Quantumized Graph Cuts in Portfolio Construction and Asset Selection

W. Bernard Lee*
HedgeSPA Limited, 29 Mediapolis Circle #04-14, Singapore
School of Accounting and Finance
Hong Kong Polytechnic University, Hong Kong
bernard.lee@hedgespa.com
bernardl@cs.stanford.edu

Anthony G. Constantinides
Financial Signals Processing Laboratory
Imperial College London
Exhibition Road
London SW7 2AZ, United Kingdom
a.constantinides@imperial.ac.uk

ABSTRACT

This paper is concerned with two fundamental problems in investment science, namely a) the construction of a portfolio by segregating target assets into sectors representing a typical market index and b) the selection of target assets from each of those sectors. Such solutions may be applied, for example, to construct a portfolio of 50 assets (say) that aims to outperform the S&P 500 index by selecting the most promising performer in each applicable sector or subsector. The formulation of this investment objective is non-trivial due to the following reasons:

- For Problem 1, the cohorts of assets should be selected based on not only a fundamental classification such as the Global Industry Classification Standard (GICS)[5] but also appropriate statistical characteristics of the assets. The classic argument is that Tesla should be classified as a Technology firm instead of an Automotive manufacturer.
- For Problem 2, the selection of assets with each sector can be driven by factors that may be non-uniform across different sectors. The classic example here is that Inventory is usually not important in the Software Technology sector but it is considered important for the Industry Manufacturing sector.

A computationally efficient approach to solve a similar problem using Graph Theory has been described in Chapter 13 of *Data Analytics on Graphs*[4], contributed by a co-author of this paper. Problem 1 can be solved in the Graph Theory sense by cutting the Set A of assets into two disjoint subsets to maximize the difference in a certain performance metric $f(x_1, x_2, \ldots, x_n)$ of one subset $\{x_1, x_2, \ldots, x_n\}$, where $n < N$, to the same metric applied to the remaining subset of $N-n$ elements. This problem, known as finding the "maximum graph cut" or Max-Cut, sounds intuitively straightforward for a small number of exemplar assets, but its complexity can be prohibitive when applied to real-world, industrial-scale problems. To illustrate the complexity of the problem, if $N = 500$ (e.g. the S&P 500 stock components), the number of combinations to split the vertices into two subsets is $C = 1.6 \times 10^{150}$, and we have to continuously subdivide the leaves in the tree of portfolio cuts to arrive at a suitable sector partitioning that combines fundamental classifications with asset statistical characteristics.

Once we have arrived at the suitable sectors (currently there are 11 sectors for the S&P 500 index, resulting in the average of 45 assets in each sector, but the computer may choose say 20 sectors with 25 assets each on average), in theory, we can apply a similar methodology to the factor sets to fit an asset selection model, so that the model (usually via using a neural network) is only allowed to choose one key factor from each subset of similar factors as a workaround to the well-known problem of overfitting models with too many similar factors, resulting in low *ex-ante* predictive powers. Using Graph Theory to enhance neural networks is a technique known as Graph Neural Networks, which has successfully addressed challenging missing data issues for cross-sectional and panel-data models. However, if there are say 40 factors in a single time slice, we will be working with an average of 1000 factors or $N = 1000$ for a single cross-section in time, before even introducing the time dimension to the problem, which may grow each factor set to a few thousand factors.

Given the computational complexities shown above, these problems are not tractable within a meaningful timeframe using classical computers. In a paper presented at the 2021 Annual Meeting of the American Statistical Association[3], the authors have proposed an alternative approach that would allow novel computational means using the quantum computing variant of the Max-Cut algorithm. Following the work done by quantum physicists at MIT[2], we propose to transform the Max-Cut problem that can be solved by "brute force" combinatorial optimization into its equivalent representation using the Ising Hamiltonian function. Instead of millions of CPU cycles to create a single vector of random samples, a quantum computer is designed to generate a vector of simultaneous random samples in a single cycle. This paper aims to describe the initial results of a test implementation from solving this problem on:

- Classical computer as the base performance benchmark,
- Accelerated classical architectures, and
- IBM quantum computers and/or simulators[6].

CCS CONCEPTS

• **Mathematics of Computing**; • **Discrete Mathematics**; • **Graph Theory**; • **Graph algorithms**;

KEYWORDS

Portfolio Cuts, Graph Theory, Time Series Estimates of Expected Returns, Combinatoric Optimization and Simulations, Quantum Computing, Maximum Graph Cuts

*Corresponding author.

Authors' addresses: W. Bernard Lee, HedgeSPA Limited, 29 Mediapolis Circle #04-14, Singapore; School of Accounting and Finance; Hong Kong Polytechnic University, Hong Kong, bernard.lee@hedgespa.com, bernardl@cs.stanford.edu; Anthony G. Constantinides, Financial Signals Processing Laboratory, Imperial College London; Exhibition Road, London SW7 2AZ, United Kingdom, a.constantinides@imperial.ac.uk.

1 PROBLEM STATEMENTS

Problem 1. We aim to create an indicator vector, in which each asset is represented as an element in the vector, 1 for present and 0 for not present, that will yield a portfolio under a risk-return metric such as the highest Sharpe ratio. It should be noted that portfolio objective functions more complicated than the textbook Sharpe ratio are often chosen in practice for managing institutional investment portfolios. The first step in solving the problem, known as finding "portfolio cuts", sounds intuitively straightforward for a small number of exemplar assets, but its complexity can be prohibitive when applied to solve real-world, industrial-sized problems. Let's assume that A is the set of all admissible assets. Our goal is to find a subset $\{x_1, x_2, \ldots, x_n\}$ of A such that:

- the weight attributed to each asset is great than or equal to zero,
- all weights sum up to one, i.e. $\sum_{i=1}^{n} w_i = 1$
- the number of assets chosen is smaller than or equal to some prescribed maximum $n \leq N$, and
- while a given objective function $f(w_1, w_2, \ldots, w_n)$ attains its maximum within the feasible set.

Practically, this problem is typically solved by stepping w_i on a certain discrete grid with increments of (say) 0.5%, starting from zero. This is computationally time-consuming and not tractable for a large set of available assets to choose from. An alternative way to think of the problem in a Graph Theory sense and essentially to seek a solution in which the objective is maximized is to cut the Set A into two disjoint subsets so that one subset $\{x_1, x_2, \ldots, x_n\}$ can maximize the objective function $f(w_1, w_2, \ldots, w_n)$. To illustrate the complexity of the problem, if $N = 500$ (e.g. the S&P 500 stock components), the number of explicit combinations to split the vertices into two subsets is $C = 1.6 \times 10^{150}$, and that is the complexity even before we allow the objective function to be further maximized based on the chosen subset! However, contrary to the computationally intensive approach sketched in the previous paragraph, a computationally efficient approach to solve a similar problem using Graph Theory has been described in Chapter 13 of *Data Analytics on Graphs*[4].

Problem 2. Once we have identified the possible sectors, the returns of each asset within the sectors can be driven by a set of factors such as $\{r_1, r_2, \ldots, r_m\}$. Typically, each sector or sub-sector contains 20 to 30 assets while the factor set can easily exceed over 100 (say from the accounting factors). Allowing all such factors to be used in any alpha selection model will result in the well-known problem of model overfitting, as in one may find a model with good adjusted R^2 but with relative weak *ex-ante* predictive power. The classic solutions are a) LEAPS regression to try all possible combinations of factors or, b) certain "branch and bound" style heuristic to select a subset of factors. The goal is to identify a small collection of dissimilar factors to form a model showing good predictive power. However, these are pure numerical methods. Using these methods, the computer is not using the knowledge that say Dividends and Earnings Per Share can be grouped together with the goal of picking only one factor from this group. In fact, using similar factors in one model may result in linear dependency among the factors which may hurt the predictive powers of the resulting model. In other words, the desirable approach is to choose at most one factors from each subset $\{r_1, \ldots, r_{m_1}\}, \{r_{m_1+1}, \ldots, r_{m_2}\}, \{r_{m_3+1}, \ldots, r_{m_4}\}, \cdots$ where $m_i \leq m$. In one sentence, the problem is about cutting the set $\{r_1, r_2, \ldots, r_m\}$ into the appopriate subsets based on both the fundamental grouping information of the factors, along with other statistical characteristics such as the historical correlation coefficients observed among the factors.

2 SOLUTION TO PROBLEM 1 - PORTFOLIO GRAPH CUTS

In this section, we use $n = 1000$ from the universe of Russells 1000 components with daily closing prices from the beginning of January 2018 to the end of May 2021. If we invest in the entire universe of roughly 1000 stocks scaled by market capitalization, by definition there will be no out-performance, which is uninteresting to investors. For each portfolio cut, we try to produce a subset with similar characteristics, relative to traditional Global Industry Classification Standard[5] (or "GICS"), which are often chosen based on fundamental criteria (e.g. Tesla is classified as automotive although some may argue that its asset prices behave more like a technology stock).

The alternative idea is to invest in the best-performing stock(s) in each portfolio cut so that we create out-performance relative to the underlying index while preserving index diversification benefits to some extent. Next, we describe the precise algorithm required. We use the notation of an n-node undirected graph $G = (V, E)$ where $|V| = n$ with edge weights $w_{ij} > 0, w_{ij} = w_{ji}$ for $(i, j) \in E$, which is defined as the absolute correlation coefficient, $|\rho_{ij}|$, or

$$w_{ij} = \frac{|\sigma_{ij}|}{\sqrt{\sigma_{ii}\sigma_{jj}}} = |\rho_{ij}|$$

where $\sigma_{ij} = cov(r_i(t), r_j(t))$ is the covariance of returns between assets i and j. The degree matrix, $\mathbf{D} \in \mathbb{R}^{N \times N}$, is a diagonal matrix with its elements defined as $D_{mm} = \sum_{n=1}^{N} w_{mn}$. Then, the $N \times N$ graph Laplacian matrix, $\mathbf{L} \in \mathbb{R}^{N \times N}$, defined as $\mathbf{L} = \mathbf{D}$-$\mathbf{W}$, serves as an operator for evaluating the curvature, or smoothness, of the graph topology. Now we group the n-vertex market graph, $G = \{V, E\}$, into $K = 2$ disjoint subsets of vertices, $V_1 \subset V$ and $V_2 \subset V$, with $V_1 \cup V_2 = V$ and $V_1 \cap V_2 = \emptyset$. A cut of this graph, for the given clusters, V_1 and V_2, is equal to a sum of all weights that correspond to the edges which connect the vertices between the subsets, V_1 and V_2, i.e,

$$Cut(V_1, V_2) = \sum_{m \in V_1} \sum_{n \in V_2} w_{nm}.$$

The normalized cut is regularized by an additional term to enforce that the subsets V_1 and V_2 should be simultaneously as large as possible:

$$CutN(V_1, V_2) = \left(\frac{1}{N_1} + \frac{1}{N_2}\right) \sum_{m \in V_1} \sum_{n \in V_2} w_{nm}$$

Further, it can be shown that if an indicator vector is defined as

$$x(n) = \begin{cases} \frac{1}{N_1}, & \text{for } n \in V_1 \\ -\frac{1}{N_2}, & \text{for } n \in V_2 \end{cases}$$

then the normalized cut, i.e. $CutN(V_1, V_2)$, can be found by the Rayleigh quotient of $\mathbf{L}$ and $\mathbf{x}$, i.e.

$$CutN(V_1, V_2) = \frac{\mathbf{x}^T \mathbf{L} \mathbf{x}}{\mathbf{x}^T \mathbf{x}}.$$

The indicator vector, $\mathbf{x}$, which minimizes the normalized cut also minimizes $CutN(V_1, V_2)$. We rewrite this minimization problem, for the unit-norm form of the indicator vector, as

$$\min_{\mathbf{x}} \mathbf{x}^T \mathbf{L} \mathbf{x}, \quad s.t. \quad \mathbf{x}^T \mathbf{x} = 1$$

which can be solved through the eigenanalysis of $\mathbf{L}$, i.e. $\mathbf{L}\mathbf{x} = \lambda_k \mathbf{x}$. After neglecting the trivial solution $\mathbf{x} = \mathbf{u}_1, (k = 1)$, since it produces a constant eigenvector, we next arrive at $\mathbf{x} = \mathbf{u}_2, (k = 2)$. Thus, the membership of a vertex, n, to either the subset V_1 or V_2 is uniquely defined by the sign of the indicator vector $\mathbf{x} = \mathbf{u}_2$, i.e.

$$\text{sign(x(n))=} \begin{cases} 1, & \text{for } n \in V_1 \\ -1, & \text{for } n \in V_2 \end{cases}$$

Notice that scaling of $\mathbf{x}$ by any constant would not impact the clustering of elements of $\mathbf{x}$ into subsets of V_1 or V_2. It is straightforward to generalize to $K \geq 2$ disjoint sub-graphs through the method of repeated bisections. For instance, $K = 4$ portfolio cuts generate $2^4 = 16$ leaves of the market graph, with each statistically-derived sector averaging 62.5 assets for $n = 1000$. Since the size of each portfolio cut is not evenly distributed, we will target the maximally acceptable size of a proper portfolio cut to be 125, or twice the number of average assets in each cut. This heuristic criterion will turn out to be quite helpful in subsequent discussions.

2.1 Mixing GICS and Absolute Correlation in Graph Laplacian Matrix

An alternative definition of edge weight v_{ij} is given by

$$v_{ij} = \begin{cases} 1, & \text{if assets } i \text{ and } j \text{ are in the same GICS sector} \\ 0, & \text{if assets } i \text{ and } j \text{ are in different GICS sectors.} \end{cases}$$

GICS[5] is an industry-standard taxonmy used to describe asset sectors such as:

No	GICS Classification	Sector
1	10	Energy
2	15	Materials
3	20	Industrials
4	25	Consumer Discretionary
5	30	Consumer Staples
6	35	Health Care
7	40	Financials
8	45	Information Technology
9	50	Communication Services
10	55	Utilities
11	60	Real Estate

Table 1: GICS Sector Names.

Next, we use this linear combination of v_{ij} and w_{ij}, i.e. $\kappa \times v_{ij} + (1 - \kappa) \times w_{ij}$, to construct the $N \times N$ graph Laplacian matrix $\mathbf{L}$. When $\kappa = 1$, the graph Laplacian contains only the GICS sector information of the assets, so the portfolio graph cuts can only return the partitioning by sectors. When $\kappa = 0.9$, the graph Laplacian is still dominated by the GICS sector information of the assets, but the graph cuts are beginning to be influenced by how certain assets may strongly correlate with each other, as shown in how "financials" are split into two cuts in Figure 1.

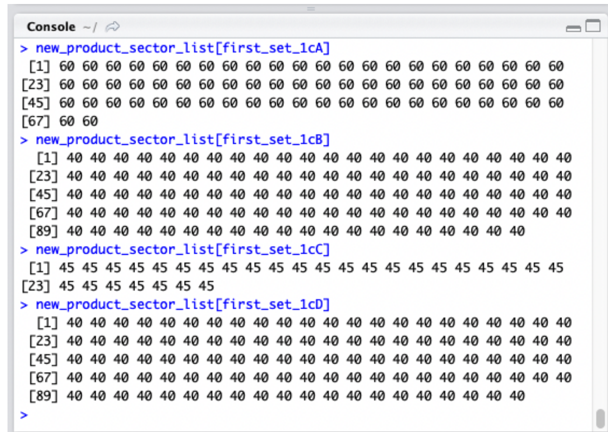


Figure 1: Sector information with portfolio cuts generated with $\kappa = 0.9$.

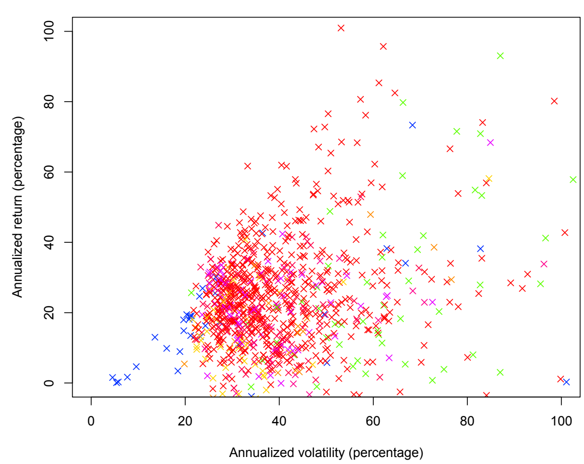
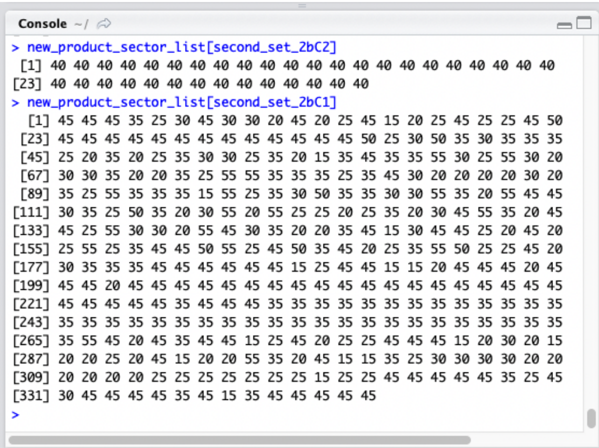


Figure 2: Rainbow-colored return/volatility plot with portfolio cuts generated with $\kappa = 0.9$.

When $\kappa = 0.3$, the graph Laplacian is now driven by both the GICS information and absolute correlation among the assets as shown in Figure 2. Now we can see that the portfolio cuts may allow a mix of sectors with mostly "new economy" stocks mixed with a handful of "old economy" stocks, as shown in Figure 5.



```
Console ~/
> new_product_sector_list[second_set_2bC2]
 [1] 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40 40
[23] 40 40 40 40 40 40 40 40 40 40 40 40 40 40
> new_product_sector_list[second_set_2bC1]
  [1] 45 45 45 35 25 30 45 30 30 20 45 20 25 45 15 20 25 45 25 25 45 50
 [23] 45 45 45 45 45 45 45 45 45 45 45 45 45 50 25 30 50 35 30 35 35 35
 [45] 25 20 35 20 25 35 30 30 25 35 20 15 35 45 35 35 55 30 25 55 30 20
 [67] 30 30 35 20 20 35 25 55 55 35 35 35 25 35 45 30 20 20 20 20 30 20
 [89] 35 25 55 35 35 35 15 55 25 35 30 50 35 35 30 30 55 35 20 55 45 45
[111] 30 35 25 50 35 20 30 55 20 55 25 25 20 25 35 20 30 45 55 35 20 45
[133] 45 25 55 30 30 20 55 45 30 35 20 20 35 45 15 30 45 45 25 20 45 20
[155] 25 55 25 35 45 45 50 55 25 45 50 35 45 20 25 35 55 50 25 25 45 20
[177] 30 35 35 35 45 45 45 45 45 45 15 25 45 45 15 15 20 45 45 45 20 45
[199] 45 45 20 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45
[221] 45 45 45 45 45 35 45 45 45 35 35 35 35 35 35 35 35 35 35 35 35 35
[243] 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35
[265] 35 55 45 20 45 35 45 45 15 25 45 20 25 25 45 45 45 15 20 30 20 15
[287] 20 20 25 20 45 15 20 20 55 35 20 45 15 15 35 25 30 30 30 30 20 20
[309] 20 20 20 20 25 25 25 25 25 25 25 15 25 25 45 45 45 45 45 35 25 45
[331] 30 45 45 45 45 35 45 15 35 45 45 45 45 45
>
```

Figure 3: Sector information with sample portfolio cuts generated with $\kappa = 0.3$.

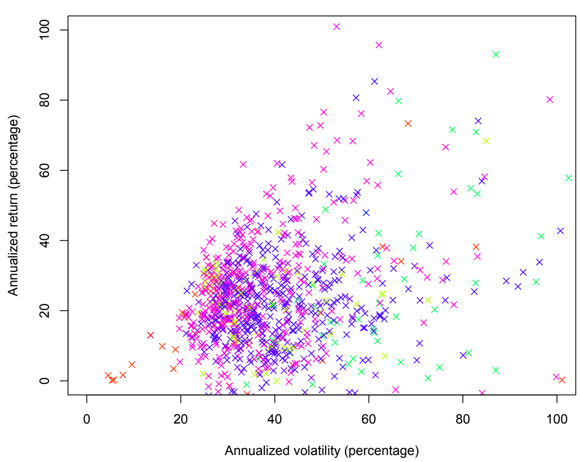
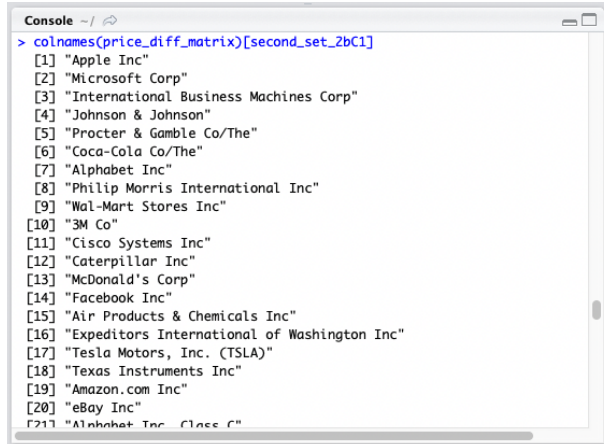


Figure 4: Rainbow-colored return/volatility plot with sample portfolio cuts generated with $\kappa = 0.3$.

2.2 Criteria for Intuitive Portfolio Cuts

Since GICS sector information carries only 30% of the edge weights, for the remaining 70%, we can mix in as much information as desirable when defining the edge weights, such as certain correlation measures of financial information reported by the candidate companies. The practical issue is whether the results generated will be reasonably intuitive, which may include the following criteria:

(1) The largest portfolio cuts should be no more than 125 assets;
(2) Either no "orphan" cuts (cuts with only one or two assets) or effective ways to eliminate them by recombining leaves; and
(3) "Intelligence" to group companies such as Tesla with the technology universe based on asset price statistical characterisitics.

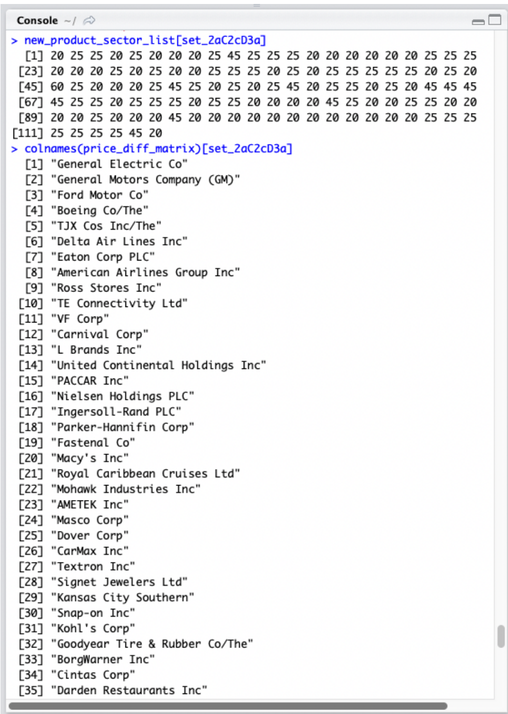


```
Console ~/
> colnames(price_diff_matrix)[second_set_2bC1]
 [1] "Apple Inc"
 [2] "Microsoft Corp"
 [3] "International Business Machines Corp"
 [4] "Johnson & Johnson"
 [5] "Procter & Gamble Co/The"
 [6] "Coca-Cola Co/The"
 [7] "Alphabet Inc"
 [8] "Philip Morris International Inc"
 [9] "Wal-Mart Stores Inc"
[10] "3M Co"
[11] "Cisco Systems Inc"
[12] "Caterpillar Inc"
[13] "McDonald's Corp"
[14] "Facebook Inc"
[15] "Air Products & Chemicals Inc"
[16] "Expeditors International of Washington Inc"
[17] "Tesla Motors, Inc. (TSLA)"
[18] "Texas Instruments Inc"
[19] "Amazon.com Inc"
[20] "eBay Inc"
[21] "Alphabet Inc  Class C"
```

Figure 5: Membership in portfolio cut generated with $\kappa = 0.3$ in Figures 3 and 4.

For a universe of 1000 assets, the eigenvectors associated with the smallest 50 or so non-zero eigenvalues are reasonably smooth (let's call the Set S), we can ask the computer to choose a subset of S to do the cuts so that we keep progress on the leaves until the largest cut contains no more than 125 assets. Empirical experience suggests that we can do so with 17 eigenvectors chosen from a set S of 34 eigenvectors. We have also eliminated one out of any two neighboring eigenvectors with less than 10 variations in signs, which will almost surely result in a trivial if not an identical cut. Choosing 17 eigenvectors out of a set of 34, i.e. $\binom{34}{17}$, can result in over 2 billion total combinations. The goal is to sample only a few hundred combinations so that the computer will satisfy the criteria as stated above while generating cuts that reflect either sector characteristics or asset price correlations, but ideally both. That was the case in one trial in which the largest portfolio cut contains 116 assets selected from a small mix of sectors ("industrials", "consumer discretionary" and "technology"), as shown in Figure 6, and the second-largest portfolio cut contains 72 assets from the "new economy" technology stocks including Facebook and Alphabet, as shown in Figure 7.

Interestingly enough, the largest portfolio cut also contains the ride-sharing company Lyft which is widely thought of as a technology company, which should be grouped with the second-largest portfolio cut. If we run (say) 2000 samples (or 0.0001% of the 2 billion combinations), we may get more effective portfolio cuts, but doing so requires generating much larger sampling sets. Generating correlated random vectors is complicated on a classical computer but almost trivial on a quantum computer. Do note that any complicated matrix computations are already completed at the point of the random sampling, so the remaining steps are about "scoring" the cuts, which are all simple arithmetics with linear complexity. In addition, only the eigenvector information needs to be sent to the random sampling engine at that point. The next section will discuss how we can achieve "quantum leaps" in computational efficiency by solving what would have been a time-consuming computational problem using classical computers.



```
Console ~/
> new_product_sector_list[set_2aC2cD3a]
  [1] 20 25 25 20 25 20 20 20 25 45 25 25 25 20 20 20 20 20 20 25 25 25
 [23] 20 20 20 25 20 25 20 20 25 25 25 20 25 20 25 25 25 25 25 20 25 20
 [45] 60 25 20 20 20 25 45 25 20 25 20 25 45 20 25 25 20 25 20 45 45 45
 [67] 45 25 25 20 25 25 25 20 25 25 20 20 20 20 45 25 20 20 25 25 20 20
 [89] 20 20 25 20 20 20 45 20 20 20 20 20 20 20 20 20 20 20 20 25 25 25
[111] 25 25 25 25 45 20
> colnames(price_diff_matrix)[set_2aC2cD3a]
  [1] "General Electric Co"
  [2] "General Motors Company (GM)"
  [3] "Ford Motor Co"
  [4] "Boeing Co/The"
  [5] "TJX Cos Inc/The"
  [6] "Delta Air Lines Inc"
  [7] "Eaton Corp PLC"
  [8] "American Airlines Group Inc"
  [9] "Ross Stores Inc"
 [10] "TE Connectivity Ltd"
 [11] "VF Corp"
 [12] "Carnival Corp"
 [13] "L Brands Inc"
 [14] "United Continental Holdings Inc"
 [15] "PACCAR Inc"
 [16] "Nielsen Holdings PLC"
 [17] "Ingersoll-Rand PLC"
 [18] "Parker-Hannifin Corp"
 [19] "Fastenal Co"
 [20] "Macy's Inc"
 [21] "Royal Caribbean Cruises Ltd"
 [22] "Mohawk Industries Inc"
 [23] "AMETEK Inc"
 [24] "Masco Corp"
 [25] "Dover Corp"
 [26] "CarMax Inc"
 [27] "Textron Inc"
 [28] "Signet Jewelers Ltd"
 [29] "Kansas City Southern"
 [30] "Snap-on Inc"
 [31] "Kohl's Corp"
 [32] "Goodyear Tire & Rubber Co/The"
 [33] "BorgWarner Inc"
 [34] "Cintas Corp"
 [35] "Darden Restaurants Inc"
```

Figure 6: Largest portfolio cut for sample portfolio cuts generated with $\kappa = 0.3$.

```
Console ~/
> new_product_sector_list[set_6aC2cD1a]
 [1] 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45
[23] 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45
[45] 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45
[67] 45 45 45 45 45 45
> colnames(price_diff_matrix)[set_6aC2cD1a]
 [1] "Alphabet Inc"
 [2] "Facebook Inc"
 [3] "Texas Instruments Inc"
 [4] "Alphabet Inc. Class C"
 [5] "salesforce.com Inc"
 [6] "Cognizant Technology Solutions Corp"
 [7] "Adobe Systems Inc"
 [8] "Juniper Networks Inc"
 [9] "Xerox Corp"
[10] "Motorola Solutions, Inc."
[11] "Analog Devices Inc"
[12] "QUALCOMM Inc"
[13] "NetApp Inc"
[14] "HP Inc"
[15] "Broadcom Ltd"
[16] "NVIDIA Corp"
[17] "Xilinx Inc"
[18] "Micron Technology Inc"
[19] "KLA-Tencor Corp"
[20] "Microchip Technology Inc"
[21] "Qorvo Inc"
[22] "Advanced Micro Devices Inc"
[23] "Benchmark Electronics Inc"
[24] "GoDaddy Inc"
[25] "II-VI Inc"
[26] "Nice Ltd"
[27] "3D Systems Corp"
[28] "Arista Networks Inc"
[29] "CACI International Inc"
[30] "Cadence Design Systems Inc"
[31] "Ciena Corp"
[32] "Fair Isaac Corp"
[33] "Fortinet Inc"
[34] "GrubHub Inc"
[35] "Guidewire Software Inc"
[36] "IAC/InterActiveCorp"
[37] "j2 Global Inc"
```

Figure 7: Second-largest portfolio for sample portfolio cuts generated with $\kappa = 0.3$.

3 QUANTUMIZED VARIANT TO THE ALGOIRTHM

Based on the algorithm described above, we use $n = 1000$ and target 16 cuts. Each sector has 62.5 assets on average. Assuming that we have a reasonably robust model for finding asset alpha, we limit the problem to picking 3 assets from (say) the top 5 identified in each cut. Then, the total number of portfolio combinations to check will be $\binom{5}{3}^{16} = 10^{16}$, which is still a very large number, but is a significant reduction in the order of magnitude from the initial combinations of 1.6×10^{150} as stated in the very beginning of this paper. Thus, we have aggressively reduced the dimensionality of the problem by controlling the number of desirable assets allowed in each portfolio cut. In each of the target cuts, we choose only 3 assets from the top 5 assets screened out by the alpha model to create a portfolio of roughly 48 stocks, which is a portfolio containing the number of assets that is consistent with industry practice. Such a problem is at least computationally tractable using modern computers.

This problem bears resemblance to the more generalized Maxcut and MaxCut-SAT problems. Their solutions have quantum analogs (e.g. “Quantum Approximate Optimization Algorithm QAOA” by Farhi, Goldstone, and Gutmann[2]). Essentially, a quantum computer can produce a simultaneous random vector far more efficiently than a classical computer can without the typical time-consuming matrix inversion step to compute Cholesky decompositions to generate correlated random vectors, so we use the quantumized random vector as the tool for performing an extremely fast random sampling. The key to practical application is to find reliable bounds for convergence. Specifically, the authors are keen to explore whether bounds derived for classical combinatoric optimization problems can be applied to their quantum equivalent solutions[1].

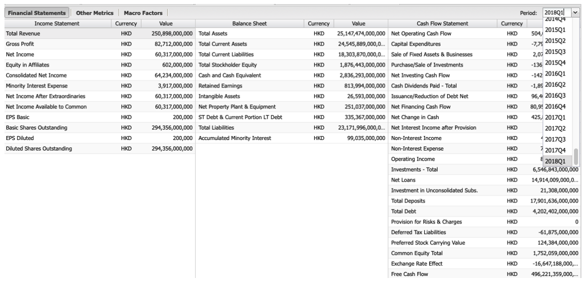
To give a more specific example, MaxCut is an NP-complete problem with the following characateristics. The problem is a 1000-node undirected graph $G = (V, E)$ where $|V| = n$ with edge weights $w_{ij} > 0, w_{ij} = w_{ji}$ for $(i, j) \in E$, The ultimate goal is to find partitions of the original set V into n subsets, namely $V_1, V_2, \ldots, V_n$, where each subset contains fewer than the maximally allowed 125 assets. As mentioned earlier, this is done by applying the method of repeated bisections until the leave of each cut in a market graph reaches the desired size. The cost function to be miminized in each cut is the sum of weights of edge-connected points in the two different subsets (crossing the cut). Assign $x_i = 0$ or $x_i = 1$ to each node i to *maximize* the global objective function

$$\hat{C}(x) = -\sum_{i,j} w_{ij} x_i (1 - x_j),$$

or more generally as

$$C(x) = -\sum_{i,j} w_{ij} x_i (1 - x_j) - \sum_{i} w_i x_i$$

to incorporate both quadratic and linear terms in the objective function C. To map this algorithm to a quantum circuit, map the objective to an Ising Hamiltonian: $x_i \to (1 - Z_i)/2$, where Z_i is the



Income Statement	Currency	Value	Balance Sheet	Currency	Value	Cash Flow Statement	Currency	Value
Total Revenue	HKD	250,898,000,000	Total Assets	HKD	25,147,474,000,000	Net Operating Cash Flow	HKD	504,
Gross Profit	HKD	82,712,000,000	Total Current Assets	HKD	24,545,889,000,0...	Capital Expenditures	HKD	-7,79
Net Income	HKD	60,317,000,000	Total Current Liabilities	HKD	18,303,870,000,0...	Sale of Fixed Assets & Businesses	HKD	2,07
Equity in Affiliates	HKD	602,000,000	Total Stockholder Equity	HKD	1,876,443,000,000	Purchase/Sale of Investments	HKD	-136
Consolidated Net Income	HKD	64,234,000,000	Cash and Cash Equivalent	HKD	2,836,293,000,000	Net Investing Cash Flow	HKD	-142
Minority Interest Expense	HKD	3,917,000,000	Retained Earnings	HKD	813,994,000,000	Cash Dividends Paid - Total	HKD	-1,89
Net Income After Extraordinaries	HKD	60,317,000,000	Intangible Assets	HKD	26,593,000,000	Issuance/Reduction of Debt Net	HKD	96,4
Net Income Available to Common	HKD	60,317,000,000	Net Property Plant & Equipment	HKD	251,037,000,000	Net Financing Cash Flow	HKD	80,95
EPS Basic	HKD	200,000	ST Debt & Current Portion LT Debt	HKD	335,367,000,000	Net Change in Cash	HKD	425,
Basic Shares Outstanding	HKD	294,356,000,000	Total Liabilities	HKD	23,171,996,000,0...	Net Interest Income after Provision	HKD	
EPS Diluted	HKD	200,000	Accumulated Minority Interest	HKD	99,035,000,000	Non-Interest Income	HKD	[illegible]
Diluted Shares Outstanding	HKD	294,356,000,000				Non-Interest Expense	HKD	[illegible]
						Operating Income	HKD	[illegible]
						Investments - Total	HKD	6,546,843,000,000
						Net Loans	HKD	14,914,009,000,0...
						Investment in Unconsolidated Subs.	HKD	21,308,000,000
						Total Deposits	HKD	17,901,636,000,000
						Total Debt	HKD	4,202,402,000,000
						Provision for Risks & Charges	HKD	0
						Deferred Tax Liabilities	HKD	-61,875,000,000
						Preferred Stock Carrying Value	HKD	124,384,000,000
						Common Equity Total	HKD	1,752,059,000,000
						Exchange Rate Effect	HKD	-16,647,188,000,...
						Free Cash Flow	HKD	496,221,359,000,...

Figure 8: Typical Choice of Factors.

Pauli Z operator with eigenvalues ± 1:

$$C(Z) = -\sum_{i,j} \frac{w_{ij}}{4}(1 - Z_i)(1 + Z_i) - \sum_i \frac{w_i}{2}(1 - Z_i),$$

which can be rewritten as:

$$C(Z) = \frac{1}{2}(\sum_{i<j} w_{ij} Z_i Z_j + \sum_i w_i Z_i) + const.$$

Thus, the weighted MaxCut problem is equivalent to *minimizing* the following Ising Hamiltonian:

$$H = -\sum_i w_i Z_i - \sum_{i<j} w_{ij} Z_i Z_j$$

A solution solving for this functional form has been implemented by the IBM Qiskit optimization module which can generate the Ising Hamiltonian for the objective function C[6]. The computation as described in Section 2 has taken hours if not days when performed on a state-of-the-art 8-core computer with RISC architecture. The goal of our future research is to reduce the graph cut computation to minutes if not seconds by using a quantum computer on the cloud so that prospective users at financial institutions can visualize the new portfolio cuts almost instantly in each rebalancing cycle.

3.1 Empirical Results

[TO BE INCLUDED]

4 SOLUTION TO PROBLEM 2 - ESTIMATING EXPECTED RETURNS

A further application of Graph Theory is to derive time-series estimates of expected returns for each of the assets in Set A as inputs to the portfolio optimization algorithm. This is typically done as either a cross-sectional regression or a panel regression, but the pre-condition is that an effective set of factors must be chosen from customary financial and market data available on each asset first. A typical user interface to choose such data (as extracted from the HedgeSPA Institutional Investment Platform) was presented in the 2018 Annual Meeting of the American Statistical Association[3], as shown in Figure 8.

We "cut" the set of factors $\{r_1, r_2, \ldots, r_m\}$ into different subsets $\{r_1, \ldots, r_{m_1}\}, \{r_{m_1+1}, \ldots, r_{m_2}\}, \{r_{m_3+1}, \ldots, r_{m_4}\}, \cdots$ where $m_i \leq m$. We also use a linear combination of v_{ij} and w_{ij}, i.e. $\kappa \times v_{ij} + (1-\kappa) \times w_{ij}$, to construct the $N \times N$ graph Laplacian matrix $\mathbf{L}$. The edge weight v_{ij} is given by

$$v_{ij} = \begin{cases} 1, & \text{if factors } i \text{ and } j \text{ are in the same fundamental category} \\ 0, & \text{if factors } i \text{ and } j \text{ are in different fundamental category.} \end{cases}$$

and w_{ij} is the absolute correlation coefficient between factors i and j. κ will be a parameter chosen based empirical experience.

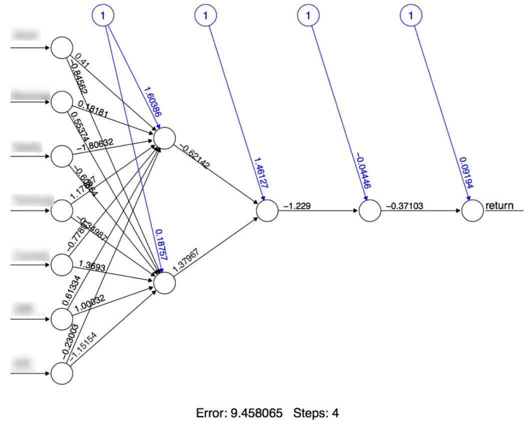


Figure 9: Neural Network.

Once the facors are segregated into different cuts, we choose one factor from each cut to fit a linear model or a more advanced neural net, as shown in Figure 9. Formally, from the set B_i of all $n_i + m_i$ factors related to asset i, we want to choose a subset of n_i factors that is most effective in explaining asset returns. The total available combinations will be $\sum_{k=1}^{n_i+m_i} \binom{n_i+m_i}{k}$. This is still a very large number of combinations even for day-to-day factor sets constructed from typically available accounting and economic data. For instance, the computer will have to check 1.1×10^{15} combinations for a set of 50 factors and 1.3×10^{30} combinations for a factor set of 100. The success of the alternative method as proposed above will depend on:

- Ability to generate a large number of graph cuts as trials without performing explicit sampling; and
- Ability to reject a potentially unhelpful graph cut as quickly as feasible.

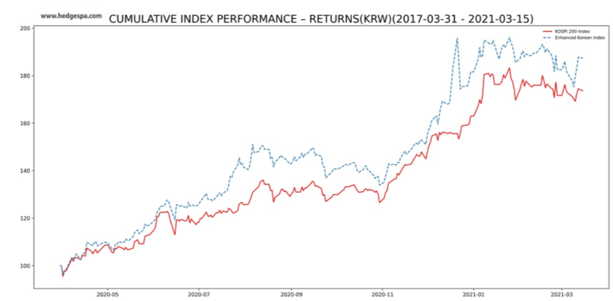
4.1 Empirical Results for Classical and Quantum Computing

[TO BE INCLUDED]

5 ROLE OF QUANTUM COMPUTING

Real-world financial constraints are expected to be far more complex than those with idealized constraints in the basic MaxCut-SAT published by Farhi, Goldstone, and Gutmann. We are modifying MaxCut (as implemented on Qiskit by the IBM Quantum Lab) to solve the problems of interest under constraints. Using a quantum computer would greatly increase the speed of the algorithm described in earlier sections in this paper. Firstly, the intensive use of a random number generator in the first step of our algorithm is extremely cumbersome on a classical computer. With a quantum computer, this can be generated in one single cycle. A quantum circuit can generate a vector of correlated random numbers based on the quantum entanglement property in one cycle, as compared to the millions of cycles required on a classical computer to compute correlated quasi-random numbers based on Cholesky decompositions. In addition to speeding up the initial random number generator, using a quantum computer will help find faster samples of portfolio combinations in the second stage of the algorithm, which will also help eliminate excess steps required to reach an optimal within practical precision bounds.

There is strong objective evidence why this line of research holds major promises. We have been able to produce superior time-series estimates that have resulted in significant investment out-performance not seen in the past. The performance graph to follow is one example of applying our methodology to create an enhanced alternative to the KOSPI 200 Index (the main equity market index in South Korea) for a leading global institutional asset manager to achieve almost double-digit out-performance per year *without* any use of leverage. Please note that this is an independently verifiable out-performance track record created based on professional performance reporting standards, *not* a backtesting of a hypothetical portfolio:



For comparison, the following describes the massive scale required to perform the rebalancing computation for the Enhanced Korean Index if we sample exhaustive combinations on classical computers. Let's say that our goal is to select 25 stocks out of the 200 components in the KOPSI 200 Index, roughly representing 20 sectors in total, or:

- Number of Portfolio Cuts = $\binom{200}{25} = 2.5 \times 10^{31}$
- Number of Asset Selection Trials for 20 Sectors = $20 \times \sum_{k=1}^{100} \binom{100}{k} = 4.5 \times 10^{31}$
- Total Combinations by Exhaustive Sampling = 7.1×10^{31}

6 CONCLUSIONS AND FUTURE RESEARCH

To conclude, the problems as discussed in this paper cannot be solved effectively in the real world on a meaningful scale for industrial applications without using advanced hardware such as supercomputers and quantum computers as well as a smart algorithm such as Graph Theory to reduce the dimensionality to that of a tractable problem. This paper presents our initial efforts in such research, and the authors hope that the results coupled with the new perspective on wider and more pragmatic portfolio selection will inspire similar R&D efforts by publicizing this work.

REFERENCES

[1] Nicos Christofides, "Bounds for the Travelling Salesman Problem", *Operations Research*, Vol. 20, No. 5 (Sep. - Oct., 1972), pp. 1044-1056 (13 pages). https://www.jstor.org/stable/169166?seq=1.

[2] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann, "A Quantum Approximate Optimization Algorithm", MIT-CTP/4610, MIT Center for Theoretical Physics, 2014. https://arxiv.org/pdf/1411.4028.pdf.

[3] W. Bernard Lee, Emma T. Carney, and Anthony G. Constantinides, "Computational Results from Portfolio Graph Cut Simulations", *Proceedings of Joint Statistical Meeting 2021*.

[4] Ljubiša Stanković, Danilo Mandic, Miloš Daković, Miloš Brajović, Bruno Scalzo, Shengxi Li and Anthony G. Constantinides, "Chapter 13: Portfolio Cuts", *Data Analytics on Graphs Part III: Machine Learning on Graphs, from Graph Topology to Applications*, Foundations and Trends® in Machine Learning: Vol. 13: No. 4, pp 499-511. http://dx.doi.org/10.1561/2200000078-3.

[5] "Global Industry Classification Standard", Wikipedia. https://en.wikipedia.org/wiki/Global_Industry_Classification_Standard.

[6] "Max-Cut and Traveling Salesman Problem", *Qiskit Tutorials*, IBM Quantum Lab. https://qiskit.org/documentation/tutorials/optimization/6_examples_max_cut_and_tsp.html.