Conditional Portfolio Optimization: Using machine learning to adapt capital allocations to market regimes

By Ernest Chan*, Haoyu Fan, Sudarshan Sawal, and Quentin Viville.

Predictnow.ai, Inc.

First version: 2022-09-30
Second version: 2023-03-08
Third version: 2023-03-27
Current version: 2023-06-27

ABSTRACT

Conditional Portfolio Optimization is a portfolio optimization technique that adapts to market regimes via machine learning. Traditional portfolio optimization methods take summary statistics of historical constituent returns as input and produce a portfolio that was optimal in the past, but may not be optimal going forward. Machine learning can condition the optimization on a large number of market features and propose a portfolio that is optimal under the current market regime. We call this Conditional Portfolio Optimization (CPO). Applications on portfolios in vastly different markets suggest that CPO can outperform traditional optimization methods under varying market regimes.

Keywords: Portfolio optimization, machine learning, factor models.

JEL Classification: G11, G17
AMS Classification: 49M37, 68T20, 90C30, 91G10, 91G60

*Founder and CEO, Predictnow.ai Inc. Email: ernest@predictnow.ai

Regime changes obliterate traditional optimization methods

In a previous paper, we discussed a machine-learning-based parameter optimization technique we invented, called Conditional ***Parameter*** Optimization, or CPO (Chan, Belov, & Ciobanu, Conditional Parameter Optimization: Adapting Parameters to Changing Market Regimes, 2021). While CPO showed promise in optimizing the operating parameters of trading strategies, we discovered that its greatest potential lies in its potential to optimize *portfolio allocations*. We refer to this approach as Conditional ***Portfolio*** Optimization (which, fortuitously, shares the same acronym).

To recap, traditional optimization methods involve finding the parameters that generate the best results based on past data for a given business process, such as a trading strategy. For instance, a stop loss of 1% may have yielded the best Sharpe ratio for a trading strategy tested over the last 10 years, or stocking a retail shelf with sweets generate the best profit over the last 5 years. Although the numerical optimization process can be complex due to various factors such as the number of parameters, the nonlinearity of objective functions, or the number of constraints on the parameters, standard methods are available to handle such difficulties.

However, when the objective function is dependent on external time-varying and stochastic conditions, traditional optimization methods may not produce optimal results. For example, in the case of a trading strategy, the optimal stop loss may depend on the market regime, which may not be clearly defined. In the case of retail shelf space optimization, the optimal selection may depend whether it is a back-to-school season or a holiday season. Furthermore, even if the exact set of conditions is specified, the outcome may not be deterministic. Machine learning is a better alternative to solve this problem.

Machine learning can detect regime changes

Using machine learning, we can approximate this objective function using a neural network[1], by training its many nodes using historical data. The inputs to this neural network will not only include the parameters that we originally set out to optimize, but also the vast set of features that measure the external conditions. For example, to represent a “market regime”, we may include market volatility, behaviors of different market sectors, macroeconomic conditions, and many other input features. The output of this neural network would be the outcome you want to optimize. For example, maximizing the future 1-month Sharpe ratio of a trading strategy is a typical outcome. In this case you would feed historical training samples to the neural network that include the trading parameters, the market features, plus the resulting forward 1-month Sharpe ratio of the trading strategy as “labels” (i.e. target variables). Once trained, this neural network can then predict the future 1-month Sharpe ratio based on any hypothetical set of trading parameters and the current market features.

[1] Recall that while a neural network is able to approximate almost any function (Hornik, Stinchcombe, & White, 1989), there are other machine learning algorithms that can be used for this task as well. We primarily use gradient-boosted regression trees, but for simplicity we will continue to call the machine learner a “neural network”.

With this method, we “only need” to try different sets of hypothetical parameters to see which gives the best Sharpe ratio and adopt that set as the optimal. We put “only need” in quotes because, of course, if the number of parameters is large, it can take a very long time to try out different sets of parameters to find the optimal. This is especially true when the application is *portfolio optimization*, where the parameters represent the capital allocations to different components of a portfolio. These components could be stocks in a mutual fund, or trading strategies in a hedge fund. For example, in Table 1 we display the input features for one day of a portfolio optimization problem:

GOOG	MSFT	AAPL	*VIX*	*Oil 30d return*	*GDP growth*	*HML*	...
20%	60%	20%	15.3	0.456%	1.23%	4.56%	...
25%	60%	15%	15.3	0.456%	1.23%	4.56%	...
30%	60%	10%	15.3	0.456%	1.23%	4.56%	...
...							
40%	50%	10%	15.3	0.456%	1.23%	4.56%	...
45%	50%	5%	15.3	0.456%	1.23%	4.56%	...
50%	50%	0%	15.3	0.456%	1.23%	4.56%	

Control features Market features

Table 1: A 1-day slice of features table as input to the neutral network

Assuming that the features that measure market regimes (denoted “Market features” in Table 1) have a daily frequency, a 1-day slice of the features table nevertheless contains many rows (samples). Each row represents a unique capital allocation – we call them “control features”. For a portfolio that holds S&P 500 stocks, for instance, there will be up to 500 parameters (if cash is included). In this case, we are supposed to feed into the neural network *all possible combinations* of these 500 parameters, plus the market features, and find out what the resulting forward 1-month Sharpe ratio (or whatever performance metric we want to maximize) is. *All possible combinations*? If we represent the capital weight allocated to each stock as $w_s \in [0, 1]$,

assuming we are not allowing short positions, the search space has $[0, 1]^{500}$ combinations.

Even discretizing to a grid search, our computer will need to run till the end of time to finish. And that is just for one day – training the neural net will require many days of such samples that include the features and the resulting labels (forward 1-month Sharpe ratio). Overcoming this curse of dimensionality by intelligently sampling the grid is one of the major breakthroughs our team has accomplished. Intelligent sampling involves, for example, not sampling those parts of the 500-dimensional grid that are unlikely to generate optimal portfolios, or not sampling those parts of the grid that result in portfolios that are too similar to an existing sample.

Perhaps the following workflow diagrams can illuminate the process:

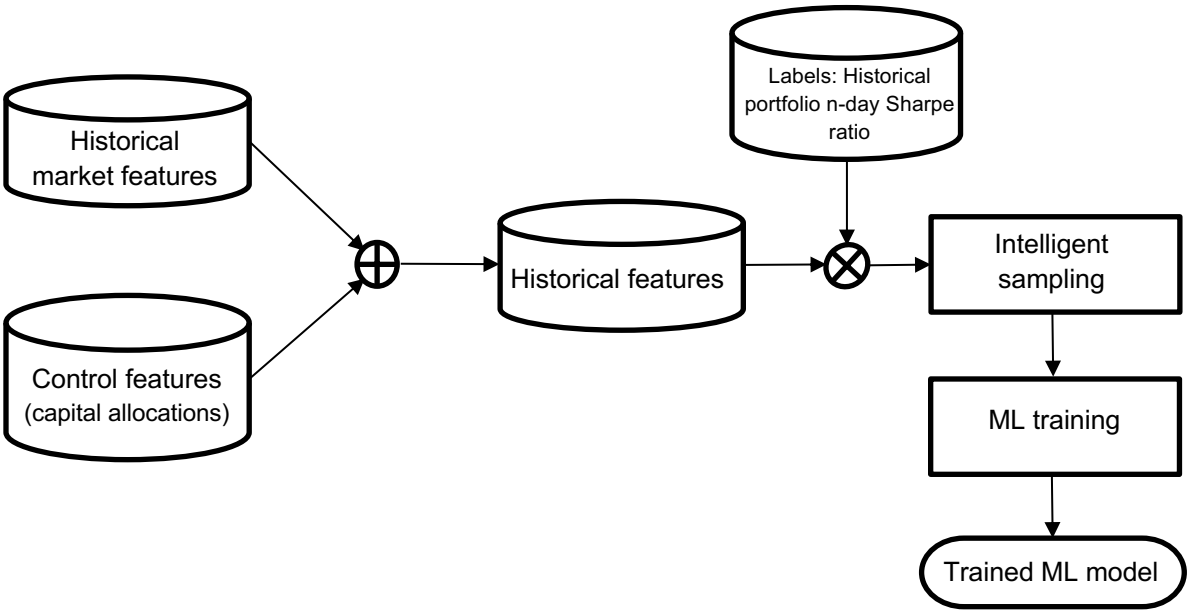

Figure 1: Training a portfolio performance prediction machine

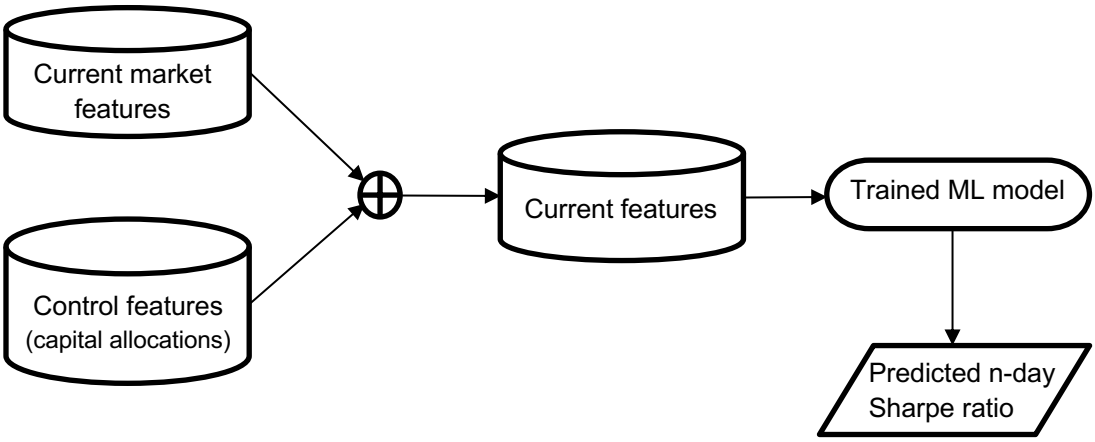

Figure 2: Live prediction of portfolio performance (inferences)

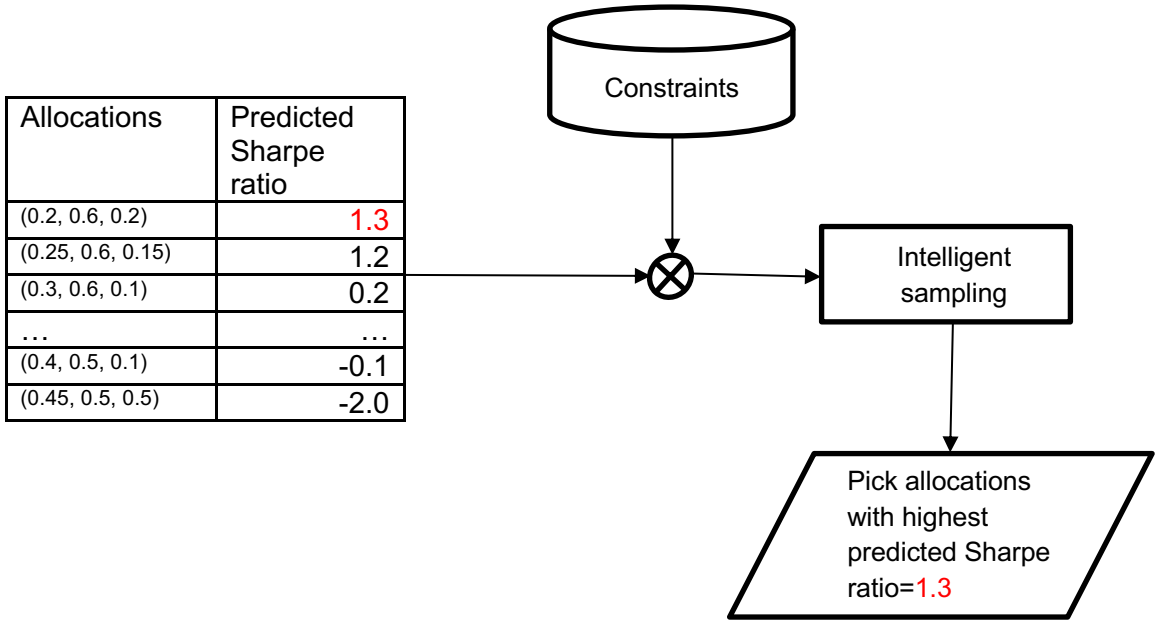

Figure 3: Optimization step

Ranking is easier than predicting

Some readers may argue that if a neural network can predict the Sharpe ratio of a portfolio based on its parameters and market features, why not use the network to directly predict the underlying assets' returns and replace the original portfolio altogether? However, with our CPO method, you don't need to predict the portfolio's Sharpe ratio accurately - you just need to predict which set of parameters give the *best* Sharpe ratio. Even if the Sharpe ratio predictions come with huge error bars, it is the *ranking* of predicted Sharpe ratio that matters. The situation is analogous to many alpha models for stock selection: to form a profitable long-short portfolio of stocks, an alpha model doesn't need to predict the cross-sectional returns accurately, but only needs to rank them correctly. We don't have an alpha model to predict or even rank such cross-sectional returns. (If we did, we wouldn't be offering that to clients as a technology product – we would offer it as a performance-fee-based managed account service.) Our client has used its own alpha model to select the portfolio components for us already. All we need to do is to rank the various capital allocations applied to a client's portfolio based on their predicted Sharpe ratios. It is a much less demanding predictive model if we only care that it gets the ranking of the labels correctly (Poh, Lim, Zohren, & Roberts, 2020).

The Fama-French lineage

With CPO, not only is it unnecessary to predict cross-sectional returns, but we also don't need to use any cross-sectional features as input to our optimizer. Instead, we only use market features that are sometimes called "time series factors" (Ruppert & Matteson, 2015). Can neural networks really predict portfolio returns without any cross-sectional features? Let's draw an analogy with the Fama-French 3-factor model (Fama & French, 1995). Recall that Fama and French proposed that we can explain (not predict) a portfolio's returns using just 3 factors: the market index return, SMB which measures the outperformance of small cap stocks over large cap stocks, and HML which measures the outperformance of value stocks over growth stocks. (Note that these factors can be negative, where "outperformance" becomes "underperformance".) The explanatory model is just a linear regression fit using these three factors as independent variables against the current-period portfolio return, hopefully with a decent R^2. But if we use these as predictive factors (or features) to forecast the next-period portfolio return, the R^2 of such a regression fit will be poor. This may be because we have too few factors. We can try to improve it by adding hundreds of factors (e.g. using our "factor zoo" (Nautiyal & Chan, 2021)), capturing all manners of market regimes and risks. But many of these factors will be collinear or insignificant and will continue to cause high bias and variance (Murphy, 2012). So finally, we are led to the application of nonlinear machine learning model that can deal with such multicollinearity and insignificance, via standard techniques such as features selection ((Man & Chan, The Best Way to Select Features? Comparing MDA, LIME, and SHAP, 2021) and (Man & Chan, Cluster-based Feature Selection, 2021)). If we also add to the input "control features" that *condition* the prediction on the capital allocation in the portfolio, we have come full circle to Table 1 and arrive at Conditional Portfolio Optimization.

Comparison with conventional optimization methods

To assess the value Conditional Portfolio Optimization adds, we need to compare it with alternative portfolio optimization methods. The default method is Equal Weights, which involves allocating equal capital to all portfolio components. Another simple method is Risk Parity, where the capital allocation to each component is inversely proportional to its return volatility. It is

called Risk Parity because each component is supposed to contribute an equal amount of risk, as measured by volatility, to the overall portfolio's risk. This method assumes zero correlations among the components' returns, which is of course unrealistic. Then there is the Markowitz method, also known as Mean-Variance optimization. This well-known method, which earned Harry Markowitz a Nobel prize, maximizes the Sharpe ratio of the portfolio based on the historical means and covariances of the component returns through a quadratic optimizer. The optimal portfolio that has the maximum historical Sharpe ratio is also called the *tangency portfolio*. One of us wrote about this method in a previous blog post (Chan, 2014) and its equivalence to the Kelly Formula. It certainly doesn't take into account market regimes or any market features. It is also a vagrant violation of the familiar refrain, "Past Performance is Not Indicative of Future Results", and is known to produce all manners of unfortunate instabilities (see (Ang, 2014) and (López de Prado, 2020)). Nevertheless, it is the standard portfolio optimization method that many asset managers use. Finally, there is the Minimum Variance portfolio, which uses Markowitz's method not to maximize the Sharpe ratio, but to minimize the variance (and hence volatility) of the portfolio's returns. Even though this approach does not maximize a portfolio's past Sharpe ratio, it often achieves better forward Sharpe ratios than the tangency portfolio! Another case of "Past Performance is Not Indicative of Future Results".

Some researchers compute expected cross-sectional returns using an alpha model and then use Markowitz's optimization by inputting these returns (Tomasz & Katarzyna, 2021). However, in practice most alpha models do not produce expected cross-sectional returns accurately enough as input for a quadratic optimizer. As we explained before, the beauty of our method is that we don't need cross-sectional returns nor cross-sectional features of any kind as input to the neural network or the optimizer. Only "time series" market features are used.

Let's see how our Conditional Portfolio Optimization method stacks up against these conventional methods.

(Note that transaction costs are not included in the following comparative studies. Our goal is to demonstrate whether CPO improves the optimal solution compared to conventional methods, and we do not expect CPO to transact more than, for e.g., the Markowitz method. In any case, the difference is a second order effect. Also, when CPO is applied to an actively traded portfolio, the transaction costs are primarily due to updating the portfolio components, not to capital reallocation. In other words, the transaction cost in that case is due to alpha generation, not to portfolio optimization due to CPO. In fact, if the actively traded portfolio is a portfolio of trading strategies, transaction cost analysis by us is impossible since those strategies aren't disclosed to the CPO algorithm.)

We test how our CPO performs for an ETF (TSX: MESH) given the constraints that we cannot short any stock, and the weight w_s of each stock s obeys $w_s \in [0.5\%, 10\%]$, but we can allocate a maximum of $w_c = 10\%$ of the portfolio to cash, with $\sum_s w_s + w_c = 1$.

Period	Method	Sharpe Ratio	CAGR
2021-08 to 2022-07 (Out-of-sample)	Equal Weights	-0.76	-30.6%
	Risk Parity	-0.64	-22.2%
	Markowitz	-0.94	-30.8%
	Minimum Variance	-0.47	-14.5%
	CPO	-0.33	-13.7%

Table 2: MESH ETF

In the bull market, CPO performed similarly to the Markowitz method. However, it was remarkable that CPO was able to switch to defensive positions and outperformed the Markowitz method in the bear market of 2022. Overall, it improved the Sharpe ratio of the Markowitz portfolio by more than 60%. That is the whole rationale of *Conditional* Portfolio Optimization - it adapts to the expected future external conditions (market regimes), instead of blindly optimizing on what happened in the past. Because of the long-only constraint and the tight constraint on cash allocation, the CPO portfolio still suffered negative returns. But if we had allowed the optimizer to allocate a maximum of 50% of the portfolio NAV to cash, it would have delivered positive returns. The dramatic effect of cash allocation will be evident in the next example.

In this example, we tested the CPO methodology on a private investor's tech portfolio, consisting of 7 US and 2 Canadian stocks, mostly in the tech sector. We call this the Tech Portfolio. The constraints are that we cannot short any stock, and the weight w_s of each stock s obeys $w_s \in [0\%, 25\%]$ and we can allocate a maximum of $w_c = 50\%$ of the portfolio to cash, with $\sum_s w_s + w_c = 1$.

Period	Method	Sharpe Ratio	CAGR
2021-08 to 2022-07 (Out-of-sample)	Equal Weights	0.39	6.36%
	Risk Parity	0.49	7.51%
	Markowitz	0.40	6.37%
	Minimum Variance	0.23	2.38%
	CPO	0.70	11.0%

Table 3: Tech Portfolio

CPO performed better than both alternative methods under all market conditions. It improves the Sharpe ratio over the Markowitz portfolio by 75% as the market experienced a regime shift around January 2022**.** Here are the comparative equity curves:

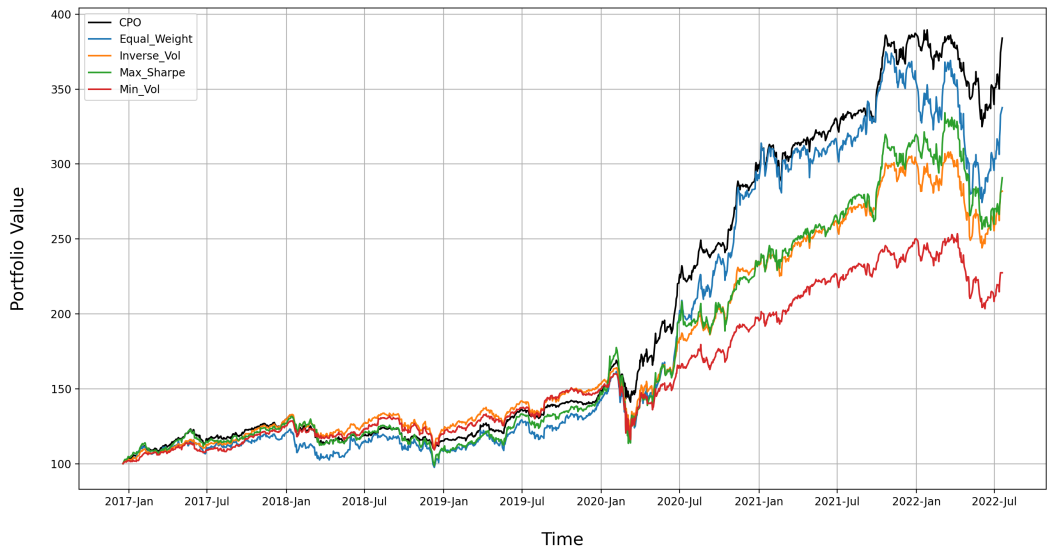

Figure 4: Comparative performances of various portfolio optimization methods on Tech Portfolio. (Out-of-sample period starts August 2021)

Even though this portfolio is tech-heavy, it was able to generate a positive return during this trying period. The reason is that it can allocate 50% of the NAV to cash, as one can see by looking at the time evolution of the cash component:

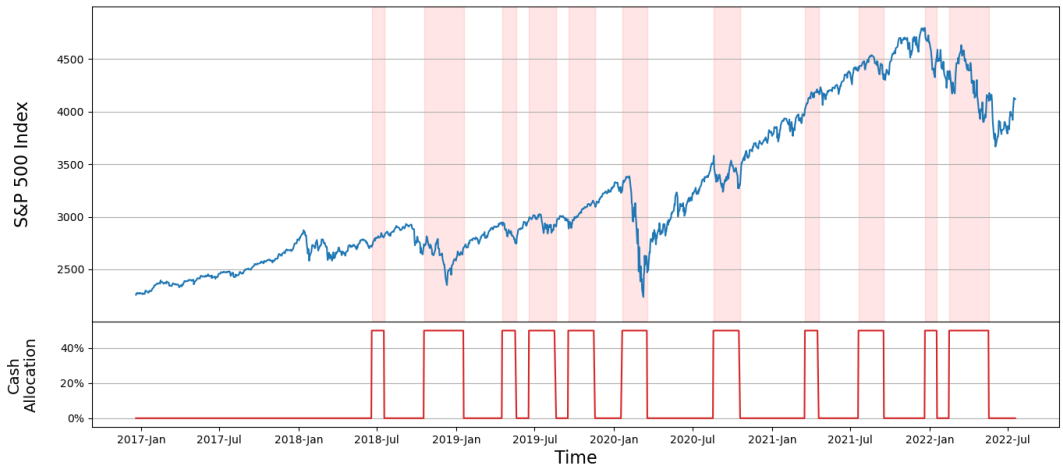

Figure 5: Cash allocation vs market regime of Tech Portfolio

In Figure 5, the highlighted time periods indicate when CPO allocated maximally to cash. The overlay of the S&P 500 Index reveals that these periods are highly correlated with the drawdown periods of the market index, even during out-of-sample testing that started in August 2021. This supports our hypothesis that CPO can rapidly adapt to market regime changes.

We also tested how CPO performs for some non-traditional assets - a portfolio of 8 crypto currencies, again allowing for short positions and aiming to maximize its 7-day forward Sharpe ratio,

Method	Sharpe Ratio
Markowitz	0.26
CPO	1.00

Table 4: Crypto portfolio

(These results are over an out-of-sample period from January 2020 to June 2021, and the universe of cryptocurrencies for the portfolio are BTCUSDT, ETHUSDT, XRPUSDT, ADAUSDT, EOSUSDT, LTCUSDT, ETCUSDT, XLMUSDT). CPO improves the Sharpe ratio over the Markowitz method by a factor of 3.8.

Finally, to illustrate that CPO doesn't just work on portfolios of assets, we apply it to a portfolio of FX trading strategies managed by a FX hedge fund WSG. (WSG is our client and we published these results with their permission.) It is a portfolio of 7 trading strategies s, and the allocation constraints are $w_s \in [0\%, 40\%]$, $w_c \in [0\%, 100\%]$, with $\sum_s w_s + w_c = 1$.

Method	Sharpe Ratio
Equal Weights	1.44
Markowitz	2.22
CPO	2.65

Table 5: WSG's FX strategies portfolio

(These results are over an out-of-sample period from January 2020 to July 2022). CPO improves the Sharpe ratio over the Markowitz method by 19%. WSG has decided to deploy CPO in production starting July 2022. Since then, CPO has added about 60bps per month to the portfolio over their previous proprietary allocation method.

In all 4 cases, CPO outperformed both the naive Equal Weights portfolio and the Markowitz portfolio during a market downturn, while generating similar performance during the bull market. We do not claim that CPO can outperform *all* other allocation methods for *all* portfolios in *all* periods. Some portfolios may be constructed to be so risk-neutral that CPO can't improve on an Equal Weights allocation. For other portfolios, CPO may underperform a conventional allocation method for a certain period with the benefit of hindsight (*ex-post*), but nevertheless outperform the best conventional allocation method selected at the beginning of the period (*ex-ante*). We provide an illustration of this effect through a model portfolio below.

A model tactical asset allocation portfolio

The purpose of studying a model tactical asset allocation (TAA) portfolio is not only to investigate whether CPO can outperform conventional allocation methods, but also to observe how CPO allocates across different asset classes over evolving market regimes. The model portfolio we selected comprises of five ETFs representing various asset classes: GLD (gold), IJS (small cap stocks), SPY (large cap stocks), SHY (1-3 year Treasury bonds), and TLT (20+ year Treasury bonds). This portfolio is inspired by the Golden Butterfly portfolio created by (Tyler, 2016). To train our machine learning model, we use the period January 2015 to December 2018, while the out-of-sample test period covers January 2019 to December 2022. The portfolio rebalances every 2 weeks, and CPO aims to maximize the Sharpe Ratio over the forward 2-week period. The constraint is $w_s \in [0\%, 100\%]$, $\sum_s w_s = 1$, with no cash allocation allowed (since SHY is practically cash).

Period	Method	Sharpe Ratio	CAGR
2015-01 to 2018-12 (In-sample)	Equal Weights	0.51	3.60%
	Risk Parity	0.62	1.87%
	Markowitz	0.59	5.26%
	Minimum Variance	0.47	1.13%
	CPO	0.63	3.93%

Period	Method	Sharpe Ratio	CAGR
2019-01 to 2022-12 (Out-of-sample)	Equal Weights	0.62	6.61%
	Risk Parity	0.22	1.16%
	Markowitz	-0.13	-2.09%
	Minimum Variance	-0.05	0.39%
	CPO	0.42	3.83%

Table 6: In- and Out-of-sample performance of TAA portfolio

Table 6 shows during the out-of-sample period, CPO generates the second-highest Sharpe ratio, trailing only the Equal Weights method. However, selecting Equal Weights *ex-ante* would not have been an obvious choice, since it generated the second-lowest Sharpe ratio during the in-sample period. If we were to choose a conventional allocation method *ex-ante*, Risk Parity would have been our choice, but it underperformed CPO out-of-sample as measured by both the Sharpe ratio and CAGR, the latter by more than threefold.

To gain more transparency into the CPO method, we can examine its allocations at various times:

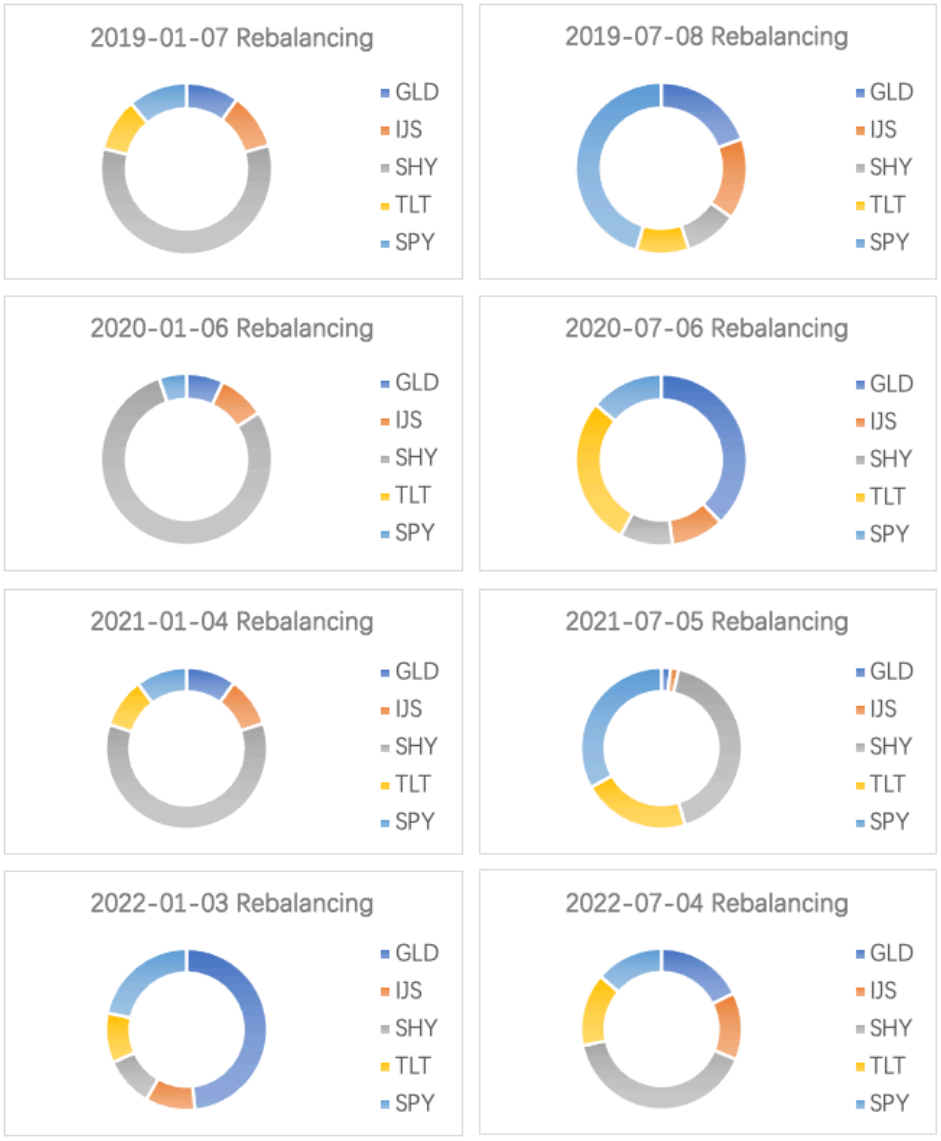

Figure 6: Time evolution of allocations of TAA portfolio (GLD is deep blue)

It is noteworthy that the portfolio had a high allocation to large-cap stocks beginning in July 2019, just before the market experienced a period of calm appreciation over the next six months. The high allocation to short-term treasuries in January 2020 proved to be prescient in light of the COVID-induced financial crisis that followed. The portfolio also had a high allocation to gold at the beginning of 2022, which fortuitously anticipated the surge in commodity prices due to the war in Ukraine. Finally, the allocation to small-cap stocks increased in mid-2022, which performed better than large-cap stocks during that year.

For further interpretability, we list the most important 10 input features picked by our algorithm at each rebalance date in Table 7. The features in this table are listed in decreasing importance rank, as scored by the SHAP algorithm (Man & Chan, 2021). Though the top 10 features vary

over time, we can see that intraday volatility of SPX (SPX_HL_20_realVol_20), the return of gold price (e.g. GLD_logMA5_250), the implied skewness of the SPX options (e.g. SKEW_MA5), the implied volatility of volatility of the SPX (e.g. VVIX_MA20), and the treasury bond yield (e.g. TR10_logMA5_20) are often picked.

Date	Features	Description
2019/1/7	SPX_HL_20_realVol_20	20 days Risk adjusted volatility of SPX_logDayRange_20
	GLD_logMA5_250	Logarithimic Delta of GLD_Close 5 day Moving Average vs GLD_Close 250 Day Moving Average
	SKEW_MA5	SKEW Moving Average of 5 days (rolling mean)
	GLD_logMA5_20	Logarithimic Delta of GLD_Close 5 day Moving Average vs GLD_Close 20 Day Moving Average
	SKEW_MA20	SKEW Moving Average of 20 days
	GLD_realVolatility_20	20 day Volatility of GLD_logD1
	VVIX_MA20	VVIX Moving Average of 20 days
	TR10_realVolatility_20_60	Delta of TR10_realVolatility_20 vs TR10_realVolatility_60
	VVIX_MA5	VVIX Moving Average of 5 days (rolling mean)
	Canary_1	If BND has negative momentum
2019/7/8	RUT_logMA5_20	Logarithimic Delta of RUT_Close 5 day Moving Average vs RUT_Close 20 Day Moving Average
	TR10_logMA5_20	Logarithimic Delta of TR10Y_Close 5 day Moving Average vs TR10Y_Close 20 Day Moving Average
	HYG_realVolatility_20_60	Delta of HYG_realVolatility_20 vs HYG_realVolatility_60
	GLD_realVolatility_20_60	Delta of GLD_realVolatility_20 vs GLD_realVolatility_60
	SPX_HL_20_realVol_20	20 days Risk adjusted volatility of SPX_logDayRange_20
	GLD_logDayRet	Single Day Logarithmic returns - log delta of GLD_Close vs GLD_Open
	GLD_volumeRatio_20	Average Volume for 20 days
	SKEW_MA5_20	Delta of SKEW vs SKEW_MA20
	SKEW_MA20	SKEW Moving Average of 20 days
	TR10_TR3M	Delta of TR10Y_Close vs TR3M_Close (TR3M - 3 Month Treasury Yield)
2020/1/6	SKEW_MA20	SKEW Moving Average of 20 days
	TR10_realVolatility_20_60	Delta of TR10_realVolatility_20 vs TR10_realVolatility_60
	HYG_realVolatility_20_60	Delta of HYG_realVolatility_20 vs HYG_realVolatility_60
	SPX_logMA20_250	Logarithimic Delta of SPX_Close 20 day Moving Average vs SPX_Close 250 Day Moving Average
	GLD_volumeRatio_250	Average Volume for 250 days
	Canary_1	If BND has negative momentum

	HYG_logD5	Five logarithmic difference of HYG_Close and HYG_Close (t-5)
	SPY_volumeRatio_250	Average Volume for 250 days
	SKEW_MA1_5	Delta of SKEW vs SKEW_MA5
	TR10_RSI5	TR10Y_Close - 5 days Relative Strength Index
2020/7/6	TR10_realVolatility_20_60	Delta of TR10_realVolatility_20 vs TR10_realVolatility_60
	SKEW_MA20	SKEW Moving Average of 20 days
	Backward_Risk_MV	Risk factor of defensive strategy in the past cycle
	RUT_logDayRange_5	5 Day Moving Average of RUT_logDayRange_1
	GLD_logMA1_250	Logarithimic Delta of GLD_Close vs GLD_Close 250 day Moving Average
	Backward_Risk_EW	Risk factor of equal weights investment in the past cycle
	GLD_logMA5_250	Logarithimic Delta of GLD_Close 5 day Moving Average vs GLD_Close 250 Day Moving Average
	HYG_logMA5_20	Logarithimic Delta of HYG_Close 5 day Moving Average vs HYG_Close 20 Day Moving Average
	SPX_HL_20_realVol_20	20 days Risk adjusted volatility of SPX_logDayRange_20
	HYG_realVolatility_20_60	Delta of HYG_realVolatility_20 vs HYG_realVolatility_60
2021/1/4	GLD_logD2	Two Day logarithmic difference of GLD_Close and GLD_Close (t-2)
	TR10_realVolatility_20	20 day Volatility of TR10_logD1
	NOPE_MAD_SPY	Mean absolute deviation of net options pricing effect for SPY
	HYG_realVolatility_20_60	Delta of HYG_realVolatility_20 vs HYG_realVolatility_60
	GLD_realVolatility_20_60	Delta of GLD_realVolatility_20 vs GLD_realVolatility_60
	Backward_Risk_EW	Risk factor of equal weights investment in the past cycle
	GLD_logD5	Five Day logarithmic difference of GLD_Close and GLD_Close (t-5)
	TR10_RSI20	TR10Y_Close - 20 days Relative Strength Index
	SPX_HL_20_realVol_20	20 days Risk adjusted volatility of SPX_logDayRange_20
	TR10_logMA5_20	Logarithimic Delta of TR10Y_Close 5 day Moving Average vs TR10Y_Close 20 Day Moving Average
2021/7/5	RUT_logMA20_250	Logarithimic Delta of RUT_Close 20 day Moving Average vs RUT_Close 250 Day Moving Average
	TR10_realVolatility_20_60	Delta of TR10_realVolatility_20 vs TR10_realVolatility_60
	GLD_logMA5_20	Logarithimic Delta of GLD_Close 5 day Moving Average vs GLD_Close 20 Day Moving Average
	RVX_MA20	RVX Moving Average of 20 days
	VVIX_MA5_20	Delta of VVIX_MA5 vs VVIX_MA20

	SPX_HL_5_realVol_20	20 days Risk adjusted volatility of SPX_logDayRange_5
	SPY_volumeRatio_20	Average Volume for 20 days
	HYG_realVolatility_60	60 day Volatility of HYG_logD1
	SKEW_MA5	SKEW Moving Average of 5 days (rolling mean)
	GLD_realVolatility_20	20 day Volatility of GLD_logD1
2022/1/3	SPX_HL_20_realVol_20	20 days Risk adjusted volatility of SPX_logDayRange_20
	GLD_logD5	Five Day logarithmic difference of GLD_Close and GLD_Close (t-5)
	SKEW_MA5	SKEW Moving Average of 5 days (rolling mean)
	TR10_realVolatility_60	60 day Volatility of TR10_logD1
	VVIX_MA20	VVIX Moving Average of 20 days
	VXN_MA5_20	Delta of VXN_MA5 vs VXN_MA20
	TR10_TR1	Delta of TR10Y_Close vs TR1Y_Close (TR1Y - 1 Year Treasury Yield)
	GLD_logMA5_20	Logarithimic Delta of GLD_Close 5 day Moving Average vs GLD_Close 20 Day Moving Average
	SKEW_MA1_5	Delta of SKEW vs SKEW_MA5
	HYG_realVolatility_20	20 day Volatility of HYG_logD1
2022/7/4	GLD_logDayRet	Single Day Logarithmic returns - log delta of GLD_Close vs GLD_Open
	GLD_realVolatility_20_60	Delta of GLD_realVolatility_20 vs GLD_realVolatility_60
	TR10_realVolatility_20_60	Delta of TR10_realVolatility_20 vs TR10_realVolatility_60
	Backward_Risk_MV	Risk factor of defensive strategy in the past cycle
	Backward_Risk_MS	Risk factor of aggresive strategy in the past cycle
	Backward_Sharpe_MV	Sharpe ratio of defensive strategy in the past cycle
	Backward_Risk_IV	Risk factor of risk parity strategy in the past cycle
	HYG_dayRangeRatio_250	250 day moving average ratio of HYG_Range (HYG_High - HYG_Low)
	SPX_HL_20_realVol_20	20 days Risk adjusted volatility of SPX_logDayRange_20
	HYG_realVolatility_60	60 day Volatility of HYG_logD1

Table 7: Features with decreasing importance rank at each rebalance date

CPO software-as-a-service

For clients of our CPO software-as-a-service platform, we can optimize any objective function, not just Sharpe ratio. For example, we have been asked to minimize Expected Shortfall, UPI, etc. We can also add specific constraints to the desired optimal portfolio, such as average ESG rating, maximum exposure to various sectors, or maximum turnover during portfolio rebalancing. The only other input we require from them is the historical returns of the portfolio components (unless these components are publicly traded assets, in which case clients only need to tell us

their tickers). If these components changed over time, we will also need the historical components. We will provide pre-engineered market features (Nautiyal & Chan, 2021) that capture market regime information. If the client has proprietary market features that may help predict the returns of their portfolio, they can merge those with ours as well. Clients' features can remain anonymized. We will be providing an API for clients who wish to experiment with various constraints and hyperparameters (such as the frequency of portfolio rebalancing) and their effects on the optimal portfolio.

Just like ChatGPT, our CPO product will improve in optimality regularly due to our ever-increasing number of pre-engineered features and the ever-enlarging search space. That is the main advantage of machine-learning-based products – they will learn to improve themselves.

Conclusion

It is intuitively obvious that the optimal solution to a problem depends on the environment in which it occurs, whether the problem is the optimal way to stock a retail shelf (back-to-school or Christmas sales) or optimal asset allocation (risk-on or risk-off). Unfortunately, most conventional optimization methods cannot consider the environmental context, as it often is often ill-defined and may involve hundreds of variables. However, machine learning algorithms excel in dealing with big data inputs that may contain redundant and insignificant variables. Our CPO method leverages machine learning and big data to provide an optimal solution to many commercial problems such as portfolio optimization that adapts to the environment. We have demonstrated in multiple use cases that it can outperform conventional portfolio optimization methods and have shown an example in tactical asset allocation where historical allocations were timely.

Acknowledgements

We thank Pavan Dutt, Akshay Nautiyal, and Jai Sukumar for their assistance in features engineering and software implementation of the CPO method, and to Sergei Belov and Guillaume Goujard for their mathematical insights. We also thank Crispin Clarke, Erik MacDonald, and Jessica Watson for their comments on the manuscript. Finally, we appreciate the insightful questions raised by the audiences at UBS & Cornell Financial Engineering Manhattan AI Speaker Series, NYU Mathematical Finance and Financial Data Science seminar, CIBC's Finance-AI seminar, Fidelity AI Asset Management group, and many other public and private forums where we presented CPO.

References

Ang, A. (2014). *Asset Management: A Systematic Approach to Factor Investing.* Oxford University Press.

Chan, E. (2014, August 18). From http://epchan.blogspot.com/2014/08/kelly-vs-markowitz-portfolio.html

Chan, E., Belov, S., & Ciobanu, R. (2021, April 1). From https://predictnow.ai/conditional-parameter-optimization-adapting-parameters-to-changing-market-regimes/

Fama, E. F., & French, K. R. (1995). Size and book-to-market factors in earnigns and returns. *Journal of Finance, 50*, 131-155.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer Feedforward Networks are Universal Approximators. *Neural Networks, 2*, 359–366.

López de Prado, M. (2020). *Machine Learning for Asset Managers.* Cambridge University Press.

Man, X., & Chan, E. (2021). The Best Way to Select Features? Comparing MDA, LIME, and SHAP. *The Journal of Financial Data Science, 3*(1), 127-139.

Man, X., & Chan, E. P. (2021). Cluster-based Feature Selection. *Market Technician, 90*, 11-22.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective.* The MIT Press.

Nautiyal, A., & Chan, E. (2021, September 9). From https://predictnow.ai/new-additions-to-the-predictnow-ai-factor-zoo/

Poh, D., Lim, B., Zohren, S., & Roberts, S. (2020). Building Cross-Sectional Systematic Strategies By Learning to Rank. *https://arxiv.org/abs/2012.07149*.

Ruppert, D., & Matteson, D. S. (2015). *Statistics and Data Analysis for Financial Engineering with R examples, Second Edition.* Springer.

Tomasz, K., & Katarzyna, P. (2021). Building portfolios based on machine learning predictions. *Economic Research-Ekonomska Istraživanja*.

Tyler. (2016). From https://portfoliocharts.com/portfolio/golden-butterfly/