

# 2018 INFORMS O.R. & Analytics Student Team Competition – ENTRY FORM

Entry Number: [2018ORASTC252]

Executive Summary (not to exceed 2 pages)

Team Makeup & Process

Framing the Problem

Data

## 1. The Structure of the Data

The data from Principal consists of 3 parts which are timeseries data, riskmodels data, and result template data.

- Time Series Data :
- Riskmodels Data :
- Result Template Data :

## 2. Data Pre-processing and Rescaling

There are some difficulties in applying the data given from Principal directly to the model. Therefore, we present the following data preprocessing process. To make the data composed of three parts more flexible, a process of data preprocessing formed dataframe using the Pandas Library(<http://pandas.pydata.org>) in Python.

In particular, as rebalancing is carried out, it is configured to extract not all time series data, but only the data needed for the iteration. The parameters used in model were extracted from data frames formed for each period, when each parameter (Alpha Score( $\alpha$ ), Beta( $\beta$ ), 4 Weekly Returns ( $r$ ), etc) was set with the 'SEDOL' index as the key value. This dictionary data type is proper to consider a list of assets that may change every period.

The parameter Omega ( $\Omega$ ) is specified as the covariance matrix for a given data. In this case, the index in the columns and rows are 'SEDOL', the key value of the time series dictionary derived from the above, and are constructed in the form of full matrix.

### 3. The Analysis of the Data

## Methodology Approach & Model Building

### 1. Formulation Approach

**Model Building:** Before buiding the model, we tried to solve the Quadratic Constraint Quadratic Programming (QCQP) problem by using the IBM CPLEX, which is well known Optmization tool, to determine the difficulty of the problem. Since CPLEX can not deal with non-linear constraints,the given formulation is required to be reformulated. First, the constraints (9) are reformulated as follows:

$$\begin{aligned}
 (9^*) \quad & y_i \geq w_i, \quad \forall i \in N \\
 & y_i \leq w_i + 0.999 \quad \forall i \in N \\
 & 50 \leq \sum_{i \in N} y_i \leq 70 \\
 & y_i \in \{0, 1\}, \quad \forall i \in N
 \end{aligned}$$

The decision variables  $y_i$  is 1 if asset  $i \in N$  is selected where  $w_i$  is bigger than 0. Because  $w_i \leq 0.001$  is considered to  $w_i = 0$ ,  $y_i = 0$  if  $w_i = 0$ . The opposite is the same as well. Constraints (10) and (11) are also non-linear. However, if non-convex constraints are linearized, they significantly impact speed by increasing the number of variables to determine and the number of constraints to consider. In other words, it is very inefficient to reformulate constraints (10) and (11), which represent the limits of Active Share and Tracking Error, respectively. Therefore, after solving the problem without reflecting constraints (10) and (11), we tried to identify the following observations.

**Observation1.** In most cases, the range of Active Share is satisfied.

**Observation2.** Tracking Error is less than lower bound (0.05).

According to observation2, Tracking Error should be raised. At this time,  $d^T \Omega d$  can be expressed as  $w^T \Omega w - w_{\text{bench}}^T \Omega w_{\text{bench}}$ , which means that the difference between the benchmark weight and the portfolio weight is too small that TE does not satisfy the lower bound. In other words, the difference between the two weights must be increased to some extent to satisfy the constraints. Since the following  $w_{\text{bench}}^T \Omega w_{\text{bench}}$  is fixed, it is required to satisfy the constraints by changing  $w^T \Omega w$ . Because the weight value of all of the unselected assets from approximately 500 candidates is zero, the weight value is equal to or less than the benchmark weight value. On the other hand, the weight value of the 50 to 70 selected assets may be larger or smaller than the benchmark weights. If the portfolio weight is further increased in the case of the portfolio weight is bigger than benchmark weight, the difference between the two weights will be positively increased. In the opposite case, portfolio weight is less than benchmark weight, the difference between to two weights(Active Weight) will be negatively increased by reducing portfolio weight.

**Bisectional Algorithm:** We applied the idea to the bisectional search algorithm. The bisectional search algorithm is an iterative algorithm that narrows the range and finds a feasible solution. When narrowing the range, if the feasible solution exists in the range, search in the right range based on the median of the range, and search in the left range if the feasible solution does not exist.

An example is shown in 1. Let  $\sum_{p \in P} w_p = 0.6$ , where  $P$  is a set of assets whose  $d$  is greater than zero. First, the solution is searched by dividing the interval between 0.6 and 1.0. If the solution is a feasible solution that satisfies both the range of Active Share and Tracking Error, narrow the range and search again in the left section. The search is stopped when the width of the range( $\underline{Z} \sim \bar{Z}$ ) is less than the end condition which is set to 0.001.

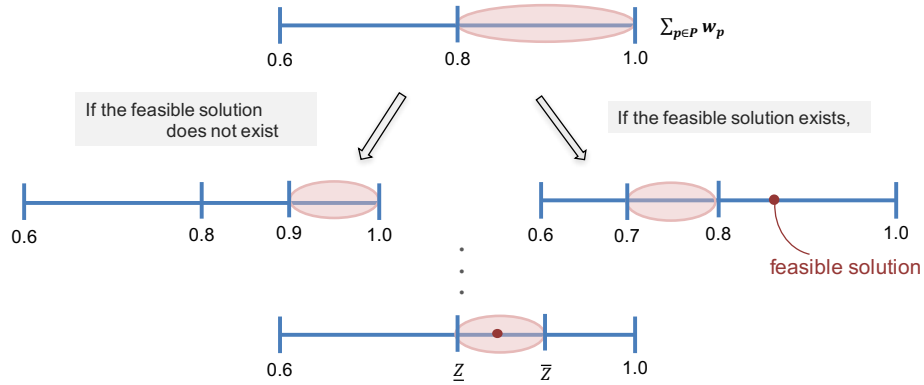


Figure 1: An Example of Bisectional Search

## 2. Neural Network Approach

**The application of GAN :** Generative adversarial networks (GANs) are well known deep neural net(NN) architectures comprised of two networks. One neural network, called the **Generator(G)**, generates new data instances, while the other, the **Discriminator(D)**, decides whether each instance of data is real dataset or not. Both neural networks learn to alternate with each other. The G trains to generate data similar to real data, and the D trains to distinguish real data from fake data generated by G.

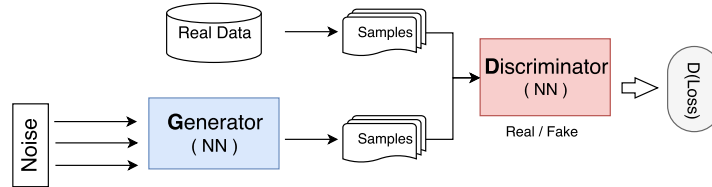


Figure 2: The structure of basic GAN

See 2. G generates fake data when arbitrary noise  $z$  is given as a input through a NN composed of layers which are a input layer, multiple hidden layers and a output layer. G tries to make

$D(x) = 1$  for sample  $x$  extracted from fake data, and  $D$  tries to make  $D(G(z)) = 0$  for sample  $G(z)$ , where  $z \sim p_z(z)$ . Therefore, the loss function  $V(D, G)$  of GAN can be expressed as follows :

$$\min_G \max_D V(D, G) = \mathbb{E}_x p_{data}(x) [\log D(x)] + \mathbb{E}_x p_x(x) [\log(1 - D(G(z)))]$$

We use the GAN's basic idea of learning networks with adversarial relationships between two networks to solve the problem of portfolio optimization.  $G$  determines the decision variable  $w$  of the formulation, and  $D$  determines if the data generated from  $G$  is feasible.  $D$  is based on the formulation, and the sample data  $G(z)$  generated in  $G$  checks both the feasibility of the solution and the quality of objective value. Therefore, the value of loss in  $D$  is the sum of the degree of violation from the constraints and objective value.  $G$  is a deep NN structure consisting of one input layer, three hidden layers and one output layer.  $G$  tries to minimize the loss of  $D$  for the sample  $G(z)$  generated from the random input noise  $z \sim \mathcal{U}(-1, 1)$ . That is, the decision variable  $w$  generated in  $G$  is trained to be feasible for all constraints and the objective value to be minimized. Unlike the basic GAN structure, GAN for portfolio optimization has  $G$  for NN structure, but  $D$  is not, and only  $G$  is required to train. The loss function  $V(G)$  of GAN can be expressed as follows :

$$\min_G \mathbb{E}_x p_{data}(x) [\log(D(G(z)))]$$



Figure 3: The structure of GAN for portfolio optimization

### 3. GAN-MP Hybrid Approach

We propose a new **GAN(Generative adversarial networks)-MP(Mathematical Programming) Hybrid approach** that complements the disadvantages of formulation approach and neural network approach and enhances their advantages. Several limitations could be derived from the three approaches we present above. When solving the problem with the formulation approach, we could not find a reasonable solution because to consider about 500 assets per iteration the number of decision variables to decide and the number of constraints to be satisfied are too high. Especially, constraints (9) which select cardinalities considering the target range on number of stocks make the problem difficult. In the case of Neural Network Approach, there is a great advantage in that a solution that does not get significantly out from the whole constraints and that has a good objective value can be considered at the same time. However, it is difficult to find a solution satisfying the feasibility for all the constraints and the fact that the training time takes too much time. The GAN-MP hybrid approach is a

highly efficient approach that maximizes the advantages of GAN and formulation-based optimizations. This approach consists of three steps as shown in Figure 4. Step1 is the process of finding the cardinality set which is the combination of the selected assets. The cardinality set selected in Step1 solves the QCQP problem with the selected assets fixed and finds the local optimal solution. According to the result of Step2, the update algorithm finds cardinality set that can reduce the objective value and improves the solution.

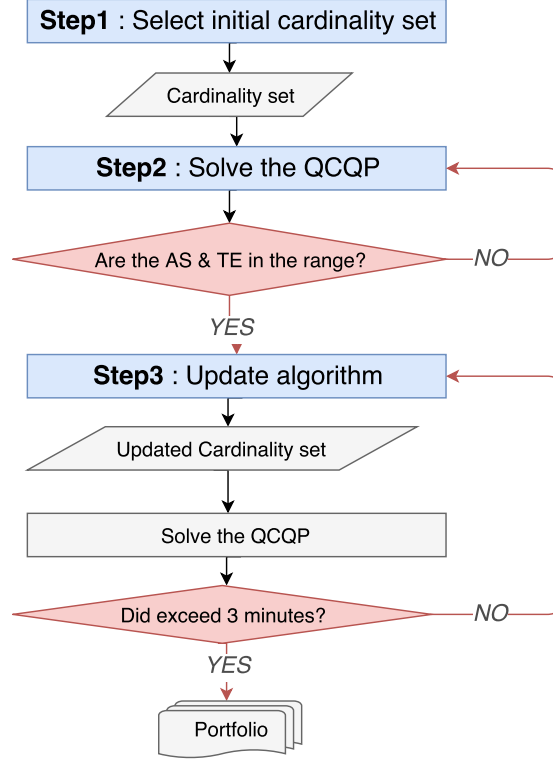


Figure 4: A Flowchart of Hybrid Approach

**Step1. Initial Cardinality Set** : When the number of assets to be considered is large, the problem becomes large as the number of constraints increases as well as decision variables. Thus, step1 filters out good assets that are crucial for minimizing the objective value. The combination of assets is chosen to minimize the risk and maximize return. In order to do this, the whole assets are required to consider. We propose a formulation approach that finds the solution through Mixed Integer Programming (MIP) and a neural network-based GAN as a way to satisfy constraints and to take low values of objective value globally. The following experiments were conducted to evaluate the performance of the algorithms.

- Experiment Environment
  - \* Number of Experiments : 50 times
  - \* Experiment Time of MIP and GAN : 60 seconds
  - \* Solver of MIP : Cplex

- \* Deep Learning Library : Pytorch
- \* The range of assets selected : 60 ~ 70
- Experiment Results :

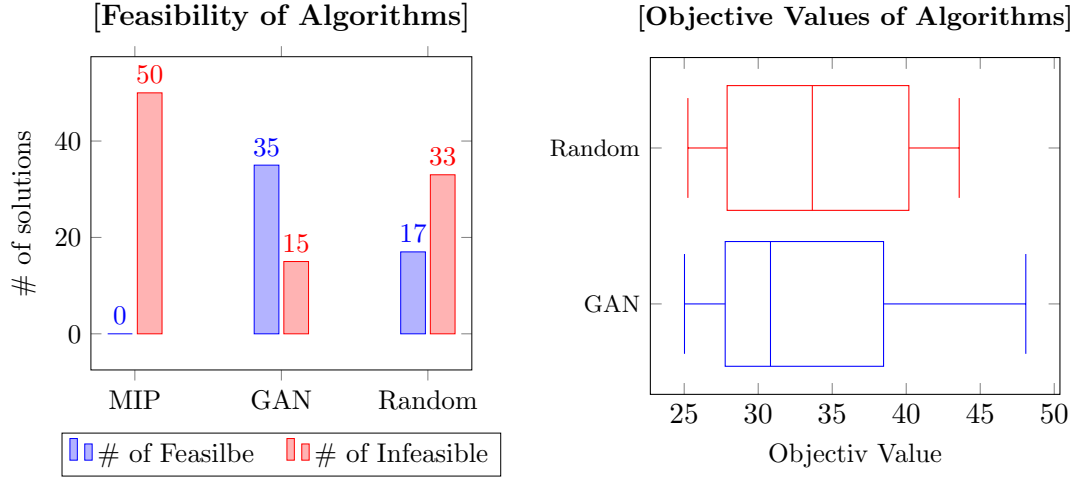


Figure 5: Result of Algorithms

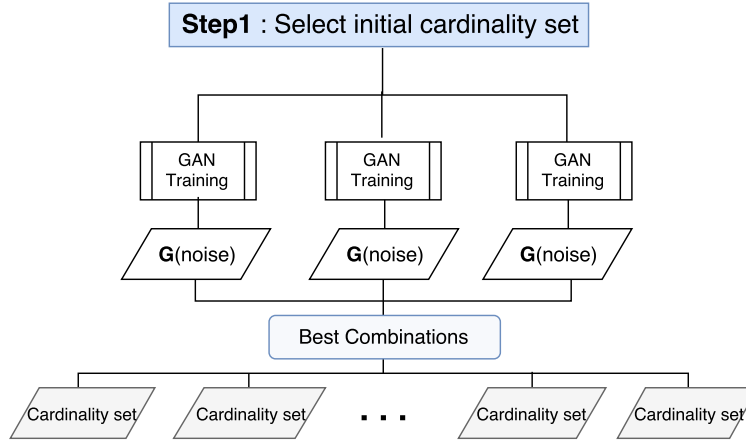


Figure 6: A Flowchart of Hybrid Approach

## Analytics Solution and Results

- This Entry Form: Present your solution and results in this section, including completion of the Portfolio Performance Statistics chart below. You may supplement your analysis with additional charts, diagrams and/or other visualization; these supplements must be incorporated into this section of the Entry Form.

<b>2007-01-01 to 2016-12-31</b>	<b>Portfolio</b>	<b>Benchmark</b>
Cumulative Return	%	%
Annualized Return	%	%
Annualized Excess Return	%	–
Annualized Tracking Error	%	–
Sharpe Ratio		
Information Ratio		–

### Portfolio Performance Statistics

- Results Template: Populate and submit your numerical results using the Results Template. This template is provided as a separate file on the Competition download site. You can use either the Excel or CSV version.

### References

Please follow guidelines in the *Chicago Manual of Style*, 16<sup>th</sup> Edition. Here are examples:

- Journal article: Flynn J, Gartska SK (1990) A dynamic inventory model with periodic auditing. *Oper. Res.* 38(6):1089–1103.
- Book: Makridakis S, Wheelwright SC, McGee VE (1983) *Forecasting: Methods and Applications*, 2nd ed. (John Wiley & Sons, New York).
- Edited Book: Martello S, Toth P (1979) The 0-1 knapsack problem. Christofides N, Mingozzi A, Sandi C, eds. *Combinatorial Optimization* (John Wiley & Sons, New York), 237–279.
- Online reference, fictional example: American Mathematical Institute (2005) Better predictors of geospatial variability. Retrieved June 14, 2005, [www.mathematicsinstitute](http://www.mathematicsinstitute).