package: rrv 0.4.0

Random Return Variables (Rewrite, Rough Draft)

Charlotte Maia

September 12, 2011

This vignette gives an overview of the rrv package. The package is partly based on Markowitz (1952), however considers empirical distributions. There's a strong emphasis on modelling conditional portfolio returns as functions of weight. Various "conditional parameters" are considered, including expected returns and quantile returns.

Introduction

The rrv package provides an object oriented framework for modelling portfolio returns as random variables, with empirical distributions. It's partly based on Markowitz (1952), who provides a theoretical framework for modelling portfolio returns as random variables, with normal-like distributions.

The author defines a portfolio as a set of sources and a source as any process that generates return. In general, sources are assumed to be weighted, with weights summing to one. Under this assumption, a portfolio's return is a weighted average of it's source returns.

Currently, the package has the following over-arching goals:

1. To model portfolio returns as random variables.
2. To model conditional portfolio returns as functions of weight.
3. To consider various kinds of portfolios.
4. To consider portfolio optimisation.

The package is based on object oriented principles and contains two main classes of objects. Firstly, random return variable (rrv) objects, representing unknown future returns, which are constructed from known historical returns. Secondly, conditional portfolio return (cpr) objects/functions, representing portfolio returns as functions of weight, which are also constructed from historical returns.

Future versions may also contain a portfolio class with summary methods.

The term "conditional portfolio returns" is used ambiguously and can refer to conditional random variables (random variables with conditional distributions) or conditional parameters. The term "conditional parameters" is used to describe parameters of conditional random variables, which can include expected values, variances and quantiles.

Getting Started

First, we need to load the rrv package:

```
> library (rrv)
```

The package makes use of the dataset from Markowitz (1959), which gives discounted returns for nine securities over an eighteen year period.

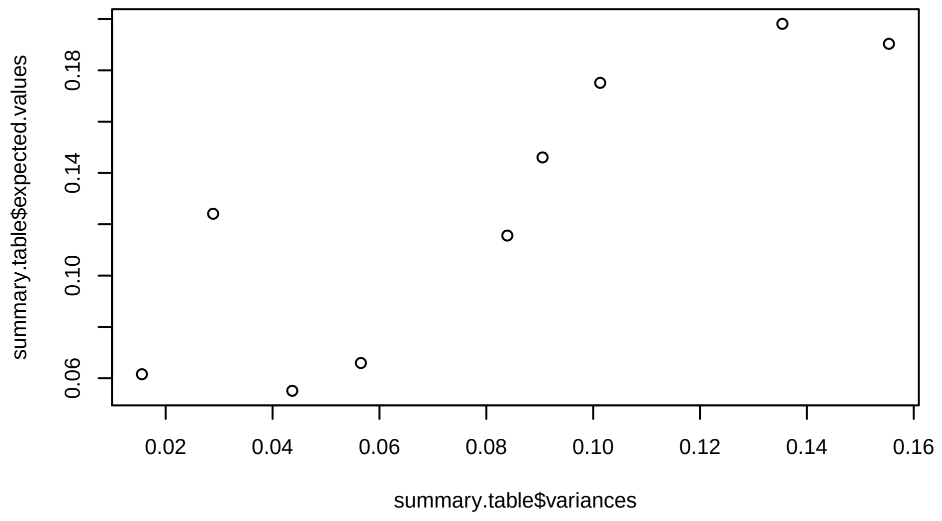
```
> data = read.package.data ("rrv", "markowitz.csv")
> samp (data)
   Year  Am.T. A.T. & T. U.S.S.   G.M. A.T. & Sfe   C.C.   Bdn. Frstn.   S.S.
1  1937 -0.305    -0.173 -0.318 -0.477     -0.457 -0.065 -0.319 -0.400 -0.435
2  1938  0.513     0.098  0.285  0.714      0.107  0.238  0.076  0.336  0.238
3  1939  0.055     0.200 -0.047  0.165     -0.424 -0.078  0.318 -0.093 -0.295
16 1952  0.128     0.083  0.131  0.390      0.434  0.079  0.109  0.175  0.062
17 1953 -0.010     0.035  0.006 -0.072     -0.027  0.067  0.210 -0.084 -0.048
18 1954  0.154     0.176  0.908  0.715      0.469  0.077  0.112  0.756  0.185
```

Consistent with Markowitz, lets consider expected values and variances.

```
> summary.table = data.frame (expected.values=apply (data [,-1], 2, mean),
        variances=apply (data [,-1], 2, var) )
> summary.table = summary.table [order (summary.table [,1], decreasing=TRUE),]
> round (summary.table, 3)
           expected.values variances
A.T. & Sfe           0.198     0.135
Frstn.               0.190     0.155
G.M.                 0.175     0.101
U.S.S.               0.146     0.091
Bdn.                 0.124     0.029
S.S.                 0.116     0.084
Am.T.                0.066     0.057
A.T. & T.            0.062     0.016
C.C.                 0.055     0.044
```

A plot of variance versus expected values, supports the cliche notion in finance, that high return requires high risk.

```
> plot (summary.table$variances, summary.table$expected.values)
```

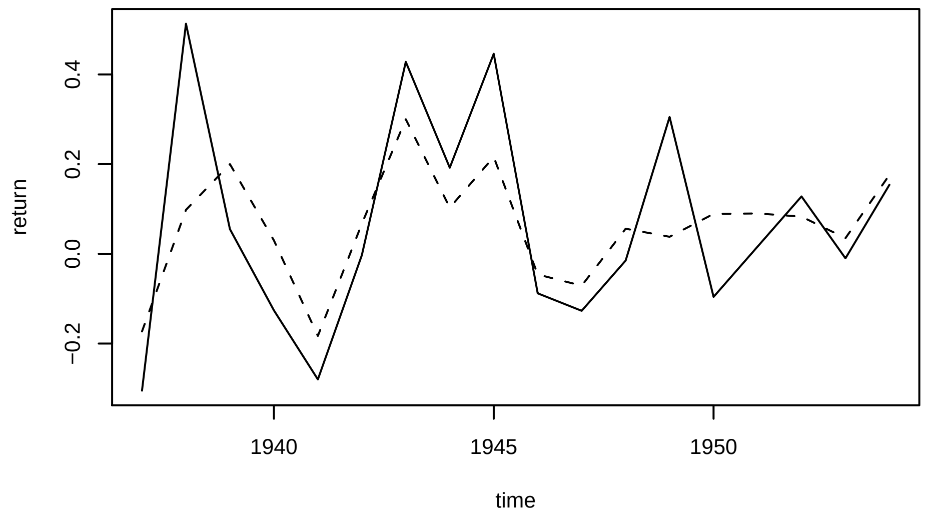


However, returns are often correlated, so lets consider pairwise correlations.

```
> correlation.matrix = cor (data [,-1])
> round (correlation.matrix, 1)
           Am.T. A.T. & T. U.S.S. G.M. A.T. & Sfe C.C. Bdn. Frstn. S.S.
Am.T.        1.0       0.8    0.4  0.7        0.2  0.7  0.6    0.5  0.6
A.T. & T.    0.8       1.0    0.5  0.7        0.2  0.4  0.7    0.5  0.6
U.S.S.       0.4       0.5    1.0  0.7        0.4  0.2  0.2    0.6  0.5
G.M.         0.7       0.7    0.7  1.0        0.5  0.5  0.4    0.8  0.4
A.T. & Sfe   0.2       0.2    0.4  0.5        1.0  0.2  0.4    0.7  0.4
C.C.         0.7       0.4    0.2  0.5        0.2  1.0  0.4    0.4  0.4
Bdn.         0.6       0.7    0.2  0.4        0.4  0.4  1.0    0.5  0.4
Frstn.       0.5       0.5    0.6  0.8        0.7  0.4  0.5    1.0  0.5
S.S.         0.6       0.6    0.5  0.4        0.4  0.4  0.4    0.5  1.0
```

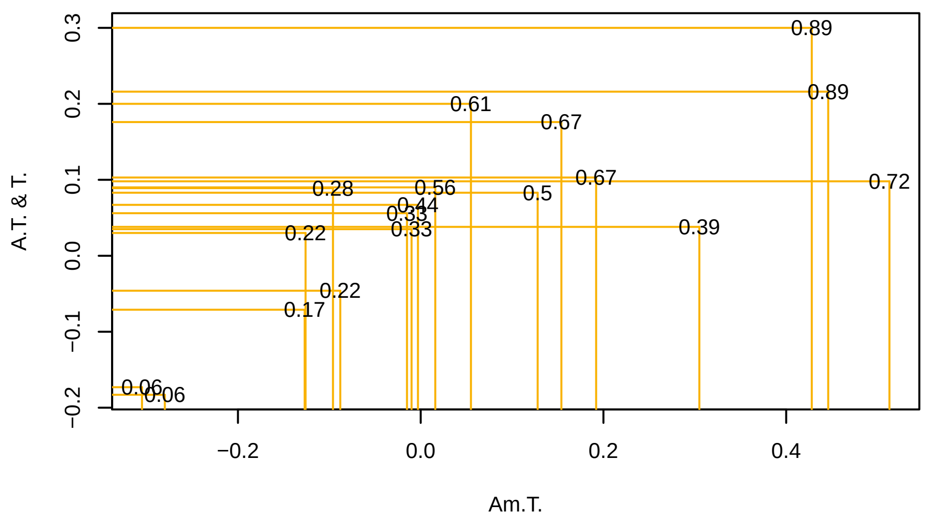
An important feature of this data, is that all pairwise combinations are positively correlated. Using the first two sources (which are strongly correlated), we can create a timeseries plot.

```
> time = data [,1]
> plot (time, data [,2], type="l", ylab="return")
> lines (time, data [,3], lty=2)
```



Alternatively, we can create a bivariate rrv object (from the data) and plot it.

```
> x = rrv (data [,2:3])
> plot (x)
```



Source returns are discussed in more detail later.

Conditional Portfolio Return

For a given portfolio, we shall denote source returns as $\mathbf{X}$ or $(X_1, X_2, ..., X_k)$, portfolio return as Y and weights as $\mathbf{w}$ or $(w_1, w_2, ...w_k)$. Both portfolio returns and source returns are random variables. k is used to denote the number of sources. Portfolio return is weighted sum of source returns, so:

$$\begin{aligned} Y &= \mathbf{wX} \\ &= w_1 X_1 + w_2 X_2 + ... + w_k X_k \end{aligned}$$

The rrv package implements cpr_rrv objects, for this purpose. A cpr_rrv object, follows some object oriented conventions and is created using a constructor. However, these objects are also functions and map a vector of weights to an rrv object. If we denote our constructor as G, our cpr_rrv object as f and our data (a matrix or an rrv object) as $\mathbf{x}$, then we have:

$$\begin{aligned} \mathrm{G} &: \mathbf{x} \mapsto \mathrm{f} \\ \mathrm{f} &: \mathbf{w} \mapsto Y \end{aligned}$$

Whilst this may seem cumbersome at first, it's important to note that the function f, only requires one argument, weight.

Let's consider an example. In practice, the package doesn't use G, the constructor has the same name as the class of the object so:

```
> #construct the object (also a function)
> f = cpr_rrv (x)
> f
FUNCTION (w)
{  rprv(.$x, w)
}
object attributes:
nv, names, x
```

We can evaluate this object, however rrv and rprv objects are discussed later, so the output probably won't make much sense at this stage.

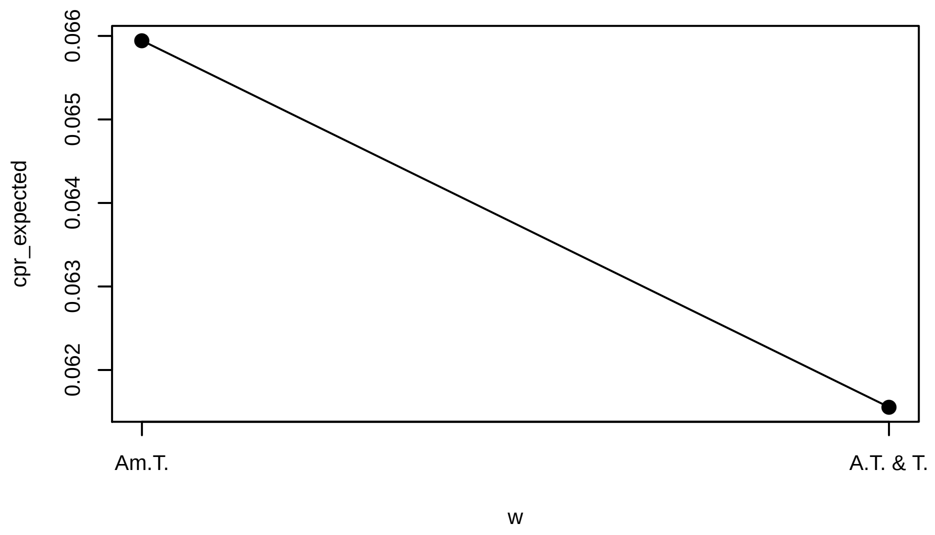
The rrv package also implements cppr objects. Which are similar to cpr_rrv objects, however simpler and more practical. Instead of mapping a vector of weights to a rrv object, they make a vector of weights to what we shall regard as a conditional parameter, such as expected return. There are a few subclasses, each with a different constructor.

Let's consider another example:

```
> #construct the object (also a function)
> f_expected = cpr_expected (x)
> f_expected
FUNCTION (w)
{  sum(w * .$x.mean)
}
object attributes:
nv, names, x.mean
```

We can plot and evaluate the object:

```
> #plot of expected return as function of weight
> plot (f_expected)
```

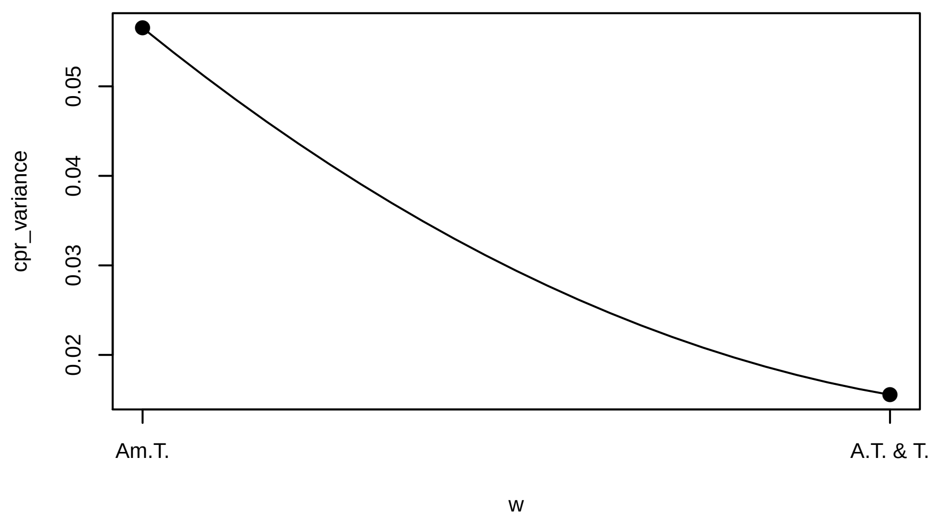


```
> #evaluate it, for an equally weighted portfolio
> f_expected (c (0.5, 0.5) )
[1] 0.06375
```

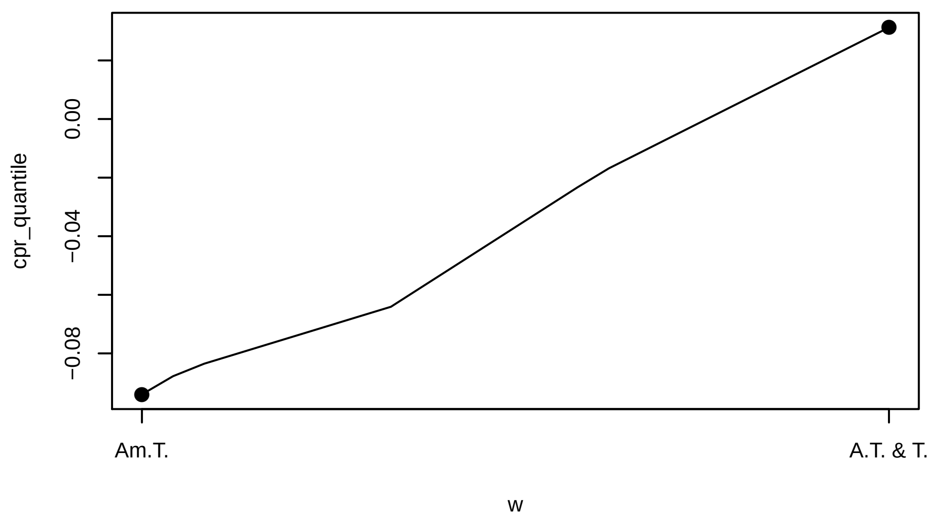
Similarly for variance and quantiles, noting the cpr_quantile constructor has an extra argument.

```
> f_variance = cpr_variance (x)
> f_q0.25 = cpr_quantile (0.25, x)
```

```
> plot (f_variance)
```



```
> plot (f_q0.25)
```



There are also a cpr_median and cpr_sd classes.

Conditional Portfolio Return for 3-Source Portfolios

In general, weight represents a triangle (potentially a hyper-triangle). This is the most intuitive in the 3-source case, where triangular heatmaps, show the value of a conditional parameter given a vector of 3 weights. In future contour plots may be used instead.

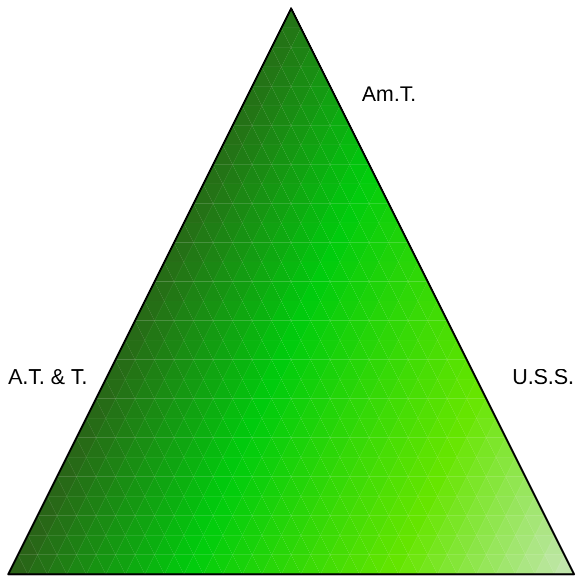
Note that each plot, uses a separate scale (in order to achieve maximum colour variation). Hence, two points with the same colour, each from a separate plot, may represent quite different values.

```
> x = as.matrix (data [,2:4])
> samp (x)
       Am.T. A.T. & T. U.S.S.
[1,] -0.305    -0.173 -0.318
```

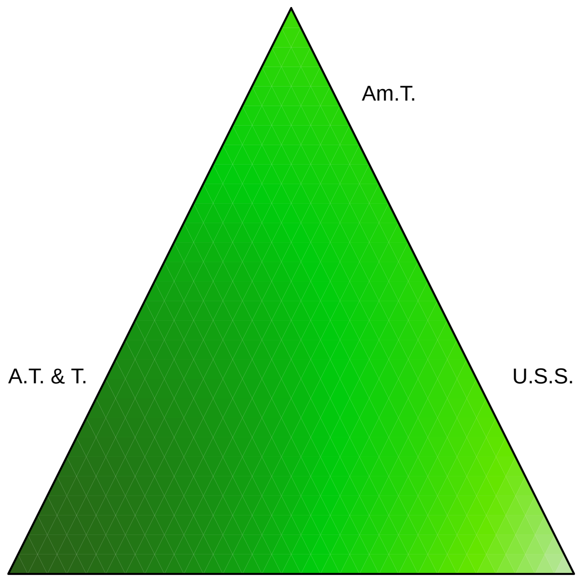
```
[2,]  0.513     0.098  0.285
[3,]  0.055     0.200 -0.047
[4,]  0.128     0.083  0.131
[5,] -0.010     0.035  0.006
[6,]  0.154     0.176  0.908
```

```
> f_expected = cpr_expected (x)
> f_variance = cpr_variance (x)
> f_q0.25 = cpr_quantile (0.25, x)
> f_q0.50 = cpr_quantile (0.50, x)
> f_q0.75 = cpr_quantile (0.75, x)
```

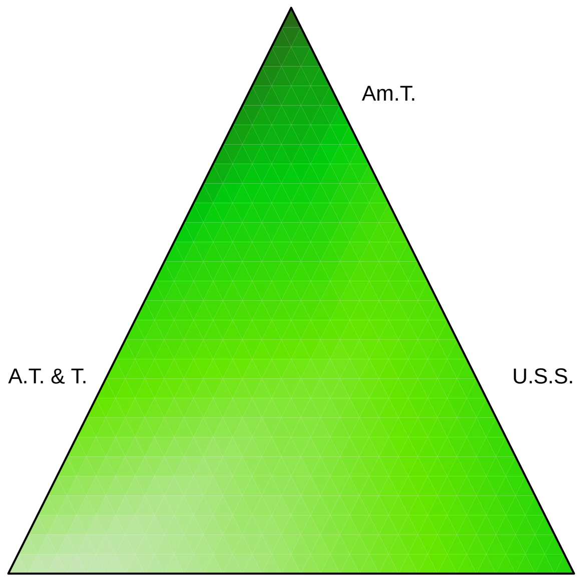
```
> #expected portfolio return
> plot (f_expected)
```



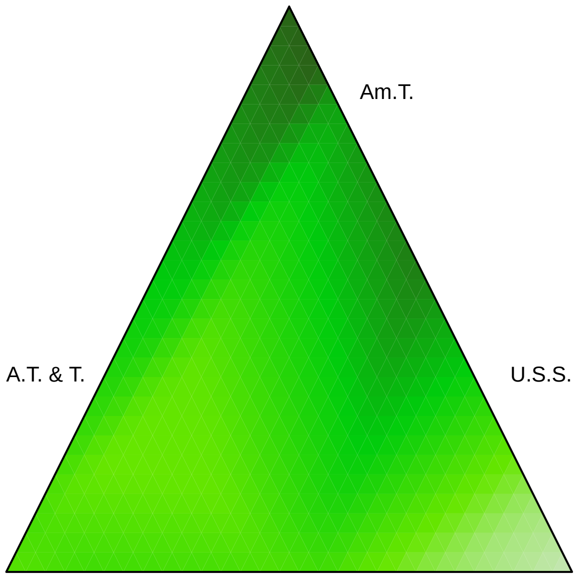
```
> #portfolio return variance
> plot (f_variance)
```



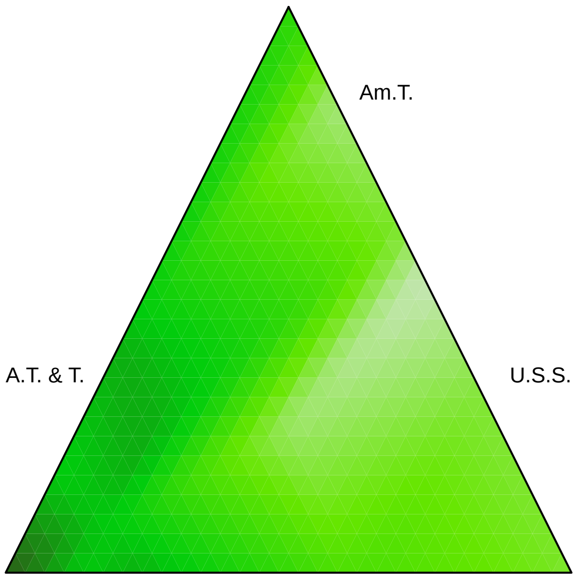
```
> #quantile (0.25) portfolio return
> plot (f_q0.25)
```



```
> #median portfolio return
> plot (f_q0.50)
```



```
> #quantile (0.75) portfolio return
> plot (f_q0.75)
```



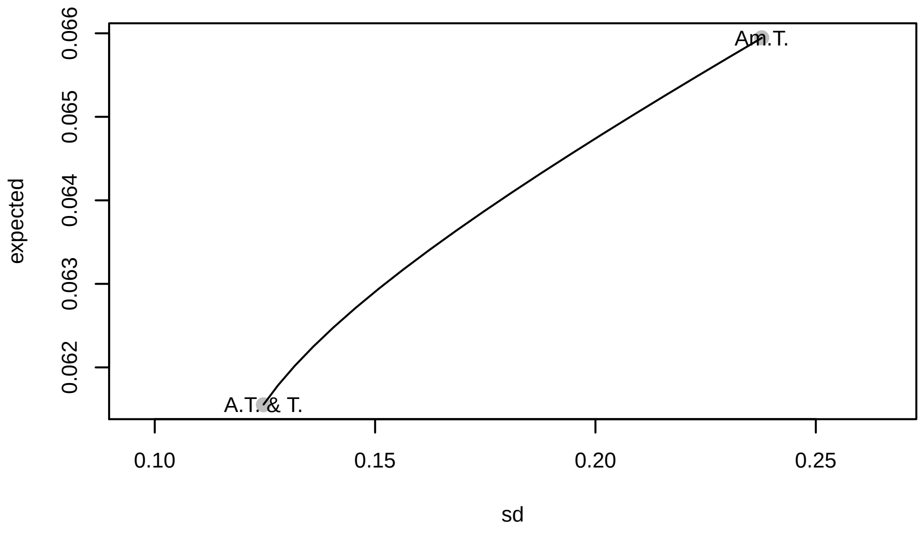
Text Book Plots

Up until now, we have regarded conditional portfolio returns as functions of weight, plus plotted them as functions. Most textbooks (on finance) plot standard deviation (or variance) versus expected return. A prototype function, unimaginatively named, .textbookplot (with a dot), was added to the package, just before releasing the current version.

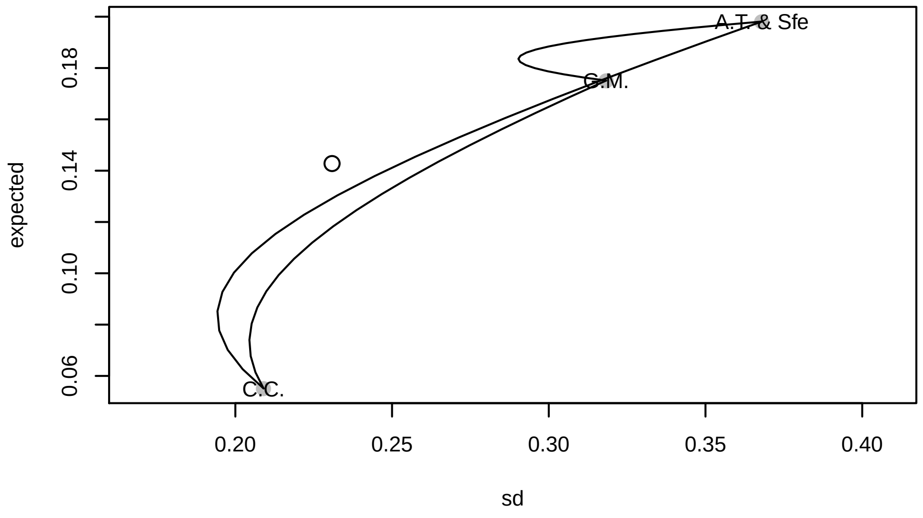
Currently, the plot produces incorrect output. For 3-source and 4-source portfolios, it plots 2-source subsets. Plus the function isn't tested well.

Despite these problems, we will still consider some examples.

```
> #2-source, first two variables
> .textbookplot (data [,2:3])
```



```
> #3-source, incorrect, giving 2-source subsets
> m = as.matrix (data [,5:7])
> .textbookplot (m)
> #superimpose equally-weighted portfolio
> w = c (1, 1, 1) / 3
> x = cpr_sd (m)(w)
> y = cpr_expected (m)(w)
> points (x, y, cex=1.5)
```



More on Source Returns

In mainstream probability, a random variable is characterised by parameters (which in turn characterise it's distribution). However here (with empirical distribution functions), a random variable characterised by its realisations.

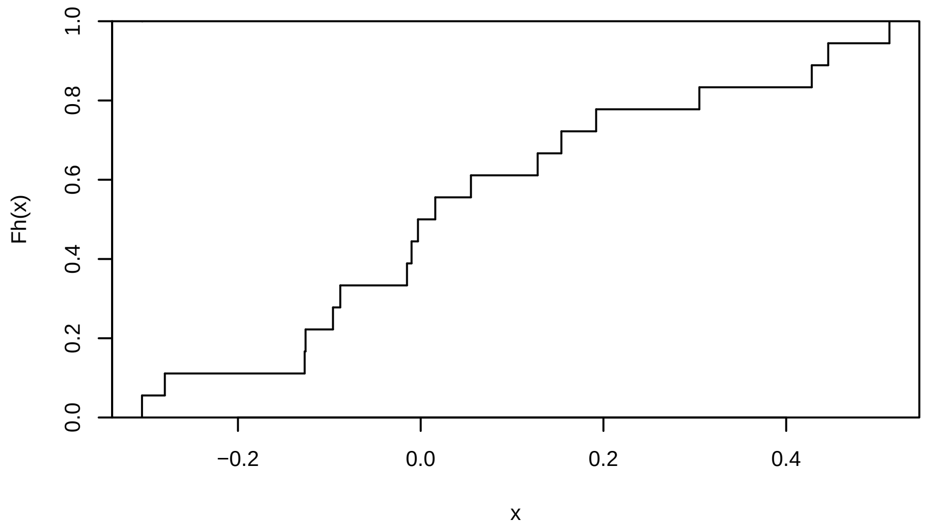
The rrv class, is used to represent random return variables, including random vector-return variables. An rrv object is simply an extended matrix, containing the realised values.

Currently, the package provides little support for modelling source returns.

As mentioned earlier, a portfolio class, may be added to future versions, for summary or exploratory purposes.

We can create rrv objects and plot them, here using an univariate example.

```
> r = rrv (data [,2])
> plot (r)
```



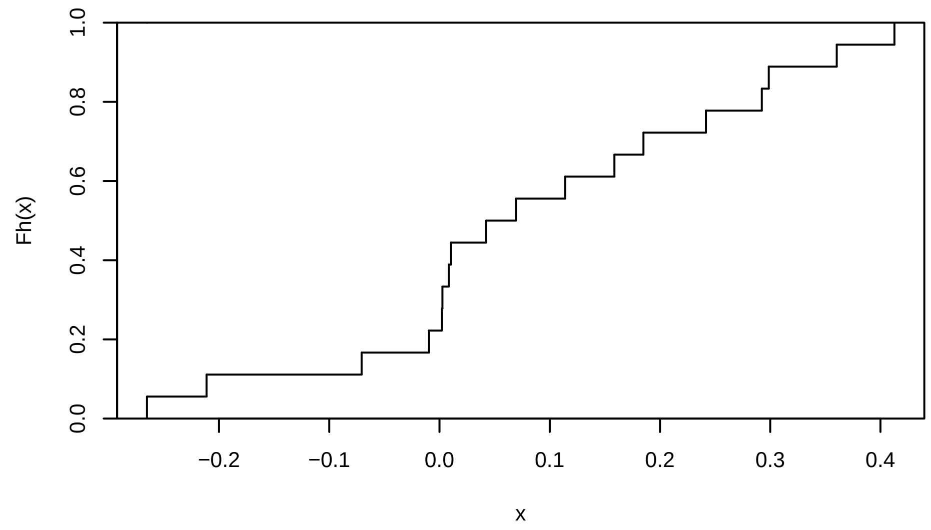
More on Portfolio Returns

The rrv class is extended by the rprv class, which is univariate version of rrv, created from an rrv object (or a matrix), along with vector of weights. Using the first three variables:

```
> y = rprv (as.matrix (data [,2:4]), c (1, 1, 1) / 3)
```

The object y represents a conditional random return variable or a random portfolio return variable. We can plot our rprv object, which looks similar to a univariate source return.

```
> plot (y)
```



The values of y are trivial to compute. Let $\mathbf{r}_i$ be the ith row vector of $\mathbf{x}$, then

$$\begin{aligned} y_1 &= \mathbf{w}\mathbf{r}_1 \\ y_2 &= \mathbf{w}\mathbf{r}_2 \\ &\vdots \\ y_n &= \mathbf{w}\mathbf{r}_n \end{aligned}$$

References

Markowitz, H.M. (1952). Journal of Finance. Portfolio Selection.
Markowitz, H.M. (1959). Cowles Foundation. Portfolio Selection Efficient Diversification of Investments.