NONPARAMETRIC MARKOWITZ OPTIMIZATION

Jon Huml

1 Introduction

In contrast to the standard two-moment Markowitz portfolio optimization, this paper proposes an optimization method rooted in nonparametric probability density functions so as the capture all moments of a specified investor return target distribution. Daily return data is binned, smoothed, and used as an input for a nonlinear programming problem that minimizes Kullback-Liebler divergence of the specified investor return target distribution and a calculated portfolio distribution. The methodology examines a risk parity technique to bolster potential shortcomings of this optimization process.

This paper makes no claims as to the exact nature of the investor specification. The usefulness of the methodology is precisely in its flexibility concerning investor beliefs about market returns, as well as its applications to financial engineering. This paper explores the latter implementation, recreating the Dow Jones Industrial Average (DJIA) with just a handful of assets.

2 Motivation and Prior Literature

The motivation behind this paper is twofold. The first addresses the flaws in the normality assumption and the use of variance as a measure of volatility. Normality can be a reasonable approximation to returns that renders variance a meaningful quality, though empirical evidence supports a heavy-tailed nature of return distributions [1]. The interpretation of variance is less meaningful in the face of non-symmetric distributions unlike the Gaussian bell curve. Focusing on the first two moments might be correct on average. More relevant questions in modern portfolio theory, however, may not be concerned with average correctness but on the tails (and the higher order moments, in context) of any given distribution. Taking the standard two-moment utility function as an axiom, we can see that a 10% loss is not equivalent to a 10% gain. For skewed or heavy-tailed distributions, this utility is not accurately represented. Significant daily losses are more likely in observed return distributions than the normal approximation by implication of the heavy-tailed nature. We thus look to recalculate density functions that are not necessarily normal. As the number of assets increases, normality becomes true at a price to fund managers. The main focus of this paper will be to recreate the distribution of the DOW30 or equivalent with a small handful of assets so as to decrease transaction and labor costs for fund managers. We propose a question: how can we recreate or fit a specified probability density function using a linear combination of assets? The paper presents a set of steps to answer this question building off of the motivation of Lassance [2], who proposes a method to capture all moments of a desired portfolio.

The choice of using a probability density function is part of the second motivation, which is to make an easily visualized method of optimization. Density functions are intuitive and require little explanation to clients. This upside is, of course, in addition to their usefulness in capturing the entire distribution of asset returns. Investors can create any specified probability density function that they seek to mimic, and optimize with respect to this function. This implies that investors can use their own beliefs about return distributions in a nonparametric fashion. This also allows financial engineers to recreate all moments of desired indices for index funds.

3 Data

The only data required for the proposed methods are the daily returns for the Dow Jones constituents, as well as the changes in DJIA itself. There are numerous sources for this data, none of which are necessarily superior to the others. This paper uses the Yahoo Finance API and the R package "tidyquant" to obtain the prices for each ticker and convert to log returns.

The data here are chosen, daily, for six years from January 2014 to January 2020. This time frame was chosen to capture the average length of one full business cycle, plus one year to allow for changes in market conditions.

Two notable changes happened in the DJIA during this time period. After 110 years on the index, General Electric (GE) was dropped from the DJIA in 2018. Given that the price of GE more than halved from January 2014 to January 2020, GE is simply dropped for from the dataset. Only two stocks in the dataset had negative average daily returns (IBM and XOM). A company in steep decline like GE is not representative enough of the DJIA to warrant keeping or changing the time period over which the data is pulled. Since the index itself decided that the company does not accurately reflect the current state of the largest and most viable companies in the economy, this change seems justified.

Another series of events concerns the DuPont-Dow Chemical merger in 2015. In 2019, DowDuPont split into three companies. Given the difficulty of arranging for these different splits, this paper elects to drop the ticker DOW (not to be confused with the DJIA) from the dataset instead of inferring null values with the mean or median value of the returns, for example. This dropping is less justified than that of General Electric. However, it must be noted that the current DOW average daily returns are roughly on par with the median of the entire dataset. A 3% information loss is acceptable enough to save focus for the optimization process.

4 Methodology

There is often a trade off between simplicity and rigor at the cost of computational complexity. This papers opts for the former. There are a variety of ways to fit probability density functions from empirical data. The most optimal method for the use of nonparametric returns is some form of kernel density estimation (KDE), which smooths over the data using a bandwidth parameter. KDE was explored for this paper (see repository) with unsuccessful results. Bandwidth selection frames an optimization problem within an optimization problem, making estimates from the second nonlinear program less stable and more difficult to obtain. Such was the case when attempted in the exploratory data analysis. Not only must the divergence between the specification and computed return densities be minimized, but the loss for any given estimated density must be minimized as well during bandwidth selection. A simple rule of thumb for bandwidth selection, such as that of Silverman, may ease the cross-validation process but simultaneously make assumptions about the underlying structure of the data.

A simple, perhaps more crude method of overcoming the meta-optimization is to simply bin the data according to some pre-specified rule and optimize on the weights of the returns based on these binned frequencies.

4.1 Binning

Of course, binning itself requires some assumptions (not necessarily about the actual data) and approximations. However, the choice of kernel or bandwidth rule of thumb itself may infringe on our motivation to construct nonparametric densities, so we oblige to such assumptions while benefiting from their computational simplicity. The binning method is as follows:

1. Determine the number of bins by Rice's Rule ($b = \lceil 2\sqrt[3]{n} \rceil$)
2. Determine the step size s (Range of daily return data / b)

The binning algorithm iterates down the column, across the rows, and through each of the pre-determined number of bins b. The daily return must be greater than the minimum plus s * (b - 1) and less than the minimum plus s * b to fall into bin b, which indexes from one. After counting each value and sorting into an array of size $b \times n$, the frequencies of each bins can be easily computed by element-wise division of this counted matrix by the length of the data, which is the number of days.

The motivation for the binning method, at the cost of some accuracy, will become apparent during the computational process.

4.2 Smoothing

The motivation of KDE is to output a "smoothed" probability density function from empirical data. The binning method presented in this paper requires a non-analogous form of ϵ-smoothing, not to be confused with the smoothing of KDE. The binning motivation behind such ϵ-smoothing is not necessary for the actual frequency binning unlike KDE, but for the actual optimization process concerning Kullback-Liebler divergence given that logarithms are undefined at zero frequencies. If we take the bins of the two distributions as sets, the investor specification S_I and the computed density S_C, where $\neg(S_I \cap S_C)$ has some element, the distributions will infinitely diverge and the nonlinear program will be unbounded. We thus apply absolute discounting as a smoothing method to investor-specified and calculated probability density functions I(x) and C(x), respectively, introducing ϵ (an arbitrarily small positive constant; in this paper, taken to be 0.00001) and redefining the previous probability density functions as I'(x) and C'(x). Each value in the set can be regularized according to an iterative process.

First, define the union of the sets S_I and S_C as S_{IC}. Each element in I'(x) is then regularized by ℓ_i (new value = old value - ℓ_i) defined as

$$\epsilon * (|S_{IC} - S_I|/|S_I|)$$

Similarly, for Q'(x), we define ℓ_c

$$\epsilon * (|S_{IC} - S_C|/|S_C|)$$

Notice that all elements of $\neg(S_I \cap S_C)$ will be in the differences of the set S_{IC} and the sets S_I or S_C, respectively. Thus, all values that are not in both I'(x) and C'(x) will simply be assigned the value of the smoothing parameter ϵ. For example, consider a simple 4-bin density function with one zero value. Since we add one ϵ, the above formulas will subtract ϵ / (4 - 1) from each of the three nonzero probabilities, leaving our original density function equal to one but with no zero values. The optimization problem is now guaranteed to converge.

4.3 Optimization

The nonlinear program seeks to minimize the relative entropy or Kullback-Liebler divergence between a specified investor return distribution, I(x), and a computed portfolio, C(x). If an investor believes that market returns are normally distributed, then the specification could be a simple two-moment Gaussian distribution. The methodology goes beyond theory, however. Empirical distributions can be extracted and fitted for financial products such as index funds. By minimizing the area between two distributions, it is possible to mimic the daily return behavior of an index like the DJIA with far less than 30 assets. The objective function, which is the Kullback-Liebler divergence of two probability density functions, is as follows:

$$\min D_{KL}(P||Q) = \int_{x_a}^{x_b} P(x) log(\frac{P(x)}{Q(x)}) dx$$

The only necessary constraint we add is that the sum of the weights must equal 1 (fully invested). While P(x) is specified by the investor, Q(x) contains the weights that we rebalance to minimize the divergence. This can be formulated as simple matrix multiplication *Ax*=*b*, where *A* is our return frequency matrix, *x* is our weight matrix, and *b* is Q(x) itself.

$$\begin{bmatrix} r_{11} & r_{12} & \cdots \\ \vdots & \ddots & \\ r_{K1} & & r_{KJ} \end{bmatrix} \begin{bmatrix} w_{11} \\ \vdots \\ w_{K1} \end{bmatrix} = \begin{bmatrix} b_{11} \\ \vdots \\ b_{K1} \end{bmatrix}$$

As calculated earlier in §4.2, *A* makes no reference to any specified distribution. Empirical distributions can easily be fit to known distributions in Matlab. Since the motivation behind this methodology is to avoid making assumptions such as normality, however, the optimization problem is less straightforward. The desired output is not a transformed probability distribution, but the weight matrix *x* that will reshape a combination of assets to minimize KL divergence. For the task of fitting the DJIA, *J* will equal 30. *K* is the desired number of bins, which can be calculated in any arbitrary number of ways.

We may also experiment with flipping certain weights on and off by adding binary constraints on the number of allowable chosen assets. For this, we may simply add another matrix $y \in \{0, 1\}$ where the sum of y_i's must equal some constant less than 30. It should be noted, however, that nonlinear programs in the Matlab Optimization Toolbox cannot add integer constraints directly. The combinations of assets can be selected as $\binom{N}{k}$, with $N = 30$ and k equal to the desired number of assets. Of course, for a larger index like the S&P 500, this methodology becomes intractable. This paper will present a risk parity method to alleviate the factorial time of large indices as well as smaller ones. Even for the DJIA, the maximum number of combinations (155,117,520) occurs at 15 assets. 155 million optimizations requires much more than 155 million calculations. However, as noted in the findings, we may be able to fit the distributions using far less than even 15 assets.

The differences in optimization techniques between Markowitz and density estimation here become apparent. The combinations in density estimation make no reference to any covariances between the assets, whereas such information is a key insight of modern portfolio theory. With the normality assumption and a symmetric distribution, risk can be equated with variance (the second moment) without much explanation needed. Certain modifications, such as adjusting for downside risk (Sortino ratio instead of Sharpe ratio) can be made to remedy the obvious flaw that upside "risk" is desirable, not to be penalized. In a nonparametric distribution, as assets are usually found to conform to, this becomes a treacherous assumption.

This paper centers on investing from an optimization perspective, but future work with a similar motivation might use kernel density estimation, a methodology that can account for covariances along the dimensions (which are the assets themselves). This is in contradiction to our goal of making an easily visualizable method. With two or three assets, the combined probability distribution can be understood in two or three spatial dimensions. Beyond this, visualization becomes more difficult and the specification becomes almost meaningless. Investors cannot specify return distributions in multidimensional space beyond combinations of known distributions like the Gaussian, where the combinations themselves are *not* Gaussian. By using the binning methodology instead of this extension, we sacrifice accuracy in favor of explainability and ability to visualize the results.

4.4 Hierarchical Risk Parity with Unsupervised Learning

The factorial time of the selection process for optimization is unacceptable for personal computing purposes. A risk parity process that filters or separates stocks based on risk profiles can significantly reduce the computational complexity and improve the fit to the desired index. For investors using this methodology to construct a hypothetical distribution that seeks to maximize returns, this may also filter out assets with undesirable distribution profiles. Consider an asset such as Exxon (XOM), which has a negative average daily return and above-average variance in the examined time period from January 2014 to January 2020. Such an asset might be relevant for recreating an index distribution, but should be left out of a portfolio that seeks to put capital to best use.

A hierarchical process may also assist to modify foundations of mean-variance optimization, not disregard them entirely. Although variance may not be mathematically precise for nonsymmetric distributions, the findings presented in this paper will show that even after just a few assets the combination converges to approximately normal. Underlying covariances (not accounted for in the methodology up to this point) may leave a potential for increased hidden risk. Decreasing the number of assets, while potentially useful for decreasing transaction and labor costs for fund managers, will decrease diversification. For an index fund, diversification is precisely part of the appeal to many potential retail investors. By filtering stocks based on risk profiles, a number of different risk profile assets can be chosen in tandem to decrease unnecessary risk and converge to systematic risk at a faster rate.

In any use case, clustering is useful for computational complexity. Consider a 15-asset portfolio for the purpose of fitting the DJIA. The brute force methodology will perform $\binom{30}{15}$ or over 150 million optimizations. A clustering process that finds 15 different risk profiles and chooses one asset from each risk profile will perform $\binom{15}{1}^5$ or about 17,000 optimizations instead, about 200 times smaller of a problem.

4.4.1 K-Means

Because the focus of this paper is on the optimization process, and that screening can be done in any number of ways, we give a short explanation of the unsupervised learning method, k-means clustering, that was used to assign certain risk profiles in this paper. The objective is to form k centroids, which are themselves the risk profile classes, that minimize the within-cluster sum of squared errors. This is logically equivalent to maximizing between-cluster sum of squared errors. The function is as follows:

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 \tag{1}$$

No extra data is required for the clustering method. The risk profiles are calculated from the first four moments of the daily return data. Thus, the data is four dimensional along the mean, variance, skewness, and kurtosis. The intermediate step to find an optimal k leads us to a notable, empirical difference between the nonparametric optimization motivation and the standard mean-variance optimization process. To visualize the four-dimensional dataset of the four moments computed from the daily return data, a simple principal component analysis (PCA) shows the following:

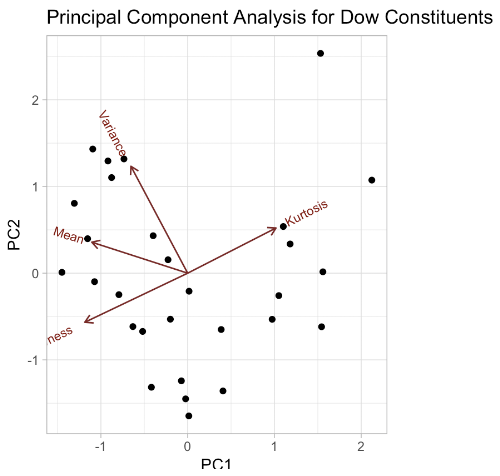


Figure 1: First two dimensions of the four-dimensional moments dataset

The first two components, PC1 and PC2, account for about 70% of the cumulative variance of the data in four dimensions (not to be confused with the *asset* moment of variance). In contrast to simple mean-variance optimization, PCA shows the need for higher order moments to capture the full picture of the return distributions. Indeed, this risk parity uses only the first four moments of infinitely many. While each successive moment captures less of the most important dimensions of the data, the methodology that captures an entire probability density function is motivated by these findings.

PCA helps us to visualize the results of selecting k. To find an optimal number of centroids, we look to minimize error while selecting as few centroids as possible (this is just another way of formulating the function given in 1). We thus look for the "elbow" of the function in Figure 2, which seems to occur around $k = 5$.

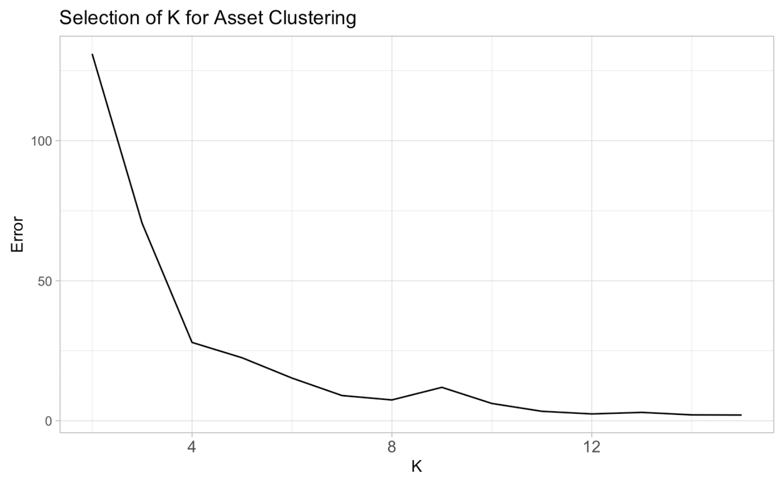


Figure 2: The "elbow" of the function seems to occur around the 5th iteration

Five centroids does not give equally sized profiles. The profiles range from 2 to 10 assets, with most profiles around 6 assets. An extensive treatment of each classification for the constituents is given in the code itself. For the purpose of extracting results for the DJIA, we use a standard procedure to build portfolios from the clustering method. If the size of the calculated portfolio is between 2 and 5, we randomly pick the index number of one of the five centroids, and pick one stock from each index number. Thus, not all profiles will be picked but all profiles that are picked will have only one asset each. If the size of the calculated portfolio is 5, we randomly pick one asset from each centroid. If the size of the calculated portfolio is greater than 5 and less than 10, we equally divide the number of assets divisible by 5 and randomly assign the indices of the remaining portfolio size.

The maximum portfolio size of 10 is chosen for a number of reasons. For the sake of comparison to the combination method, the combination method becomes quite slow at a certain portfolio size. Furthermore, we should hope to recreate the DJIA with as few assets as possible in keeping with the motivation to decrease transaction and labor costs.

5 Findings

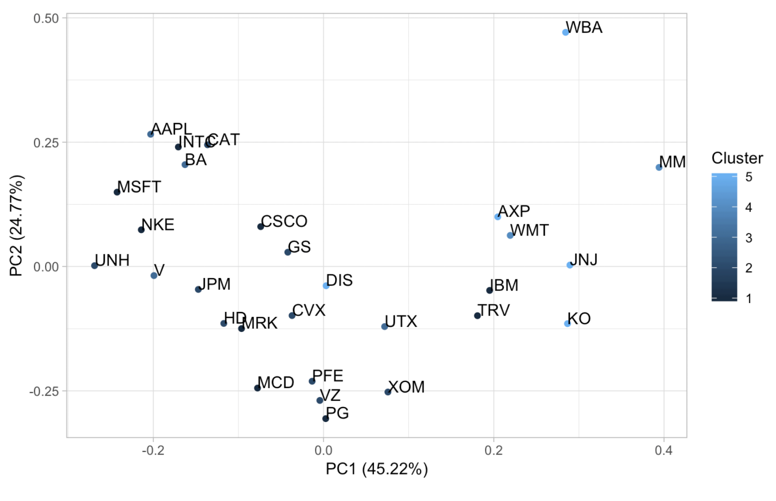


Figure 3: Clustering

The asset clusters are shown in Figure 3 within the context of the principal component analysis in the previous section, which squeezed the first four moments of each asset distribution to two dimensions, explaining roughly 70% of the dataset "variance" (not to be confused with the second moment of the asset probability distributions). The process affirms much of our current intuition about the composition of the DJIA itself in a mathematical way. Companies like Apple and Microsoft occupy roughly the same quadrant of the principal components, as do JP Morgan and Goldman Sachs, for example.

With the clusters in hand, we are able to evaluate the methodology concerning the risk parity method and the brute force method. The risk parity method is characterized by an extra binning step that uses the above clusters as a basis for the nonlinear program. The brute force method simply runs $\binom{30}{k}$ combinations of assets for portfolio building. In light of the desire for computational efficiency and the ability to run this code on a personal computer, the brute force method looks especially inferior. Past 7 assets, the optimization process was simply too slow to be calculated within a span of days on a reasonable amount of RAM offered in most laptops or desktops. The trend for the Kullback-Liebler divergence was simply extrapolated and predicted based on past trends after the seventh iteration for the brute force methodology for Figure 4, which shows the average Kullback-Liebler divergence of each successive iteration of portfolio size.

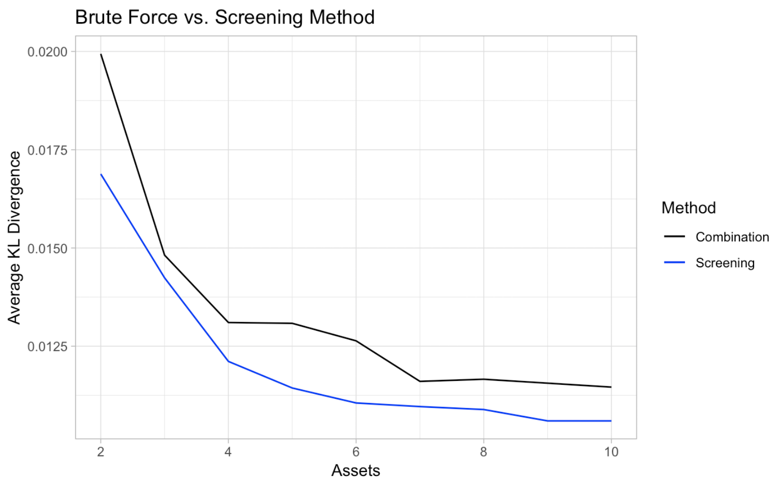


Figure 4: Convergence of methods. Note that past 7 assets, the average KL divergence for the combination method is inferred from previous rates due to the slowness of computation

The two methods clearly differ in their rate of convergence to a minimum average KL divergence. Coincidentally, the elbow of this function also seems to occur around 5 assets for the screening method, and around 4 assets for the brute force combination method. Adding more assets mostly continues to decrease the average value of the objective function from the nonlinear program, but at a decreasing rate.

The risk parity method outperforms the combination method. This is not necessarily an endorsement of the clustering model used in this paper. Rather, the finding motivates the necessity of using some grouping method in the pre-processing stages of the optimization process. Even a simple grouping by sector or industry will suffice. For example, while Apple and Microsoft are not in the same cluster, their respective cluster distances seem close in two dimensions.

Another important note in the differences between the two methods is the use of randomness for algorithmic efficiency. The risk parity method groups assets by distributional profile and randomly selects an number of assets from each profile in the most equal fashion possible. An arbitrary number of 200 random combinations for the screening method (constant across all portfolio sizes) still vastly outperforms the factorial time of the combination method. In a less-sightly three dimensional graph that accounts for the computational complexity of each method, the screening method would separate itself in greater superiority to the simple combination method.

5.1 Instability and Local Minima

A significant shortcoming of the optimization process concerns the stability of the weighting scheme of the nonlinear program. There are a variety of potential reasons for this, all of which may have a non-zero degree of influence.

5.1.1 Pre-Processing

The first reason concerns the pre-processing steps presented in the binning and smoothing sections. The assumptions and sometimes arbitrary choices made, especially for the binning technique, may have presented a significant amount of information loss. The true distributions are biased in some direction that may affect certain distributional profiles more than others. However, while the information loss may have been more significant than previously thought, the uniform application of the binning process to all assets should present a consistent form of bias. Since the DJIA was also binned according to this process, we should expect a relatively constant information loss. The binning process is also quite similar to built-in Matlab functions. Indeed, the only true difference from an integrable function itself (the empirical representations are closer to a piecewise function) is that the number of bins does not approach infinity. When increasing the bin count parameter, we do not see an appreciable difference in objective values (about a 3% difference). Increasing the bins seems to help improve the rate of convergence, but not the actual instability of the weights. With smaller bins, the local minima hazard likely *increases* since the frequencies at each bin become smaller. Since the weights are based on the frequency bins, the smaller frequencies resulting from more bins will make the weights more sensitive. The motivation of Rice's Rule for the binning process (similar to other choices like Sturge's Rule) thus seems unlikely to have an inordinate amount of influence on the weight instability.

5.1.2 Objective function

The objective function may be a source of instability. As noted in §4.3, Kullback-Liebler divergence is a continuous measure of relative entropy between two probability distributions. Again, we must approximate the continuity given the empirical nature of the data. This should not present too much difficulty since continuous functions are often often approximated by Taylor series in calculators or computers anyway, for example. The expected results of the Kullback-Liebler divergence should show that, as the number of assets increases, the fit of the calculated and specified probability distributions eventually become the same function. This does not occur at times for multiple reasons. Here, we examine an inexplicable subset of results from the combination portfolios. The exact method of iteration on the measure of goodness of fit seems highly unstable as given below in Figure 5.

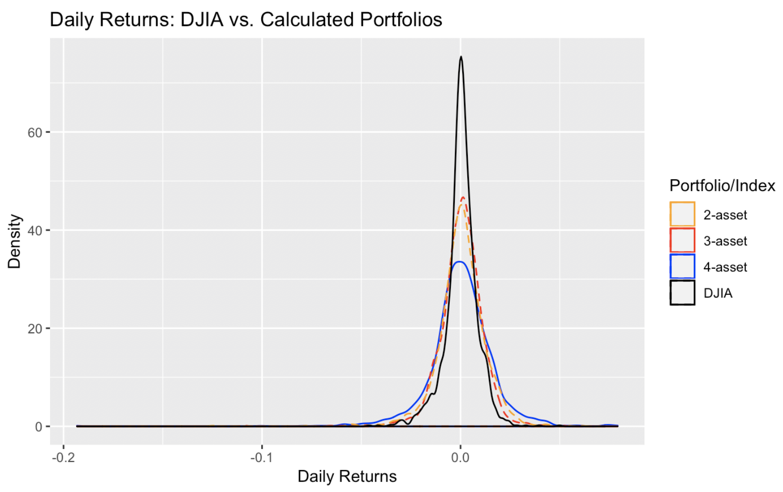
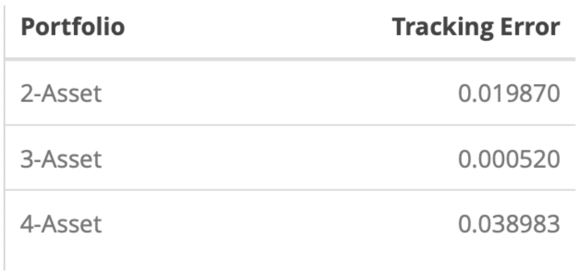


Figure 5: An odd outlier in the fittings

We select the lowest divergences of all combinations of assets for the 2-asset and 3-asset portfolios. The 4-asset portfolio is selected from all combinations of assets such that its divergence is only 15% lower than the 3-asset and 52% lower than the 2-asset. While we would expect superior or at least comparable performance, perhaps proportional to the differences in divergences between the assets, the resulting figure below is wildly different. The 4-asset portfolio captures the distribution far worse than the preceding portfolio sizes. This differs from the overall trend that shows mostly decreasing tracking errors as the portfolio size increases, as one would expect. This odd result is shown below in Figure 6.



Portfolio	Tracking Error
2-Asset	0.019870
3-Asset	0.000520
4-Asset	0.038983

Figure 6: Tracking error ratio for the outlier portfolio as compared to the DJIA (annualized)

5.1.3 Starting Points

The objective function and pre-processing steps each influence the stability of the weights to some degree. The greatest influence is likely the starting point in the optimization process.

This can be tested easily. For all optimizations in this paper, we started with an equal weighting scheme: $1/n$ for n assets. We can test the stability of the starting points by fitting the index to itself with 30 assets, formulating a $\binom{30}{30}$ process, and changing the initial weighting scheme. For the task of fitting the DJIA, we already have the known asset weights based on the pricing model of the index. These are calculated by dividing each asset price by the DJIA, which is calculated by $\frac{\sum p_i}{d}$, where p_i is each asset price and d is the Dow divisor (updated frequently), most recently given by 0.14744568353097. The error between the weights of the optimization process and the known weights should be zero if the optimization process is able to perfectly recreate the DJIA.

Using all 30 assets should give close approximation to the actual Dow. However, as shown in the code, some of the asset weights are negative. No shorting is allowed in the Dow, so this is incorrect. The root mean squared error of the two weight vectors, one calculated from the optimization process and the other from the Dow weights of the last trading day of the time period in the data, is 0.0461.

We reformulate the problem by starting with all weight in one asset. The first asset in this case is Apple, so we simply assign a weight of 1 to AAPL. The resulting solution is highly skewed. Most weights remain close to zero after the optimization, with over 60% of the weight in AAPL. The root mean squared error between the real Dow weights nearly quadruples. Thus, the weighting scheme seems to heavily influence the outcome of the optimization process. The objective function seems to be quite "hilly" in that there are many local minima. Where we place the "ball" to roll down the error function matters significantly for minimizing the Kullback-Liebler divergence.

5.2 Comparisons

Despite the shortcomings of the methodology associated with the local minimum problem, the results are still promising enough to warrant further inspection of density optimization. As seen in Figure 6, even with just three assets, the methodology approaches a tracking error of just 0.052%.

Given that a tracking error should approach zero for the purpose of recreating an index fund, the ability to roughly recreate the probability density function of a 30-asset index with just one-tenth of the assets shows the potential of the method.

Risk factors associated with the active pursuit of decreasing diversification may penalize smaller portfolios, however. One difficulty with a risk (or covariance) adjusted tracking error is that the nonparametric nature of these asset returns is an argument against the use of variance in the optimization process in the first place. If we were to use such an error, we might as well use a traditional mean-variance optimization process since this will address a similar goal. Within the context of the method's goal to recreate the probability density function of an index, the proposed process performs quite well. Within the context of traditional Markowitz portfolio optimization, the results may seem irrelevant.

6 Conclusion

This paper presented an optimization process to minimize Kullback-Liebler divergence between an investor-specified and a calculated portfolio probability density function. Given that asset returns have long been known to be "approximately normal," we make no reference to any known distributions. The methodology presented is nonparametric. Daily return data was binned into return frequencies, smoothed as to guarantee convergence of the nonlinear program, and screened into risk profiles using k-means clustering. The grouped risk profiles were randomly selected to form portfolios of (roughly) equal-sized proportions from these risk profiles and used as input into the nonlinear program. The computational load was not only lightened by the screening process, but also converged to a minimum Kullback-Liebler divergence faster than a brute force combination of assets in factorial time.

The results suffered from the local minimum problem. Some outliers show inexplicable results relating to different aspects of the problem formulation, including the pre-processing stages, objective function, and starting points for the portfolio weights.

The findings were still promising, however. The tracking error approaches zero using just one-tenth of the assets of the DJIA. Future extensions, perhaps using kernel density estimation, may find less information loss and more precision in return frequencies. Different objective functions may be tested to minimize the differences between the desired probability density function.

References

[1] Fama, Eugene. *The Behaviour of Stock Market Returns.* 38(1). *The Journal of Business.* Jan. 1965.

[2] Lassance, Nathan. *Information-Theoretic Approaches to Portfolio Selection.* Jan. 2020.