Applications in Trading System Development

Before upcoming chapters dig into details regarding applications of Monte-Carlo permutation tests and bootstrap tests in trading system development, it's useful to get a head start by briefly presenting an overview of some topics that will be covered. These are *some* of the situations in which these tests are immensely useful:

- We have already developed and tested a trading system. We just evaluated it on an out-of-sample (*OOS*) dataset and it performed well enough that we are inclined to trade it with real money. But what are the chances that the impressive OOS results we just found could have been due to good luck? The truth is that in order to confidently put a trading system into practice, it must satisfy *two* qualifications:
    1) Its measured OOS performance must be good. Everybody knows about and tests this quality.
    2) There must be small probability that this performance could have been due to good luck. Shockingly few developers take the vital extra step of computing this probability. It doesn't matter how wonderful its OOS performance is if there is a significant probability that this impressive performance could have arisen from good luck. Believe it!

- We are in the earliest stages of developing a trading system. We have a generally defined system but it has one or more optimizable parameters that must be tweaked. Perhaps we have a sophisticated predictive model with numerous model parameters to optimize, or perhaps we have a simple algorithm system such as a moving-average-crossover system with optimizable lookbacks. The problem is that if we have too many or too strong parameters we may *overfit* the data; our system may learn random noise as if it is an authentic pattern. We can usually detect this by OOS testing, but that is a waste of time and valuable OOS data. *Once we use OOS data, it is no longer truly OOS and, to be strictly correct, can never again be considered OOS.* Thus, it is a treasure to preserve. Luckily, the MCPT provides an excellent way to evaluate the susceptibility of our system to overfitting at the earliest stage of development.

- We have developed what may be called a *model factory*. This is typically a computer program that may try different models, perhaps with different hyper-parameters, perhaps specializing in different market conditions (high/low volatility, up/down trend, etc), perhaps trading different markets or market sectors, and so forth. In other words, this is a machine that throws spaghetti against the wall, watching for something to stick. Invariably, a key part of deciding if a candidate trading system is worth further evaluation is walkforward testing. We already saw that good OOS performance is a necessary but not sufficient condition for having confidence in a trading system. We also need very low probability of the performance having been possibly the product of good luck. The same caveat applies to walkforward testing. The MCPT for walkforward testing is considerably more complex than that for testing a single OOS dataset, but just as necessary. A model factory that does not have walkforward permutation testing built in is a poor and unreliable factory.

- We have a large set of indicators that may or may not have predictive power. For each individual indicator we don't know if its predictive power is restricted to just unusually large values, or unusually small values, or both. We don't even know if that indicator is best used for long systems, or short systems, or both. We just have a lot of indicators and a lot of hope. If we were to try every combination of predictive region and long/short ability for every indicator in our list, we would almost certainly find some amazing 'predictive power' in our trials, even if all indicators are completely worthless. This is because some of our trials are bound to be lucky even though worthless; the chance of winning a major lottery is tiny, but someone still wins despite not having any skill at picking good lottery numbers. An MCPT solves this problem by computing the probability that the clear 'winners' in our ranked performance list could have achieved their exalted position by sheer good luck. This makes it easy to identify potentially powerful indicators while rejecting junk.

- We have multiple competing trading systems. These might be different algorithms operating on the same market, or a single supposedly universal algorithm operating on a variety of markets, or both (multiple systems on multiple markets). There are two questions that should be answered:
    1) Are the best systems, which probably have good apparent performance, really that good or just the luckiest systems?
    2) Is the single best system truly superior to its competitors? More generally, is the rank ordering of the competitors by performance well defined or is quite possibly random?

  It must be made clear that despite superficial similarities, these are completely different questions. The first regards true predictive power versus temporary good luck that will not continue. This is the more important question. In fact, the second question doesn't even matter unless the answer to the first question indicates true power, not just good luck. The second question concerns the best system's ranking relative to the other systems. Is it truly superior to its competitors? Suppose several of our employees present us with different trading systems. Should we focus exclusively on the best performer, or is there evidence that its performance superiority is far from certain? Or suppose we apply our 'universal' trading system to several markets. Should we focus on whichever market has the best performance, or is its superiority uncertain? More generally, how much importance should we give to the quality ranking? An MCPT answers question 1 above, and an interesting randomization test answers question 2.

- Suppose we have a significant block of truly OOS daily (or finer resolution) returns, perhaps from single-use withheld data, perhaps pooled OOS returns from a walkforward, or even from realtime trading results. For example, we may have a year of returns. We can use bootstraps to compute probability-based estimates of performance in the future, assuming that market behavior does not change (a haughty assumption!). We can say, for example, that there is a 90 percent chance that the system's mean annualized daily return is at least some computed value. Using a very complex algorithm we can even compute approximate probabilities for catastrophic drawdowns, an extremely useful thing to know.