# A proposal to build and efficient portfolio using HRP

Juan Román Martínez López
Gerardo Zambrano Rosales
Kevin Emmanuel Soto Hernández
Víctor Hugo Herrera Tamez
Team 4
Facultad de Ciencias Físico Matemáticas

May 28, 2018

**Abstract**

Given a data set of securities HRP assigns to each one an specific weight according to its covariance matrix, however, sometimes is not accurate to invest in to the complete set of securities because there might be a solution with less securities than the total with better expected utility. In this paper is addressed a proposal that first filters the securities according to their nature and gets subsets to work with, then, applies HRP to one of those subsets to get a local optimum that is going to be compared to a desired number of solutions randomly obtained to get the best combination of securities with highest expected utility possible. Finally, a linear programming provides the best amount to invest in to every subset considering the legal constraints and the expected return obtained in every case.

***Keywords***— *filtering, utility, local optimum, random solutions of securities, legal constraints*

# 1 Filtering securities

The set of securities will include securities with different characteristics, these can be equity or debt. The debt securities, are normally attached to a risk that is represented by a percentage or by a classification that is defined by the issuer's capacity to pay debt. If the issuer is a Federal Government, the risk is close to zero, this is the reason why they are usually called risk-free assets, in addition, the debt´s price is varying according to an specific rate, and this variation does not give reliability to the historical change of it. Moreover, if the debt is bought, it allows a completely different scenario, because selling it in the secondary market becomes an option to get a better return.

Now, according to the article 43 of the law of retirement institutions, the investment regime shall have as its main objective to grant the greatest security and profitability of workers' resources. In addition, the investment regime will tend to increase domestic savings and the development of a market for long-term instruments in accordance with the pension system. For this purpose, it will provide that the investments are channeled preponderantly, through their placement in securities, to encourage:

a) The national productive activity;
b) The greatest generation of employment;
c) Housing construction;
d) The development of the country's strategic

From all of the above, it is decided that the risk-free assets must not be analyzed by HRP, because their historical information does not really help, and that the used data must be mainly from the Mexican Equity Market.

From now on, securities will be considered as securities with risk and with no risk attempting to be from the national market.

# 2 What to do with risk securities?

Every security has an individual risk and return and therefore an utility, also every combination of securities has its own utility, suppose now that a data set of n securities is given, the number of all the possible combinations of securities is given by:

$$\sum_{i=1}^{n} \binom{n}{i} = 2^n - 1$$

It is clear that one of those combinations will be the global optimum, the global optimum could be considered as the subset of utilities that after being processed by the HRP algorithm and obtained the respective weights, will offer the best utility. But there is one trouble, if $n = 30$, the combinations of all possible securities is 1073741823, it means that HRP has to be run that quantity of times to get the best utility. That is a huge number, and it keeps increasing every time that a security is added to the set, and not only the number increases, but also the time to get an answer.

To avoid this computational work, in the next lines is presented a pair of algorithms that together, try to find the global optimum without running HRP so many times.

## 2.1 Finding the local optimum

The first step in the search of the global optimum, is try to find a local one, the process to get it is the next one:

First, set the whole securities with their historical information in $D$ and get the security with the best utility, let it be called $D_1$ and remove it from $D$, then, add to $D_1$ one by one the securities left in $D$ and get the pair of securities with best utility, let this pair be called $D_2$ ($D_1$ remains fixed in $D_2$), update $D$ again removing $D_2 \cap D$ from it. Now the same process is going to be repeated again until the total number of securities time, it means, after saved $D_2$, it gets the three securities with the best utility and saves them in $D_3$ ($D_2$ remains fixed in $D_3$), updates $D$ removing $D_3 \cap D$ and search the four securities with the best utility, and so on consecutively, until it gets the utility of the whole set of securities. The subset with better utility, is going to be considered as the local optimum. The pseudocode is shown in exhibit 1.

## 2.2 Diversification algorithm

After getting the local optimum, it has to be set a number of iterations to run this algorithm, and for every time it creates a key of random numbers between 0 and 1. The key has the same number of components as the total number of securities, and each component of the key is representing a security,

so, if the random number in the ith position is grater than .5, the ith security is going to be considered to create a subset of securities, it is easy to note that for every key corresponds a determinated combination of securities, and for that combination it gets the utility, if the utility is better than the one of the local optimum, it saves the new utility and consider it as the best one, if it finds in another iteration a better utility, it saves it again and for the given number of iterations, it repeats the process. The result is going to be a better subset of securities with better utility than the utility of the local optimum if it finds it, the bigger the number of iterations, the better the approach gets. Go to exhibit 2 to find the pseudocode.

## 2.3 Utility

The local optimum and the diversification algorithms completely depend of the utility used. The utility function that is going to be used is $U = R - \frac{1}{2}V$, where $R$ is the expected return of the subset of securities and $V$ the expected variance of the subset of securities. To calculate $R$ and $V$ it is required a vector of weights $w$ that indicates the percentage from the total that is going to be invested in to each security, $R = w^T m$, $V = w^T \Sigma w$, with $m$ the vector with the means of the historical data of every security and $\Sigma$ their covariance matrix. To get a good value of $w$ it is going to be used HRP.

## 2.4 The Hierarchical Risk Parity algorithm

Why HRP? Hierarchical risk parity deals with the three main difficulties in the problem of optimization of a quadratic function (the utility function is dependent on the variance of the portfolio, which is quadratic by definition). These three difficulties are listed below:

1. Markowitz's curse.
2. CLA's instability as a quadratic optimization problem.
3. The lack of hierarchical structure in the allocation of weights.

The hierarchical portfolio construction method uses the information contained in the covariance matrix without requiring its inversion or positive-definitiveness. In fact, HRP can compute a portfolio based on a singular covariance matrix, an impossible feat for quadratic optimizers. The algorithm operates in three stages: Tree clustering, quasi-diagonalization and recursive bisection.

The main step of the HRP will be the input of the nxn covariance matrix to get the correlation matrix from the securities, the tree clustering stage combine the n column-vectors into a hierarchical structure of clusters, so that allocations can flow downstream through a tree graph. To do this, the algorithm creates a matrix of distances between the input data and clusters the data that is closer, in this part, there are some metrics for distances to consider, and choosing one of them must be a decision to take.

The different metrics between numerical objects (distances) are shown in the image below:

Agglomerative clustering schemes.

| Name | Distance update formula<br>FORMULA for $d(I \cup J, K)$ | Cluster dissimilarity<br>between clusters $A$ and $B$ |
|---|---|---|
| single | $\min(d(I,K), d(J,K))$ | $\min_{a \in A, b \in B} d[a,b]$ |
| complete | $\max(d(I,K), d(J,K))$ | $\max_{a \in A, b \in B} d[a,b]$ |
| average | $\frac{n_I d(I,K) + n_J d(J,K)}{n_I + n_J}$ | $\frac{1}{\lvert A\rvert \lvert B\rvert} \sum_{a \in A} \sum_{b \in B} d[a,b]$ |
| weighted | $\frac{d(I,K) + d(J,K)}{2}$ | |
| Ward | $\sqrt{\frac{(n_I + n_K)d(I,K) + (n_J + n_K)d(J,K) - n_K d(I,J)}{n_I + n_J + n_K}}$ | $\sqrt{\frac{2\lvert A\rvert\lvert B\rvert}{\lvert A\rvert + \lvert B\rvert}} \cdot \lVert \vec{c}_A - \vec{c}_B \rVert_2$ |
| centroid | $\sqrt{\frac{n_I d(I,K) + n_J d(J,K)}{n_I + n_J} - \frac{n_I n_J d(I,J)}{(n_I + n_J)^2}}$ | $\lVert \vec{c}_A - \vec{c}_B \rVert_2$ |
| median | $\sqrt{\frac{d(I,K)}{2} + \frac{d(J,K)}{2} - \frac{d(I,J)}{4}}$ | $\lVert \vec{w}_A - \vec{w}_B \rVert_2$ |

Figure 1. Agglomerative clustering schemes

Based on the reference [4], the next things can be said about them:

1.- For single linkage clustering, the MST-algorithm is clearly the fastest one. Together with the fact that it has only half the memory requirements of the other algorithms

2.- The clustering schemes without inversions ("complete", "ward", "average" and "weighted") have very similar performance. The NN-chain algorithm is the only one with guaranteed $O(n^2)$ performance

here. We can conclude that the good worst-case performance can be had here without any cutbacks to the use-case performance.

3.- "centroid", "median": The generic clustering algorithm is the best choice, since it can handle inversions in the dendogram and the performance exhibits quadratic complexity in all observed cases

Single linkage is the fastest metric if it is related to computational time, and because of that reason, that metric is the one that is going to be chosen for the tree clustering in the HRP.

### 2.5 The optimization problem

HRP is a modern version of what Markowitz proposed in 1952, both centralizes on the idea that expected return is desirable and variance of return is undesirable, however both centralizes in the diversification of a portfolio, based on the variance of return. The HRP output is $w$, and it is built by HRP supported in the result of an optimization problem. In reference [6], the one that explains HRP, the optimization problem used to get $w$ is the minimization of the variance of the portfolio, subject to an expected return, let it be the optimization problem 1. Using only the variance means that the optimization problem can be improved even further because the return can also be considered, and this is what brings the next optimization problem:

$$\max U(w) = w^T m - \frac{1}{2} w^T \Sigma w$$

$$s.t.\ 1_n^T w = 1\ w_i > 0$$

Let it be the optimization problem 2 where $w^T m$ is the portfolio return and $w^T \Sigma w$ the variance of the portfolio. Solving that problem and using the results in HRP instead of the ones from the optimization problem 1 will bring a different values of $w$ that mathematically maximize the utility. The mathematical development of this problem and what HRP does with the result is shown in exhibit 3.

## 3 Testing the algorithms

In the next link there is a data set with 22 risk securities and the codes in R for local optimum algorithm and diversification algorithm.

url → https://drive.google.com/drive/folders/1igPS32Cj3U6-luPjYD9pgqNavQWynCQt?usp=sharing

Both algorithms work together but there are variables in each one that save specific information. The best subset of securities of the local optimum algorithm are saved in datosmax and the weights are saved in outf. In the algorithm 2, there is a variable named indicador, it tells how many times was improved the subset of securities with best utility, the best utility obtained after all the iterations is saved in the variable Umax, RVhdiversificado saves the best return and volatility respectively, Outdiv the weights given by HRP, and datosdiv the securities involved.

The algorithms were run for different number of iterations and twice to see the performance of the optimization problems, the results from the first and the second optimization problem can be found in the next table where utility 1 is the utility that corresponds to the first optimization problem and utility 2 to the second:

| Iterations | Utility 1 | Utility 2 |
|---|---|---|
| 100 | 2.025935e-05 | 0.0003004917 |
| 1000 | 6.06737e-05 | 0.0001655417 |
| 10000 | 0.0001776614 | 0.0002509697 |
| 20000 | 0.0001690564 | 0.0002494978 |

Tabla 1: Results using each optimization problem

The table shows that for the second optimization problem the utility obtained at every number of iterations is better than all of the utilities obtained by the first problem. In exhibit 4 is shown an example of the weights behavior related to each optimization problem.

# 4 What to do with risk-free securities?

There are different kinds of risk-free securities, and according to their kind, everyone should be analyzed properly with an specific method that will provide better information than an HRP. An easy thing to do, is, invest in to the other securities the bigger quantity possible according to law and a risk level predefined, and dedicate the remaining to the risk-free securities prioritizing the bonds that are not susceptible to inflation, without exceed the percentage indicated by the law.

# 5 Percentages of investment

The maximum capital that can be invested in certain securities considering 4 kinds of SIFORES is provided in the next table:

| Security | SB1 | SB2 | SB3 | SB4 |
|---|---|---|---|---|
| Variable income | 10% | 30% | 35% | 45% |
| Fibras | 5% | 10% | 10% | 10% |
| Valores extranjeros | 20% | 20% | 20% | 20% |
| Udibonos | 25% | 25% | 25% | 25% |

Tabla 2: Maximum Percentages of securities

# 6 Finally, how to invest?

Let $\alpha$ be the expected return from the risk securities estimated with the algorithms already proposed and $\beta_1$, $\beta_2$ ..... $\beta_n$ the expected return from the risk free securities, then, the final weights to invest at each kind of the securities are going to be given by the next linear programming,

$$\text{Max } X_1\alpha + X_2\beta_1 + \ldots\ldots X_{n+1}\beta_n$$
Subject to

$$\sum_{i=1}^{n+1} X_i = 1$$

The other constraints are directly set by the table 2, for example, suppose that $X_2$ is an UDIBONO, then
$25 \geq X_2$ if it is a SB1, and so on for each one.

# 7 Conclusions

After testing the algorithms, it can be observed that the second optimization problem outperforms the first one because the utility of the portfolio obtained with the second problem is clearly greater than the first problem and also its vector of weights $w$ is more diversified and satisfies better the purpose of HRP. In addition, if the number of iterations of the diversification algorithm grows, it will be more probably to get better results, however, the more securities had and the bigger the number of iterations, the more important the computational time becomes, because it will take more time to give an answer, so, the dataset size and the iterations have to be considered. It is also important to know that the best $w$ obtained might be the best solution, and if it is not, it is very probably to be closer to the best solution, so, whatever the solution obtained is, it has to be taken just as an advice, because mathematical models and tools do not model reality completely, and common sense has to be always present.

# 8 Exhibits

**Exhibit 1**

**Algoritmo 1** Finding the local optimum

**Require:** Set of securities

$D \leftarrow$ Set of securities
$D_1 \leftarrow$ Security with best individual utility
$U_1 \leftarrow$ Utility of $D_1$
$D \leftarrow D - D_1$ Update $D$ without $D_1$
**for** $(k = 1 \ \ to \ \ ncol(D))$ **do**
  $D_2 \leftarrow D_1 D[k]$ add to $D_1$ another security ($D_1$ is fixed)
  Run HRP for $D_2$ and get $U_2'$
  **if** ($U_2'$ is the highest utility) **then**
    $U_2 \leftarrow U_2'$
  **end if**
**end for**
$D_2 \leftarrow$ The pair of securities with highest utility
$D \leftarrow D - D_2$ Update $D$ without $D_2$
Help variable $p \leftarrow 1$
Help variable $l \leftarrow 3$
**while** $(p < ncol(D)$ **do**
  $U_n \leftarrow$ Short value help variable
  **for** $(k = 1 \ \ to \ \ ncol(D))$ **do**
    $D_l \leftarrow D_1 D_{l-1} D[k]$ add to $D_l$ another security ($D_1 D_{l-1}$ is fixed)
    Run HRP for $D_l$ and get $U_l'$
    **if** ($U_l'$ is the highest utility) **then**
      $U_l \leftarrow U_l'$
    **end if**
  **end for**
  $D_l \leftarrow$ The securities with highest utility
  $D \leftarrow D - D_l$ Update $D$ without $D_l$
  $p \leftarrow p + 1$
  $l \leftarrow l + 1$
**end while**

[1]Everytime the algorithm saves an utility, it must also save the weights from HRP, the indexes of the securities that it choose and the best expected return and variance.

Exhibit 1 - Pseudocode to find the local optimum.

**Exhibit 2**

**Algoritmo 2** Diversification algorithm

**Require:** Set of securities and maximum utility gotten from algorithm 1

$D \leftarrow$ Set of securities
$U_{max} \leftarrow$ Maximum utility gotten from algorithm 1
$v \leftarrow$ Number of times to run the algorithm
$H \leftarrow$ Empty variable
$c \leftarrow 0$ Help variable
**for** $(j = 1 \ \ to \ \ v)$ **do**
  $key \leftarrow 1xncol(D)$ matrix of random numbers between 0 and 1
  **for** $(i = 1 \ \ to \ \ ncol(D)$ **do**
    **if** $(key[1, i] > .5)$ **then**
      $c \leftarrow c + 1$
    **end if**
  **end for**
  **if** $(c >= 2)$ **then**
    **for** $(i = 1 \ \ to \ \ ncol(D))$ **do**
      **if** $(key[1, i] > .5)$ **then**
        $H \leftarrow$ D[,i] Add to H a selected security from D
      **end if**
    **end for**
  **end if**
  Run HRP for H and get the utility $U_H$ for it
  **if** $(U_H > U_{max})$ **then**
    $U_{max} \leftarrow U_H$ Set the new maximal utility
  **end if**
**end for**

[2]Everytime the algorithm saves an utility, it must also save the weights from HRP, the indexes of the securities that it choose and the best expected return and variance.

Exhibit 2 - Diversification pseudocode.

**Exhibit 3. Optimization problem 2, mathematical development and HRP work.**

$$\max U(w) = w^T m - \frac{1}{2} w^T \Sigma w$$

$$s.t.\ 1_n^T w = 1\ w_i > 0$$

U(w) is a well known scalar-valued quadratic function whose image is convex (i.e. there exist only one minimum inflection point). This function can be easily derived by gradient operator in order to find its inflection point through Lagrange multipliers

$$\frac{\partial U}{\partial w} = \Sigma w - m = \lambda 1_n$$

$$w = \Sigma^{-1}(\lambda 1_n + m)$$

substituting w in constraint, $\lambda$ is given as

$$1_n^T \Sigma^{-1}(\lambda 1_n + m) = 1$$

$$\lambda = \frac{1-\mu}{\sigma}\ (1)$$

where $\sigma = 1_n^T \Sigma^{-1} 1_n$ and $\mu = 1_n^T \Sigma^{-1} m$

Since it is desired to avoid the inversion of covariance matrix because its instability, we choose to diagonalize the covariance matrix (i.e. dropping covariances factors and let only variances in main diagonal) in order to invert covariance matrix in a trivial way. Finally $w$ vector is given as:

$$w = diag(\Sigma)^{-1} \left[ \left( \frac{1-\mu}{\sigma} \right) 1_n + m \right]\ (2)$$

thus $\sigma = tr(diag(\Sigma)^{-1})$ and $\mu = 1_n^T diag(\Sigma)^{-1} m$

HRP's stage 3 (Recursive Bisection) recursively splits the list of indexes into two lists of equal length following the famous method "divide and conquer". The reason behind such an approach is to ensure the assignment of a non-zero weight to each of the securities. One way to explain how this stage works is to think that both left and right sub-list indexes represent only one asset, each one with its variance and its expected return defined as:

$$\sigma_{i,t}^2 = var(L_i^{(t)}) = w_i^{(t)T} \Sigma_{i,t} w_i^{(t)}$$

$$r_{i,t} = E(L_i^{(t)}) = w_i^{(t)T} m_{i,t}$$

Let $\Sigma_{i,t}$ and $m_{i,t}$ be the sub covariance matrix and sub expected return vector respectively which only concern indexes from $L_i^{(t)}$ list. In case $n = 2$ the $w$ vector can be rewritten as $w = [w_i^{(1)}\ w_i^{(2)}]$ and according to (2):

$$w_i^{(1)} = \frac{\sigma_{i,2}^2 + r_{i,1} - r_{i,2}}{\sigma_{i,1}^2 + \sigma_{i,2}^2}\ and\ w_i^{(2)} = \frac{\sigma_{i,1}^2 + r_{i,2} - r_{i,1}}{\sigma_{i,1}^2 + \sigma_{i,2}^2}\ (3)$$

Notice that $w_i^{(2)} + w_i^{(1)} = 1$ and first constrain holds. However, both $r_{i,1}$ and $r_{i,2}$ can be negative or positive even bigger than $\sigma_{i,t}^2$ which can lead to generate negative weigths. This does not hold second constrain ($w_i > 0$) so in order to ensure non-negatives weights, $r_{i,1}$ and $r_{i,2}$ must be dropped.
Finally, a unitary vector $w$ is created in order to multiply the obtained weights $w_i^{(t)}$ in (3) for each of the positions that is represented by the list $L_i^{(t)}$. The reader can see this procedure as follows:

$$Unitary\ vector\ \rightarrow w = [1,\ 1,\ 1,\ ...\ 1,\ 1,\ 1]$$

$$First\ partition\ i = 1\ \rightarrow w = [w_1^{(1)},\ w_1^{(1)},\ w_1^{(1)},\ ...\ w_1^{(2)},\ w_1^{(2)},\ w_1^{(2)}]$$

$$\textit{Second partition } i = 2 \ \rightarrow w = [w_1^{(1)} w_2^{(1)}, \ \ldots \ , w_1^{(1)} w_2^{(1)}, \ w_1^{(2)} w_2^{(2)}, \ \ldots \ , \ w_1^{(2)} w_2^{(2)}]$$

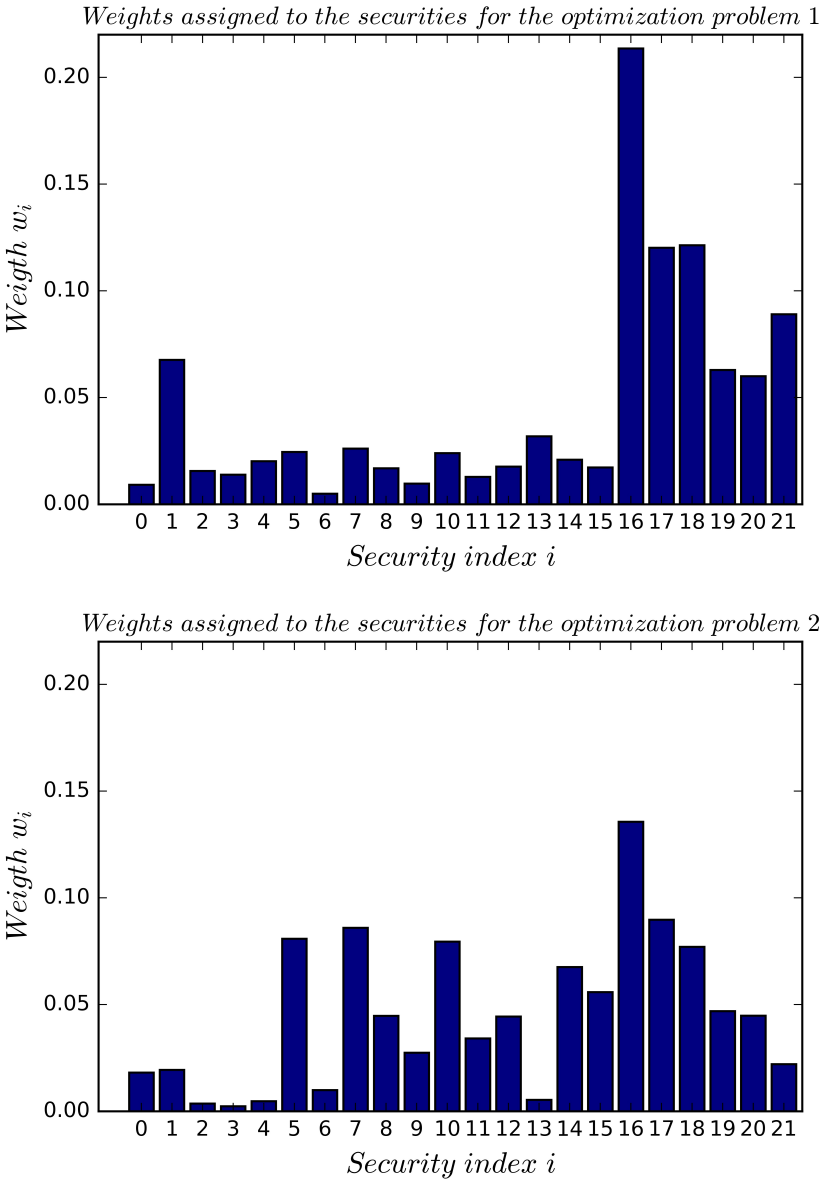
$$\ldots$$

$$\ldots$$

$$\textit{kth partition } i = k \ \rightarrow w = [w_1^{(1)} w_2^{(1)} \ldots w_k^{(1)}, \ \ldots \ , \ w_1^{(2)} w_2^{(2)} \ \ldots \ w_k^{(2)}]$$

The algorithm stops until $|L_i^{(t)}| = 1 \ \forall i = 1, 2, 3 \ldots k \ t = 1, 2$. In such a case we will find that $||w|| = 1$ is the vector of the weights assigned to the securities.

**Exhibit 4. Weights behavior**




From the histograms it can be seen that the optimization problem 1 gives more weight to the securities in the right and the optimization problem 2 diversifies giving more weight to some securities in the left and also giving weight to the securities in the right, however both give the highest weight to the same security.

# References


[1] *LEY DE LOS SISTEMAS DE AHORRO PARA EL RETIRO* **2007**, *https://www.gob.mx/cms/uploads/attachment/file/63042/normatividad_ley_sar.pdf* , pp. 26-27.

[2] *DISPOSICIONES DE CARÁCTER GENERAL EN MATERIA DE OPERACIONES DE LOS SISTEMAS DE AHORRO PARA EL RETIRO* **2017**, *https://www.gob.mx/cms/uploads/attachment/file/242827/CUO_CONSAR_COMPILA_20170704.pdf*

[3] Mohammed J. Zaki & Wagner Meira *Data Mining and Analysis: Fundamental Concepts and Algorithms* **2014**, *Cambridge University Press.*[Tesis].

[4] Daniel Müllner *Modern Hierarchical, Agglomerative Clustering Algorithms.* **2011**, *Stanford University Department of Mathematics.* http://math.stanford.edu/~muellner [Tesis].

[5] Harry Markowitz *Portfolio Selection* **1952**, *The Journal of Finance* http://links.jstor.org/sici?sici=0022-1082%28195203%297%3A1%3C77%3APS%3E2.0.CO%3B2-1 [Tesis].

[6] Marcos López de Prado *BUILDING DIVERSIFIED PORTFOLIOS THAT OUTPERFORM OUT-OF-SAMPLE* **2015**, *Lawrence Berkeley National Laboratory* [Tesis].