

Portfolio Probe: Changing Fund Management



Patrick Burns
<http://www.burns-stat.com>

May 2010

This talk was given at the LondonR meeting 2010 May 04.

The website for the group is <http://www.londonR.org>

Portfolio Probe is commercial software produced by Burns Statistics.

Optimise

```
trade.optimizer(price.vector,  
variance.matrix, ...)
```

Here is a call to an R function to do a portfolio optimisation. The function takes a vector of prices for the assets, (possibly) a variance matrix, and a number of other arguments.

The additional arguments relate to:

Constraints

Utility

Trading costs

~~trade.optimizer~~(price.vector,
variance.matrix, ...)

The title talks about changing fund management, but this is quite standard stuff.

If we keep the same arguments but change the function that is used, then we do get something different.

Generate Random Portfolios

```
random.portfolio(number.rand,  
price.vector,  
variance.matrix, ...)
```

The `random.portfolio` function takes the same arguments as `trade.optimizer` plus two more: the number of portfolios that should be generated, and whether you want the portfolios or the trades in the result.

This will ignore the arguments pertaining to utility, and pay particular attention to the constraint arguments.

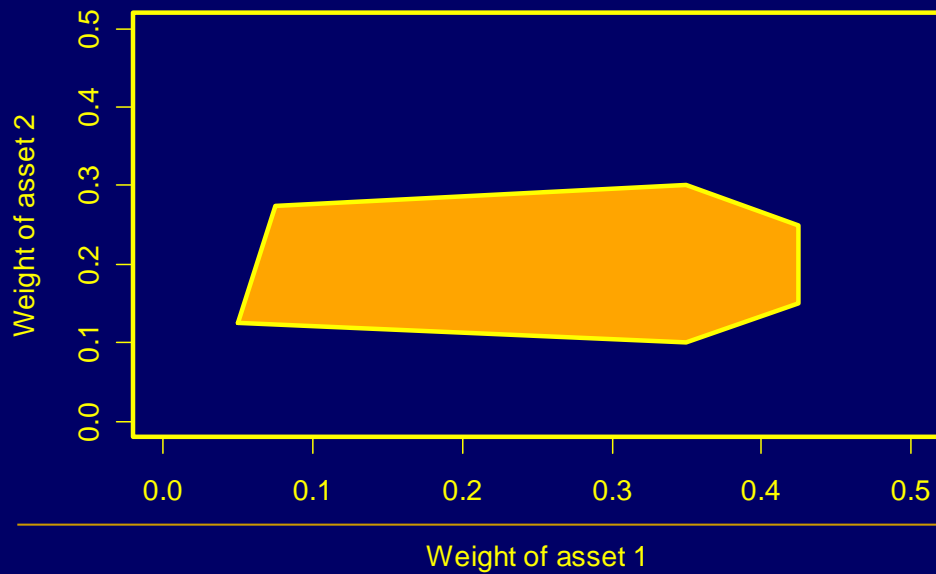


CONSTRAI_{nt}

The idea of random portfolios is very simple. We just take a random sample from the set of portfolios that obey a set of constraints.

(The not-so-simple part is how to actually do the sampling.)

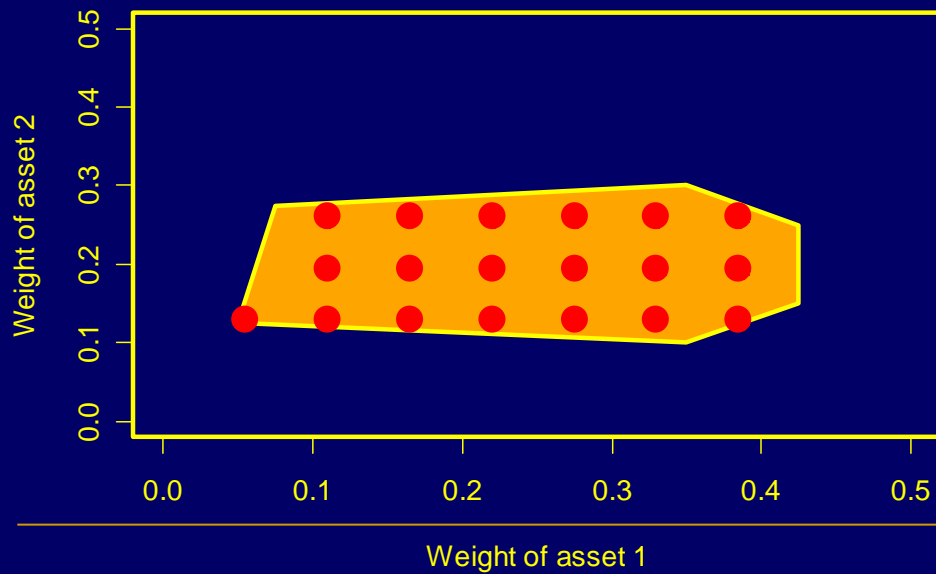
Constrained weights



Here is a picture from a three-asset problem. We want to sample from the orange area.

This picture implies that we only have linear constraints (there are only straight lines on the boundary). Constraints don't have to be linear.

Constrained weights



The picture also implies that we have a continuous space. In reality we are on a lattice because, for instance, you can not buy 2.68 shares of BP.

In practice it wouldn't matter if we were using random portfolios with continuous values or not. But when thinking about random portfolios it is sometimes easier to remember that we only have a finite number of choices.

Example Random Portfolio

[[1]]

asset23	asset12	asset37
14	-73	129

[[2]]

asset93	asset44	asset12
201	25	-68

Here is what a random portfolio might look like in R.

This will be a list with the length of the list equal to the number of portfolios that were generated.

Each component of the list will be a named numeric vector. The names are the asset ids, the numbers are the number of shares or lots or contracts or whatever.

This list will have some attributes, including a class.

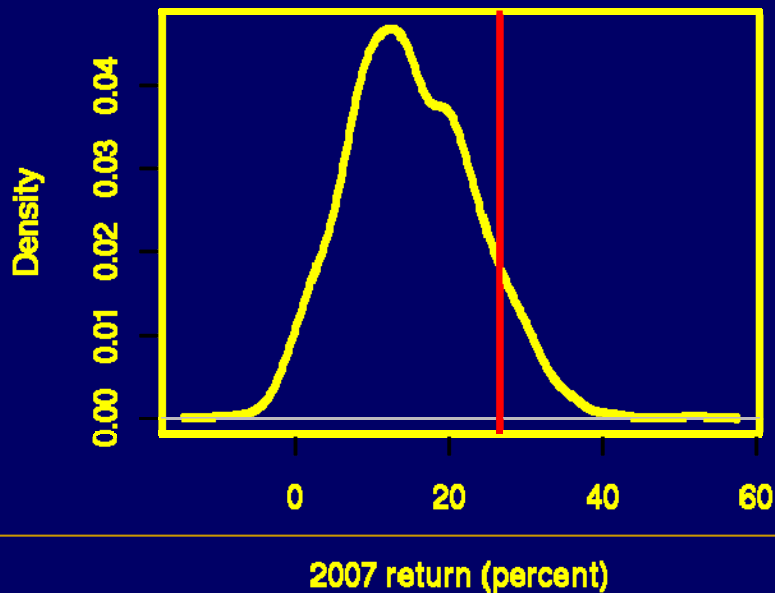
Changing Fund Management

- **Performance measurement**
- **Evaluating trading strategies**
- **Tracking error constraints**

There are more things that should be changed in fund management, and in fact there are more things in fund management that random portfolios can help change.

Performance Measurement

Simple Performance Measurement

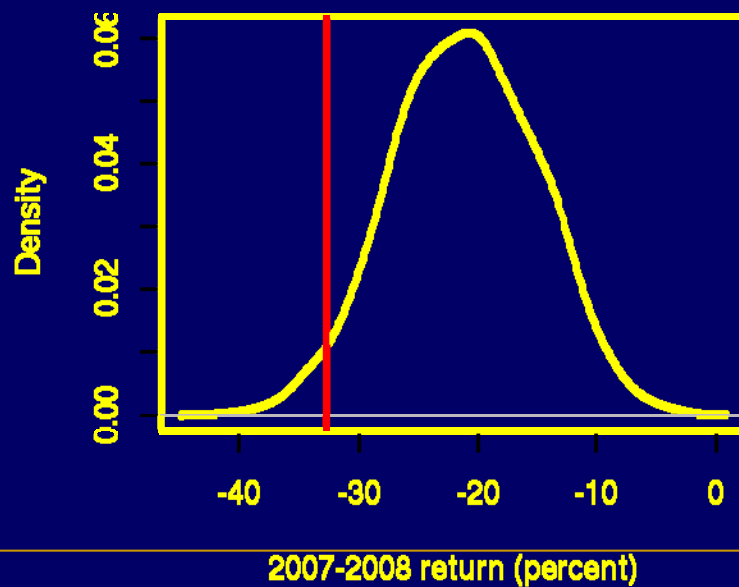


The performance measurement has a period of time of interest. In this case it is calendar year 2007.

At the beginning of the year we generate a number of random portfolios that obey the constraints of the fund. We hold those portfolios throughout the time period, calculate their returns, and find the density of the random portfolios. This gives us the distribution of having zero skill – in the vernacular: what would have happened just by chance. We then compare that density to the actual return of the fund.

In this case the fund is doing reasonably well – it is in the upper tail.

Simple Performance Measurement

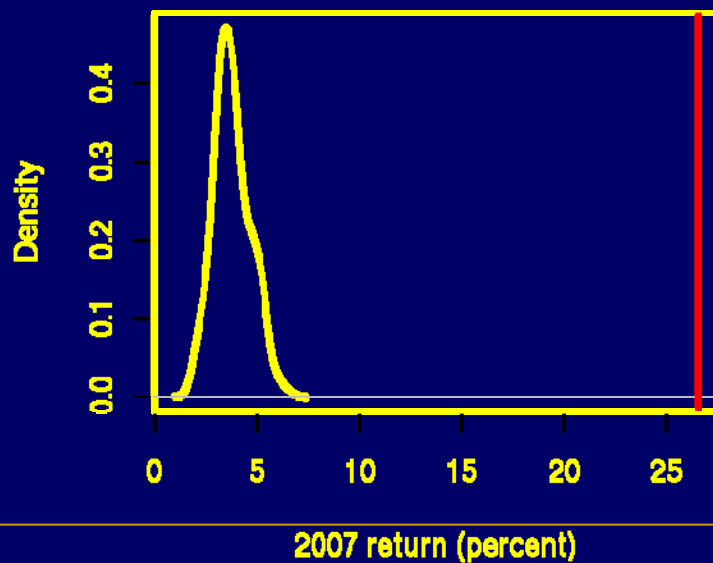


If we add a year to the time period, now our fund is doing reasonably badly – it is in the lower tail. We can infer that if we looked at just 2008, the fund would have done very badly.

This sort of performance measurement with random portfolios is decidedly better than how it is currently done with peer groups or with a benchmark.

Just on the basis of this form of using random portfolios, the other methods of performance measurement should be dropped. But there is another form of using random portfolios for performance measurement that is an even more dramatic improvement.

Smarter Performance Measurement



If we use more information, then we can get a tighter distribution of the portfolios with zero skill.

We sometimes know some or all of the positions of the fund at the beginning of the period along with a sense of the turnover of the fund.

In the example, we know all of the initial positions and that there is daily trading with annual turnover of 200% (buys plus sells).

We generate one random portfolio by starting with the initial portfolio and doing random trades throughout the year. We repeat this a number of times and then draw the density of the returns.

We can think of this as how the proverbial monkeys are doing. A typical monkey is getting a 4% return, a really good monkey gets 7%.

Test Trading Strategies

Backtest Results

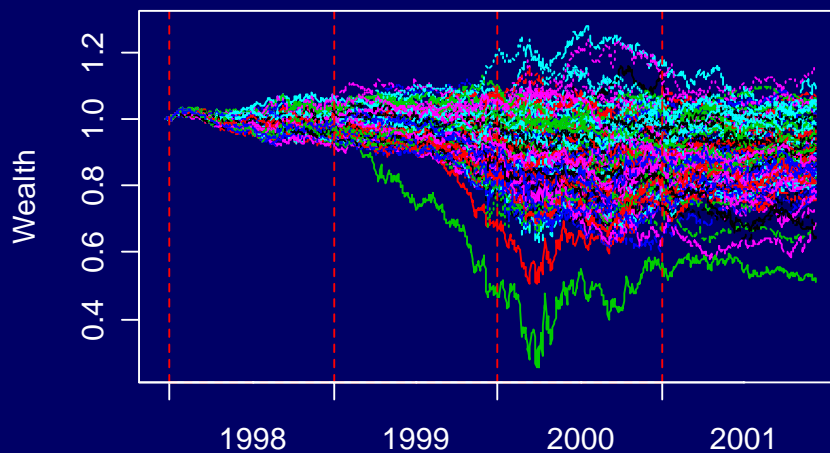


In a backtest we start with an initial portfolio, we then trade throughout the test period obeying the constraints we've set up and adding in the information that we think will make us beat the market.

In this case we have a dollar neutral strategy and all of the constraints are symmetric in terms of long and short. So being in positive territory means we're good. The question is: How good?

From what you're about to see, you should question the assumption that positive territory implies "good".

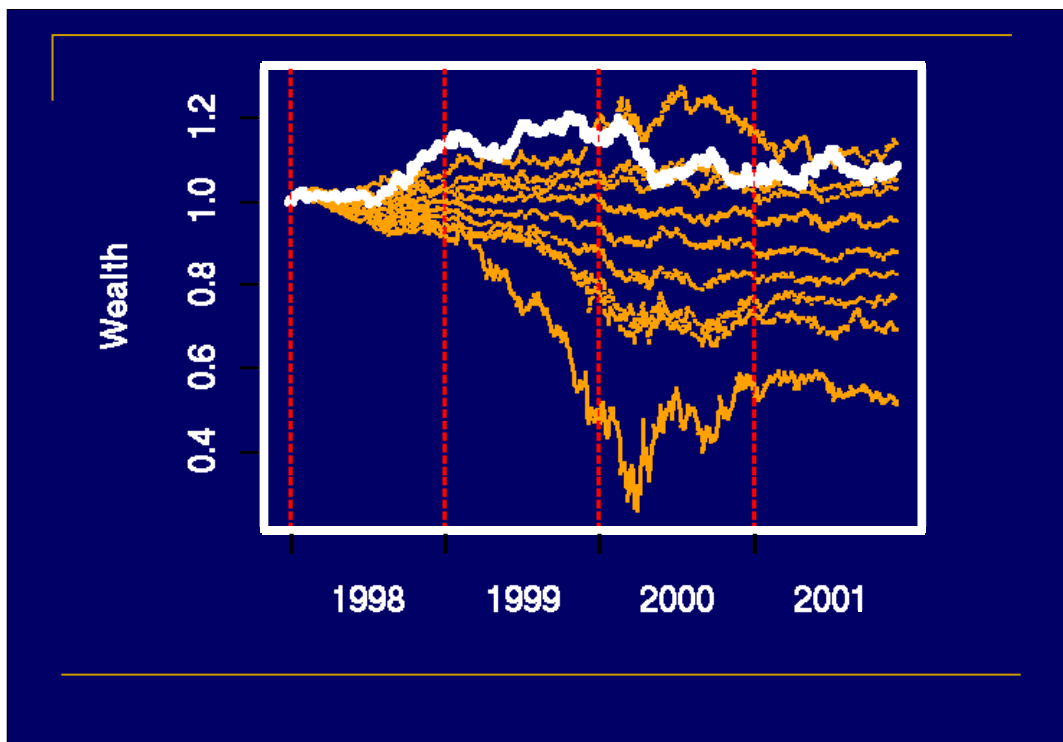
The Random Paths



To test this we do just the same thing as we did with the smart performance measurement. We start with the initial portfolio, then we trade just like the actual backtest trades, obeying the constraints, but paying no attention to things that might beat the market. We just trade randomly within the constraints.

Here we have 100 random paths that correspond to the backtest result we just saw.

Note that the paths are not centered on wealth 1 as we might expect. The initial portfolio performed very badly through this time period, and the random paths retain that for a surprisingly long time.

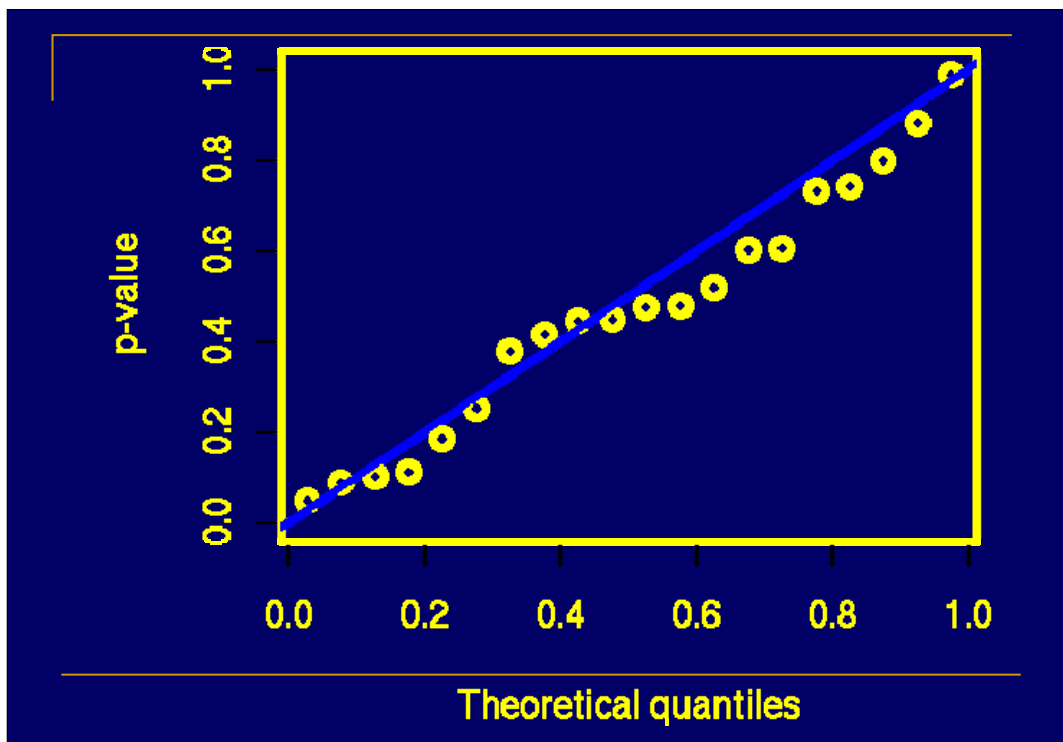


Here we have the original backtest result along with selected quantiles of the random paths at each point in time.

But for our evaluation of the backtest, we will only save one number from this: the fraction of random paths that end the period better than the actual backtest. In this case that is two or three out of 100. So the p-value is 2% or 3%. (If you want to be pedantic about it, it would be 3% or 4%.)

This tells us that this strategy over this period of time starting at that particular initial portfolio does pretty well. We have reasonable evidence that it is doing better than chance.

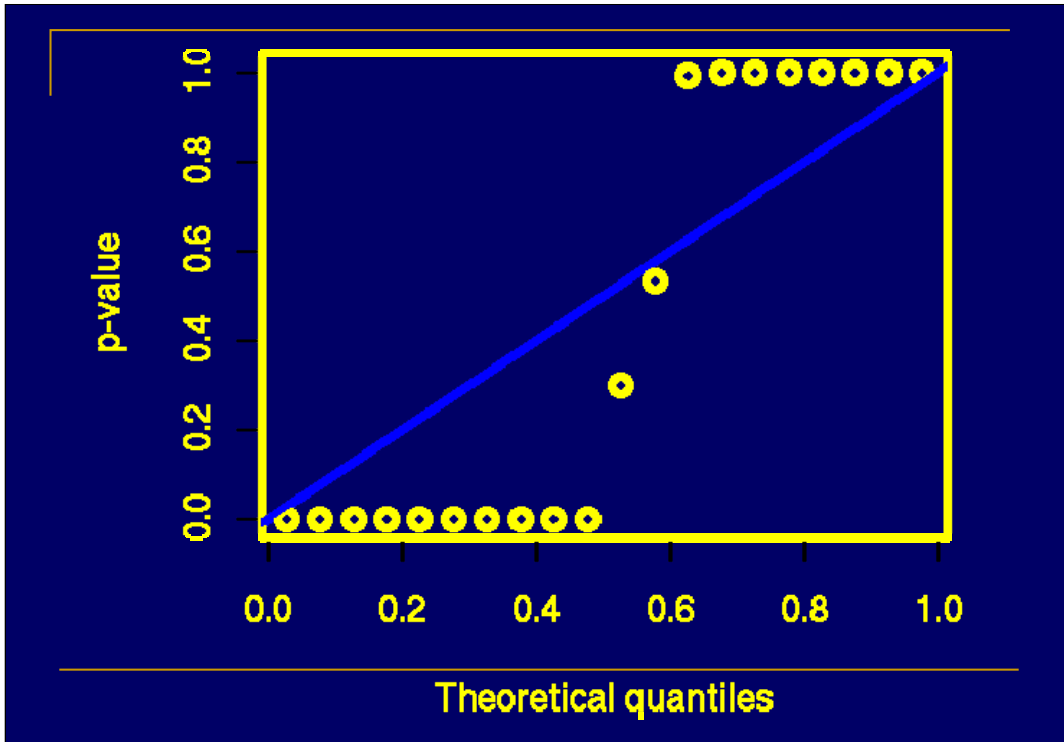
But what we care about is not starting at that particular initial portfolio, but starting anywhere. So we want to do this whole process a number of times.



If we did the process 20 times and created a QQ-plot of the p-values, then we might get a picture like this.

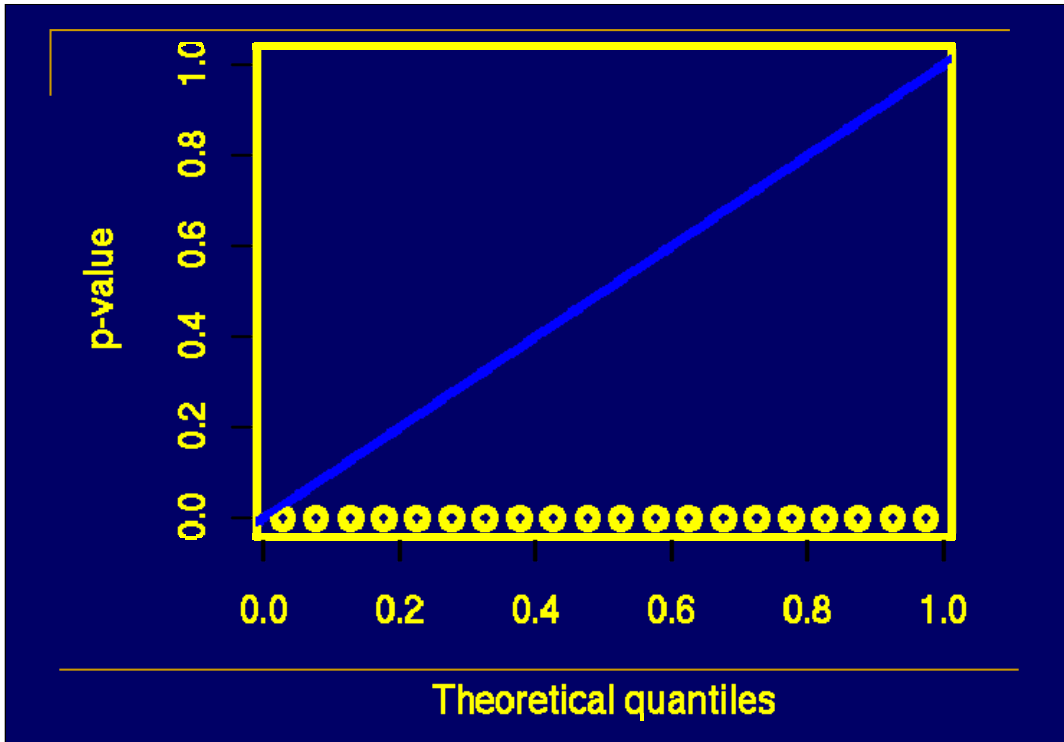
If the picture is like this, then that means that our strategy is acting just like one of our random paths. The p-values are like a sample from the uniform distribution. There is zero skill.

In fact this picture *is* a random sample from the uniform.



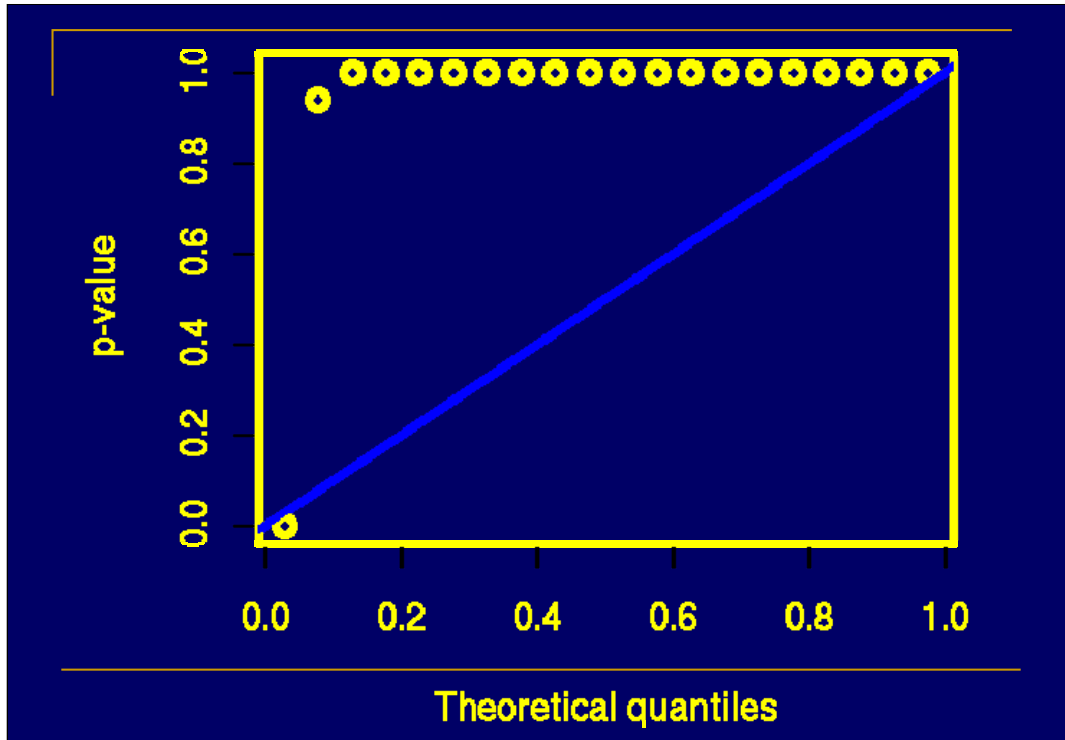
This is real data that also says “zero skill” but in a different sense.

Here with half of the initial portfolios we definitely outperform but with the other half we definitely underperform.



Again real data.

This is what we want to see: where ever we start, we definitely outperform.



Yet again real data.

Here is what we definitely do not want to see.

Almost everywhere we start we definitely underperform.

But if we did this only once and happened to pick that one initial portfolio, we would think we have a winning strategy when in fact we have a losing strategy.

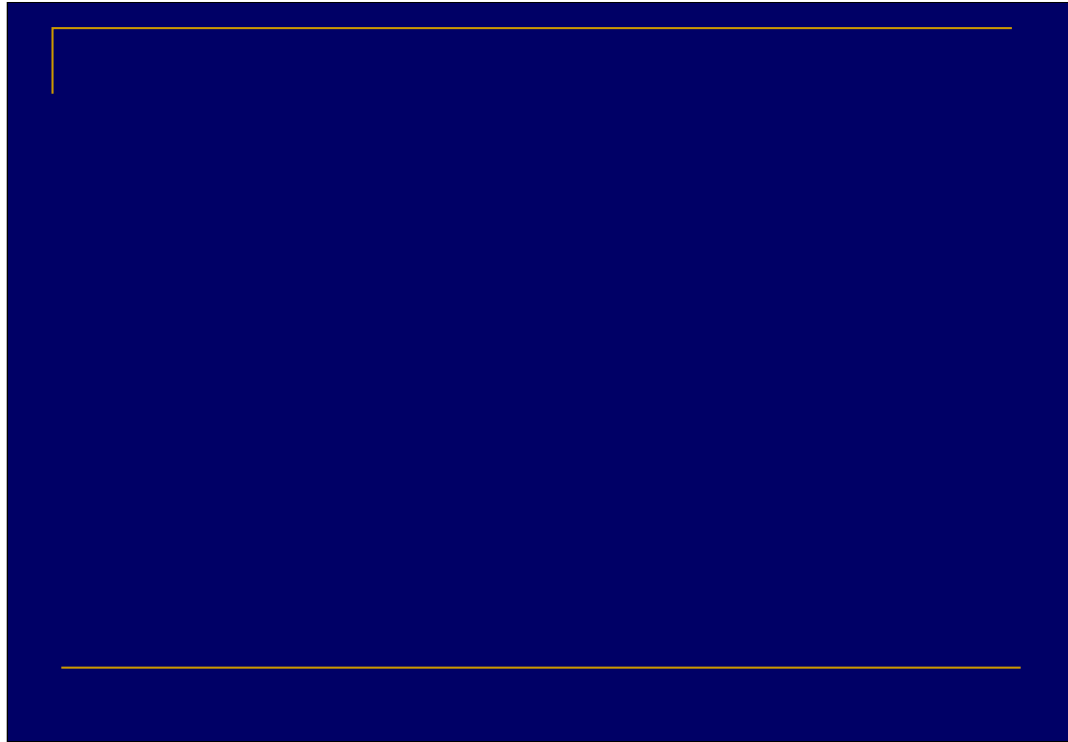
Tracking Error Constraints

- **Annualised standard deviation of the difference in returns between fund and a benchmark**
- **Common to have a maximum tracking error constraint**

It is quite common for investment mandates to impose a maximum tracking error constraint on the fund manager.

The day of the talk I bought a sandwich at Boots that cost £1.50. I quite liked the sandwich so I decided to make reservations for the next day at The Ivy. At The Ivy I had them make me a sandwich as close as possible to the Boots sandwich. I had to send the first one back because it was too good. The cost was substantially more than £1.50.

That is precisely what we are doing with tracking error constraints. You can buy an index essentially for free. Or you can pay a lot of money to get something ever so much like what you can buy for free.



Let's simplify reality and say there are two possibilities: the fund manager we hired has skill or they don't have skill.

If they don't have skill, then we are paying a bunch of money for nothing. Or possibly we are paying a bunch of money to get less than nothing.

If they do have skill, then we are stopping them from using that skill. It is like buying a Ferrari and never putting air in the tires.

Why Tracking Error?

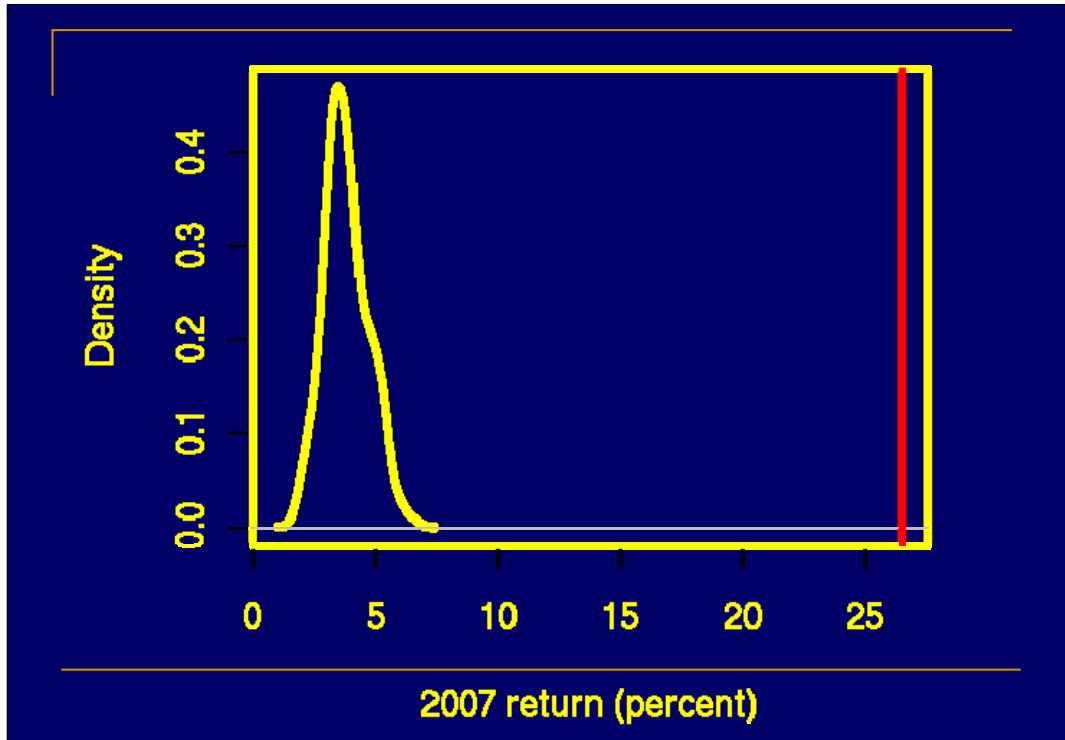
- **Illusion of testing value added**

The reason that we have tracking error constraints is so that we can have the illusion that we can tell if the fund manager is adding value or not.

With a tracking error constraint if the fund manager outperforms the benchmark, we can congratulate ourselves on selecting such a good fund manager. If they underperform, we can say that they aren't doing their job properly and take our money back from them.

Never mind that it will take decades using this procedure to know if the fund manager really is adding value or not.

If we don't impose a tracking error constraint, then how do we know if the fund manager is doing well? One idea is to just trust them. That isn't everyone's idea of a good idea.



Random portfolios allow us to see if a fund manager is doing well whether or not a tracking error constraint is imposed.

Random portfolios give us the freedom to have a more rational investment process.

Better Tracking Error Constraints

- **Minimum tracking error constraints make sense**

You might think I am entirely down on tracking error constraints. Not true. I just think that we should use better tracking error constraints.

A better tracking error constraint is a minimum tracking error constraint instead of a maximum tracking error constraint.

If you go to an expensive restaurant, you should order something that you can't buy on the corner at your lunch hour.

Why R?

- **Not a spreadsheet**
- **Made for data analysis**
- **S3 object-orientation**

This talk is to an R group, so what does R have to do with it?

First off, R is not a spreadsheet. Random portfolios could be used in a spreadsheet. Your sanity is in question if you think that is a good idea.

R was specially designed for data analysis. What we do when we use random portfolios is to use one set of data, create a big bloom of new data, and then compress that into a third bit of data. That is precisely what R is good at.

The object orientation is a handy feature that simplifies some operations for the user.

Random Portfolio Values

```
valuation(rand.ports, asset.price.matrix,  
collapse=TRUE)
```

The `valuation` function is a generic function in Portfolio Probe. It has a method for random portfolios.

The command above takes a matrix of asset prices: the rows correspond to time, the columns to assets. The result of the command is also a matrix. It has the same rows, but the columns will correspond to random portfolios rather than assets.

Optimise

**trade.optimizer(price.vector,
variance.matrix, ...)**

If we use R and we use random portfolios, the same old stuff will be better.

More information on:

R

The danger of spreadsheets

Random portfolios

Portfolio Probe

can be found on <http://www.burns-stat.com>