Yule-Simon: Application Note

Bayesian Sharpe ratio estimation with volatility clustering

The Sharpe ratio is a metric for assessing investment performance with respect to a risk-free asset. It is defined as the expected return (less the risk-free return) divided by the risk (i.e. standard deviation) of an asset. However, the expected return and standard deviation are not directly observable from the empirical asset return data so point estimates are typically used.

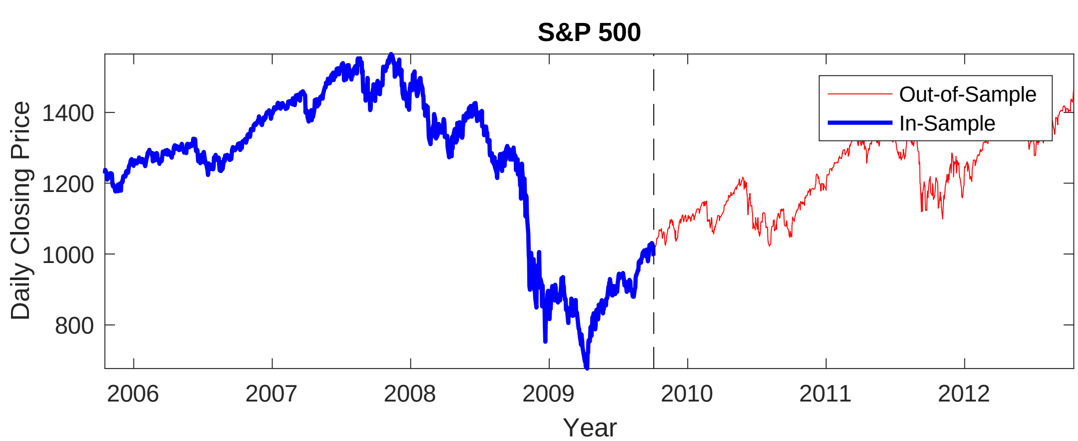
The dynamics of financial price data are such that returns with similar standard deviations tend to “cluster” together in time, a phenomenon known as volatility clustering. This volatility clustering effect can cause errors in the Sharpe-ratio calculation when point estimates based on historical data are used. For example, the Sharpe ratio can be underestimated if a standard deviation point estimate is made from a sample containing highly volatile clusters which have since ended.

The Yule-Simon model computes (or rather samples) the Sharpe ratio posterior distribution based on a generative model which takes into account volatility clustering and long-term memory effects. Therefore, detecting market recovery after crashes and periods of high market volatility can be made sooner using Yule-Simon than traditional point estimation methods based on historical data.

Example: Estimate the Sharpe-ratio of the S&P 500 shortly after the 2008 financial crisis

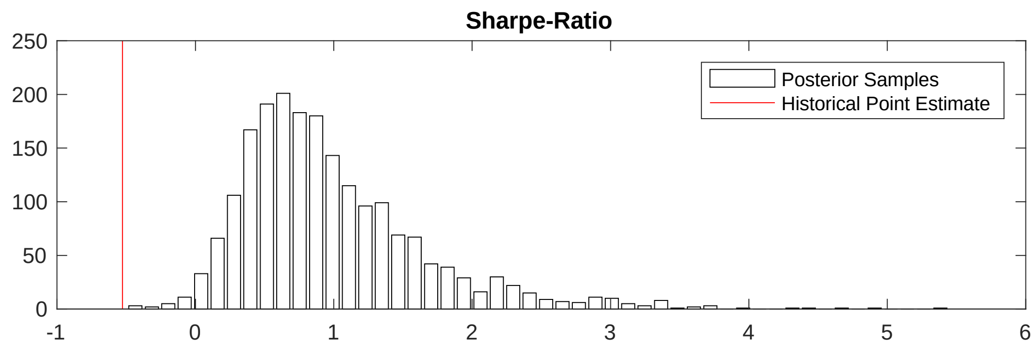
As an example, we will compute the Sharpe ratio of the S&P 500 shortly after the 2008 financial crisis and compare the result to point estimate based on historical data. Code for this example can be found in the file: SCRIPT.m.

First, we import and plot the data:

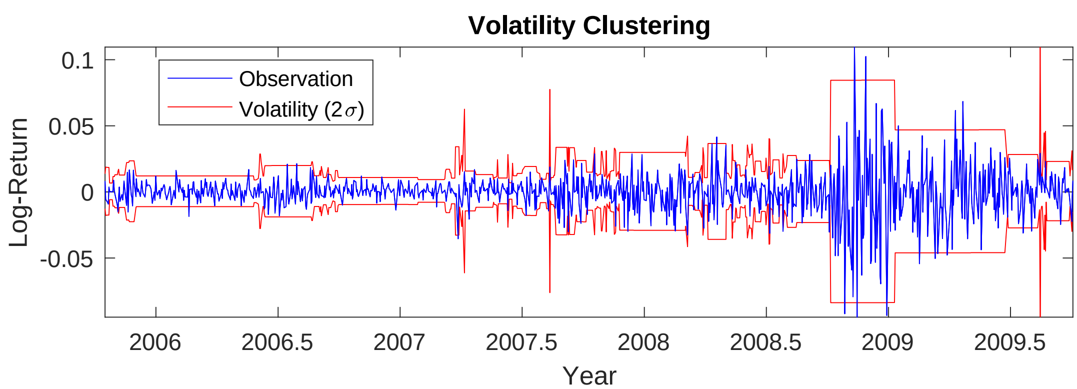


We will use the blue segment to compute the Sharpe ratio at the time of the vertical dashed line (the red curve is there for reference). Next, we run yulesimon.m using the default parameters and plot the results using sharpe_ratio_plots.m. The whole process takes about 2 minutes.

The Sharpe-ratio posterior is plotted below alongside the historical point estimate based on the previous 252 days. Clearly the point estimate is biased by the crash, however the Bayesian posterior suggests this may be a good investment.



To better understand this phenomenon, we can inspect the best-fit volatility clustering solution returned by yulesimon.m as shown below:



The volatility clustering solution indicates we have switched to a lower volatility regime at the time of the posterior inference which allows us to make a more accurate Sharpe ratio estimate. The historical point estimate does not know this detail and computes the standard deviation from a sample including the regime of high volatility.

The yulesimon algorithm is based on Markov chain sampling so solutions need to be checked for model fit and convergence. These plots are automatically generated by the sharpe_ratio_plots.m routine and are shown and described below.

Model Fit: This plot shows how well the generative model posterior fits the data. In general, the standardized log returns should be perfectly Gaussian and follow a linear trend on this quantile plot. This is quantified by the p-value which should be close to 1 for good fits. Here we see the model fit is quite good.

Convergence: This plot shows the log-posterior of the generative model as a function of iteration. Here we are looking for a steep increase followed by a plateau to indicate the algorithm has converged. By inspecting the plot, we can confirm this is the case.

Volatility Parameters: This plot shows the steps taken by the Markov chain to sample the shape and rate volatility hyper-parameters. It's not necessary to understand what these parameters are, it just necessary to verify the chain converges to a consistent region of the state space as shown by the plot.

State-Space Parameters: This plot shows the convergence of the state-space model parameters as a function of iteration. Note that this is a dual-axis plot with process noise and power-law exponent shown on the left and right (respectively). Again, it is not necessary to understand what these parameters are, we are just looking for each curve to converge to relatively constant values. We can confirm this is the case for the power-law exponent, however for the default configuration of yulesimon, the process noise is set to a constant thus no curve is shown. This can be easily changed by modifying the call the yulesimon (see comments).