overcome these difficulties by simply collecting enough data points so that these uncertainties are no longer an issue. Unfortunately, due to the different statistical properties of the mean and variance, this is not the case.

Estimation of Mean and Variance

We have T time periods and wish to estimate the associated mean and variance. The sample mean is computed by

$$\hat{\mu}_i = \bar{r}_i = \frac{1}{T}\sum_{t=1}^{T} r_i(t)$$

Assuming a Normal distribution, the variance (and standard deviation) of the that statistic is

$$\sigma^2[\hat{\mu}_i] = \frac{{\sigma_i}^2}{T} \quad \Longrightarrow \quad \sigma[\hat{\mu}_i] = \frac{\sigma_i}{\sqrt{T}}$$

The sample variance (and standard deviation) of this estimate is

$$\hat{\sigma}_i^2 = s_i^2 = \frac{1}{T-1}\sum_{t=1}^{T}(r_i(t) - \bar{r}_i)^2$$

Note that an unbiased estimate divides by $T-1$ rather than T. This because we lose a degree of freedom by using the sample mean above, which is estimated from the same data as the variance. An intuitive way to look at this is that we only begin to get information about the dispersal about the mean of the data with the second data point.

The sample variance (and standard deviation) of this estimate is

$$\sigma^2[\hat{\sigma}_i^2] = \frac{2\,\sigma_i^4}{T-1} \quad \Longrightarrow \quad \sigma[\hat{\sigma}_i^2] = \sqrt{\frac{2}{T-1}}\,\sigma_i^2$$

Comparing Daily, Monthly, and Yearly Returns

Returns compound, thus the relationship between, e.g., annual and monthly returns is

$$(1 + r_y) = (1 + r_m(1))\,(1 + r_m(2))\,\ldots\,(1 + r_m(11))\,(1 + r_m(12))$$

If the values of r_m are not too large then we can use the approximation

$$r_y \approx r_m(1) + r_m(2) + \ \ldots\ + r_m(11) + r_m(12)$$

Alternately, we can use the log return $l = \log[1 + r]$ in which case the yearly log return is exactly the sum of the monthlies.

If we look at the computation of the mean it is obvious that

$$\mu_y = 12\,\mu_m$$
$$\sigma_y{}^2 = 12\,\sigma_m{}^2 \quad \Longrightarrow \quad \sigma_y = \sqrt{12}\ \sigma_m$$

If we look at the means and standard deviations of return, then

$$\mu_m = \frac{1}{12}\,\mu_y$$
$$\sigma_m = \frac{1}{\sqrt{12}}\,\sigma_y$$

Example — Comparing Dispersal of the Mean

If the yearly mean and standard deviation of return are both 0.1 for an asset then its monthly mean and standard deviation are

$$\mu_y = 0.1$$
$$\sigma_y = 0.1$$
$$\mu_y / \sigma_y = 1.0$$

$$\mu_m = 0.00833$$
$$\sigma_m = 0.02887$$
$$\mu_m / \sigma_m = 0.2887$$

and, assuming 250 trading days in a year, its daily are

$$\mu_d = 0.000400$$
$$\sigma_d = 0.006325$$
$$\mu_d / \sigma_d = 0.06325$$

As can be clearly seen, as the measurement period decreases the mean decreases much faster than the standard deviation does.

Mean Blur

When we look at the estimated mean and its sample standard deviation, we have

$$\sigma[\hat{\mu}] = \frac{\sigma}{\sqrt{T}}$$

If we want to decrease the statistical error of our estimate by a factor of 2, then we need 4 times as much data. A factor of 4, then 16 times as much data. In general, given the typical values of returns for most assets such as stocks, to get an acceptable level of accuracy demands an unrealistic amount of data. If we try to sample the data more frequently, then although the number of data increases, the relative size of the standard deviation decreases more slowly than the mean so the relative error is largely unaffected.

The estimation problems faced with estimating the variance (and standard deviation) are not as serious. Decreasing the measurement period yields more data and the sampling error of the variance scales with $\sqrt{2/(T-1)}$ so the relative error is not too extreme if T is large. This is in contrast with the mean where increasing the sampling frequency does us little good and we need a long duration of observations to bring the sampling error to a manageable size.

Factor Blur

In factor models, the effects similar to that of mean blur occur with the intercept terms of the model—what is often called α—which are related to the means of sources and target variables. The estimates of the factor loadings b are related to the covariance between the sources and target and, like the variance, are better behaved.

Numerical Stability of the Covariance Matrix

The direct estimation of the covariance matrix involves the estimation of $n(n+1)/2$ distinct parameters for n assets. Unfortunately, to accomplish this effectively requires a great deal of data and the result may be singular. Even if non-singular, unless the number of data points is large, then the matrix will be poorly conditioned with small changes in the observation leading to large changes in the values of the inverse matrix. One measure of stability is the condition number (see http://en.wikipedia.org/wiki/Condition_number.) which is the ratio of the absolute value of the largest to smallest eigenvalues.

$$\kappa(M) = \frac{|\lambda_{\max}|}{|\lambda_{\min}|}$$

Example — Simulations of Condition Number

In[]:=
```
vnM = Array[0. &, 100];
mnS = Array[If[#1 == #2, 1., 0.5] &, {100, 100}];
```

In[]:=
```
MatrixForm[mnS[[1 ;; 5, 1 ;; 5]]]
```

Out[]//MatrixForm=

$$\begin{pmatrix} 1. & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 1. & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 1. & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 1. & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 1. \end{pmatrix}$$

In[]:=
```
iNumbParams = n (n + 1) / 2. /. n → 100
```

Out[]=
```
5050.
```

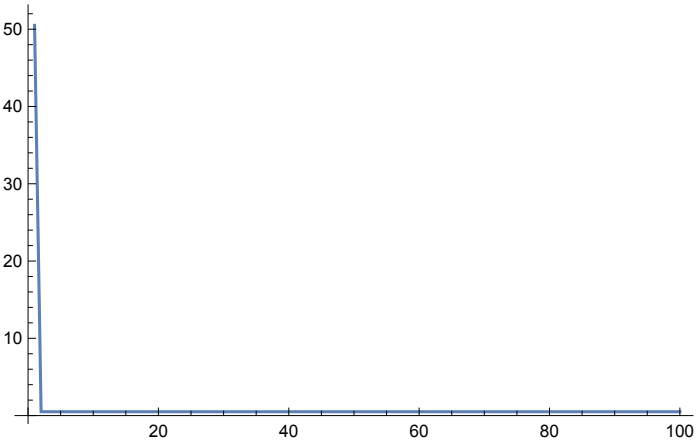
In[]:=
```
xConditionNumber = First[#] / Last[#] &[Abs[Through[{Max, Min}[Eigenvalues[#]]]]] &;
```

In[•]:=
```
nConditionNumber = xConditionNumber[mnS]
```

Out[•]=
```
101.
```

In[•]:=
```
ListLinePlot[Eigenvalues[mnS]]
```

Out[•]=



In[•]:=
```
mnConditionNumbers = Table[
   mnX = RandomVariate[MultinormalDistribution[vnM, mnS], i];
   {i / 12., xConditionNumber[Covariance[mnX]]},
   {i, {60, 120, 240, 480, 960, 1920, 3840}}
  ];
```

In[•]:=
```
mnConditionNumbers = Append[mnConditionNumbers, {∞, nConditionNumber}];
```

In[•]:=
```
TableForm[mnConditionNumbers,
 TableHeadings → {None, {"Monthly Data in Years", "Condition Number"}}]
```

Out[•]//TableForm=

Monthly Data in Years	Condition Number
5.	2.33734×10^{16}
10.	13 180.2
20.	699.147
40.	310.994
80.	206.474
160.	160.982
320.	149.907
∞	101.

$$\mathbf{\Sigma} = \mathbf{x}^T\left(\mathbf{B}\,\mathbf{B}^T + \mathbf{D}\right)\mathbf{x}$$

$$\mathbf{\Sigma}:\ n(n+1)/2$$

$$\left(\mathbf{B}\,\mathbf{B}^T + \mathbf{D}\right):\ n\,m + n = n(m+1)$$

In[•]:=
```
nNoF = 6;
nNoA = 1000;
nNoA (nNoA + 1) / 2
```

Out[•]=
```
500 500
```

```
In[•]:= nNoA (nNoF + 1)
Out[•]= 7000

In[•]:= 7000.
        500 500.
Out[•]= 0.013986
```

The Challenge of *Ex Ante* Portfolio Optimization

When we are faced with a mean-variance estimation problem, the statistical effects we must deal with mean that performance predictions based on historical estimates tend to outperform the ideal ones we would make if we knew the true mean and covariance exactly. And the actual performance, based on suboptimal allocations, tends to underperform the theoretical.

Historic ≥ Theoretical ≥ Actual

Thus, we are dangerous ground. Our analyses based on available data give us unrealistic expectations, while the actual performance usually falls far short of those expectations.

Ex Post vs. *Ex Ante*

In practice we have available to us *ex post* or in-sample results based on our observation of history, but what we are interested in are *ex ante* or out-of-sample results, the results we will actually experience from applying our analysis. Even when we take steps to prevent it, we inevitably over-fit our results to the past and produce results that are suboptimal for the future.

To explore the problem of in-sample vs. out-of-sample performance we will examine three cases: the in-sample historic, the theoretical optimal, and out-of-sample future. We will be using the following portfolio optimization which in its pure theoretical form is:

$$\mathcal{T} = \max_{\mathbf{x}} \left\{ \boldsymbol{\mu}^T \mathbf{x} \mid \mathbf{x}^T \boldsymbol{\Sigma} \mathbf{x} \le \sigma^2 \wedge \mathbb{1}^T \mathbf{x} = 1 \right\}$$

Let $\boldsymbol{\eta}$ be a random variable representing the deviations of the return from the mean, then one way to view the above is

$$\mu_{\mathcal{T}} = \max\left(\mathrm{E}\left[(\boldsymbol{\mu} + \boldsymbol{\eta})^T \mathbf{x}\right]\right) = \max\left(\mathrm{E}\left[\boldsymbol{\mu}^T \mathbf{x} + \boldsymbol{\eta}^T \mathbf{x}\right]\right) = \max\left(\boldsymbol{\mu}^T \mathbf{x}\right)$$

For our purposes here, we will assume that we know $\boldsymbol{\Sigma}$ exactly and focus our attention on the uncertainties associated with $\boldsymbol{\mu}$.

Historical vs. Theoretical

When we have estimates based on historical data (these are the *ex post* results) then they equal theoretical value of the mean plus a random sampling error $\boldsymbol{\eta}$ which has mean $\mathbf{0}$:

$$\mathcal{H} = \max_{\mathbf{x}} \left\{ (\boldsymbol{\mu} + \boldsymbol{\eta})^T \mathbf{x} \mid \mathbf{x}^T \boldsymbol{\Sigma} \mathbf{x} \le \sigma^2 \wedge \mathbb{1}^T \mathbf{x} = 1 \right\}$$

The expected performance using historical performance is

$$\mu_{\mathcal{H}} = \mathrm{E}\left[\max\left((\boldsymbol{\mu} + \boldsymbol{\eta})^T \mathbf{x}\right)\right]$$

We can think of the theoretical solution as reversing the order of execution of the expectation and maximization.

It is easy to show, using Jensen's inequality (http://en.wikipedia.org/wiki/Jensen%27 s_inequality), that

$$\mathrm{E}[\mu_{\mathcal{H}}] \geq \mu_{\mathcal{T}}$$

Intuitively, one can think of the $\mu_{\mathcal{H}}$s as each being over-fit to the particular errors $\boldsymbol{\eta}$ in each history's realization of the returns, rather than to the true expected value itself. The $\mathcal{H}$-solutions are fine-tuned to history's particular realization rather than the likely future.

Theoretical vs. Actual

Clearly the best estimate, on average, for the unknown future realization of returns is the theoretical solution. Unfortunately, we cannot access that theoretical solution and our estimated allocations will, therefore, be suboptimal. Hence, the expected actual, or *ex ante*, performance will be worse

$$\mu_{\mathcal{T}} \geq \mathrm{E}[\mu_{\mathcal{A}}]$$

It is important to note that these results are to the expectations of performance. It is possible that luck will be on our side and our actual performance will exceed the expectations based on our historic analysis, but the probabilities are quite strongly not in our favor. We have to live with the fact that

$$\mathrm{E}[\mu_{\mathcal{H}}] \geq \mu_{\mathcal{T}} \geq \mathrm{E}[\mu_{\mathcal{A}}]$$

Improving *Ex Ante* Performance

We will cover 3 approaches to handling the problem of *ex ante* performance: enforcing diversification, shrinkage estimates, and combining estimates.

Enforcing Diversification

The effects of over-fit tend to produce solutions which are less diversified than the theoretical optimal. Sampling errors in the mean tend to increase allocations when positive and decrease them when negative so that positions tend to be over- and under-represented. This is usually a material effect and many have noted that a minimum variance portfolio (which is equivalent to assuming all assets have the same return) often outperforms one based on historic returns.

We return to our usual portfolio optimization, minimizing risk with a target return and no shorts,

$$\mathcal{M} = \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x} \;\middle|\; \boldsymbol{\mu}^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\wedge\; 0 \leq \mathbf{x} \right\}$$

One method is to introduce a quadratic penalty based on a positive definite matrix $\mathbf{P}$, which often takes the value of the identity matrix $\mathbf{I}$:

$$\mathcal{M} = \min_{\mathbf{x}} \left\{ \frac{1}{2} \left(\mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x} + \lambda\, \mathbf{x}^T \mathbf{P}\, \mathbf{x} \right) \;\middle|\; \boldsymbol{\mu}^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\wedge\; 0 \leq \mathbf{x} \right\}$$

For example, for the identity matrix and n assets the solution is towards diversifying $\mathbf{x}$.

$$\mathcal{M} = \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T (\boldsymbol{\Sigma} + \lambda\, \mathbf{I})\, \mathbf{x} \;\middle|\; \boldsymbol{\mu}^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\wedge\; 0 \leq \mathbf{x} \right\}$$

The penalty parameter λ is arbitrary so we drop the 1/2 when expanding to

$$\mathcal{M} = \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x} + \lambda\, \mathbf{x}^T \mathbf{x} \;\middle|\; \boldsymbol{\mu}^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\wedge\; 0 \leq \mathbf{x} \right\}$$

Note that the objective function as $\lambda \to \infty$ is driven towards

$$\min_{\mathbf{x}} \left\{ \mathbf{x}^T \mathbf{I}\, \mathbf{x} \right\} \quad \Longrightarrow \quad x_i \propto \frac{1}{n}$$

Thus, as λ increases, the push to diversify becomes stronger and stronger. The precise value of λ relative to $\boldsymbol{\Sigma}$ is usually tuned by cross-validation, i.e., dividing existing data into separate calibration and hold-out periods to simulate out-of-sample performance.

An alternative to using a fixed penalty represented by the identity matrix is to use the diagonalized covariance; *i.e.*, to assume that the assets are uncorrelated.

$$\mathcal{M} = \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T ((1-\lambda)\, \boldsymbol{\Sigma} + \lambda\, \text{DiagonalMatrix}[\boldsymbol{\Sigma}])\, \mathbf{x} \;\middle|\; \mu^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\wedge\; 0 \leq \mathbf{x} \right\}$$

Note that the objective function as $\lambda \to \infty$ is driven towards

$$\min_{\mathbf{x}} \left\{ \mathbf{x}^T\, \text{DiagonalMatrix}[\boldsymbol{\Sigma}]\, \mathbf{x} \right\} \quad \Longrightarrow \quad x_i \propto \frac{1}{{\sigma_i}^2}$$

Another approach is to add upper bounds to the positions as below:

$$\mathcal{M} = \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x} \;\middle|\; \boldsymbol{\mu}^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\wedge\; 0 \leq \mathbf{x} \leq \mathbf{u} \right\}$$

These upper bounds must, of course, on average be greater than $1/n$ or the optimization either tends to evenly allocate across assets or ends up infeasible. The idea is prevent a few large positions from taking over the portfolio.

There is no reason that a diversification penalty and upper bounds cannot both be used in the same optimization.

Shrinkage Estimation

A shrinkage estimate of the mean takes the observed estimate $\hat{\boldsymbol{\mu}}$ and shrinks it towards a common value u to produce $\tilde{\boldsymbol{\mu}}$ which is produced by the convex combination

$$\tilde{\boldsymbol{\mu}} = (1-w)\, \hat{\boldsymbol{\mu}} + w\, u\, \mathbb{1}, \quad 0 \leq w \leq 1$$

The idea is to take a complex model which takes full advantage of the statistical estimates of the parameters and balance it out against a simple model which will be more resistant to over-fitting. A frequent choice, though by no means the only reasonable choice, of u above is the mean mean. If $\hat{\boldsymbol{\mu}}$ is a vector of dimension n, then u might be estimated by

$$u = \frac{\mathbb{1}^T \hat{\boldsymbol{\mu}}}{n}$$

The James-Stein shrinkage estimate (See http://en.wikipedia.org/wiki/James–Stein_estimator.) computes the weight w as follows:

$$w = \min\left[1, \; \frac{n-2}{T\, (\hat{\boldsymbol{\mu}} - u\, \mathbb{1})^T\, \boldsymbol{\Sigma}^{-1}\, (\hat{\boldsymbol{\mu}} - u\, \mathbb{1})}\right]$$

where T is the number of periods, the sample size, and n is the number of securities.

Another possible choice is to set u to the return of the minimum variance portfolio. The computation can be specialized to the form below which guarantees that $0 \leq w \leq 1$:

$$w = \frac{n+2}{n+2+T\left(\hat{\boldsymbol{\mu}} - u\,\mathbb{1}\right)^T \boldsymbol{\Sigma}^{-1}\left(\hat{\boldsymbol{\mu}} - u\,\mathbb{1}\right)}$$

There are also shrinkage estimates for the covariance which will not be covered here (See, however, http://www.ledoit.net/ole2.pdf.).

In an earlier lecture we have covered using correlation denoising based on the eigenvalues using the Marchenko-Pastur distribution.

Combining Estimates

Another approach is combine two different estimation approaches and weighs them based on their variance and covariance. If the two estimates are uncorrelated then combining, e.g., an estimate based on history $\mu_i^{(\mathcal{H})}$ and another based on a factor model $\mu_i^{(\mathcal{F})}$, is simpler. The combined estimate $\mu_i^{(\mathcal{H}+\mathcal{F})}$ weighs each inversely proportionally to their sampling variances.

$$\mu_i^{(\mathcal{H}+\mathcal{F})} = \left(\frac{\mu_i^{(\mathcal{H})}}{\sigma^2\left[\mu_i^{(\mathcal{H})}\right]} + \frac{\mu_i^{(\mathcal{F})}}{\sigma^2\left[\mu_i^{(\mathcal{F})}\right]}\right) \Big/ \left(\frac{1}{\sigma^2\left[\mu_i^{(\mathcal{H})}\right]} + \frac{1}{\sigma^2\left[\mu_i^{(\mathcal{F})}\right]}\right)$$

The idea here is that each estimate individually has its over-fitting, but the nature of the over-fit will be different in each case. Combining them, therefore, tends to reduce the over-fitting effects.

If the factor model were the CAPM, then a possible estimate for the sampling variance of the estimate of the mean might be estimated by

$$\sigma^2\left[\mu_i^{(\mathcal{F})}\right] = \frac{(\beta_i\,\hat{\sigma}_M)^2}{T_{\mathcal{F}}}$$

and for the historic estimate by

$$\sigma^2\left[\mu_i^{(\mathcal{H})}\right] = \frac{\hat{\sigma}_i{}^2}{T_{\mathcal{H}}}$$

Note that we've subscripted the number of observations T to indicate that the number of observations used in each case may not be the same, although one would often finds them to be equal.

A Simulation

A simulation experiment can demonstrate these effects. The issues of *ex post* against *ex ante* portfolio optimization is pervasive and unavoidable, but there are techniques for dealing with them.

The Portfolio Optimization Problem

We will use as our standard the problem the allocation which selects from the set of efficient portfolios the one which has the highest Sharpe ratio ψ. We will generate the efficient set using the form of the quadratic program shown below.

$$\mathbf{x}_{\max[\psi]} = \underset{\mathbf{x}}{\operatorname{argmax}} \left\{ \frac{\boldsymbol{\mu}^T\mathbf{x} - r_f}{\sqrt{\mathbf{x}^T\boldsymbol{\Sigma}\,\mathbf{x}}} \;\middle|\; \min_{\mathbf{x}} \left\{ \frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}\,\mathbf{x} \;\middle|\; \boldsymbol{\mu}^T\mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T\mathbf{x} = 1 \;\wedge\; 0 \leq \mathbf{x} \leq \mathbf{1} \right\} \right\}$$

In what follows, recall that the *ex post* returns are the known, in-sample past and the *ex ante* returns are the unknown, out-of-sample future. The *ex post* performance takes the solution generated by the *ex post* returns and

evaluates it with those same *ex post* returns. In contrast, the *ex ante* performance takes the solution generation by the *ex post* returns but evaluates it with the *ex ante* returns.

Generating the Ex Post (In-Sample) and Ex Ante (Out-of-Sample) Returns

The CAPM is used as the model to generate the stock returns for the simulation study. We will generate random returns from the model

$$r_i(t) = r_f + \beta_i(r_M(t) - r_f) + \epsilon_i(t)$$

where $r_M \approx$ StudentT[μ_M, σ_M, ν] and $\epsilon_i \approx$ StudentT[0, $\sigma[\epsilon_i]$, ν]. We assume that the market is stationary, *i.e.*, the values of these parameters do not change over the study (in-sample and out-of-sample) periods.

We also need to generate reasonable values for the betas and error variances. We will generate the fixed set of βs by $\beta \approx$ U[0.5, 1.5] and the $\sigma^2[\epsilon]$s by $\sigma^2[\epsilon] \approx \frac{2}{5}\sigma_M{}^2\,\chi^2[5]$. Thus, the betas have a mean of 1 spread uniformly over the interval [0.5, 1.5] and the error variances have a mean twice that of the market's variance's and are drawn from a suitably transformed χ^2-distribution with 5 degrees of freedom. This means that a "typical" stock's mean is the market mean. Its systematic variance is the same as the market's and is one-third of its total variance. Its idiosyncratic variance is two-thirds of its total variance.

The Historical Solution: $\mathcal{H}$

We use the return model to generate simulated *ex post* in-sample past returns.

The historical solution uses the *ex post* data to estimate the model parameters, including those of the market.

$$\mathbf{x}_{\mathcal{H}} = \underset{\mathbf{x}}{\text{argmax}} \left\{ \frac{\boldsymbol{\mu}_{\mathcal{H}}{}^T \mathbf{x} - r_f}{\sqrt{\mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x}}} \;\middle|\; \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T \boldsymbol{\Sigma}_{\mathcal{H}}\, \mathbf{x} \;\middle|\; \boldsymbol{\mu}_{\mathcal{H}}{}^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\bigwedge\; 0 \leq \mathbf{x} \leq \mathbf{1} \right\} \right\}$$

$$\mu_{\mathcal{H}} = \boldsymbol{\mu}_{\mathcal{H}}{}^T \mathbf{x}_{\mathcal{H}}$$

$$\sigma_{\mathcal{H}}{}^2 = \mathbf{x}_{\mathcal{H}}{}^T \hat{\boldsymbol{\Sigma}}\, \mathbf{x}_{\mathcal{H}}$$

The Theoretical Solution: $\mathcal{T}$

The theoretical solution yields the optimal solution given the actual parameter values. In practice, we never know their true values, but in this simulation we know them exactly.

$$\mathbf{x}_{\mathcal{T}} = \underset{\mathbf{x}}{\text{argmax}} \left\{ \frac{\boldsymbol{\mu}^T \mathbf{x} - r_f}{\sqrt{\mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x}}} \;\middle|\; \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x} \;\middle|\; \boldsymbol{\mu}^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\bigwedge\; 0 \leq \mathbf{x} \leq \mathbf{1} \right\} \right\}$$

$$\mu_{\mathcal{T}} = \boldsymbol{\mu}^T \mathbf{x}_{\mathcal{T}}$$

$$\sigma_{\mathcal{T}}{}^2 = \mathbf{x}_{\mathcal{T}}{}^T \boldsymbol{\Sigma}\, \mathbf{x}_{\mathcal{T}}$$

The Ex Ante Solution Based on $\mathcal{H}$: $\mathcal{A}$

We use the return model to generate simulated *ex ante* out-of-sample future returns. In practice, the only allocations we have to use for this future period is an *ex post* solution computed from some prior historical data. The *ex ante* performance is then an *ex post* solution applied to the *ex ante* future that emerges.

The realized *ex ante* statistics computed from the future returns, identified with a $\mathcal{A}$ subscript, are used to compute the out-of-sample performance $\mathcal{X}$ of the *ex post* solution, $\mathbf{x}_{\mathcal{H}}$.

$$\mu_{\mathcal{A}} = \boldsymbol{\mu}_{\mathcal{A}}{}^T \mathbf{x}_{\mathcal{H}}$$

$$\sigma_{\mathcal{A}}{}^2 = \mathbf{x}_{\mathcal{H}}{}^T \, \mathbf{\Sigma}_{\mathcal{A}} \, \mathbf{x}_{\mathcal{H}}$$

Other Considerations and Approaches

One could imagine any number of experiments based on different approaches to computing *ex post* statistics; e.g., shrinkage estimates, weighted combinations of multiple estimation approaches, or more complex factor models. We could also explore different approaches to setting up the optimization, e.g., quadratic penalties to increase diversification and tighter upper position bounds. Each such approach gives us another *ex post* solution that we can evaluate *ex ante*.

A Simulation, Step-By-Step

We'll step through a Historical-Theoretical-Actual simulation, using monthly returns drawn from a universe in which all stocks are described by stationary parameters and Normally distributed processes following the CAPM.

Functions Required

We will need to load the quadratic programming package.

In[•]:=
```
<< Local`QuadraticProgramming`
```

This function builds and searches the efficient frontier and reports the portfolio with the approximate maximum Sharpe ratio.

In[•]:=
```
xMaxSharpeAlloc[vnMean_, mnCov_, nRiskFree_] := Module[
    {mnEfficientFrontier, nMaxRet, nMinVarRet, iN, iTangent},
    iN = Length[vnMean];
    nMinVarRet = vnMean.Last@xQuadraticProgramming[
        {{}, 0.5 mnCov}, {Array[1. &, iN]}, {{1, 0}}, Table[{0., 1.}, {iN}]];
    nMaxRet = Max[vnMean];
    mnEfficientFrontier =
     Table[{Sqrt[#.mnCov.#], vnMean.#, #} &[Last@xQuadraticProgramming[{{}, 0.5 mnCov},
         {Array[1. &, iN], vnMean}, {{1, 0}, {nTarget, 1}}, Table[{0., 1.}, {iN}]]],
      {nTarget, nMinVarRet, 0.99 nMaxRet, (nMaxRet - nMinVarRet) / 19}];
    iTangent = First@First@Position[#, Max[#]] &[
      (#[[2]] - nRiskFree) / #[[1]] & /@ mnEfficientFrontier];
    mnEfficientFrontier[[iTangent]]
   ];
```

This is a simple diversification function. It calculates the cosine of vector $\mathbf{x}$ and a conforming vector $\mathbb{1}$ of equal values; 1 = maximally diversified.

In[•]:=
```
xDiversification = Length[#]^-1/2 Total[#] / Norm[#] &;
```

Generating The Market Parameters

We first define a set of parameters for the *ex post* and *ex ante* periods of the simulations: the number of assets, the length of the *ex post* period, and the length of the *ex ante* period:

```
In[•]:= iNoA = 20;
        iNoH = 10 × 12;
        iNoF = 12;
        nDoF = 3.5;
```

Next the ambient market parameters are chosen:

```
In[•]:= nMktMean = 0.0075;
        nMktVar = 0.0025;
        nRiskFree = 0.0004;
```

The βs of this universe are chosen to be Uniformly distributed between 0.5 and 1.5:

```
In[•]:= vnBeta = RandomReal[{0.5, 1.5}, iNoA];
```

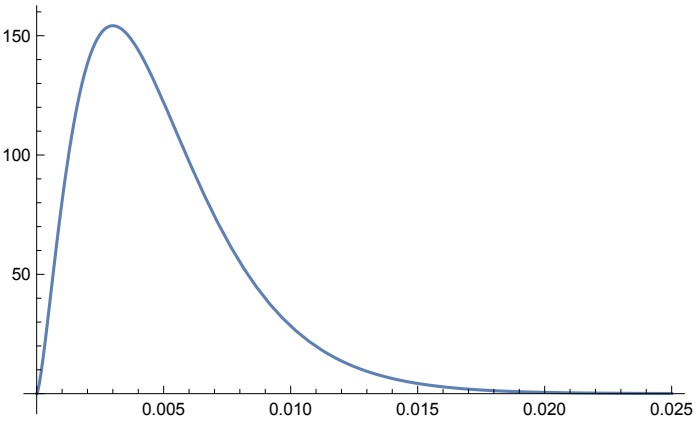
```
In[•]:= Mean[vnBeta]
Out[•]= 1.00991
```

Next, we generate the error variances, illustrating the shape of the χ^2 distribution used.

```
In[•]:= distD = TransformedDistribution[ 2 nMktVar / 5 x, x ≈ ChiSquareDistribution[5]];
```

```
In[•]:= Plot[PDF[distD, x], {x, 0, 10 nMktVar}, PlotRange → All]
```

```
Out[•]=
```



$$r_i(t) - r_f = \beta(r_M(t) - r_f) + \epsilon_i(t)$$

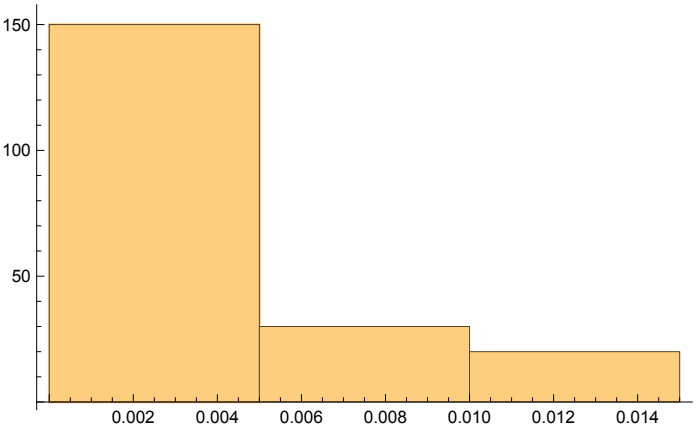
$$\mu_i = r_f + \beta(\mu_M - r_f)$$

$$\Sigma = \sigma_M{}^2 \beta \beta^T + \mathbf{D}$$

```
In[•]:= vnErrVar = RandomVariate[distD, iNoA];
```

In[•]:=
```
Histogram[vnErrVar, Automatic, "PDF"]
```

Out[•]=



Based on the CAPM we compute the theoretical mean vector and covariance matrix.

In[•]:=
```
vnMean = nRiskFree + vnBeta (nMktMean - nRiskFree);
mnCov = nMktVar KroneckerProduct[vnBeta, vnBeta] + DiagonalMatrix[vnErrVar];
```

The optimal theoretical allocation is based on the actual market parameters. In the real world we never know these parameters, only our estimates of them.

The Theoretical Solution $\mathcal{T}$

In[•]:=
```
vnAlloc𝒯 = Last@xMaxSharpeAlloc[vnMean, mnCov, nRiskFree]
```

Out[•]=
```
{0.0704505, 0.0857417, 0.0177345, 0.0227466, 0.020218, 0.0439617,
 0.011244, 0.0885346, 0.0112627, 0.0517577, 0.0222987, 0.102227, 0.0347381,
 0.0277837, 0.0481393, 0.0464127, 0.0338275, 0.202361, 0.0443932, 0.0141665}
```

A summary for each allocation containing the portfolio standard deviation, mean, Sharpe ratio and diversification is

In[•]:=
```
vnSummary𝒯 =
 {√(#.mnCov.#), vnMean.#, (vnMean.# - nRiskFree)/√(#.mnCov.#), xDiversification[#]} &[vnAlloc𝒯]
```

Out[•]=
```
{0.0603441, 0.00875655, 0.138482, 0.754111}
```

In[•]:=
```
StandardDeviation[StudentTDistribution[μ, σ, ν]]
```

Out[•]=
$$\begin{cases} \sqrt{\frac{\nu}{-2+\nu}}\,\sigma & \nu > 2 \\ \text{Indeterminate} & \text{True} \end{cases}$$

Generating Returns for the Ex Post Period

First, the market returns are generated and analyzed.

In[]:=
```
vnMktRetnℋ =
  RandomReal[StudentTDistribution[nMktMean, Sqrt[(nDoF - 2)/nDoF] Sqrt[nMktVar], nDoF], iNoH];
nMktMeanℋ = Mean[vnMktRetnℋ];
nMktVarℋ = Variance[vnMktRetnℋ];
```

then a synthetic history of stock returns based on the CAPM is generated and its statistics estimated using "standard" estimators.

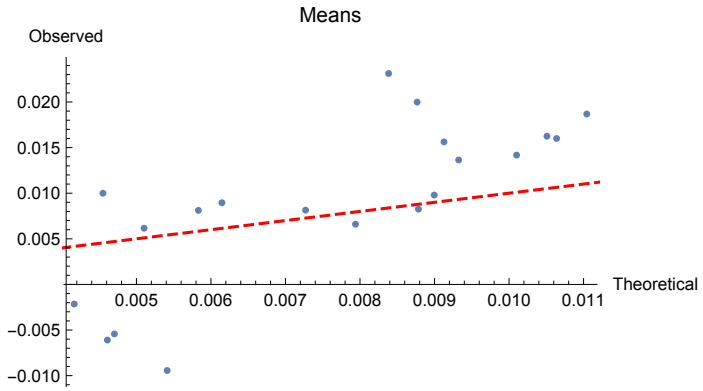
In[]:=
```
mnRetnℋ = Transpose[Table[
    nRiskFree + vnBeta[[iWho]] (vnMktRetnℋ - nRiskFree) +
     RandomReal[StudentTDistribution[0, Sqrt[(nDoF - 2)/nDoF] Sqrt[vnErrVar[[iWho]]], nDoF], iNoH],
    {iWho, 1, iNoA}
   ]];
```

In[]:=
```
vnMeanℋ = Mean[mnRetnℋ];
mnCovℋ = Covariance[mnRetnℋ];
```

In most instances, we do a poor job of estimating the means but a reasonable one of estimating the standard deviations.

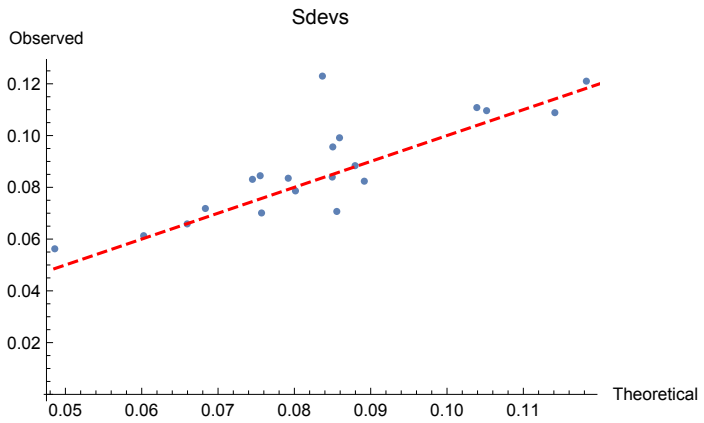
In[]:=
```
Show[
 ListPlot[{vnMean, vnMeanℋ}ᵀ, PlotLabel → "Means",
  AxesLabel → {"Theoretical", "Observed"}, PlotRange → All],
 Plot[x, {x, Min[Join[vnMean, vnMeanℋ]], Max[Join[vnMean, vnMeanℋ]]},
  PlotStyle → {Red, Dashed}]
]
```

Out[]=



In[•]:=
```
vnSdev = √Diagonal[mnCov] ;
vnSdevℋ = √Diagonal[mnCovℋ] ;
Show[
 ListPlot[{vnSdev, vnSdevℋ}ᵀ, PlotLabel → "Sdevs",
  AxesLabel → {"Theoretical", "Observed"}, PlotRange → All],
 Plot[x, {x, Min[Join[vnSdev, vnSdevℋ]], Max[Join[vnSdev, vnSdevℋ]]},
  PlotStyle → {Red, Dashed}]
]
```

Out[•]=



The Ex Post Solution $\mathcal{H}$

Next we estimate the CAPM from the synthetic history and use that model to estimate the required statistics.

The allocation and in-sample performance summary for the historical solution $\mathcal{H}$ are

In[•]:=
```
vnAllocℋ = Last@xMaxSharpeAlloc[vnMeanℋ, mnCovℋ, nRiskFree]
```

Out[•]=
```
{0.262222, 0.000126087, 0.186873, 0.0000548042, 0.0000558131, 0.212928, 0.000114668,
 0.000720868, 0.0000505123, 0.000301526, 0.00923856, 0.00109048, 0.00012913,
 0.000549296, 0.0370754, 0.258274, 0.0097463, 0.000410573, 0.0196478, 0.000391307}
```

In[•]:=
```
vnSummaryℋ =
 {√#.mnCovℋ.# &, vnMeanℋ.#&, (vnMeanℋ.# - nRiskFree)/√#.mnCovℋ.# &, xDiversification[#]&}[vnAllocℋ]
```

Out[•]=
```
{0.0607502, 0.0163911, 0.263227, 0.479277}
```

In[•]:=
```
vnSummary𝒯
```

Out[•]=
```
{0.0603441, 0.00875655, 0.138482, 0.754111}
```

Generating Returns for the Ex Ante Period

The next step is to generate the future *ex ante* returns and estimate their mean and covariance. Those estimates reflect the actual realization of those returns and will be the same regardless of which *ex post* solution we use. We'll designate these with an $\mathcal{A}$ suffix.

```
In[•]:= vnMktRetn𝒳 =
          RandomReal[StudentTDistribution[nMktMean, Sqrt[(nDoF - 2)/nDoF] Sqrt[nMktVar], nDoF], iNoF];
        mnRetn𝒳 = Transpose[Table[
            nRiskFree + vnBeta[[iWho]] (vnMktRetn𝒳 - nRiskFree) +
             RandomReal[NormalDistribution[0, Sqrt[(nDoF - 2)/nDoF] Sqrt[vnErrVar[[iWho]]]], iNoF],
            {iWho, 1, iNoA}
           ]];

In[•]:= vnMean𝒳 = Mean[mnRetn𝒳];
        mnCov𝒳 = Covariance[mnRetn𝒳];
```

The Ex Ante Results $\mathcal{X}$

The set of *ex ante* results use the *ex post* allocations $x_{\mathcal{H}}$ and are designated with an $\mathcal{X}$ suffix.

```
In[•]:= vnSummary𝒳 =
          {Sqrt[#.mnCov𝒳.#], vnMean𝒳.#, (vnMean𝒳.# - nRiskFree)/Sqrt[#.mnCov𝒳.#], xDiversification[#]} &[vnAlloc𝓗]

Out[•]= {0.0462715, 0.0169991, 0.358733, 0.479277}
```

Summary of the Experiment

Each time we run the code above we get a different sample point. In some cases the *ex ante* results will be worse than the *ex post* and sometimes better; however, if we run the simulation many time, definite patterns will emerge.

```
In[•]:= TableForm[{vnSummary𝓗, vnSummary𝒯, vnSummary𝒳}, TableHeadings →
          {{"𝓗: Ex Post", "𝒯: Theoretical", "𝒳: Ex Ante"}, {"σ", "μ", "ψ", "δ"}}]

Out[•]//TableForm=
```

	σ	μ	ψ	δ
$\mathcal{H}$: Ex Post	0.0607502	0.0163911	0.263227	0.479277
$\mathcal{T}$: Theoretical	0.0603441	0.00875655	0.138482	0.754111
$\mathcal{X}$: Ex Ante	0.0462715	0.0169991	0.358733	0.479277

$E[\mu_{\mathcal{H}}] \geq \mu_{\mathcal{T}} \geq E[\mu_{\mathcal{X}}]$ Revisited

We need to perform the *Ex Post*-Theoretical-*Ex Ante* Comparison Experiment many times to gain real insight into not only the true differences in the the expectations, but how much variability in those expectations we can expect .

The Portfolio Optimization

We will use as our standard the problem the allocation which selects from the set of efficient portfolios the one which has the highest Sharpe ratio ψ. We will generate the efficient set using the form of the quadratic program shown below.

$$\mathbf{x}_{\max[\psi]} = \underset{\mathbf{x}}{\operatorname{argmax}} \left\{ \frac{\boldsymbol{\mu}^T \mathbf{x} - r_f}{\sqrt{\mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x}}} \;\middle|\; \min_{\mathbf{x}} \left\{ \frac{1}{2} \mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x} \;\middle|\; \boldsymbol{\mu}^T \mathbf{x} \geq \tau \;\wedge\; \mathbb{1}^T \mathbf{x} = 1 \;\wedge\; 0 \leq \mathbf{x} \leq \mathbf{1} \right\} \right\}$$

In what follows, recall that the *ex post* returns are the known, in-sample past and the *ex ante* returns are the unknown, out-of-sample future. The *ex post* performance takes the solution generated by the *ex post* returns and evaluates it with those same *ex post* returns. In contrast, the *ex ante* performance takes the solution generation by the *ex post* returns but evaluates it with the *ex ante* returns.

There are any number of alternatives that we could use as our basic optimization problem: We could use the minimum variance portfolio. We could chose the portfolio mid-way between the minimum variance and the maximum Sharpe ratio. We could select a target return mean or a target return variance. We could a diversification penalty to our objective function or adjust the maximum position bounds in order to encourage greater diversification.

We could employ a number of different approaches to estimating our *ex post* estimates, *e.g.*, the James-Stein shrinkage estimates, the estimates generated from different factor models, the estimates based on combining multiple methods, or the test multiple estimates using techniques such as bootstrapping.

Investigating these alternatives certainly involves a great deal of work. The basic framework of the problem is relatively simple. We can think of the skeleton of our portfolio optimization problem as having the follow design decisions:

- The construction of our objective function $f : \{\mathbf{x}, \boldsymbol{\Omega}\} \xrightarrow{f} \mathbb{R}^1$.
  - Minimize variance penalized by a weighed return: $\frac{1}{2} \mathbf{x}^T \boldsymbol{\Sigma}\, \mathbf{x} - \lambda\, \boldsymbol{\mu}^T \mathbf{x}$.
  - Minimize variance (either to solve for the min variance portfolio or with a mean target constraint).
  - Build the efficient frontier and select an efficient solution that satisfies certain goals, *e.g.*, the maximum Sharpe ratio.
  - Adding additional component to the basic objective function, such as a penalty to encourage diversification.
  - Maximize mean return (typically with a variance target constraint).
  - Using a more exotic objective function such as portfolio VaR or CVaR.
- The selection of the distribution $\mathcal{D}$ of the returns over n assets, the random variable $R \in \mathbb{R}^n$, as parameterized by $\boldsymbol{\Omega}$: $R \approx \mathcal{D}[\mathbf{r} \mid \Omega]$.
  - The most common choice is a Multivariate Normal distribution.
  - Building an elliptical multivariate distribution out of non-Normal marginal distributions.
  - Using copulas to build a multivariate distribution out of non-Normal marginal distributions.
  - Using α-Stable or tempered α-Stable distributions that have been found to fit financial returns better than the Normal.
  - Using a finite mixture of multivariate distributions. This can be combined with any of the distributional choices above. *Question: If you have a finite mixture of multivariate Normal distributions and an allocation vector* $\mathbf{x}$*, then how does one compute the univariate distribution of portfolio returns that are the result of this allocation?*
- The method $\mathcal{M}$ employed in the statistical estimation of the parameters given a history matrix $\mathbf{R} \in \mathrm{M}^{T \times n}$ of T observations each of dimension n: $\mathbf{R} \xrightarrow{\mathcal{M}} \Omega$.
  - Traditional statistical techniques, based on MLE or related techniques, are able to directly estimate the parameters.
  - More complex approaches that are able to handle missing data or more complex MLE that require a numerical technique such as the EM algorithm.

- Cases beyond numerical approaches such as the EM algorithm which require simulation, such as Markov chain Monte Carlo (MCMC).
- Bayesian estimation, shrinkage estimates, combining independent estimate approaches with an eye towards improving *ex ante* performance.

- The set of constraints C to which we restrict our valid portfolios. This is meant to include any lower and upper bound constraints.
  - Constraints is tied closely to the choice of objective function.
  - Operational constraints and bounds that encode trading or policy restrictions, *e.g.*, implementing a no-short condition, limiting the maximum position allowed in a given position. or limiting the total amount invested in a given investment sector.
  - Accounting constraints which enforce certain necessary connections between variables, *e.g.*, the accounting constraints which relate portfolio positions to factor exposures.

We can think of the skeleton of our portfolio optimization problem (with "opt" representing the natural choice of maximize or minimize depending on our objective function) as having the abstract form:

$$\mathbf{x}^* = \underset{\mathbf{x}}{\text{argopt}} \left\{ \text{opt } f[\mathbf{x} \mid \mathbf{\Omega}] \;\middle|\; \mathbf{x} \in C \right\}$$

The Simulation (Again)

Other Considerations and Approaches

One could imagine any number of experiments based on different approaches to computing *ex post* statistics; e.g., shrinkage estimates, weighted combinations of multiple estimation approaches, or more complex factor models. We could also explore different approaches to setting up the optimization, e.g., quadratic penalties to increase diversification and tighter upper position bounds. Each such approach gives us another *ex post* solution that we can evaluate *ex ante*.

A Simulator with Statistical Analysis (Max Sharpe)

Code

In[•]:=
```
<< Local`QuadraticProgramming`
```

```
In[•]:= xMaxSharpeAlloc[vnMean_, mnCov_, nRiskFree_] := Module[
          {mnEfficientFrontier, nMaxRet, nMinVarRet, iN, iTangent},
          iN = Length[vnMean];
          nMinVarRet = vnMean.Last@xQuadraticProgramming[
              {{}, 0.5 mnCov}, {Array[1. &, iN]}, {{1, 0}}, Table[{0., 1.}, {iN}]];
          nMaxRet = Max[vnMean];
          mnEfficientFrontier = Table[Last@xQuadraticProgramming[{{}, 0.5 mnCov},
              {Array[1. &, iN], vnMean}, {{1, 0}, {nTarget, 1}}, Table[{0., 1.}, {iN}]],
            {nTarget, nMinVarRet, 0.99 nMaxRet, (nMaxRet - nMinVarRet) / 19}];
          iTangent = First@First@Position[#, Max[#]] &[
            (#[[2]] - nRiskFree) / #[[1]] & /@ mnEfficientFrontier];
          mnEfficientFrontier[[iTangent]]
         ];

In[•]:= xDiversification = Length[#]^-1/2 Total[#] / Norm[#] &;
```

Parameters

```
In[•]:= nDoF = 3.5;
        iNoA = 100;
        iNoH = 10 × 12;
        iNoF = 12;
        nMktMean = 0.0075;
        nMktVar = 0.0025;
        nRiskFree = 0.0004;
```

Simulation

```
In[•]:= iSamples = 25;

In[•]:= Print["Begin >>> ", DateString[]];
        anSimMaxSharpe = Table[
          Print["Starting iteration ", i, " of ", iSamples, " >>> ", DateString[]];
          vnBeta = RandomReal[{0.5, 1.5}, iNoA];
          distD = TransformedDistribution[2 nMktVar / 5 x, x ≈ ChiSquareDistribution[5]];
          (* T *)
          vnErrVar = RandomVariate[distD, iNoA];
          vnMean = nRiskFree + vnBeta (nMktMean - nRiskFree);
          mnCov = nMktVar KroneckerProduct[vnBeta, vnBeta] + DiagonalMatrix[vnErrVar];
          vnAllocT = xMaxSharpeAlloc[vnMean, mnCov, nRiskFree];
          vnSummaryT =
           {Sqrt[#.mnCov.#], vnMean.#, (vnMean.# - nRiskFree) / Sqrt[#.mnCov.#], xDiversification[#]} &[vnAllocT];
```

```
    (* ℋ *)
    vnMktRetnℋ =
     RandomReal[StudentTDistribution[nMktMean, Sqrt[(nDoF - 2)/nDoF] nMktVar], nDoF], iNoH];
    nMktMeanℋ = Mean[vnMktRetnℋ];
    nMktVarℋ = Variance[vnMktRetnℋ];
    mnRetnℋ = Transpose[Table[
       nRiskFree + vnBeta[[iWho]] (vnMktRetnℋ - nRiskFree) +
        RandomReal[StudentTDistribution[0, Sqrt[(nDoF - 2)/nDoF] vnErrVar[[iWho]]], nDoF], iNoH],
       {iWho, 1, iNoA}
      ]];
    vnMeanℋ = Mean[mnRetnℋ];
    mnCovℋ = Covariance[mnRetnℋ];
    vnAllocℋ = xMaxSharpeAlloc[vnMeanℋ, mnCovℋ, nRiskFree];
    vnSummaryℋ = {Sqrt[#.mnCovℋ.#], vnMeanℋ.#,
         (vnMeanℋ.# - nRiskFree)/Sqrt[#.mnCovℋ.#], xDiversification[#]} &[vnAllocℋ];
    (* 𝒜 *)
    (* --- generate returns *)
    vnMktRetn𝒜 =
     RandomReal[StudentTDistribution[nMktMean, Sqrt[(nDoF - 2)/nDoF] nMktVar], nDoF], iNoF];
    mnRetn𝒜 = Transpose[Table[
       nRiskFree + vnBeta[[iWho]] (vnMktRetn𝒜 - nRiskFree) +
        RandomReal[StudentTDistribution[0, Sqrt[(nDoF - 2)/nDoF] vnErrVar[[iWho]]], nDoF], iNoF],
       {iWho, 1, iNoA}
      ]];
    (* --- estimate parameters *)
    vnMean𝒜 = Mean[mnRetn𝒜];
    mnCov𝒜 = Covariance[mnRetn𝒜];
    (* --- compute sumary *)
    vnSummary𝒜 = {Sqrt[#.mnCov𝒜.#], vnMean𝒜.#,
         (vnMean𝒜.# - nRiskFree)/Sqrt[#.mnCov𝒜.#], xDiversification[#]} &[vnAllocℋ];
    (* experiment result *)
    {vnSummaryℋ, vnSummary𝒯, vnSummary𝒜},
    {i, 1, iSamples}
   ];
Print["End >>> ", DateString[]];
```

```
Begin >>> Sat 24 Mar 2018 18:36:36
```

```
Starting iteration 1 of 25 >>> Sat 24 Mar 2018 18:36:36
Starting iteration 2 of 25 >>> Sat 24 Mar 2018 18:37:35
Starting iteration 3 of 25 >>> Sat 24 Mar 2018 18:38:34
Starting iteration 4 of 25 >>> Sat 24 Mar 2018 18:39:34
Starting iteration 5 of 25 >>> Sat 24 Mar 2018 18:40:34
Starting iteration 6 of 25 >>> Sat 24 Mar 2018 18:41:34
Starting iteration 7 of 25 >>> Sat 24 Mar 2018 18:42:36
Starting iteration 8 of 25 >>> Sat 24 Mar 2018 18:43:37
Starting iteration 9 of 25 >>> Sat 24 Mar 2018 18:44:38
Starting iteration 10 of 25 >>> Sat 24 Mar 2018 18:45:38
Starting iteration 11 of 25 >>> Sat 24 Mar 2018 18:46:37
Starting iteration 12 of 25 >>> Sat 24 Mar 2018 18:47:37
Starting iteration 13 of 25 >>> Sat 24 Mar 2018 18:48:36
Starting iteration 14 of 25 >>> Sat 24 Mar 2018 18:49:36
Starting iteration 15 of 25 >>> Sat 24 Mar 2018 18:50:36
Starting iteration 16 of 25 >>> Sat 24 Mar 2018 18:51:35
Starting iteration 17 of 25 >>> Sat 24 Mar 2018 18:52:34
Starting iteration 18 of 25 >>> Sat 24 Mar 2018 18:53:32
Starting iteration 19 of 25 >>> Sat 24 Mar 2018 18:54:32
Starting iteration 20 of 25 >>> Sat 24 Mar 2018 18:55:32
Starting iteration 21 of 25 >>> Sat 24 Mar 2018 18:56:31
Starting iteration 22 of 25 >>> Sat 24 Mar 2018 18:57:31
Starting iteration 23 of 25 >>> Sat 24 Mar 2018 18:58:30
Starting iteration 24 of 25 >>> Sat 24 Mar 2018 18:59:33
Starting iteration 25 of 25 >>> Sat 24 Mar 2018 19:00:34
End >>> Sat 24 Mar 2018 19:01:34
```

Statistical Analysis

Note the dimensions of the simulation are: *run_number* $\times \{\mathcal{H}, \mathcal{T}, \mathcal{A}\} \times \{\mu, \sigma, \psi, \delta\}$

In[•]:=
```
Dimensions[anSimMaxSharpe]
```

Out[•]=
```
{25, 3, 4}
```

Most statistical tools that take vectors and return a scalar result work along the first dimension, Thus, Mean[anSim] yields a matrix of means for $\{\mathcal{H}, \mathcal{T}, \mathcal{A}\} \times \{\mu, \sigma, \psi, \delta\}$.

Here are the sample means

```
In[•]:= TableForm[Mean[anSimMaxSharpe],
          TableHeadings → {{"ℋ", "𝒯", "𝒜"}, {"μ", "σ", "ψ", "δ"}}]
```

Out[•]//TableForm=

	μ	σ	ψ	δ
$\mathcal{H}$	0.0352555	0.00731351	0.182003	0.310772
$\mathcal{T}$	0.0502634	0.00731623	0.136291	0.435594
$\mathcal{A}$	0.0448618	0.000910393	0.0819375	0.310772

Here are the sampling errors (sample standard deviation) of the means.

```
In[•]:= TableForm[StandardDeviation[anSimMaxSharpe] / √First@Dimensions[anSimMaxSharpe],
          TableHeadings → {{"ℋ", "𝒯", "𝒜"}, {"μ", "σ", "ψ", "δ"}}]
```

Out[•]//TableForm=

	μ	σ	ψ	δ
$\mathcal{H}$	0.00327645	0.00147772	0.0241836	0.0140656
$\mathcal{T}$	0.00351667	0.000517053	0.000947282	0.0291081
$\mathcal{A}$	0.00691279	0.0019659	0.0574867	0.0140656

Note that, *e.g.*, anSim⟦All, 2, 3⟧ gives the vector of all of $\mathcal{T}$ results' Sharpe ratios. Use can use such indexing in designing more focused analyses.

Some Interesting Questions

- Copy the simulation code, the section "E[$\mu_{\mathcal{H}}$] ≥ $\mu_{\mathcal{T}}$ ≥ E[$\mu_{\mathcal{A}}$] Revisited" into a new notebook and rerun the simulation with the same parameters.
- Given the null hypothesis H_0; E[$\psi_{\mathcal{H}}$] − E[$\psi_{\mathcal{A}}$] ≥ 0, is H_0 rejected or not rejected at a 95% confidence level?
- Given the simulation results, what is the probability that $\psi_{\mathcal{H}} < \psi_{\mathcal{A}}$ for a randomly chosen simulation run.
- Plot the sample PDFs of $\mu_{\mathcal{H}} - \mu_{\mathcal{A}}$, $\sigma_{\mathcal{H}} - \sigma_{\mathcal{A}}$, $\psi_{\mathcal{H}} - \psi_{\mathcal{A}}$, and $\delta_{\mathcal{H}} - \delta_{\mathcal{T}}$. Use Histogram[< *data* >, Automatic, PDF].
- Was iSamples = 25 too low?