A Case for iPCA in Financial Forecasting

Christopher Fougner,* Ruoxi Wang,† Michael D'Angelo‡

Abstract

Principle Component Analysis (PCA) is used to reduce dimensionality and noise, while still preserving the majority of the variance in the data. It however gives little guarantee on the predictive value of the remaining data. This paper proposes an inverted Principle Component Analysis (iPCA), to achieve dimensionality and noise reduction, by removal of the *largest* principle components. In addition a three-part decomposition of the financial markets is proposed and its validity is tested. By applying iPCA we were able remove the market component that is less predictable, while at the same time keeping the components that carry most of the predictive information. Our method is applied to various regression models including recurrent neural networks and an improvement of 57% is observed.

1 Time Series Prediction in Finance

A time series is a sequence of observations at successive points in time. Time series prediction consists of predicting future values of a time series, given past history of the same series. In finance, the return $r^{(t)} = \frac{p^{(t)} - p^{(t-1)}}{p^{(t-1)}}$ is typical to viewed as the variable, where $p^{(t)}$ is the stock price at time t. The objective is to predict $r^{(\tau)}$ ($\tau > t$), given $r^{(0)}, .., r^{(t)}$. Frequently one has multiple stocks, in which case each stock is viewed as a variable.

Time series prediction in finance is a constantly evolving field. The proliferation of electronic trading means that an individual can invest in more instruments and markets than ever before. It is not uncommon for an investor to maintain a portfolio of derivatives, ETFs, and equities in markets all over the world. This behavior has made individual stocks and markets as a whole significantly more correlated. The additional advancement of algorithmic trading has removed many of the inefficiencies in the markets, making stock prices look increasingly noisy. Whereas simple trend following strategies worked well in the 20th century, increasingly sophisticated algorithms and trading systems are required to maintain an edge over the market.

To deal with the large number of financial instruments and the high degree of what can be classified as noise, both dimensionality reduction and filtering are necessary preprocessing steps. Principle Component Analysis (PCA) has a long history of being used for both these purposes in many fields of statistics including time series prediction. PCA projects the data into the d-dimensional subspace, that accounts for the most variability in the data, among all spaces of dimension d. Doing so, implicitly assumes that any noise in the data either has low variance or is uncorrelated among the different variables. In finance, neither of these assumptions can be made. Data can be extremely noisy and unpredictable events can have highly correlated effects on the stock market. We hypothesize that the directions of largest variation in the data in fact correspond to directions of most noise and greatest unpredictability. If this is true, then one would be better off removing the largest principle components and keeping the smaller principle components. This is opposite to the way PCA is typically employed, accordingly we refer to this procedure as inverted PCA (iPCA). We seek to verify this hypothesis through application to high frequency data of the 100 largest[1] stocks by market capitalization, listed on the New York Stock Exchange (NYSE).

2 Model of Financial Markets

We first introduce some notation. Let $X \in \mathbb{R}^{m \times n}$ be the data matrix[2] of returns ($r^{(t)}$), where each column represents one stock, and each row is the return of each individual stock at a specific time t, where the row number corresponds to t:

$$X = \begin{bmatrix} - & {r^{(1)}}^T & - \\ - & {r^{(2)}}^T & - \\ & \vdots & \\ - & {r^{(T)}}^T & - \end{bmatrix}$$

Next, we propose the following decomposition of the return of a stock i:

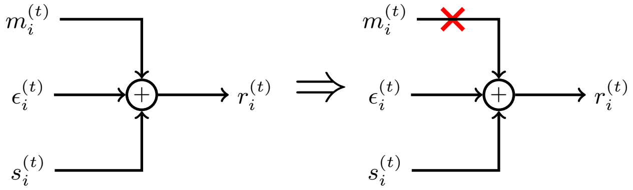

Figure 1: ***Left:*** *Decomposition of one stock.* ***Right:*** *Our stock model after market component removed by iPCA*

where $r_i^{(t)}$ is the return of the i'th stock at time t, $m_i^{(t)}$ represents the influence of the market, $s_i^{(t)}$ represents a signal component and $\epsilon_i^{(t)}$ is noise. We make the following five assumptions

1. $s^{(t)}$ shows at least some correlation in time.
2. $\epsilon^{(t)} \sim \mathcal{N}(0, D)$, where D is diagonal. We assume at each point of time, the noise for each stock has zero mean and are independent of each other. Furthermore, we assume that the noise is uncorrelated in time.
3. $m^{(t)} \sim \mathcal{D}(0, \Sigma_M)$, where $\mathcal{D}$ is some unknown distribution, and the individual components are highly correlated at each point in time. This leads to a near low-rank assumption of covariance matrix Σ_M. In addition, we assume that $m^{(t)}$ shows little correlation in time and is very difficult, if not impossible to predict. We formalize this notion as being equivalent to $\mathbb{E}[m^{(t+1)}|r^{(0)}, ..r^{(t)}] = 0$. This directly leads to the assumption that the market component has zero mean.
4. $||\text{Cov}(s^{(t)}, m^{(t)})|| \ll ||\text{Cov}(s^{(t)}, s^{(t)})||$. In other words, the leading direction (the main subspace) of the market must be sufficiently different from any leading direction that the signal may have (as shown in Figure 2).

*e-mail:fougner@stanford.edu
†e-mail:ruoxi@stanford.edu
‡e-mail:mdangelo@stanford.edu

[1] Seven of these had to be removed due to missing data.

[2] For consistency with Machine Learning notation, we use X – rather than R – to represent the data matrix.

5. $||\text{Cov}(s^{(t)}, s^{(t)})|| \ll ||\text{Cov}(m^{(t)}, m^{(t)})||$.

Intuitively, this decomposition represents the notion that the movement of a single stock is mostly influenced by the movement of the market as a whole, yet each stock has nuances that makes it somewhat predictable. A pictorial description of our model is shown in Figure 2.

Since the market contributes most of the variance in our data, the largest principle components of the data will primarily consist of the subspace spanned by the eigenvectors of Σ_M. Therefore by removing the largest principle components we are able to remove the "market noise" and restrict ourselves to a subspace consisting of mainly signal and uncorrelated noise. The proposed model is shown in Figure 1. This is promising since the market is not easily predictable while the signal is predictable if only marginally.

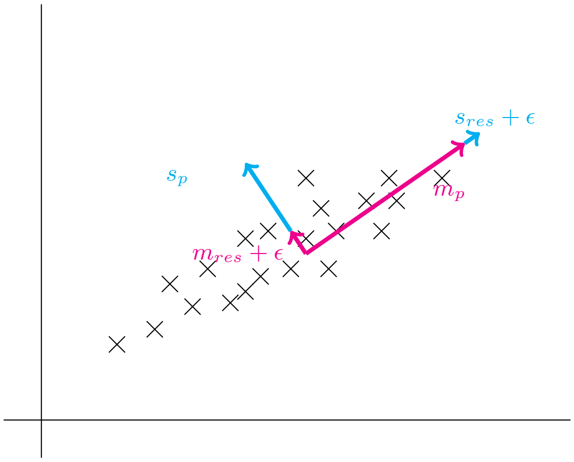

Figure 2: *The crosses represent data points, pink vectors represents the market movement, and the blue vectors represents the noise and signals. Subscript p indicates that the subscripted variable moves principally in the indicated subspace. Subscript res indicates residual movement in the indicated direction*

3 Metrics

To evaluate whether financial forecasting benefits from iPCA, we must formalize a metric, by which performance can be measured. RMSE is typically used in Machine Learning, however it is poorly suited for financial forecasting. Typically predicting the exact return of a stock is extremely difficult, and in most cases it is sufficient that the sign of the return is predicted correctly. In other words, an algorithm, which can correctly predict whether a stock will go up or down, is very valuable. Likewise, an algorithm which correctly predicts when there will be large swings in the market, but generates wildly incorrect predictions the rest of the time, can still be a very useful algorithm. In lieu of reporting high RMSE, many research papers resort to reporting accuracy metrics that are a function of the predicted price $\hat{p}^{(t)} = (1+\hat{r}^{(t)})p^{(t-1)}$. Such metrics are extremely misleading, because the price tends to show small fluctuations, which leads to very low reported errors, without actually being indicative of performance. In many cases, simply predicting $\hat{p}^{(t)} = p^{(t-1)}$ will result in low reported errors.

Rather than focusing on error metrics, we instead direct our attention to how well a model would perform if implemented in practice. To do so, we use the industry standard metric for risk-adjusted returns, the Sharpe ratio, defined as

$$SR = \frac{R_p - R_f}{\sigma_p}$$

where R_p is the portfolio return, R_f is the risk free rate and σ_p is the portfolio volatility. In more detail,

- R_p is the percentage return of your portfolio if you follow a policy $\delta^{(t)} \in [-1, 1]^n$. A policy simply states how heavily your portfolio is invested in each stock at time t. It is frequently enforced that $||\delta^{(t)}||_1 \leq 1$, which ensures that the portfolio cannot be leveraged. Mathematically:

$$R_p = \prod_t (1 + \delta^{(t)} \cdot r^{(t)}) - 1$$

- R_f is the return your assets could achieve if they were invested in a "risk-free" asset. This risk-free asset is generally taken to be the 3 month T-Bill[3].

- σ_p is the standard deviation of the returns that your policy achieves:

$$\sigma_p^2 = \frac{1}{m}\sum_{t=1}^{m}\left(\delta^{(t)} \cdot r^{(t)} - \frac{1}{m}\sum_{t'=1}^{m}\delta^{(t')} \cdot r^{(t')}\right)^2$$

The Sharpe ratio quantifies the notion that increased performance should come from smarter investments, rather than riskier investments.

The space of possible policies $\delta^{(t)}$ is large, portfolio optimization is even a field in itself. Since, the focus of this report is not on optimizing returns, but rather on evaluating iPCA, we choose a simple policy:

$$\delta^{(t)} = \frac{1}{n}\text{sign}(\hat{r}^{(t)})$$

where n is the number of instruments in your portfolio and $\hat{r}^{(t)}$ is the predicted return at time t (recall that $\hat{r}^{(t)}$ can be a function of $r^{(1)}, ..., r^{(t-1)}$)

4 Regression Models

Since iPCA is a pre-processing step, we also need a regression model to be able to make any observations about performance improvements. Consider the general form of linear regression:

$$Y = g(X)\beta + \epsilon$$

where Y is the response variable, g is an arbitrary function, β is an unknown parameter, and ϵ is a noise with zero mean. Finding the β can be posed as an optimization problem:

$$\min_\beta \; ||Y - g(X)\beta||_p^p + \lambda||\beta||_q^q$$

where $\lambda||\beta||_q$ is a penalization term to avoid over-fitting. The degree of regularization is governed by $\lambda \in \mathbb{R}$. Different choices of g, p and q lead to different regression models. We will focus on the case when $p = 2$ (ℓ_2 regression), for the following reasons:

[3] Rather than using the actual risk-free rate (which was near zero at the time of writing this report), we use the market return over the period 24th of April to November 27th (which was close to 7%).

- Optimization is computationally inexpensive, which means the model can be retrained frequently and it can be applied to many datasets. This is important when dealing with large financial datasets consisting of large numbers of instruments and time series that can span decades.
- Decay can easily be applied to the training data by a weighted least squares extension. This allows developments of newer market dynamics to be more prominent.
- The least squares objective, not only attempts to find an $\hat{r} \approx r$, but also the $\hat{r}$ that minimizes the variance of the errors. This follows because $\mathbb{E}[||r - \hat{r}||_2^2] \propto \sigma_p^2$, under the standard statistical assumption that errors follow a normal distribution with zero mean. We see that least squares attempts to decrease σ_p and increase r_p, and by implication increases the Sharpe Ratio.

The regularization term prevents over-fitting by forcing the q-norm of β to be small. The most common choices are $q = 1$ and $q = 2$. Choosing $q = 1$ encourages β to be sparse, which can be viewed as a method to perform feature selection. On the other hand, $p = 2$ is unlikely to set any element of β to zero, but it is the optimal choice assuming a Bayesian prior on β. The case $p = 2, q = 1$ is commonly known as Lasso and the case $p = 2, q = 2$ is known as Ridge Regression.

Various functions g can be chosen, but we have opted to look at two specific ones in this report. The first is simply a linear function $g(X) = X$. The second function is a randomized approach to recurrent neural networks (RNNs), known as the echo state network (ESN). ESNs relies on sparsely connected neurons, where sparsity and edge weights are chosen at random. In essence it is a method to generate a highly non-linear function with time dynamics from a time series matrix X. A diagram of an ESN is shown in Figure 3.

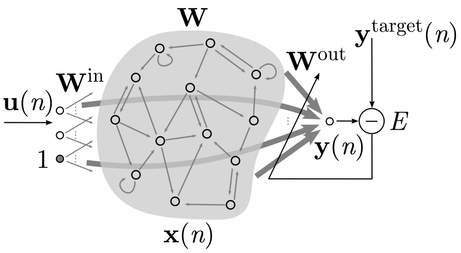

Figure 3: *Echo state network architecture*

Mathematically the ESN function (g_{esn}) can be defined recursively as,

$$x(n+1) = f(W^{in}u(n+1) + Wx(n) + W^{fb}y(n))$$
$$y(n+1) = W^{out}[1;\ u(n+1);\ x(n+1)]$$
$$g_{esn}(X) = \begin{bmatrix} 1 & u(0) & x(0) \\ 1 & u(1) & x(1) \\ & \vdots & \\ 1 & u(n) & x(n) \end{bmatrix}$$

where $u(n+1)$ is the input (and hence synonymous with $r^{(t)}$), $x(n)$ is the reservoir neuron activations, $y(n)$ is the network output and f is a non-linear activation function. W_{in}, W, W_{out} and W_{fb} are weight matrices for input, internal connections, output and feedback respectively. f is typically chosen to be a sigmoid function, such as the logistic function. We notice that g does not act on each row independently, instead the dependence of g can be viewed as follows:

$$g_{esn}(X) = \begin{bmatrix} \tilde{g}(r^{(1)}) \\ \tilde{g}(r^{(2)}, r^{(1)}) \\ \tilde{g}(r^{(3)}, r^{(2)}, r^{(1)}) \\ \vdots \end{bmatrix}$$

The advantage of such a recurrent function is that if the stock returns show a non-linear dependence on previous returns, then such a function may be able to capture part of the dynamics.

To summarize, we compare the iPCA applied to the following

- $g(X) = X$ with ℓ_1 and ℓ_2 regularization,
- $g(X) = g_{esn}(X)$ with ℓ_1 and ℓ_2 regularization.

We also use a re-training method, whereby the X is initially split into three component X_{train}, X_{test} and X_{rest}. Time-wise, all observations in X_{train} strictly precede those in X_{test}, which strictly precede those in X_{rest} (order of the data points is maintained). The model is then trained on X_{train} and tested on X_{test}. Thereafter X_{test} and X_{train} are combined to form a new $\tilde{X}_{train}$. Lastly X_{rest} is split into $\tilde{X}_{test}$ and $\tilde{X}_{rest}$. This process is continued until X_{rest} is empty.

5 Data

For the empirical evaluation of iPCA we chose to analyze the 100 largest companies by market capitalization on the NYSE. Stock prices at one minute intervals were obtained from Bloomberg for the period April 24th to November 26th 2013, which corresponds to approximately 50'000 data points. We chose to use data at one minute intervals because there is surprisingly little academic literature which mentions it and much of algorithmic trading is done at high frequency. Bloomberg is the industry standard data provider for financial data, so we can be confident that the data we are using is "clean".

As an input to the various regression models, we simply used the first $m - 1$ time points of the data matrix, and as the output we used the data matrix shifted by one. Using MATLAB notation:

```
X = bb_returns(1:end-1,:)
Y = bb_returns(2:end,:)
```

In Section 2, we made the assumption (1.) that part of our data shows correlation in time. To understand the degree to which this assumption can be justified, the autocorrelation of each stock return was computed and indeed, the 1-lag autocorrelation was statistically significantly different from zero. An excerpt of these correlations can be seen in Figure 4.

We also made the assumptions (4. and 5.) that the the largest principle component(s) consist of market movements $m^{(t)}$ and that the market movement has significantly larger variance than either the signal $s^{(t)}$ or noise $\epsilon^{(t)}$. Two very interesting observations were made, which both greatly support this claim:

- The principle direction $v_1 \approx \frac{1}{\sqrt{n}}\mathbf{1}$, where $\mathbf{1}$ is a vector consisting of all ones. The projection of X into the subspace spanned by v_1, simply averages the columns (stocks) of X. This averaging can naturally be viewed as extracting the underling market direction. Additionally we see that the corresponding eigenvalue is significantly larger than the rest, in fact accounting for 32% of the variance in the data (Figure 5). Both these observations support our model.

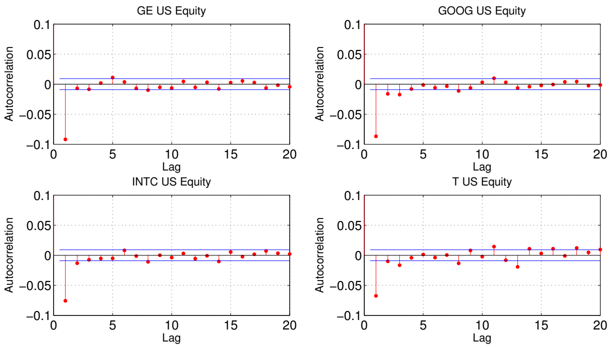

Figure 4: *Autocorrelation of stock return for a selection of 4 companies. Each lag is equal to one minute.*

- The contribution of each stock to the second principle direction v_2 were ordered from most negative, to most positive. Of these, the top and bottom 10 stocks were extracted; the results can be seen in Table 1. We see that the subspace spanned by v_2 is essentially the difference between the average movement of the Oil, Gas & Coal industry and the average movement of the Consumer Products industry. Such movements can clearly be classified as market movements, further justifying our assumptions.

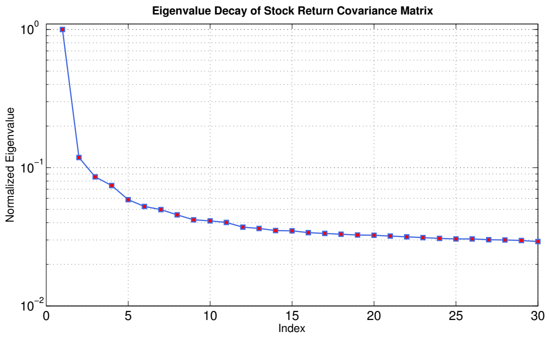

Figure 5: *Eigenvalue decay of X^TX, the eigenvalues are normalized using the largest eigenvalue.*

Positive Contrib.	Industry	Negative Contrib.	Industry
Chevron	Oil, Gas & Coal	PepsiCo	Consumer Prod.
Occidental Petroleum	Oil, Gas & Coal	Altria Group	Consumer Prod.
Freeport-McMoRan	Metals & Mining	Philip Morris	Consumer Prod.
ConocoPhillips	Oil, Gas & Coal	Coca-Cola	Consumer Prod.
Schlumberger	Oil, Gas & Coal	Procter & Gamble	Consumer Prod.
Apache	Oil, Gas & Coal	Colgate-Palmolive	Consumer Prod.
Anadarko Petroleum	Oil, Gas & Coal	Eli Lilly & Co	Biotech & Pharma.
EOG Resources	Oil, Gas & Coal	Wal-Mart	Retail Staples
Halliburton	Oil, Gas & Coal	Merck & Co	Biotech & Pharma.
National Oilwell Varco	Oil, Gas & Coal	Abbott Lab.	Medical Equip.

Table 1: *Stocks contributing most positively and most negatively to the second principle component of X.*

6 Results

To test the performance of iPCA, we implemented the four regression models outlined in Section 4. These models were then tested on the whole dataset (X, Y) and the dataset $(\hat{X}, \hat{Y}) = (X\hat{V}, Y\hat{V})$, where $\hat{V}$ is obtained by removing the b largest eigenvectors from V (V satisfies $X^TX = V\Lambda V^T$). Various values for b were tested, and the optimal value b^* was reported. The results can be found in Table[4] 2. The Baseline Performance column shows how well the model performed when simply predicting Y using X and the iPCA Performance shows the results after applying iPCA.

Regression Model	Baseline	iPCA Performance	Improvement
	(SR)	(b^*, SR)	
Ridge	22.90	(16, 26.37)	15%
Lasso	26.57	(1, 40.82)	54%
ESN w/ Ridge	24.83	(15, 27.00)	9%
ESN w/ Lasso	28.49	(1, 40.73)	43%

Table 2: *Comparisons of best performances among different models.*

Performance improved across the board when applying iPCA. Most notably when training with Lasso and applying iPCA, the Sharpe ratio shot up by 54%. To achieve this performance improvement, only the very first principle component had to be removed. The cumulative portfolio return can be seen in Figure[4] 6.

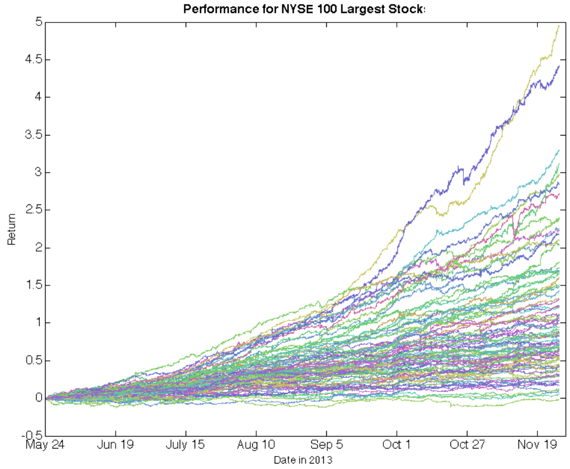

Figure 6: *Cumulative market-adjusted return when applying iPCA to minute data and training using Lasso. Notice that no returns are available for the period 24th of May to 24th of June, since this period was used for training.*

The effect of annealing eigenvalues can be seen in Figure[4] 7. We make two additional observations

- When removing smaller principle components, the performance immediately worsens, implying that small principle components do not correspond to noise. This even suggests that adding more stocks to the model could improve performance.
- Removing principle components 2 through 18 have negligible impact on the Sharpe Ratio, suggesting that these components don't add much in terms of predictive value.

Intuitively, since removing components 2 through 18 doesn't impact performance significantly, one would expect that the corresponding rows in β are reasonably close to zero (when $(\hat{X}, \hat{Y})$ are obtained by removing the single largest principle component from

[4] These results do not take trading costs, bid-ask spread or slippage into account.

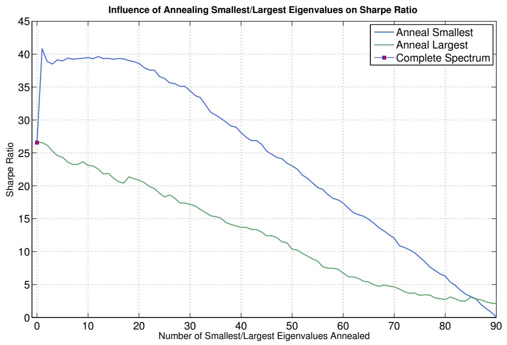

Figure 7: *Effect on Sharpe Ratio of annealing largest and smallest principle components of X.*

(X, Y)). This hypothesis was justified by looking at the magnitude of the entries in the β matrix (Figure 8). We quite clearly see that the rows corresponding to the largest principle components (the last rows) contain only very few elements that are different from zero. Surprisingly, we also see that β is nearly diagonal. This indicates that the cross correlation matrix of $\hat{X}$ and $\hat{Y}$ is nearly diagonal. In other words, the correlation between two columns $\hat{x}_i \in \text{col}(\hat{X})$, $\hat{y}_j \in \text{col}(\hat{Y})$ where $i \neq j$, is zero. If this is the case, then regressing each column of $\hat{Y}$ onto each column of $\hat{X}$ independently should yield better performance (since there is less noise). Additionally it is much less computationally intensive to perform linear regression of 1 variable onto 1 variable, n times, than it is to perform linear regression once of n variables onto n variables.

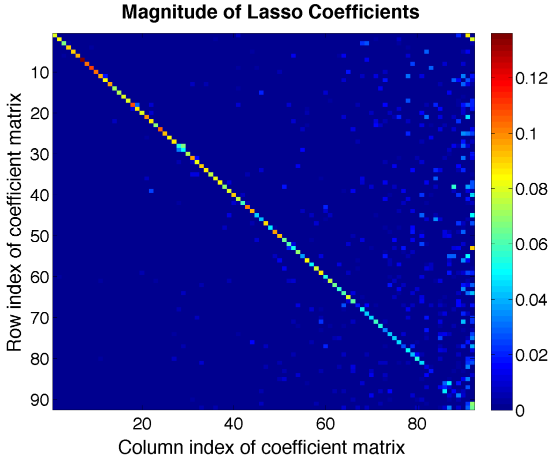

Figure 8: *Maginutde of Lasso Coefficients when regressing the columns of $\hat{Y}$ onto the columns of $\hat{X}$. iPCA was used to remove the largest principle component. Higher indices correspond to larger principle components.*

By performing column-wise regression of $\hat{X}$ onto $\hat{Y}$, we were able to increase the Sharpe Ratio by at least 8-10%. We also see that ESN with Lasso overtook plain Lasso, indicating that there are non-linearities in the data that Lasso could not account for.

7 Conclusion

We see that by applying iPCA to Financial Forecasting, we are not only able to increase the Sharpe Ratio from 28.49 to 44.79 (a 57% improvement), but we also showed that it was possible to train each column of $\hat{X}$ independently, which reduces the computational cost and yields even better performance . Furthermore, we find that using iPCA as a pre-processing step improves the performance of all the models we tested, among which Lasso performed the best.

Regression Model	iPCA column-wise regression (SR)	Improvement
Lasso/Ridge	44.07	8%
ESN w/ Ridge	44.65	65 %
ESN w/ Lasso	44.79	10 %

Table 3: *Columnwise regression of $\hat{X}$ onto $\hat{Y}$. Note that when performing column-wise regression Lasso and Ridge regression are the same.*

8 Future Work

We would like to see how well this model generalizes to time series at lower frequencies, such as hourly or daily data.

References

JAEGER, H. 2001. The" echo state" approach to analysing and training recurrent neural networks-with an erratum note'. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report 148*.

MEDSKER, L., AND JAIN, L. C. 2010. *Recurrent neural networks: design and applications*. CRC press.

PASCANU, R., MIKOLOV, T., AND BENGIO, Y. 2012. On the difficulty of training recurrent neural networks. Tech. rep., Technical Report.