Fund Asset Inference Using Machine Learning Methods: What’s in That Portfolio?

DAVID BYRD, SOURABH BAJAJ, AND TUCKER HYBINETTE BALCH

ABSTRACT: Given only the historic net asset value of a large-cap mutual fund, which members of some universe of stocks are held by the fund? Discovering an exact solution is combinatorially intractable because there are, for example, C(500, 30) or $1.4 \times 10^{48}$ possible portfolios of 30 stocks drawn from the S&P 500. The authors extend an existing linear clones approach and introduce a new sequential oscillating selection method to produce a computationally efficient inference. Such techniques could inform efforts to detect fund window dressing of disclosure statements or to adjust market positions in advance of a significant fund disclosure date to earn profit from those pursuing a replication strategy after disclosure.

Other researchers have addressed related problems: Sharpe (1992) addressed the problem of inferring fund exposures to broad asset classes. Edirisinghe (2013), Chen and Kwon (2012), and others have addressed the index tracking problem, in which the constituents of the portfolio are known a priori. However, to our knowledge, ours is the first method to infer both the number of constituent assets and their identities without having any constituent information in advance. Instead, we assume knowledge only of the daily close price for each stock in the S&P 500 plus the target exchange-traded fund (ETF) or daily NAV for a target mutual fund.

The linear clones method of Hasanhodzic and Lo (2007), on which we base our extended linear clones method, addresses the problem of estimating and evaluating portfolio weights $W$ if constituency $C$ is known. Our extension is to infer
C (including its size) prior to applying linear clones to obtain W. That is, the present state of the art takes known C and estimates W. Our extension addresses the question, “What if we do not know C?”

Discovering an exact solution by exhaustively checking all possibilities is combinatorially intractable. There are, for instance, C(500, 30) or \(1.4 \times 10^{48}\) possible portfolios of 30 stocks drawn from the S&P 500. It would accordingly require millions of years using today’s highest performance computer to infer the constituents of the Dow Jones Industrial Index via exact computation. Accordingly, we have developed algorithms that find approximate solutions. Our approach leverages Sharpe’s asset class exposure method in combination with a well-known technique from machine learning, the sequential floating forward selection (SFFS) algorithm (Somol et al. 1999), to solve these problems in a few seconds. We demonstrate the approach in a series of example problems in which we task the algorithm with inferring the constituents of ETFs for which the components are known. Because the motivation is to predict holdings of mutual funds, we treat the ETFs as a demonstrative proxy for other fund types, limiting ourselves to a single end-of-day NAV and periodic holdings disclosures, despite richer data being available for ETFs. Depending on the details of the specific problem, the algorithm runs on consumer hardware in 8 to 15 seconds and identifies the constituents of the target portfolio with an accuracy of 88.2% to 98.6%.

BACKGROUND AND RELATED WORK

In this article we use a novel adaptation of the SFFS technique (Somol et al. 1999) to infer the holdings of a fund for which daily NAVs are known. A key improvement our method provides is fitting a model of individual equity constituents rather than a broad clone based on risk factors or asset classes. To our knowledge, this is the first such solution. However, a number of related papers have given us valuable insight into the general class of problem.

Large hedge funds (over $150 million in assets under management) and all mutual funds are required by the Securities and Exchange Commission to report their significant holdings every quarter. Hedge funds not subject to mandatory reporting may also disclose some or all of their holdings as a matter of activism (Agarwal et al. 2015). The timing of these disclosures has been shown to be correlated with increased trading volume in the disclosed assets and a permanent price impact as the trading public rushes to buy or sell those equities added or dropped by high-performing funds (Croci and Petrella 2015; Agarwal et al. 2015). The ability to infer fund holdings from publicly available data prior to an announcement could allow a trader to enter or exit a position with less risk, greater potential reward, and no insider trading concerns (Frank et al. 2004).

Kacperczyk, Sialm, and Zheng (2006) studied the impact of unobserved actions in mutual funds. They calculated a projected return based on previously disclosed holdings and compared it to the fund’s actual return, considering the difference (the return gap) as a measure of value added by the fund manager. Their work evaluated the correlations among the return gap, hidden costs such as transaction fees, and hidden gains such as interim trades. It also explored the possibility of using this return gap to predict mutual fund performance.

Meier and Schaumburg (2004) identified confounding factors when using portfolio holdings as a measure of performance, including the lack of detailed enter/exit dates for holdings; lack of a model for holdings both opened and closed between disclosures; and the potential for funds to window-dress their portfolios, deliberately adjusting the contents around disclosure time to mask the fund’s real investment strategy. Previous efforts have evaluated the effect of asset disclosure, whereas we aim to provide algorithms to predict the contents of the disclosure.

Frank et al. (2004) analyzed copycat hedge funds, which mimic the holdings of actively managed funds, by analyzing the long-term difference in returns between the two. Our study differs in that we aim to infer pre-disclosure holdings and earn a profit from the impact the disclosure will have on the underlying equities, whereas Frank et al. (2004) executed post facto trades to mimic the target fund.

Sharpe (1992), from whom we draw inspiration, explored asset class factor models in which he decomposed mutual funds into asset classes such as growth stocks, bonds, large cap, and so on. We use a similar approach but proceed to the more granular level of individual equities. Fung and Hsieh (1997) applied the same technique to hedge funds using principal component analysis (PCA) to categorize the components. PCA produced well-fitting component axes but still did so at the
level of investment styles and asset class mixtures, rather than individual instruments.

Hasanhodzic and Lo (2007) explored the method of linear clones to perform a regression fit of certain risk classes to which a hedge fund may be exposed. Each factor in the regression represents an entire asset class, such as the S&P 500, the US bond market, or USD (similar to the earlier work by Sharpe), and the corresponding regression coefficient represents the allocation of that class. In their work, two variants were explored for the input data: using the entire fund history (called fixed-weight) or using 24 months just prior to the prediction time (called rolling window). Although most of the clones created were inferior to the actual hedge funds, the performance is similar enough to justify further exploration. Our study follows on most closely from their efforts at adapting Sharpe’s original work: One of our approaches extends their method of rolling linear clones to infer at a finer resolution—specific assets held by a fund—rather than at the more coarse level of asset classes.

Bertsimas, Kogan, and Lo (1997) proved that any general payoff strategy such as mutual funds could be replicated using more liquid instruments. Kat and Palaro (2005) explored the possibility of distribution-based clones. Amenc, Goltz, and Le Sourd (2009) showed that such clones could only be successful with a training period of over six years. This is quite long for hedge funds and mutual funds that are more dynamic in nature. Amenc et al. (2010) also explored nonlinear clones for hedge funds, but their performance was worse than that of their linear counterparts, hinting at overfitting or bias arising from the greater flexibility of fit.

Our effort is to fill a gap in the body of work, providing methods to directly infer the quantity and identity of constituents in a liquid fund that is priced daily, using only the daily close price or NAV of a target fund, plus the daily close prices of a universe of stocks from which constituents could have been drawn.

THE PORTFOLIO INFERENCE PROBLEM

Here we state the problem formally and introduce some notation. Our problem is as follows:

- **Given**: a target portfolio $P$ with known value over time but unknown constituents $C$, unknown allocations $W$, and a universe of candidate constituents $U$

- **Find**: a portfolio clone $\hat{P}$ with constituents $\hat{C} \subseteq U$ with allocations $\hat{W}$ that maximizes the Matthews correlation coefficient (MCC) (Matthews 1975) between $C$ and $\hat{C}$

It should be noted that although we do obtain an estimated weight vector $\hat{W}$, our primary aim in the current work is an accurate estimate $\hat{C}$ of the portfolio constituents. Once $\hat{C}$ is obtained, $\hat{W}$ can be estimated using the linear clones method of Hasanhodzic and Lo (2007) or others, as described in the previous section.

The target portfolio $P$ can be any fund for which daily prices are known. Examples include mutual funds, ETFs, indexes, or other instruments for which daily information is available. In this initial work, our objective is to infer the constituents of a long-only equity portfolio. This allows experimentation and testing with publicly available data on index-tracking ETFs that provide a suitable ground truth for testing the approach. Note that if the target portfolio is drawn from $U$, the clone $\hat{P}$ that maximizes MCC is exactly $P$.

By solving this problem, we consequently discover the number of assets in $P$, their identities $C$, and portfolio weights $W$. Our method uses in sample tracking error as a loss function to optimize $C$ and $W$. The current analysis focuses on the in-class versus out-class prediction accuracy of $C$.

APPROACH

In this section we describe the various algorithmic methods we developed to solve the portfolio inference problem. In a later section we examine their performance.

Extended Linear Clones Method

We choose the linear clones method (Hasanhodzic and Lo 2007) as a starting point to attack our problem. Recall that the linear clones method assumes knowledge of the portfolio constituents, so we must extend the approach to relax that assumption. We call our modification to the linear clones method the extended linear clones (ELC) method.

Let’s first review the original linear clones method. Recall that we intend to find a set of constituents $\hat{C}$ and their weights $\hat{W}$ that will maximize the Matthews
correlation with \( P \). The linear clones method assumes that \( C \) is known and that the only issue is to find the weights \( \hat{W} \).

The linear clones method treats this as a simple regression problem, as follows: For each day \( t \) and asset \( i \), we define \( X_i \) to be the daily return of constituent \( i \). We refer to \( X_i \) as a vector of all candidate asset returns on day \( t \). \( Y_t \) is the daily return of the target portfolio. Over a year of 252 trading days, we have 252 data points: \(<X_1, Y_t>, <X_2, Y_t>, \ldots, <X_{252}, Y_{252}>\). If we treat each day’s data as an independent observation, we can use linear regression to find the weights \( \hat{W} \) that solve the equation \( W \cdot X = Y \) subject to the constraint \( \sum W_i = 1.0 \). Daily returns are calculated from consecutive market days. This having been done, the sequence of data points given to the algorithm is now unimportant.

The linear clones method assumes that the constituent set \( C \) is known. For our problem, we do not know the constituents a priori. Accordingly, we must modify the method. We perform the same linear regression fit but allow a variable number of factor terms (each an equity in a portfolio of unknown size). Unlike Hasanhodzic and Lo, we do not constrain the sum of coefficients to be 1.0. Instead, we evaluate the coefficient sum during analysis as a test of the reasonableness of our interpretation of the coefficients, expecting that it should naturally be very near one rather than normalizing it to one. Our modification to the linear clones method is summarized as a series of algorithmic steps in Exhibit 1.

The most significant change to the linear clones method is to accommodate a portfolio of unknown size because the linear clones method assumes a fixed number of exposure factors. The computation time is very fast even for a nontrivial real-world portfolio (portfolio size 30 to 80, universe size 500 or 1000), so we simply iterate over the possible range of portfolio sizes \( 1 \rightarrow \|U\| \) and select the size that produces the least in-sample tracking error, which we define as the root-mean square error (RMSE) between series of daily NAVs for \( P \) and \( \hat{P} \). To avoid the method always selecting some weighted combination of all stocks in \( U \), we mandate that any included stock must represent at least 0.1% of the portfolio by weight.

Another challenge is that we must regress over all equities in the universe as factors to determine which are the most significant (maximum positive weights). The coefficients of the selected equities were determined in the presence of a large number of other, potentially discarded, factors. To address this issue, we run the regression one more time using only the selected top-\( N \) (\( N = \) portfolio size) significant factors. The coefficients of this second run are interpreted as the portfolio allocation weights.

### Sequential Oscillating Selection Method

Note that in the ELC method the linear coefficient for each equity is used implicitly as a measure of the importance of the asset to the portfolio. The coefficient is essentially a score, and only the highest \( N \) scoring assets are retained. During exploratory work with other approaches to portfolio inference, multiple methods of scoring an individual equity’s significance to the portfolio were examined (e.g., Coefficient \( \times \) Volatility, Coefficient \( \times \) RMSE of stock time series). In some cases, the scoring formula improved the results, but this was not the case with ELC, so the most simple Score = Weight method was retained for those studies.

A potential weakness in ELC is that the fitness of a portfolio is tested in the presence of extraneous members of the universe that will not be included in the final portfolio. For example, when selecting the best portfolio for \( N = 2 \), the regression process is performed using the full universe, with the two highest coefficients selected. If, instead, all possible combinations of two candidate constituents were independently tested, the resulting selection of \( \hat{C} \) could differ. The challenge presented by this observation is the aforementioned computational intractability of testing every possible combination of candidate constituents for every \( N = 1 \rightarrow M \).

---

**Exhibit 1**

Algorithmic Steps in ELC Method

1. Let \( U \) be the universe of candidate stocks, \( N \) be the number of constituents, and \( M \) be the maximum portfolio size.
2. For \( N = 1 \rightarrow M \):
3. Apply linear regression to candidate universe \( U \).
4. Let constituent vector estimate \( \hat{C} \) contain the \( N \) candidate stocks with largest regression coefficients.
5. Apply linear regression to revised \( \hat{C} \) to determine final portfolio weights \( \hat{W} \).
6. After evaluating portfolios of size \( 1 \rightarrow M \), select candidate portfolio \( P \) that minimizes RMSE of NAV(\( P \)) vs. NAV(\( \hat{P} \)), the in-sample tracking error.
**EXHIBIT 2**

**Algorithmic Steps in Sequential Oscillating Selection Method**

1. Initialize \( \hat{C} \) to contain the single equity from the candidate universe \( U \) that minimizes the RMSE of daily returns with respect to target portfolio \( P \). Initialize algorithm direction to forward (additive).
2. Tentatively augment \( \hat{C} \) with each currently excluded candidate equity in turn, judging fit by RMSE of daily returns of the augmented hypothesis portfolio versus the target portfolio.
   (a) If at least one augmented hypothesis portfolio scores better than the base hypothesis portfolio, retain the best augmented hypothesis portfolio as the new \( \hat{C} \) and continue from Step 2.
   (b) If no augmented hypothesis portfolio scores better than the base hypothesis portfolio, retain the previous base hypothesis portfolio and change the algorithm direction to backward (subtractive). Continue with Step 3.
3. Tentatively diminish the base hypothesis portfolio by each currently included candidate in turn, judging fit by RMSE as during augmentation.
   (a) If at least one diminished hypothesis portfolio scores better than the base hypothesis portfolio, retain the best diminished hypothesis portfolio as the new \( \hat{C} \) and continue from Step 3.
   (b) If no diminished hypothesis portfolio scores better than the base hypothesis portfolio, retain the previous base hypothesis portfolio and change the algorithm direction to forward (additive). Continue from Step 2.
4. HALT: If neither augmentation nor diminishment can further improve the portfolio, emit the current \( \hat{C} \) as the final inferred portfolio.
5. Apply linear regression to final \( \hat{C} \) to determine final portfolio weights \( \hat{W} \).

The sequential oscillating selection (SOS) algorithm was developed to improve the inferential power of the ELC method by retaining the underlying time-independent linear methods but allowing for more exploration of potential candidate portfolios. SOS is related to a family of methods from the field of machine learning used to solve the feature selection problem (Pudil, Novovičová, and Kittler 1994). That is, given a set of features that may be predictive of a future outcome, which subset of those features, when used together, provides the most predictive power?

SOS is derived from SFFS as described by Pudil, Novovičová, and Kittler (1994). SOS works by first trying each feature individually to discover which one, by itself, is most predictive. It adds that feature to the set of features to be used. It then tries each remaining feature in combination with the first feature to see which augments the set with the most predictive power. The method augments the feature set one feature at a time until results fail to improve, then diminishes one feature at a time until results fail to improve, oscillating direction until no further greedy improvement is possible.

SOS starts with an input consisting of the historical data (daily returns) of all the constituent stocks in the candidate universe \( U \) (e.g., S&P 500) individually and the target fund in aggregate for an arbitrarily selected 12-month period. The daily return of each stock is treated as a feature. The corresponding daily return of the target portfolio is assigned as the dependent variable.

In a manner similar to ELC, SOS always works from daily price changes and ignores any temporal ordering of the data. SOS tests each equity in the universe using linear regression against the target portfolio to find the single equity that most closely matches the index by RMSE. This is the initial portfolio. The forward process of the algorithm iteratively augments the current size \( N \) portfolio by each candidate equity in turn, keeping only the best size \( N + 1 \) portfolio. When no augmentation improves the RMSE over the current portfolio, the backward process is engaged, iteratively trimming the current size \( N \) portfolio by each constituent in turn, keeping only the best size \( N - 1 \) portfolio. When no trimmed portfolio improves the RMSE over the current portfolio, the forward process is engaged again. When neither process improves the RMSE, the final estimated portfolio is determined.

The SOS algorithm receives the same input data as the previously described ELC algorithm and proceeds as described in Exhibit 2.

**EXPERIMENTAL METHODOLOGY**

The ELC and SOS algorithms are implemented in Python 3.5, leveraging the numerical libraries *numpy* and *scipy* (Jones et al. 2001). Twelve months of market data and ETF constituency information are drawn from Compustat Capital IQ, Select Sector SPDRs, and Yahoo! Finance (Compustat 2018; ALPS Portfolio...
We selected several sector ETFs as target portfolios \( P \), with the S&P 500 as our candidate asset universe \( U \). The algorithms also work with target portfolios containing unknown constituents (e.g., mutual funds), but we chose sector ETFs because we can validate the output of our algorithms by comparing them with the known constituents of the ETFs.

Each sector ETF is designed to track a specific S&P sector index and accordingly contains the constituents listed in the sector index. ETFs were chosen, as opposed to sector index values, because ETFs and our historical stock price data account for dividends, whereas sector index values are based on price only.

We refer to Standard & Poor’s for lists of ETF and index constituents, which we use to judge the accuracy of our algorithms.

Experiments are executed on a BSD-based UNIX system powered by a 2.6 GHz Intel Core i7 processor and 16 GB of 1600MHz DDR3 internal memory.

### Experimental Results

Nine popular ETFs during the 12-month period beginning in October 2013 were used as inference targets. Our objective was to discover the constituents and allocation weights for an inferred portfolio \( \hat{P} \) at the end of the study period. We evaluated the accuracy of the inferred portfolio using two metrics: classification accuracy and MCC (Matthews 1975). The target ETFs included DIA (Dow Jones Industrial Average tracking fund) and eight sector ETFs (tracking S&P market sectors), listed in Exhibit 3. Runtime for ELC varied from 3.24 to 3.61 seconds, with a mean runtime of 3.38 seconds. Runtime for SOS varied from 8.82 to 15.41 seconds, with a mean runtime of 11.48 seconds.

#### Predictive Accuracy

Perhaps the simplest method with which to evaluate a Boolean classification algorithm is accuracy. Although it has limitations (discussed in the following), classification accuracy is easy to compute and has a straightforward interpretation, making it popular in the financial literature, where it is referred to as classification error in the negative case (Aitken and Frino 1996) or predictive accuracy (Edmister 1972) or prediction success rate (Henry 2006) in the positive case. In the field of statistics, it is also known as the Rand Index (Rand 1971). Terminology aside, in the general case, it analyzes the similarity of two partitions \( X \) and \( Y \) of a set \( S \). When \( X, Y \) represents the predicted class and actual class of each element in a set of predictions, it can be expressed as:

\[
R = \frac{TP + TN}{TP + 1N + FP + FN}
\]

In this interpretation, \( X \) is the set of predicted element-wise classes and \( Y \) is the set of actual element-wise classes. The two classes in the experiment are \( True \)
(the equity is in the portfolio) and False (the equity is not in the portfolio). Thus, we can define TP as the set of elements labeled True in both X and Y, TN as the set of elements labeled False in both X and Y, FP as the set of elements labeled True in X but False in Y, and FN as the set of elements labeled False in X but True in Y. R then represents the accuracy of the partitions, or in this context simply the fraction correct (Hubert and Arabie 1985).

The ELC method achieved predictive accuracy of 0.068 to 0.725 with a mean of 0.357 when using the S&P 500 as the candidate universe. The SOS method achieved predictive accuracy of 0.882 to 0.986 with a mean of 0.933 on the same universe. These results are summarized in Exhibits 3 and 4. In the mean, we found that SOS misclassified 90% fewer equities than ELC.

An issue with this simple application of predictive accuracy is sensitivity to unequal class sizes in the ground truth data, called the class imbalance problem (Japkowicz and Stephen 2002). In the face of substantial class imbalance (e.g., when the correct label for an element of the universe is True far more often than it is False), the simple accuracy calculation can produce high similarity values that do not properly account for the frequency with which each class occurs (Hubert and Arabie 1985). In our case, most equities from the candidate universe are not in the target portfolio. The proper partitioning does not come close to a 50/50 split of True and False labels, so the algorithm should predict False far more often than it predicts True.

### Matthews Correlation Coefficient

Because of the potential problems with simple predictive accuracy in the face of significant class imbalance, we chose also to assess our algorithms using the MCC (Matthews 1975). MCC assesses binary classification performance even in the face of unbalanced class size (Baldi et al. 2000) by accounting for the size of the true negative prediction set: information not captured by precision, recall, and the F-score. MCC is a contingency method of calculating the Pearson product-moment correlation coefficient and therefore has the same interpretation (Pearson 1895; Powers 2011). We follow the customary interpretation of abs(r) in which values above 0.1 indicate weak accuracy, above 0.3 indicate medium accuracy, above 0.5 strong, and above 0.7 very strong. Negative values indicate similar anti-correlation (Evans 1996). The results of our methods in terms of MCC are listed in Exhibit 4.

### Exhibit 4

**Success Rate and MCC of Predictions Per Target Symbol Per Approach**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Accuracy</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIA</td>
<td>0.645</td>
<td>0.170</td>
</tr>
<tr>
<td>XLB</td>
<td>0.068</td>
<td>0.076</td>
</tr>
<tr>
<td>XLE</td>
<td>0.725</td>
<td>0.089</td>
</tr>
<tr>
<td>XLF</td>
<td>0.451</td>
<td>0.365</td>
</tr>
<tr>
<td>XLI</td>
<td>0.489</td>
<td>0.561</td>
</tr>
<tr>
<td>XLK</td>
<td>0.429</td>
<td>0.575</td>
</tr>
<tr>
<td>XLP</td>
<td>0.074</td>
<td>0.755</td>
</tr>
<tr>
<td>XLU</td>
<td>0.144</td>
<td>0.891</td>
</tr>
<tr>
<td>XLV</td>
<td>0.186</td>
<td>0.594</td>
</tr>
<tr>
<td>MEAN</td>
<td>0.357</td>
<td>0.664</td>
</tr>
</tbody>
</table>

Notes: * ELC placed the entire universe into the candidate portfolio for XLP, resulting in an undefined MCC. We treat this as zero correlation.

The ELC method achieved an MCC of 0.000 to 0.365 with a mean MCC of 0.170, indicating a weak correlation. The SOS method achieved an MCC of 0.561 to 0.891 with a mean MCC of 0.664 (strong to very strong).

### DISCUSSION

We introduced portfolio inference, a new problem, the solution for which has a number of potential applications in finance, including detection of window dressing by fund managers and development of arbitrage strategies based on the inferred constituents of large funds. It is important to distinguish portfolio inference from the simpler index tracking problem: The objectives for each are different. In the case of index tracking the goal is to minimize tracking error, whereas for portfolio inference the objective is to accurately infer the constituents of a portfolio. When applying Boolean classification techniques, we defined the null class as those members of the universe not included in the target portfolio.

We presented two potential solutions, ELC and SOS, and we evaluated their performance along several dimensions. The ELC method, which represents a natural extension of existing index tracking solutions, is a sensible approach for determining allocations to assets to minimize tracking error, but it performs poorly at inferring portfolio constituents. According to statistical evaluations, ELC did not provide better accuracy than...
random assignment of the equity universe to the target portfolio or the null class. The linear clones method proposed by Hasanhodzic and Lo (2007) and its predecessor (Sharpe 1992) have been demonstrated to work well for portfolio performance replication and inference of exposure to a fixed number of asset classes. These methods do not seem to extend well to portfolio composition inference with a large, unknown number of asset classes. This is no failing of those efforts—they were designed to solve the allocation problem in which $C$ is already known.

The new approach presented here, SOS, provides substantially better accuracy than ELC and random assignment. SOS explores a larger subset of the candidate portfolio search space in an “intelligent” manner. As a practical example, in identifying the 30 constituents of the Dow Jones Industrial Average ETF (DIA), the SOS method assigned each member of the S&P 500 universe to the correct group (in portfolio or not in portfolio) 94.0% of the time. It predicted a portfolio of 54 constituents (correct size: 30), of which 27 were correct, and it excluded 457 stocks (correct size: 471), of which 444 were correct. In comparison, the ELM method identified 206 constituents and suffered correspondingly lower measures of accuracy. The SOS method is computationally slower than ELC, with a $3.5 \times$ runtime cost multiplier, but in typical applications it completes in as little as 10 seconds on a consumer-grade laptop, and the improvement in classification accuracy is significant.

In our evaluations, we noticed that the ELM method substantially overestimated the actual membership size of most portfolios. In fact, without any constraints applied, the method would frequently suggest that all stocks in the universe were constituents (e.g., that the S&P Consumer Staples ETF contains all of the S&P 500 stocks). Because allocations to some constituents were very small, we added a minimum inclusion threshold to the ELM process. The addition of the minimum threshold did improve performance, but it was still significantly worse than that of SOS.

Even though SOS performs much better than ELC in this task, there is still room for improvement. Anecdotally, when the experiments were repeated with a maximum portfolio size of 100 and a minimum inclusion weight of 1%, which were not considered fair initial assumptions, both methods performed substantially better. This suggests better performance could be achieved in future work with constrained weights, a dynamic rather than fixed minimum inclusion threshold, or some fairly derived hint concerning the correct portfolio size. One way to obtain such a hint would be to use knowledge of prior fund disclosures: We could presume the size will remain similar or use the previous holdings disclosure as an initialization state for the algorithm.

Another limitation is the requirement for a fund to be highly liquid and priced daily because of the periodicity of our data. This requirement could be relaxed in a future study using intraday pricing information of funds for which that is available.

**ACKNOWLEDGMENTS**

This material is based upon research supported by the National Science Foundation under grant no. 1741026.

**REFERENCES**


To order reprints of this article, please contact David Rowe at d.rowe@pageantmedia.com or 646-891-2157.
Hedge Fund Returns: You Can Make Them Yourself!
HARRY M. KAT AND HELDER P. PALARO
The Journal of Wealth Management
https://jwm.pm-research.com/content/8/2/62

ABSTRACT: The authors start with the observation that hedge fund returns are not really superior to the returns on traditional asset classes, but primarily just different, and thus that hedge funds are no longer sold on the promise of superior performance, but more and more on the back of a diversification argument. They present the critical question as the following: Is it possible to generate hedge fund-like returns ourselves by mechanically trading stocks and bonds (either in the cash or futures markets)? To answer that question, they present work which has led to the development of a general procedure that allows us to design simple trading strategies in stock index, bond, currency, and interest rate futures that generate returns with statistical properties that are very similar to those of hedge funds, or any other type of managed fund for that matter.