Sampling Financial Correlation Matrices Using Generative Adversarial Network.

C. Huang, J. Vorawathanabuncha, J. Hu, A. Cao

1 Introduction

The purpose of this project is to generate realistic financial correlation matrices from the subset of empirical correlation matrices of daily financial returns as an alternative to the already established random matrix theory. The GitHub repository for the project can be found here.

The full possible correlation matrix space can be described by an elliptope in Figure 1. However, the financial correlations are a proper subset of that. Many methods have been proposed in sampling correlation matrices, but most of them are either computationally expensive or fail to capture the characteristics of the empirical distribution entirely. In Figure 1 we show the scatter plot of the upper-triangular entries of Wishart's method of sampling correlation matrices.

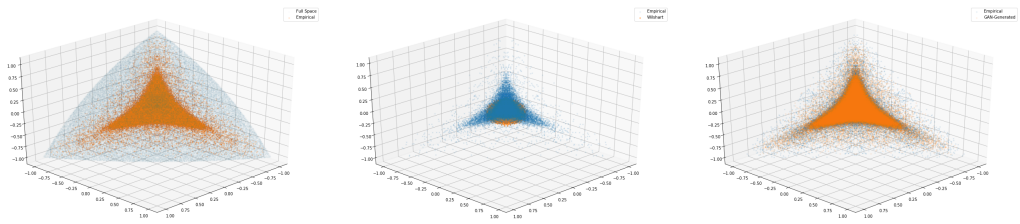

Figure 1: *(Left)* Full and Empirical 3 by 3 correlation space
(Middle) Empirical and Wishart's Correlation Samples
(Right) Empirical and GAN-Generated Samples

In this project we attempt to use GAN as an efficient way of sampling from this unknown space of 64×64 correlation matrices that can capture the distribution better than Wishart's matrices. Smaller-sized correlation matrices can easily be randomly selected from such GAN-generated samples as needed. The bulk of the project is based on Marti (2019) [1] with the exceptions of input clustering method, the underlying training steps in the GAN model, the analysis on generated correlation matrices, and the model evaluation method where we propose using modified Classifier Two-Sample Tests in evaluation as opposed to the original author using internet surveys.

The main applications of this method include generating training samples for further uses in other machine-learning models, using the correlation matrices as conditional GBM paramaters for pricing multi-asset derivatives, and validating portfolios and trading strategies.

2 Methodology

2.1 Acquiring Data and Sampling Method

We acquire daily adjusted prices on CRSP constituents from 2010 to 2019 from Wharton Research Data Services. The CRSP constituents were used as opposed to the popular S&P500 because we want high diversity in the GAN-generated samples. The log returns are calculated. The CUSIPs with more than 50 observations missing are removed after which the rows with missing data are removed entirely.

100,000 correlation matrices are sampled as training data. Training was performed on data from 2010-01-01 to 2014-12-31, where we select the starting date randomly from the range and select 64 CUSIPs without replacement. A correlation matrix is then estimated using the selected CUSIP numbers using a 250-day estimation window. The sampled matrix is then clustered and seriated using and algorithm which is explained in the next section.

The motivation for randomly selecting the starting date is to diversify the variants of the correlation matrices and to capture the most complete subsets of empirical matrices, even the outliers. This is to also make sure outdated data is not included when using long estimation windows and to ensure that not all training samples are clustered at a single point in time which could be affected by market-wide common factors.

The 100,000 correlation matrix samples in the testing set are methodically sampled and clustered from the empirical data in the same way, but within the period 2015-01-01 to 2019-31-31. The reason for the same amount of observations in the training and test sets are due to the way in which we evaluate the performance of our model, which is the be explained in later sections.

2.2 Clustering Algorithm

As 64 stocks were selected randomly without replacements to represent an 64×64 sampled correlation matrix, the matrix can have 64! permutations, all of which are still of the same correlation structure. To make the GAN invariant to input permutations, the sampled correlation matrices are clustered and seriated according to the Hierarchical Risk Parity method [2], as it is currently the most popular and well-documented and of clustering financial correlations. [3] Ward-linkage is used as opposed to single-linkage for robustness to noise and outliers according to Marti (2019).

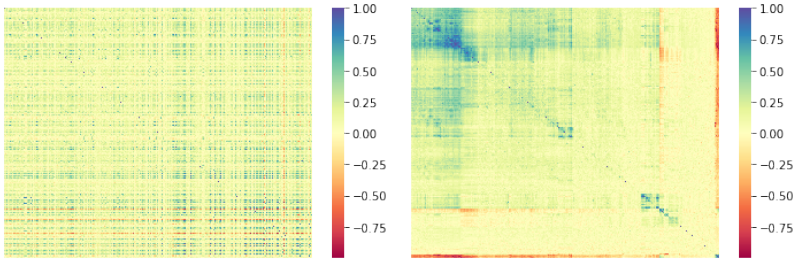

Figure 2: Empirical correlation matrix with random order and the same matrix seriated using HRP

3 Model and Initial Results

Apart from the apparent clusters in Figure 2, it has also been well-researched that financial correlations contain hierarchical structure [4], which is our main motivation for clustering the training inputs based on Hierarchical Risk Parity. A Deep Convolutional Generative Adversarial Network (DCGAN) is therefore implemented as opposed to a Vanilla GAN. This choice was made due to the fact that convolutional layers are able to capture the hierarchical structure inherent in photographic samples.

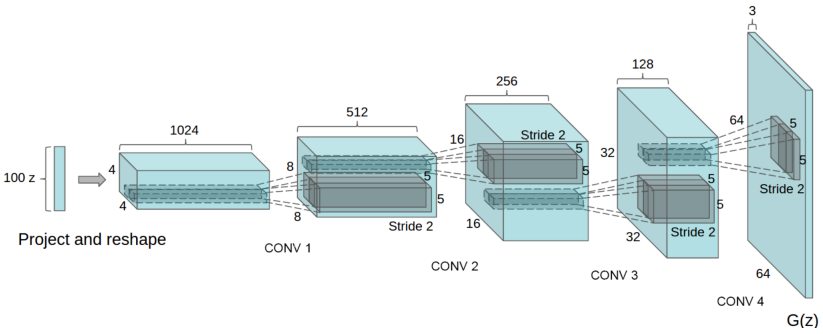

Figure 3: DCGAN Architecture

3.1 Model Setup

In Generator, we use 100-dimensional Gaussian noises as latent vectors to generate fake images. As we step through the layers[**?**], we use convolutional 2-D kernels to reduce channel size and up-sampling to expand image sizes until we get the final output of 64×64 array with channel size of one. The Discriminator is the inverse of Generator and works like normal convolutional neural networks which output the probability of the generated images being real. We use Binary Cross Entropy as the loss function, batch normalization is applied for better training efficiency, with batch size is 256 and learning rate is 0.0002.

The result of the first trial is quite unsatisfactory. The generator losses gradually increase with the each training step while those of discriminator decrease, and we were able to even notice through visual inspection that the heatmaps of generated matrices become visually worse.

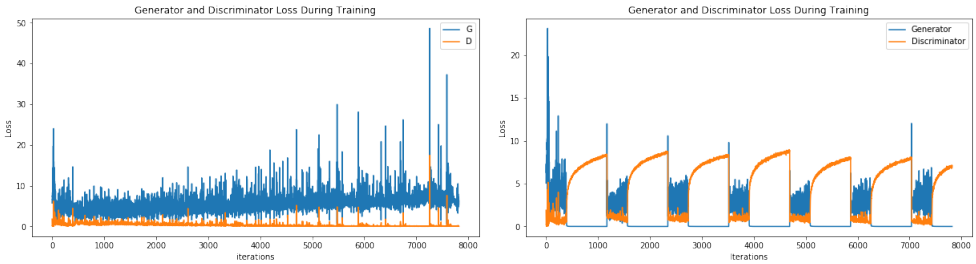

Figure 4: *(Left)* Vanilla DCGAN Training Loss
(Right) Modified DCGAN Training Loss

We deduce that the generator did not have enough chances to train against the discriminator, and therefore drew inspiration from the CycleGAN architecture by training the discriminator for 5 epochs at a time while the generator is trained constantly throughout.

The results of the modified model look very similar to actual matrices by visual inspection. This can be seen in the GAN-generated matrix of Figure 5, which is sampled from the right space in Figure 1. Unfortunately, GAN-generated matrices do not posses unit diagonal entries and are not symmetric. However, they are very close to being so. We therefore project the matrices to the nearest correlation matrix (in Frobenius norm) based on Rebonato & Jäckel (1999) [5] before proceeding to the analysis.

We make the note here that the DCGAN under-fits significantly, as the unit-diagonal and symmetry should be the most obvious characteristics to replicate.

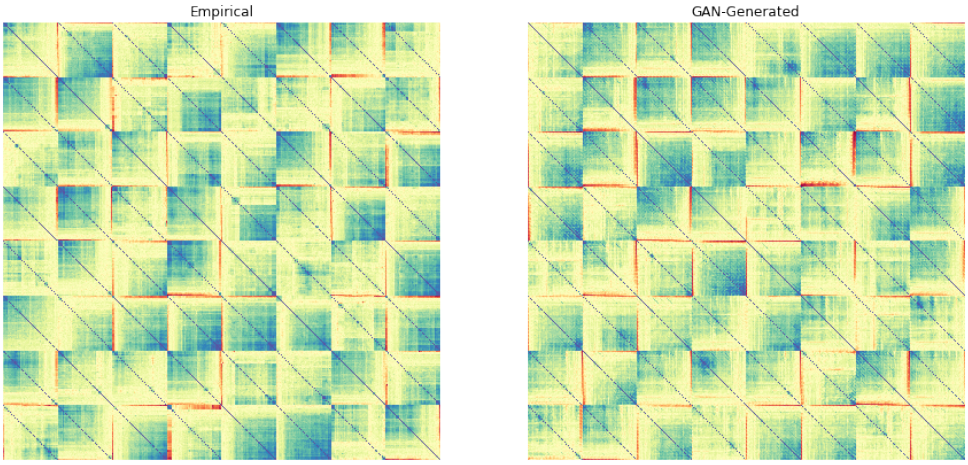

Figure 5: 64 randomly selected matrices from empirical and GAN-generated distributions

4 "Realness" Analysis

Since DCGAN is a generative model, it is tricky to evaluate its performance. While the usual applications of GAN in generating realistic images can be evaluated perceptively with visual inspection, it is much harder to quantify the "realness" of correlation matrices with just their images. Therefore, we are restricted to inspecting the statistics of the GAN-generated samples.

Following [6], a popular method of generating financial correlation matrices is to randomly generate $A \in \mathbf{R}^{N \times T}$ where N represents the number of stocks and T is the estimation window. Here, Wishart's matrix $W = \frac{AA^\top}{T}$ is the sampled correlation matrix when appropriately centered and scaled.

If the observations are serially independent, the distribution of the eigenvalues of W follows a Marchenko-Pastur (MP) density:

$$f_{MP_{N,T}}(\lambda) = \frac{T}{2\pi N} \cdot \frac{\sqrt{(\lambda_+ - \lambda)(\lambda - \lambda_-)}}{\lambda}$$

defined within strict bounds $\lambda \in (\lambda_-, \lambda_+)$ where

$$\lambda_\pm = \left(1 \pm \sqrt{\frac{N}{T}}\right)^2.$$

Since Wishart's matrices iare easily generated and well-documented, we use them as a baseline for our model. For our purposees, $N = 64$ and $T = 250$. Wishart's 3×3 matrix entries are shown against the empirical matrix entries distributions in Figure 1. The corresponding MP densities and the bounds on eigenvalues are plotted below:

We see that empirical eigenvalues do not correspond well with the MP fit. In particular, the extreme eigenvalues far exceed the infimum and supremum, $\lambda_- = 0.25$ and $\lambda_+ = 2.25$.

It is a well-known fact that financial correlation matrices have a largest eigenvalue λ_1 with magnitude much larger than that of their second-largest eigenvalue. While the overall eigenvalues of GAN-generated matrices also do not correspond perfectly with the empirical distributions, they are much closer than Wishart's. Also, the shapes of the largest eigenvalue distributions match quite well considering the high-variance nature of λ_1.

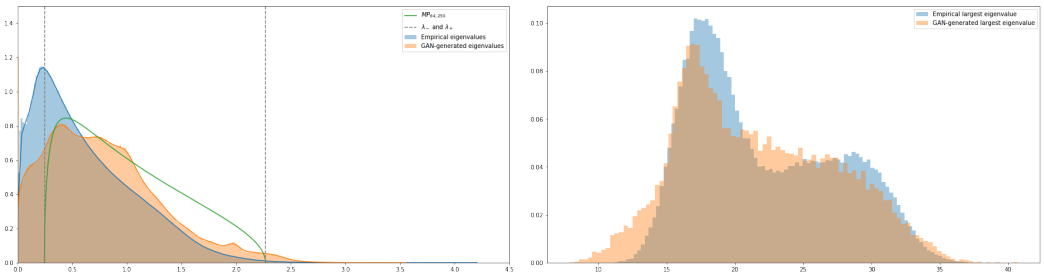

Figure 6: *(Left)* Eigenvalues distributions (largest eigenvalues removed)
(Right) The distribution of the largest eigenvalues

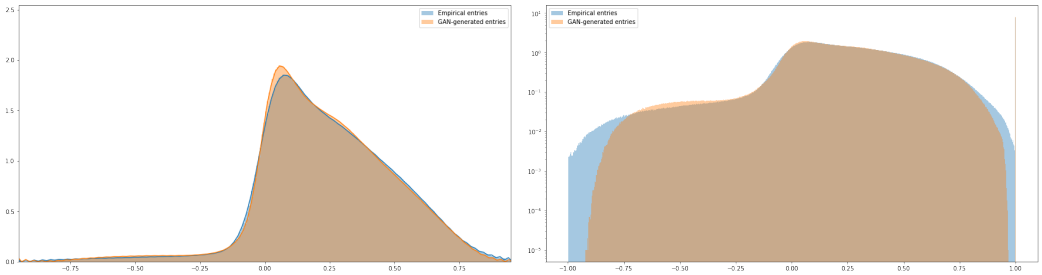

Figure 7: *(Left)* Matrix entry distributions (diagonal entries removed)
(Right) Matrix entry distributions (log-scale)

This is particularly important as λ_1 represents the "market common factor" (market mode), which is one of the most important aspects of any financial correlation matrix. This is an aspect that Wishart's matrices fail to capture ($\lambda_+ \ll \lambda_1$).

In Figure 7, the GAN fails to capture the tails of the matrix entry distribution. This is to be expected, as the empirical densities are relatively very thin in extreme values (Figure 1). As such, we see in the left figure that the distributions of the matrix entries are essentially the same shape, but GAN allocates more densities towards the mode peak.

This also explains the slight discrepancies in the largest eigenvalue distribution (Figure 6), where the GAN's largest eigenvalue distribution is more skewed towards the left. As the very high eigenvalues correspond to the market common factors in asset pricing models, it is then expected to see the failure of GAN in replicating the entries with very high or very low co-movements (the tails).

The 3×3 visualization in the right figure of Figure 1 also easily allows us to observe the much higher densities of the GAN-generated matrix's entries near the mode, and the failure in capturing the thin empirical densities in the extreme entry values.

5 Evaluation

The original paper[1] did not provide a model performance evaluation other than using internet surveys so we propose a new method.

The same DCGAN as outlined in Section 3.1 is fitted using the test data as outlined in Section 2.1. This is the motivation for bootstrapping the same number of 100,000 samples for both training and test sets. The generator on this test set is put away and never to be used after training on the test set, and we call the discriminator of the test set "test discriminator".

Then 256,000 samples were generated using the training Generator and use as inputs in the test discriminator. The percentage of the generated samples that were classified as "real" by the test discriminator is tabulated below:

	Score
Wishart's	0.0342
DCGAN	0.2082

Table 1: Proportion of generated matrices classified as "real"

The GAN-generated matrices did much better than Wishart's matrices which is to be expected from our method. But in general, it has low predictive power likely due to the changes in the shapes of the empirical distributions over time.

6 Conclusions

While the GAN has low predictive power, it is still much better than the currently used Wishart's method. However, The GAN is able to interpolate the space of empirical correlation matrices and sample from it faster than any other established methods.

Therefore, the unintended discovery of this project allows for the possibility of complete overfitting the samples and using the GAN as a very efficient sampling method (Table 2).

We note here that re-calculating correlation matrix every time a sample is drawn is the bottleneck of bootstrapping. On the other hand, generation is almost instantaneous with the GAN, but the process of projecting to the nearest correlation matrix adds significant computation time. Either way, the GAN still beats empirical sampling, which is the strongest discovery in this project.

	Time (Seconds)
Bootstrap	580.53
Wishart's	283.59
DCGAN	200.03

Table 2: Sampling time for 100,000 samples

7 Potential Usage

Exploiting the benefits of low computational time, the main benefits lie in generating for Monte-Carlo simulation. Such matrices could be used as conditional Geometric Brownian Motion parameters for pricing multi-asset derivatives. They could also be used in deriving p-values for hypothesis testing in empirical finance research or generating large training samples for other machine learning models. For these usages, one can sufficiently generate enough samples for models requiring large amount of data.

Other applications include testing the sensitivity of financial models. For market-neutral strategies, a randomly simulated correlation matrix with noise can simulate a different market condition which can test the robustness of model. Similar methodology can also be applied in risk management, especially in scenario analysis, where a randomly generated matrix can represent a random market condition. However, there are still limitations in the generated matrices, due to the fact that they are not labeled.

References

[1] Gautier Marti. Corrgan: Sampling realistic financial correlation matrices using generative adversarial networks, 2019.

[2] Marcos López de Prado. Building diversified portfolios that outperform out of sample. *The Journal of Portfolio Management*, 42(4):59–69, 2016.

[3] Gautier Marti, Frank Nielsen, Mikołaj Bińkowski, and Philippe Donnat. A review of two decades of correlations, hierarchies, networks and clustering in financial markets, 2017.

[4] R.N. Mantegna. Hierarchical structure in financial markets. *The European Physical Journal B*, 11(1):193–197, Sep 1999.

[5] Riccardo Rebonato and Peter Jaeckel. The most general methodology to create a valid correlation matrix for risk management and option pricing purposes, 2000.

[6] Hieu Nguyen, Phuong Tran, and Quang Nguyen. *An Analysis of Eigenvectors of a Stock Market Cross-Correlation Matrix*, pages 504–513. Springer Science and Business Media LLC, 01 2018.