Notes

Research Question:

How many and which equity benchmarks does an endowment portfolio really need?

How does the answer change for different periods?

With the risk-adjusted data, returns are expressed per one unit of risk, so this version emphasizes correlations as well as the relative return per unit of risk. Basically, with this method the Quantum Optimization picks the benchmarks and then as a practical application one would weight them according to perceived risk with the lower risk benchmarks getting a higher weight instead of using equal weights. This is an application of risk parity analysis.

B-Cardinality for number of B elements

GENERALLY RISK SCALING CHANGES THE DISTRIBUTIONS AND MUDDIES THEM UP

PANDEMIC SAW DIFFERENCE BETWEEN RISK SCALED AND NON SCALED FOR B=1,4,5,6,7

WHOLE PERIOD SAW DIFFERENCE BETWEEN RISK SCALED AND NON SCALED FOR B=4,5,6

PRE 2008 SAW DIFFERENCE BETWEEN RISK SCALED AND NON SCALED FOR B=5,6

QE SAW DIFFERENCE BETWEEN RISK SCALED AND NON SCALED FOR B=5

COVARIANCES DON’T EVOLVE A TON OVER THE PERIODS, WITH EXCEPTION OF THE PANDEMIC

STATEVECTOR SIMULATOR HAS LESS ERRORS THAN QASM SIMULATOR, ASSUMING SAME ANSATZ, RUNTIME, OPTIMIZER, ETC. BECAUSE QASM SAMPLES THE CIRCUIT RATHER THAN DOING THE FULL SIM, SIMILAR TO ACTUAL MEASUREMENTS

FOR LOW ANSATZ, VQE TENDS TO BE MORE INCONSISTENT

COMPARE SCALAR VALUES THAT COME FROM COVARIANCE MATRIX. PERHAPS VARIANCE/TRACE/DETERMINANT

ALLIANZ STRATEGIC ASSET ALLOCATION – lagged granger causation / covariance analysis

COMPARE COVARIANCES FOR SAME PERIOD BETWEEN DIFFERENT SCALED AND NOT SCALED FOR SAME COVARIANCE MATRIX SCALE

COMPARE COVARIANCES FOR DIFFERENT PERIODS BETWEEN SAME RISK SCALED OR NON SCALED FOR SAME COVARIANCE MATRIX SCALE

PLOT PORTFOLIOS WITH DOLLAR AMOUNTS TO COMPARE

*I'm trying to profile qiskit backends and I use randomized\_benchmarking provided by qiskit.ignis.*

*Initially I ran a circuit with 20 qubits 100 length on statevector\_simulator, which takes around 180s and qubits can go up to 32 and then it runs out of memory. Then I tried qasm\_simulator and it only takes 2s to finish and can run up to 54+ qubits.*

*Why does qasm\_simulator perform so much better than statevector\_simulator? Does this happen because the circuit only contains simple gates?*

From the outputs of these calls, you can see that both simulator backends have the same default number of maximum qubits and maximum number of shots. Also, they both accept circuits with all gates, not just the basis gates.

The reason why using the statevector simulator will take longer and then eventually run out of memory as you increase the number of qubits is because the statevector simulator has to return a block of data the size of 2^N (where N is the number of qubits, and 2^N is the size of the statevector). As the number of qubits increases, the size of the statevector increases exponentially. On the other hand, the qasm simulator returns counts, which is a sampling of the statevector of the circuit, way smaller in size and won't increase in size exponentially as the number of qubits increases. Though they return different things, the qasm and statevector simulator are both part of the same simulator.

You can also customize the qasm simulator to act like the statevector simulator by adding the parameter backend\_options to your call to execute() or run() your circuit with the value backend\_options = {"method": "statevector"}, and the qasm simulator will return a statevector.