Erasmus University Rotterdam
Erasmus School of Economics
Econometrics and Management Science

Master's Thesis Quantitative Finance

Final Version

November 9, 2022

Factor Models for the Implied Volatility Surface

Author:

Rogier Nagelkerken *497436*

Supervisors:

prof. Gustavo Freire - ESE

drs. Jeffrey Durieux - ESE

Abstract

The Implied Volatility Surface (IVS) is a key component for pricing and hedging options. We compare the performance of three different dimension reduction methods for S&P 500 index IVS; Principal Component Analysis (PCA), Instrumented PCA (IPCA), and Autoencoders (AE). The performance is assessed according to three different criteria; interpretability, modeling performance, and forecasting performance. We find that the factors of the 3-factor models of PCA and AE are easily interpretable and follow the level, skew, and term structure of the IVS closely. Next, using the Implied Volatility RMSE (*IVRMSE*), the modeling performance is measured on the balanced/smoothed and unbalanced/original test set separately. For the balanced test set, PCA outperforms all methods, with AE following closely and IPCA falling behind. However, for the unbalanced test set, AE is clearly dominant as the *IVRMSE* of PCA increases substantially more. Lastly, we measure the forecasting performance using indirect, direct, and hybrid forecasts. The indirect forecasts of PCA and AE seem to outperform direct and hybrid forecasts, with PCA being dominant for the balanced test set (with AE following closely), and AE being dominant for the unbalanced test set.

Overall, the evidence supports AE as the best method to model the IVS.

Keywords: Implied Volatility Surface - Autoencoder - Principal Component Analysis - Instrumented PCA

Note: The views stated in this paper are those of the authors and not necessarily those of the supervisor, second assessor, Erasmus School of Economics or Erasmus University Rotterdam.

Contents

List of Tables

List of Figures

1 Introduction

The Implied Volatility Surface (IVS) depicts how the Implied Volatility (IV) differs over tenor and strike price and is often used by traders and academic researchers. It is a key component for pricing and hedging options. Therefore, it is important to model and forecast the IVS accurately. The IVS is obtained by inverting the option pricing formula by Black and Scholes (1973) for different strike prices and tenors/times-to-maturity. The Black-Scholes (BS) model predicts a flat IVS, meaning that all options issued at the same time should have the same IV, independent of their strike price and tenor. However, this does not hold in practice. Namely, plotting the empirical IV against the strike price often leads to a 'volatility smile/smirk'. This means that the IV is higher for more extreme strike prices (either far In The Money (ITM) or Out of The Money (OTM)) (Rubinstein, 1985; Sircar et al., 1999). Moreover, the IVS changes over time in a highly nonlinear fashion (Andersen et al., 2015b).

The IVS consists of many points on each date. Therefore, it may be hard to model and forecast the IVS directly. Luckily, the surface can largely be explained by a few factors, which makes modeling and forecasting more straightforward (Andersen et al., 2015b; Skiadopoulos et al., 2000). We compare the performance of three different methods for obtaining such factor models: Principal Component Analysis (PCA), Instrumented PCA (IPCA), and Autoencoders (AE).

Most of the research uses PCA for constructing a factor model of the IVS (Avellaneda et al., 2020; Badshah, 2009). PCA is a popular dimension reduction method that produces factors that are linear combinations of the original data, where the weights of the linear combinations are determined by the factor loadings (Pearson, 1901). These loadings are constant and chosen such that the factors explain the most variation of the original data while being mutually orthogonal. Because of the linearity and orthogonality, the factors are generally relatively easy to interpret. However, there seems to be a nonlinear and possibly changing association between the IVS and the factors which is not captured by (constant and linear) PCA, which may hurt the modeling performance (Andersen et al., 2015b). Also, PCA is the only method of the three that does not have a built-in interpolation mechanism. This may hurt its performance as well because an important part of modeling/forecasting

the IVS is to interpolate, e.g. to determine the IV of a new option. As PCA does not have a built-in interpolation function, we use a Gaussian kernel smoother. However, this is not as well adjusted to the data as the built-in interpolation functions.

Instrumented PCA (IPCA) is a variant of PCA which does not assume constant factor loadings. IPCA makes use of a set of observable characteristics (covariates) to determine the factor loadings (Kelly et al., 2019). Because the characteristics are time-varying, the factor loadings are time-varying as well. Therefore, this method may be able to capture more of the variation in the IVS over time than the constant PCA. Furthermore, IPCA is also constructs a linear model because it assumes a linear relationship between the latent factors and the original data, and between the covariates and the factor loadings.

Lastly, we use an autoencoder (AE) to capture the nonlinearities in the data. An AE consists of two Neural Networks (NN) which are connected to each other through the latent factors, the encoder and the decoder. The encoder is a NN that generates latent factors from the original data. Then, the decoder takes these latent factors (and possibly other variables) as input and aims to recreate the original data from these inputs. We use an AE with 1 hidden layer in the encoder and decoder (AE_1) and with 2 hidden layers in both (AE_2). Because both the encoder and decoder are NNs, an AE allows for nonlinear relations between the original data and the factors and vice versa. A drawback of these nonlinearities is a possible lack of interpretability of the factors and the possibility to end up in an adverse local minimum.

We use daily European style S&P 500 index options, ranging from January 2002 to December 2021. The data consists of bid and ask quotes, which we convert into Implied Volatilities through a pre-processing procedure. This procedure makes use of proxies for the stock price, risk-free rate, and dividend yield. Because PCA and AE require a balanced grid as input, we interpolate the original (unbalanced) data to be a balanced grid of options for each point in time. We test the performance of the methods both on the balanced and unbalanced test sets. Additionally, we use a set of macroeconomic variables as covariates for IPCA. The set of covariates is largely based on Almeida et al. (2022).

The performance is split into three categories. The first is the interpretability of the constructed factors. Second, we use the ability to model the IVS in the cross-section at

a given point in time. Lastly, the forecasting ability of the methods is measured. Both the modeling performance and the forecasting performance are measured using the Implied Volatility Root Mean Squared Error (*IVRMSE*).

We test the interpretability of the 3-factor models of each method. The main findings are as follows. First, PCA is easily interpretable as it follows the level, skew, and term structure of the IVS closely. Moreover, AE_1 also follows these IVS characteristics closely, and therefore its factors have a comparable level of interpretability. This is somewhat surprising, as AE_1 can make use of complex nonlinear relations between the data and the factors. Additionally, AE_2 also has a factor that follows the level of the IVS very closely. However, the other factors are harder to interpret for AE_2. Lastly, one of the factors of IPCA behaves unexpectedly by fluctuating around 0 for most of the time, with large sudden spikes. Such factors are impossible to interpret and could harm the modeling and forecasting performance of IPCA substantially.

Next, we test the modeling performance by constructing and testing 1- to 6-factor models for all methods. For the balanced test set, PCA produces models with the lowest (best) *IVRMSE* for most numbers of factors. Moreover, both AE_1 and AE_2 closely follow PCA in terms of modeling performance. On the other hand, IPCA falls behind both methods substantially. For the unbalanced test set, which better represents performance in real-life applications, the *IVRMSE* of PCA increases substantially more than for the other methods. This is caused by the absence of a built-in mechanism for interpolation for PCA. Because the performance of PCA substantially drops, AE_1 and AE_2 are the most dominant methods for modeling the IVS using an unbalanced test set. Even though the difference between the *IVRMSE* of AE_1 and AE_2 is not large, AE_2 seems to be dominant over AE_1 for most numbers of factors. Lastly, IPCA still performs substantially worse than both AE_1 and AE_2. However, its *IVRMSE* is similar to that of PCA for the unbalanced test set.

Lastly, we test the forecasting performance by using 1- to 6-factor models for all methods to make 1-day, 1-week, and 1-month ahead forecasts. Both direct, indirect, and hybrid forecasts are made depending on what each method allows for. For direct forecasting, the parameters are directly trained to give forecasts. For indirect forecasts, the parameters are first trained for cross-sectional modeling, then the factor values are predicted using a Vector

Autoregressive (VAR) model. Lastly, hybrid forecasts are a combination of both.

Similar to the modeling performance, PCA has the lowest *IVRMSE* in most cases for the balanced test set, with $AE_{1,I}$ and $AE_{2,I}$ following closely (all indirect forecasts). Moreover, for the unbalanced test set, the *IVRMSE* of PCA increases substantially more than for the rest of the methods, which results in AE_I being the dominant method. Furthermore, the direct forecasts of AE_D perform decently well, but are outperformed by both AE_I and PCA for the balanced and unbalanced test set. For both AE_I and AE_D, the difference in forecasting *IVRMSE* between the models with 1 and 2 hidden layers is not substantial. Lastly, the *IVRMSE* of IPCA increases extremely for the models with 3 or more factors. This is likely to be caused by the aforementioned unpredictable behavior of (some of) the factors of IPCA.

All in all, AE_1 seems to be the only method that competes for first place in all three performance measures, especially when using the unbalanced test set. Therefore, AE_1 is a viable method to use, regardless of the application, both for traders and academic researchers. Lastly, IPCA in its current form is unsuitable for modeling and forecasting the IVS.

The remainder of this paper is structured as follows. Section 2 gives a brief overview of the previous literature about this subject. Section 3 describes the used option data and covariates, together with the pre-processing steps for the option data. Next, Section 4 describes the used methods for constructing factor models and the performance measures in more detail. Section 5 provides the results. Lastly, Section 6 concludes and gives implications for limitations and further research.

2 Literature

The Implied Volatility Surface (IVS) plays a big role in pricing new options and adjusting the prices of existing options. Because of this, there exists extensive research in modeling and forecasting the IVS. For modeling the IVS, numerous parametric models improve on the (basic) Black-Scholes (BS) model. Many of these models relax some of the strong assumptions which are made in the BS model. For example, Heston (1993) relaxes the assumption of constant underlying volatility by introducing stochastic volatility and Merton (1976) relaxes the assumption of continuous stock returns by including jumps in the returns. Moreover,

Bates (2000), Duffie et al. (2000), and Andersen et al. (2015a) propose models which relax both assumption, thus allowing for stochastic volatility and jumps in the returns. These parametric models generally significantly improve on the BS model in modeling the IVS. However, there is still room for improvement. With the rise of Machine Learning (ML) methods, they have also become more prominent in the asset pricing literature. Almeida et al. (2022) show that Neural Networks (NNs) outperform parametric methods relatively easily, with the ability to boost their performance even more by using a hybrid approach combining the strengths of parametric and non-parametric (NN) models.

Besides direct modeling of the IVS, there also has been extensive research on modeling the IVS indirectly using a factor structure. Andersen et al. (2015b) show that the first three Principal Components (PCs) capture approximately 99.2% of the variation in the IVS. They also show that these factors are relatively easy to interpret with the first PC (explaining 96.4% of the variation) behaving very similar to the level of the IVS. Although this is very promising, they also state that there seems to be a nonlinear association between the IVS and the factors. Moreover, there seem to be dynamic changes in the IVS over time (Cont and Da Fonseca, 2002; Mixon, 2002). Therefore, (static and linear) Principal Component Analysis (PCA), might not be the most appropriate dimension reduction method.

Kelly et al. (2020) propose a non-static alternative to PCA, Instrumented PCA (IPCA). IPCA makes use of covariates to make the model non-static. Büchner and Kelly (2022) have shown that IPCA has the potential to be successful in the asset pricing environment because of its ability to evolve over time and take outside information into account through the covariates while staying interpretable.

A nonlinear alternative to PCA is an autoencoder. Autoencoders are used in many different fields such as image recognition (Gao et al., 2015), and natural language processing (NLP) (Li et al., 2015). As of writing, the use of autoencoders in modeling the IVS is limited. Bergeron et al. (2022) find that autoencoders perform well in completing the IVS. However, to our knowledge, there does not yet exist any literature comparing the performance of different dimension reduction methods in modeling/forecasting the IVS. Gu et al. (2021) perform a similar comparison in the asset pricing domain and find that, in their case, an adjusted autoencoder outperforms both PCA and IPCA, while IPCA generally outperforms PCA.

3 Data

3.1 Option data

We use daily European-style S&P 500 index options to construct the Implied Volatility Surface (IVS). These index options are traded at the Chicago Board Options Exchange (CBOE) and obtained from OptionMetrics. The sample extends from January 2$^{\text{nd}}$, 2002, until December 31$^{\text{st}}$, 2021, consisting of 5036 trading days. The data consists of bid and ask quotes, strike prices (K), volumes (V), dates, expiration dates, and a call/put indicator. From the expiration dates and the current dates, the tenors (τ) can be computed as the number of trading days between the current dates and the expiration dates divided by 252 (average number of trading days per year). The pre-processing steps are based on Almeida et al. (2021), with some slight deviations on for example the range of dates.

Calculating the Implied Volatility (IV) of an option not only requires the option price but also (proxies of) the stock price, risk-free rate, and dividend yield. For the stock price (S), we use the closing price of the S&P 500 index using data from MarketWatch.[1] Moreover, as a proxy for the risk-free rate (r), the 3-month Treasury bill from the St. Louis Federal Reserve Economic Data (FRED) database is used.[2] Finally, the dividend yield (q) can be estimated using the option price, S, and r. However, in some cases, this estimation can not be performed. In these cases, the dividend yield is set to a proxy, namely, the S&P 500 Index Dividend Yield from OptionMetrics. We only consider options with moneyness between 0.8 and 1.5 (or $S/K \in [0.8, 1.5]$) and trading days to expiration between 5 and 300 (equivalently, $\tau \in [\frac{5}{252}, \frac{300}{252}]$).

The first pre-processing step is to construct the option prices as the mid-point of the bid and ask quotes for that option. The option prices are denoted as C for calls and P for puts. Secondly, all observations with zero volume ($V = 0$) or prices lower than 1/8 ($C < \frac{1}{8}$ or $P < \frac{1}{8}$) are dropped. Then, for each day and each τ of traded options on that day, the dividend yield is estimated by using the put-call parity on the pair of call and put closest

[1] Retrieved from https://www.marketwatch.com/investing/index/spx/download-data on June 13, 2022.

[2] Retrieved from https://fred.stlouisfed.org/series/DTB3 on June 14, 2022.

to at-the-money (ATM) ($S/K = 1$).[3] A call and a put are a pair if they share the same K and τ and are issued on the same day. For the cases where such a pair is absent, the aforementioned proxy for q is used.

After that, the observations violating the usual arbitrage conditions are dropped. These conditions are

$$\begin{aligned} C &\geq Se^{-q\tau} - Ke^{-r\tau}, \\ P &\geq Ke^{-r\tau} - Se^{-q\tau}. \end{aligned} \tag{3.1}$$

Lastly, the IV of each option is calculated. As there is no analytic solution to calculate the IV, it is calculated using the Newton-Raphson method (Algorithm 1 in Appendix A). After all these steps the data consists of 4,630,546 different options. This means that, on average, there are 919 different options in the sample each day. However, the number of options per day is not evenly distributed, with approximately 100 options per day at the start of the sample, and more than 3000 at the end (Figure 1). The sample is split up into a train and test set. We use approximately an 80% split on the dates, such that the train set extends until January 3rd 2018 consisting of 4030 trading days and a total of 2,016,377 different options, the test set consists of the 1006 trading days thereafter with 2,614,169 different options.

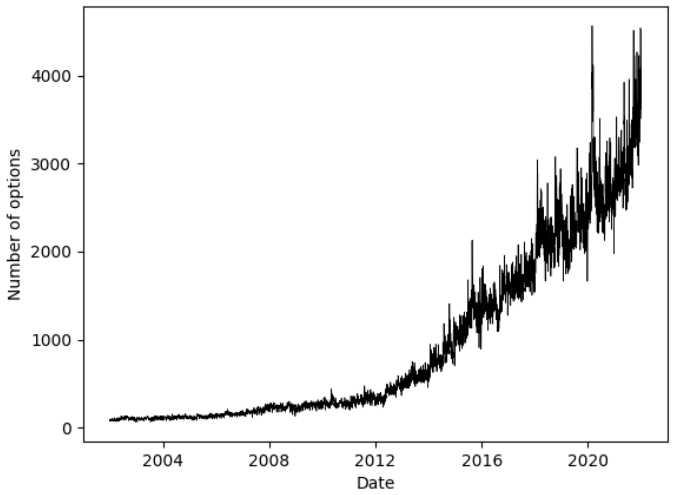


Figure 1: Number of distinct options traded on each day.

[3]The dividend yield (q) is then estimated by: $q = (1/\tau)\ln\left[(C - P + Ke^{-r\tau})/S\right]$.

The data needs to take the form of a balanced grid to train and test most of the methods. That is, there needs to be IV data available for some fixed grid of tenors and moneyness at each date. This is generally not the case in the raw data as there are many missing values and the panel is highly unbalanced. Therefore, the data needs to be interpolated (and possibly extrapolated) such that observations of a fixed grid are available at every point in time. This is done using a Gaussian kernel smoother with a fixed parameter, which makes it possible to get an estimate of each point on the IVS. For points that lay on a more dense part of the IVS (containing many observations), this estimate generally is more accurate, and vice versa. The grid consists of options with all possible combinations of τ and S/K as listed in Table 1. As there are 6 different choices for τ and 7 for S/K, the grid has a size of $6 * 7 = 42$. On each date and for every option on the grid, the Gaussian kernel smoother first calculates

Table 1: Tenors (τ) and moneyness (S/K) from which the balanced grid is constructed.

τ^*	10	21	63	126	189	252	-
S/K	0.9	0.95	1	1.05	1.1	1.2	1.3

Note: *All values for τ are divided by 252 (average number of trading days per year).

the similarity between that option on the grid and all the available options on that day using the kernel ($\mathcal{K}$):

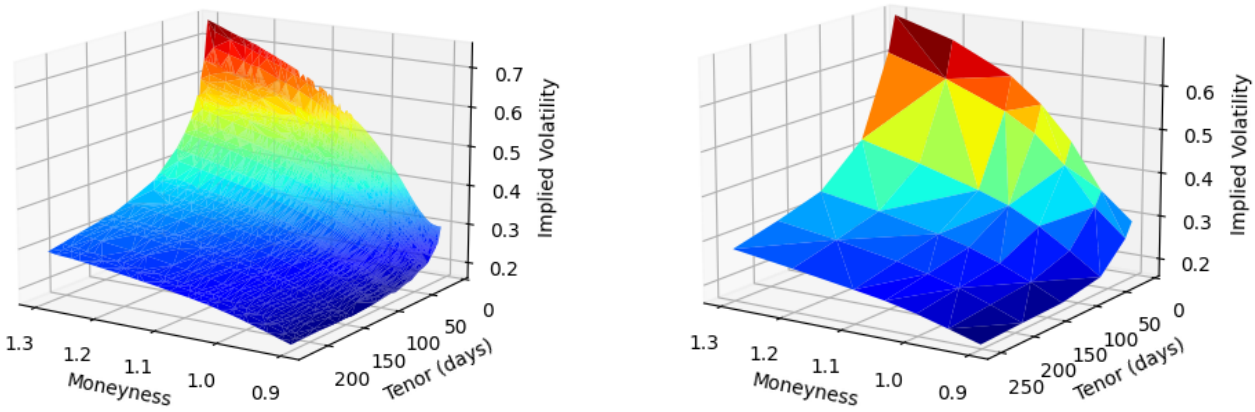
$$\mathcal{K}(S/K^*, \tau^*, S/K, \tau) = \exp\left(-\frac{(S/K^* - S/K)^2 + (\tau^* - \tau)^2}{b^2}\right), \tag{3.2}$$

where b is a hyperparameter. A lower b gives a more accurate fit to the original data, and a higher b gives a more smooth fit. Moreover, S/K^* and τ^* are the pair of moneyness and tenor for the option on the balanced grid, whereas S/K and τ are the pair from an option in the original data. Then, we compute the estimated IV by a weighted average over all the IVs of the original data on that day, where the weights are equal to the similarity measure from 3.2:

$$IV_t^* = \frac{\sum_{i=1}^{n_t} \mathcal{K}(S/K^*, \tau^*, S/K_{t,i}, \tau_{t,i}) IV_{t,i}}{\sum_{i=1}^{n_t} \mathcal{K}(S/K^*, \tau^*, S/K_{t,i}, \tau_{t,i})} \qquad t \in \{1, ..., T\}, \tag{3.3}$$

where n_t is the number of different options traded on day t. Both the moneyness and tenor

are normalized before performing the Gaussian kernel smoother such that they both have a similar influence on the smoothing.

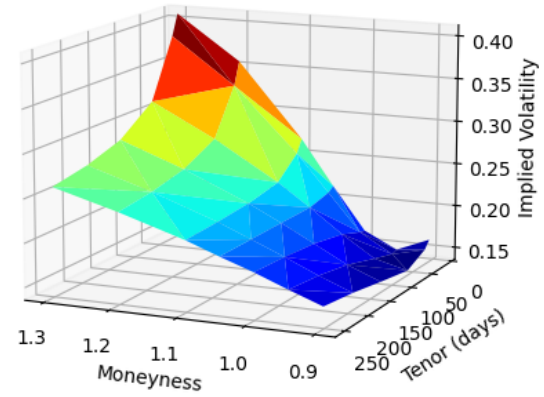


(a) Before smoother (unbalanced). **(b)** After smoother (balanced).

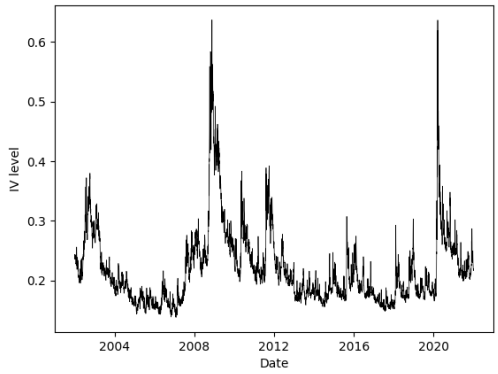
Figure 2: Plots of the IVS on February 28, 2020.

The hyperparameter b (3.2) needs to be optimized. However, as there is no way to objectively measure the performance of the smoothing algorithm, b can not be quantitatively optimized. Therefore, we perform a manual search and compare original IVS plots to their smoothed counterparts (similar to Figure 2) to determine the best value for b, which we found to be 0.05. Figure 2 shows plots of the IVS on the day with the most options traded (February 28^{th}, 2020) using both unbalanced/original data (Figure 2a), and balanced/smoothed data (Figure 2b). This figure shows the presence of a "volatility smirk", which can be seen by the IV decreasing as the moneyness decreases. However, the "volatility smile" is not as prominent, this is caused by the selected boundaries of the grid. Figure B.1 in Appendix B shows that for the full sample, this smile is indeed present in the data, because for even lower moneyness, the IV increases as the moneyness decreases.

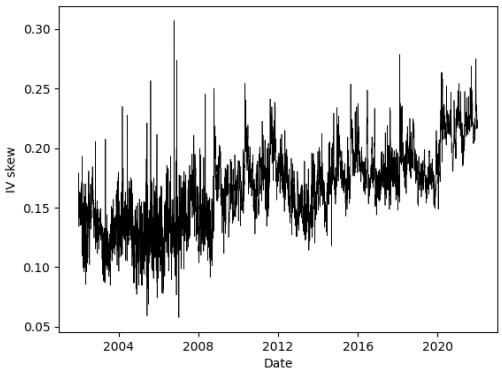
Figure 3 shows the average IVS over time using the balanced data (Figure 3a) and how the IVS evolves over time, explained using the level, skew, and term structure (Figure 3b - 3d). Figure 3a shows the presence of a volatility smile/smirk in the average IVS. However, as the tenor increases, the smile becomes less obvious. The level, skew, and term structure are all calculated using the balanced IVS, because this ensures that the calculations can be performed consistently over time. The level on date t is the average IV of the balanced IVS on date t. Next, the skew on date t is the difference between the average IV of the options



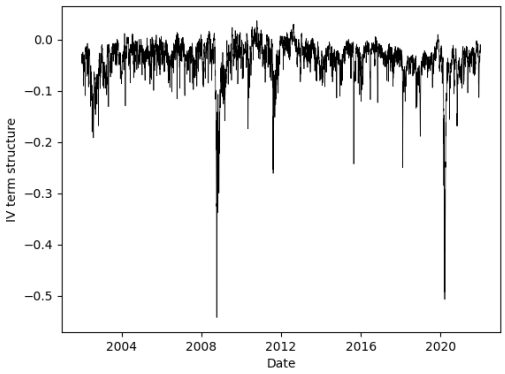
(a) Average IVS.



(b) Level of the IVS over time.



(c) Skew of the IVS over time.



(d) Term structure of the IVS over time.

Figure 3: Summary of the IVS using balanced data.

with the highest and the lowest moneyness on date t ($S/K = 1.3$ and 0.9 respectively). Lastly, the term structure on date t is the difference between the average IV of the options with the highest and the lowest tenor on date t ($\tau = 1$ and $\frac{10}{252}$ respectively). Both the level and the term structure show large peaks for both the Great Recession (December 2007 to June 2009) and the COVID Recession (February 2020 to April 2020).[4] However, the skew shows less obvious signals for these recessions.

3.2 Covariates

We use a set of covariates for the training of one of the methods (IPCA). Most of the choices for covariates are based on Almeida et al. (2022). The first is the CBOE Volatility Index

[4]Dates from https://fredhelp.stlouisfed.org/fred/data/understanding-the-data/recession-bars/.

(VIX), which is a benchmark for market risk and sentiment. Moreover, the VIX is shown to perform well in modeling the IVS (Almeida et al., 2022; Cao et al., 2020). Another risk variable that will be used is the Realized Volatility (RVOL), which is constructed by Tick Data using 5-minute intraday S&P 500 returns.

Almeida et al. (2022) also use measures of uncertainty and macroeconomic conditions. These are, the US Daily News Index (USNI) index of Baker et al. (2016), the Business Condition Index (BCI) of Aruoba et al. (2009), the first differences of the Term Spread (TMS), and the first differences of the Credit Spread (CRS) from the FRED database.

Moreover, we add the Federal Funds Effective Rate (FFER) and the Market Yield on 10-Year U.S. Treasuries (US10YMY) from the FRED database as they are connected to the short-term and long-term risk-free rate respectively. The risk-free rate influences the option prices and possibly also the IVS. Next, we include the monthly U.S. inflation rate (USCPI) from the CBOE database. To avoid forward-looking bias, we set the USCPI for each day equal to the inflation of the previous month. Lastly, we use the return on the S&P 500 index over the previous month (21 trading days) as a covariate which may give information about the current economic state (SPXM).

4 Methodology

This section contains an explanation of the used methods to construct factor models, as well as the performance measures.

4.1 Principal Component Analysis

The first method we consider is Principal Component Analysis (PCA), which assumes a linear and static relation between the data and the factors. The core process of PCA is computing the Principal Components (PCs) of the original data. The PCs are latent factors that are linear combinations of the data such that the i^{th} PC (PC_i) captures as much variation in the data as possible while being orthogonal to the first $i-1$ PCs. For IV data with k different

options, PC_i is constructed as a linear combination of the data:

$$PC_{i,t} := f_{i,t} = \beta_{i,1} IV_{1,t} + ... + \beta_{i,k} IV_{k,t} = \boldsymbol{\beta}'_{i,:}\, IV_t, \qquad i \in \{1, ..., k\} \tag{4.1}$$

where $\boldsymbol{\beta}$ is the $k \times k$ parameter matrix, $\boldsymbol{\beta}_{i,:}$ is defined as the loading vector of PC_i, and IV_t is the vector containing the IV of all options on the balanced grid, on date t. In our case, k is equal to the size of the balanced grid ($k = 42$). For PCA, the loading vectors of the PCs are equal to the eigenvectors of the covariance matrix of the data. More specifically, $\boldsymbol{\beta}_{i,:}$ corresponds to the eigenvector with the i^{th} highest eigenvalue. Moreover, from the eigenvalue corresponding to PC_i (λ_i), it can be calculated what percentage of the total variance is explained by PC_i ($\lambda_{i,\text{total}}$):

$$\lambda_{i,\text{total}} = \frac{\lambda_i}{\sum_{j=1}^{k} \lambda_j}. \tag{4.2}$$

PCA is often used as a dimension reduction method by only keeping the first M ($M << k$) PCs and discarding the rest. This is useful because the first PCs capture the most variation. Moreover, after the PCs are constructed, they can be transformed back into an approximation of the original data using the loadings $\boldsymbol{\beta}$,

$$\begin{aligned} IV_{i,t} =& \beta_{1,i} f_{1,t} + ... + \beta_{M,i} f_{M,t} + e_{i,t} = (\boldsymbol{\beta}_{1:M,i})' \boldsymbol{f}_t + e_{i,t}, \qquad i \in \{1, ..., k\}, \\ e_{i,t} :=& \beta_{M+1,i} f_{M+1,t} + ... + \beta_{k,i} f_{k,t} = (\boldsymbol{\beta}_{M+1:k,i})' \boldsymbol{f}_t, \end{aligned} \tag{4.3}$$

where $e_{i,t}$ is the error term. PCA requires a balanced panel as input. Moreover, the output of PCA only consists of the estimated values of this balanced grid. Therefore, if we need to get estimated values of the IV of options that are not on the grid, we use a Gaussian kernel smoother (3.3). Moreover, PCA normally gives more importance to covariates with higher variance. Since our goal is to estimate all options on the grid with equal importance, we standardize the data before performing PCA. The standardization is performed for each option in the balanced grid separately as follows:

$$\widetilde{IV}_{i,t} = \frac{IV_{i,t} - \overline{IV}_i}{\sigma(IV_i)}, \qquad i \in \{1, ..., k\}, \quad t \in \{1, ..., T\}, \tag{4.4}$$

where $\widetilde{IV}_{i,t}$ is the standardized IV of option i on date t, $\overline{IV}_i$ is the average IV of option i in the train set, and $\sigma(IV_i)$ is the standard deviation of the IV of option i in the train set.

4.2 Instrumented Principal Component Analysis

Instrumented Principal Component Analysis (IPCA) is a dimension reduction method that makes use of "observable characteristics" (or covariates) (Kelly et al., 2019). Similar to PCA, it uses latent factors, but instead of using constant factor loadings, IPCA uses time-varying loadings. The values of the loadings are dependent on a set of covariates. IPCA makes use of two core assumptions. The first is that there is a linear relationship between the latent factors and the original data through the factor loadings (similar to PCA). Moreover, it is assumed that the factor loadings have a linear and constant relation with the covariates. Combining these assumptions into a model, results in the following:

$$\begin{aligned} IV_{i,t} &= \boldsymbol{\beta}'_{i,t}\, \boldsymbol{f}_t + \epsilon_{i,t}, \\ \boldsymbol{\beta}_{i,t} &= \Gamma\, \boldsymbol{c}_{i,t} + \boldsymbol{\eta}_{i,t}, \end{aligned} \tag{4.5}$$

where $IV_{i,t}$ is the Implied Volatility of an option with moneyness K_i and tenor τ_i (in short, option i) at time t. Moreover, $\boldsymbol{f}_t$ is the $M \times 1$ vector of latent factors at time t which needs to be estimated, and $\boldsymbol{c}_{i,t}$ is the $L \times 1$ vector of covariates which apply to option i at time t. The constant linear relation between the covariates ($\boldsymbol{c}_{i,t}$) and the factor loadings ($\boldsymbol{\beta}_{i,t}$) is described by the $M \times L$ parameter matrix Γ. Lastly, $\epsilon_{i,t}$ and $\boldsymbol{\eta}_{i,t}$ are the error terms. Here, M is the number of latent factors and L is the number of covariates. This system of equations can be summarized in one equation as follows:

$$\begin{aligned} IV_{i,t} &= (\Gamma\, \boldsymbol{c}_{i,t})'\, \boldsymbol{f}_t + e_{i,t}, \\ e_{i,t} &\coloneqq \boldsymbol{\eta}'_{i,t}\, \boldsymbol{f}_t + \epsilon_{i,t}. \end{aligned} \tag{4.6}$$

As described, Γ and $\{\boldsymbol{f}_t\}$ need to be estimated. There is no analytical solution to this model. However, it can be speedily solved by using Alternating Least Squares (ALS) (Kelly et al., 2020). ALS iterates between minimizing the error term over Γ while holding $\{\boldsymbol{f}_t\}$ fixed, and minimizing over $\{\boldsymbol{f}_t\}$ while holding Γ fixed, until convergence. These two subproblems are

linear in the parameters, which means that they can be quickly optimized using OLS.

The covariates ($\boldsymbol{c}_{i,t}$) consist of the variables which are discussed in 3.2, complemented with $\tau_{i,t}$, $S/K_{i,t}$, $\tau_{i,t}^2$, $(S/K_{i,t}-1)^2$, and $\tau_{i,t} * S/K_{i,t}$. The nonlinear combinations are added to better capture the nonlinearities in the IVS, such as the volatility smirk. Moreover, the (nonlinear combinations of) tenor and moneyness are the only 'asset-specific characteristics', the rest of the covariates we use (from 3.2) are constant within the cross-section for each date.

Unlike PCA, IPCA does not require a balanced panel as input data as it also works on unbalanced panels. IPCA allows for interpolation by adding the tenor and moneyness (and combinations thereof) to $\boldsymbol{c}_{i,t}$. In this way, estimated values of every point on the IVS can be obtained by changing these values, while holding the others constant. We train IPCA both with the balanced panel (IPCA_B) and the unbalanced panel (IPCA_U). This can be useful to measure the effect that the interpolation/smoothing of the original data has on the training process of the model.

4.3 Autoencoder

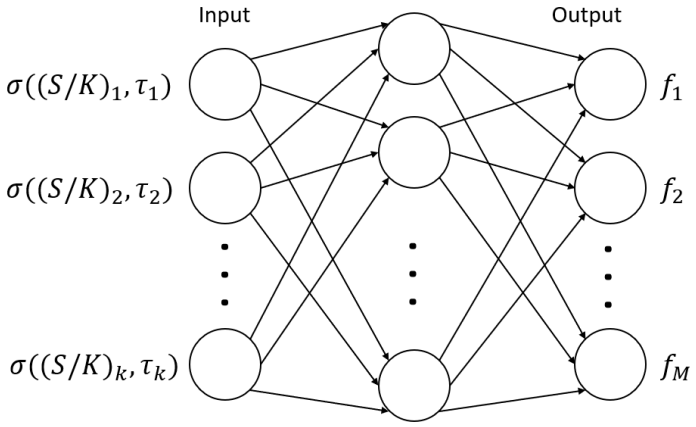
An autoencoder can be seen as a nonlinear extension of PCA and is therefore also sometimes named 'nonlinear PCA'. Instead of constructing the factors using a linear combination of the original data (PCs), an autoencoder is a NN that can capture complex nonlinear relations between the data and the factors. An autoencoder consists of two parts, the encoder, and the decoder.

The encoder is a NN that constructs the factors using the original data. The decoder is a NN that reconstructs the original data, given the factors which are estimated by the encoder. Both the encoder and the decoder generally have at least one hidden layer such that they can capture nonlinearities (Figure 4).

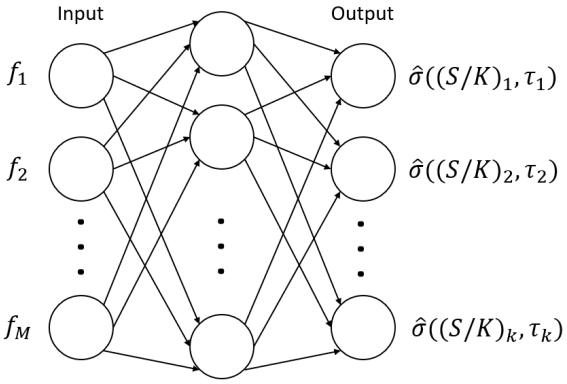
In the option pricing literature, two different decoder architectures are generally used. These are the grid-based approach and the pointwise approach. Note that the encoder architecture is the same, regardless of the choice of decoder architecture (Figure 4a). The grid-based approach is a more traditional approach, where the output is identical to the input, and the autoencoder tries to find the factors that are best able to reconstruct the

original data (Figure 4b). The pointwise approach is more specific to option pricing, as it reconstructs the original data one option at a time, adding the tenor (τ) and moneyness (S/K) to the input of the decoder (Figure 4c). This especially comes in handy for option pricing as it makes interpolation possible. Interpolation of the IVS is often needed to price new options or adjust prices, which understates its importance. Therefore, we opt for the pointwise approach in this paper.

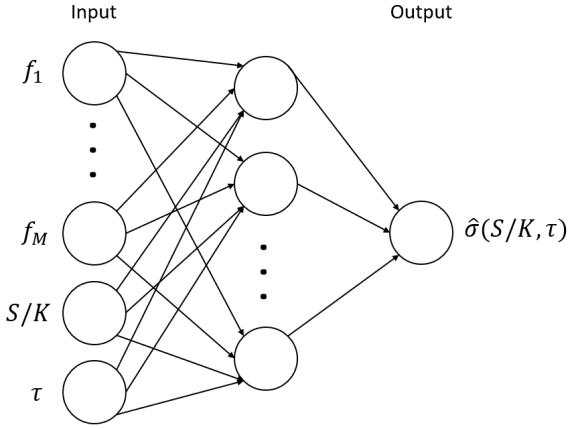
Similar to PCA, an autoencoder needs a balanced panel as input data. Also, both the input data and the output data are standardized. We use autoencoders with 1 hidden layer in the encoder and decoder (AE_1) and 2 hidden layers in the encoder and decoder (AE_2).



(a) The encoder architecture for an autoencoder.



(b) The decoder architecture for the grid-based approach.



(c) The decoder architecture for the pointwise approach.

Figure 4: Illustration of the autoencoder architectures with one hidden layer in the encoder and decoder. Based on Fung (2021).

4.4 Performance measures

The performance is split up into three aspects: Interpretability, modeling performance, and forecasting performance.

There are no quantitative methods to measure interpretability. Therefore, we compare the factors to IVS characteristics. For PCA, the interpretation should be relatively straightforward, as there exists plenty of literature covering this. The interpretation of the IPCA factors is more involved, as the factors are not constant linear combinations of the original data because of the influence of the covariates. For AE, it may be even harder because of the nonlinearities in the relation between the factors and the original data. We will investigate whether it is possible to give some interpretation to the factors of all these methods.

Both the modeling performance and the forecasting performance are measured using the Implied Volatility Root Mean Squared Error ($IVRMSE$). This error metric is defined in terms of implied volatilities (contrary to e.g. R^2), making it easily interpretable and comparable between the balanced and unbalanced test set. It is defined as follows:

$$IVRMSE = \sqrt{\frac{\sum_{(i,t)}(IV_{i,t} - \widehat{IV}_{i,t})^2}{n}}, \tag{4.7}$$

where (i,t) corresponds to option i on day t, and n is defined as the total number of options in the test set (number of elements in the sum of the numerator). Note that a lower $IVRMSE$ implies a lower error and thus a better fit/forecast. We define the modeling performance as the ability of the models to reconstruct the IVS at a given point in time, possibly having to interpolate to obtain the entirety of the IVS. This is measured pseudo-out-of-sample. That is, after training the models in-sample, we estimate the factors out-of-sample making use of the out-of-sample data as input for the estimations. By doing that, the factors at date t (out-of-sample) are estimated using the IV data at date t. Afterwards, these estimated factors are reconstructed into estimated IV data using the in-sample trained model, measuring the ability to reconstruct the IVS. Furthermore, we use two different out-of-sample test sets to measure the pseudo-out-of-sample performance. Firstly, we use the balanced test set, from which the observations are proxies of the real data using interpolation. Therefore, this may not tell us everything about the performance in practice. Second, we use the unbalanced

test set, which consists of the original observations before interpolation/smoothing. Results from the unbalanced test set better mimic real-life applications, as it requires the models to perform interpolation themselves. IV estimates from the unbalanced test set can easily be obtained by both IPCA and AE, as both methods have a built-in interpolation function. For PCA, we use a Gaussian kernel smoother to obtain estimates for this set.

Lastly, we look at the forecasting performance of the methods, which is measured out-of-sample. We make a distinction between two types of forecasts; direct forecasts and indirect forecasts. For direct forecasts, the model is directly trained to perform forecasts, usually by using lagged values as input. For indirect forecasts, the model is trained as usual (no lagged covariates). However, after training the model and obtaining the latent factors, these factors are forecasted using a Vector Autoregressive (VAR) model. We use a VAR model because the characteristics of the IVS all have a high autocorrelation with 98.5%, 88.7%, and 89.0% respectively for the level, skew, and term structure of the IVS. Moreover, the factors are usually in some way related to these characteristics, which means that the factors likely also have a high autocorrelation. Also, the characteristics are correlated to each other, though less strongly. These relations can all be captured using a VAR model. The VAR(p) model for modeling/forecasting the factors is defined as follows:

$$\boldsymbol{f}_t = \boldsymbol{c} + A_1 \boldsymbol{f}_{t-1} + ... + A_p \boldsymbol{f}_{t-p} + \boldsymbol{u}_t, \qquad \boldsymbol{u}_t \sim N(0, \Sigma_{\boldsymbol{u}}), \tag{4.8}$$

with $\boldsymbol{c}$ being the $M \times 1$ parameter vector denoting the constant and A_i being the $M \times M$ parameter matrix denoting the coefficients for the i^{th} lag. This means that for a VAR(p) model, there are a total of $M + p * M^2$ parameters to estimate, assuming $\boldsymbol{f}_t$ to be of size $M \times 1$. Lastly, $\boldsymbol{u_t}$ is the error vector.

We use the VAR(p) model to make step-by-step dynamic forecasts. That is, for an arbitrary horizon h, we start with forecasting 1-step-ahead ($\boldsymbol{f}_{t+1}$) by using the estimated parameters and setting $\hat{\boldsymbol{u}}_{t+1} = E(\boldsymbol{u}_{t+1}) = 0$. Then, for the next step, we use (among others) the estimated value of the previous period ($\hat{\boldsymbol{f}}_{t+1}$) to give a forecast 2-steps-ahead, and so

forth. Mathematically, this can be written as:

$$\hat{\boldsymbol{f}}_{t+h|t} = \hat{\boldsymbol{c}} + \widehat{A}_1 \hat{\boldsymbol{f}}_{t+h-1|t} + ... + \widehat{A}_{h-1} \hat{\boldsymbol{f}}_{t+1|t} + \widehat{A}_h \boldsymbol{f}_t + ... + \widehat{A}_p \boldsymbol{f}_{t+h-p}, \qquad \text{if } p \geq h, \tag{4.9}$$
$$\hat{\boldsymbol{f}}_{t+h|t} = \hat{\boldsymbol{c}} + \widehat{A}_1 \hat{\boldsymbol{f}}_{t+h-1|t} + ... + \widehat{A}_p \hat{\boldsymbol{f}}_{t+h-p|t}, \qquad \text{if } p < h,$$

where h is the forecasting horizon and $\hat{\boldsymbol{f}}_{t+h|t}$ is the h-step-ahead forecast of $\boldsymbol{f}$, using observations up until (and including) $\boldsymbol{f}_t$.

PCA only allows for indirect forecasts. Moreover, IPCA can only use a hybrid forecasting method, that is, Γ (and therefore also $\boldsymbol{\beta}$) is trained using lagged observed characteristics (direct), and $\boldsymbol{f}_t$ is forecasted by using a VAR model. AE forecasts can be made either directly or indirectly. For the direct forecast, the input consists of lagged IV values of the balanced grid, which means that the model is directly trained to forecast the IVS. For the indirect forecast, the AE is trained as usual (identical input and output), but the factors are forecasted using a VAR model.

The forecast horizons (h) used are; one day, one week, and one month ($h = 1, 5, 21$ respectively). We use the balanced and unbalanced test set to obtain the *IVRMSE* separately. The number of lags included in the VAR model (p) is determined using the Bayesian Information Criterion (BIC). The BIC helps with model selection by using the in-sample likelihood function and penalizing for the number of parameters in the model, which leads to a trade-off between in-sample accuracy and the complexity of the models to prevent overfitting. We determine the optimal p for each model and each number of factors (M) separately. Moreover, for IPCA the optimal p is also determined for each horizon separately because the hybrid forecasts cause the factors to change for different forecasting horizons.

4.4.1 IPCA variable selection

IPCA is prone to overfitting by including too many or highly correlated variables. Therefore, we apply variable selection to find the best configuration of variables. We use a backward step-wise selection approach using bootstrapping. Because the parameters are estimated using ALS, the model does not return standard errors and p-values for the significance of parameters. Therefore, we use the performance measure *IVRMSE* (4.7) in deciding which

variables to delete.

We use a backward step-wise selection, which means that we start with the IPCA model with all variables included. Then, the variable which harms the performance most is deleted. The performance is measured using the *IVRMSE*. This is repeated until deleting any of the variables gives a worse performance than keeping all of the remaining.

For each selection of variables, the performance of the model is measured by computing the *IVRMSE* using non-overlapping block bootstrapping. This means that the original balanced training set is divided into B equally sized, non-overlapping blocks. Then, the model is trained on $B-1$ of the blocks, and 1 of the blocks is used to test the model performance. This procedure is repeated B times such that all blocks are used as the test set exactly once. The final *IVRMSE* of the model, which is used as the performance measure, is the average of the *IVRMSE* for each of the B repetitions.

We use 5 non-overlapping blocks ($B = 5$) on the 3-factor model ($M = 3$) of IPCA_B for the variable selection. The deleted variables are SPXM, FFER, and US10YMY.

4.4.2 Autoencoder hyperparameter tuning

An autoencoder contains multiple hyperparameters which need to be tuned. This is done using a grid-search, where the performance is measured via the *IVRMSE* (4.7) using non-overlapping block bootstrapping ($B = 5$), similar to Section 4.4.1. The hyperparameters are tuned on the model with 3 factors ($M = 3$) and are tuned separately for AE_1 and AE_2. Table 2 shows which hyperparameters we choose to optimize over, which candidate values are chosen for the grid search, and which combination of values turns out to be optimal for AE_1. Table 3 displays the same information, but for AE_2. All hidden layers in the encoder and decoder are set to have the same width. Moreover, the chosen activation function is used for all layers except for the output layer of the decoder. This layer has a linear activation function because the option IV data is standardized, meaning that the data does not fall within strict boundaries. These strict boundaries are assumed for ReLU (greater than 0), Sigmoid (between 0 and 1), and Tanh (between -1 and 1), which makes them unsuitable for the output layer of the decoder. Moreover, the Adam solver (Kingma and Ba, 2014) is used for optimization for all models.

Table 2: Hyperparameter tuning for AE_1.

Hyperparameter	Candidates	Optimal
Epochs	(100, 250)	100
Batch size	(42, 84)	84
Width	(32, 64, 128)	64
Activation	(ReLU, Sigmoid, Tanh)	ReLU

Table 3: Hyperparameter tuning for AE_2.

Hyperparameter	Candidates	Optimal
Epochs	(100, 250)	100
Batch size	(42, 84)	84
Width	(32, 64, 128)	64
Activation	(ReLU, Sigmoid, Tanh)	ReLU

5 Results

The results are split up into three parts. As described in Section 4.4, these are; interpretability, modeling performance, and forecasting performance. The results are obtained using Python. More specifically, for IPCA we use the package of Kelly et al. (2019) and Kelly et al. (2020).[5]

5.1 Interpretability of the factors

For all methods, we look at the factors of the 3-factor model over the entire sample (train and test set) for the interpretability of the factors. We choose 3 factors because, in the literature, most research applying PCA to modeling the IVS uses 3 factors (Andersen et al., 2015b). We denote the i^{th} factor of method X as X-i (e.g. PCA-1 for the first factor of PCA). Note that none of the methods identify the sign of the factors. Therefore, we multiply some factors by -1 to make interpretation more convenient.

PCA For PCA, the interpretability of the factors is relatively straightforward because this has been done multiple times before in the literature. Generally, for a 3-factor model, the factors are related to the level, skew, and term structure of the IVS. This also holds for our dataset. Appendix C contains plots of each factor for every model, including the IVS

[5]The IPCA package is available at https://bkelly-lab.github.io/ipca/.

characteristic with the highest correlation for each factor. Moreover, Table 4 shows the correlation between the PCA factors and these characteristics of the IVS.

PCA-1 has a correlation of 0.999 with the level of the IVS. This means that the first factor approximately returns the average IV over the grid on each date. Moreover, PCA-2 is highly influenced by the term structure of the IVS, with a correlation of 0.856. Finally, PCA-3 comes very close to the skew of the IVS with a correlation of 0.920.

It should be noted that correlations do not always imply that two time series are related. For this to be true, the time series should be stationary. If the time series are non-stationary, having a similar trend/seasonal effect can cause high correlations, even though the two variables might not be related. To test whether the time series are stationary, we use the Augmented Dickey-Fuller (ADF) test (Dickey and Fuller, 1979). The ADF test tests for non-stationarity in a time series. The null hypothesis (H_0) states that a time series is non-stationary, whereas the alternative hypothesis states that a time series is stationary. Appendix D contains the full test results of the ADF test for the level, skew, and term structure of the IVS, and the factors of all models. It can be seen that for all factors and IVS characteristics H_0 is rejected for a 5% confidence level. Thus, the time series are stationary, and correlations can be interpreted as usual.

Table 4: Correlation between the PCA factors and characteristics of the IVS.

Factor	Level	Term str.	Skew
PCA-1	**0.999**	-0.594	0.256
PCA-2	-0.145	**0.856**	-0.329
PCA-3	0.028	0.010	**0.920**

Note: **Bold** values indicate the highest absolute correlation of each factor.

IPCA Table 5 shows the correlation between the IPCA$_B$ and IPCA$_U$ factors with the characteristics of the IVS. For IPCA$_B$, not all factors are easily interpretable. Both IPCA$_B$-2 and IPCA$_B$-3 are (somewhat) similar to the characteristics of the IVS. However, IPCA$_B$-1 seems to behave completely differently.

More specifically, IPCA$_B$-2 has a correlation of 0.978 with the term structure of the IVS, which means that IPCA$_B$-2 almost exactly follows the term structure. Moreover, IPCA$_B$-3

is also mostly correlated with the term structure of the IVS. However, with a correlation of 0.713, this relation is less strong than for IPCA$_B$-2.

Lastly, IPCA$_B$-1 has a very low correlation with all of the characteristics of the IVS. Figure C.2a in Appendix C shows the plot of IPCA$_B$-1 over time. The figure shows that, after standardization, IPCA$_B$-1 fluctuates relatively close to 0 most of the time. However, it has large spikes reaching values of over 30 multiple times. There can not be given any interpretation to such a factor. It also raises the question of whether IPCA in its current form is suitable for modeling the IVS because this unexpected behavior could potentially negatively influence the performance and seems to be impossible to forecast.[6]

Table 5: Correlation between the IPCA factors and characteristics of the IVS.

Factor	Level	Term str.	Skew
IPCA$_B$-1	0.001	**0.010**	-0.008
IPCA$_B$-2	-0.619	**0.978**	-0.204
IPCA$_B$-3	-0.478	**0.713**	0.384
IPCA$_U$-1	-0.007	0.000	**0.011**
IPCA$_U$-2	-0.602	**0.988**	-0.219
IPCA$_U$-3	-0.275	0.398	**0.705**

Note: **Bold** values indicate the highest absolute correlation of each factor.

The results for IPCA$_U$ are similar to the results for IPCA$_B$. Similar to IPCA$_B$-1, IPCA$_U$-1 also does not have a strong correlation to any of the characteristics of the IVS. Figure C.3a in Appendix C indeed shows that the behavior of IPCA$_U$-1 and IPCA$_B$-1 are very similar, with many large spikes in the data.

Moreover, IPCA$_U$-2 has a correlation of 0.988 with the term structure of the IVS, which is similar to that of IPCA$_B$-2. This means that IPCA$_U$-2 is easily interpretable as it follows the term structure very closely.

Lastly, IPCA$_U$-3 is not similar to IPCA$_B$-3. Although IPCA$_B$-3 is mostly correlated with the term structure of the IVS, IPCA$_U$-3 mostly follows the skew of the IVS with a correlation of 0.705.

[6] We tried multiple different IPCA architectures (choices of covariates), but they all gave similar results.

AE Table 6 displays the correlation between the AE factors and the level, term structure, and skew of the IVS.

AE_1-1 has a correlation of 0.997 with the level of the IVS. This is almost the same as PCA-1, their similarity is understated by the correlation of 0.999 between AE_1-1 and PCA-1. This means that, although AE_1 can make use of nonlinear combinations of the data, AE_1-1 is approximately the average IV on each date.

Moreover, AE_1-2 has a correlation of 0.881 with the term structure of the IVS, which is similar to PCA-2. Again, the correlation between AE_1-2 and PCA-2 is relatively high with 0.741.

Lastly, AE_1-3 has a correlation of 0.838 with the skew of the IVS, which again is similar to PCA-3. Their similarity is also shown by the correlation between AE_1-3 and PCA-3 being 0.929.

Concluding, the factors of AE_1 are very similar to those of PCA. This is remarkable because PCA exclusively makes use of a linear combination of the original data to construct the factors, whereas AE_1 can also use non-linear combinations.

Table 6: Correlation between the AE factors and characteristics of the IVS.

Factor	Level	Term str.	Skew
AE_1-1	**0.997**	-0.562	0.234
AE_1-2	-0.639	**0.881**	-0.670
AE_1-3	0.044	0.044	**0.838**
AE_2-1	**0.995**	-0.612	0.205
AE_2-2	**0.559**	-0.044	0.078
AE_2-3	-0.114	-0.320	**0.765**

Note: **Bold** values indicate the highest absolute correlation of each factor.

The AE_2 factors are also well interpretable, though somewhat harder than for AE_1. Similar to AE_1 and PCA, AE_2-1 is also highly correlated to the level of the IVS with a correlation of 0.995. This means that, even though more complex nonlinear relations can be made, AE_2 still prefers one of the factors to follow the level of the IVS, which understates how important the level is in modeling the IVS using a factor model.

Moreover, AE_2-3 has a correlation of 0.765 with the skew of the IVS. Although this is lower than for PCA-3 and AE_1-3, it is still obvious that AE_2-3 is largely influenced by the

skew of the IVS.

Finally, AE_2-2 does not have a large correlation with the term structure of the IVS, unlike PCA-2 and AE_1-2. This factor has some correlation with the level of the IVS (0.559) but is not easily interpretable. It is not entirely surprising that the factors of AE_2 are somewhat harder to interpret, as there are two neural networks with 2 hidden layers underlying AE_2, these can construct complex nonlinear relations between the data and the factors.

5.2 Modeling performance

After the qualitative analysis of the interpretability of the factors, we now focus on the quantitative modeling performance. As discussed, the modeling performance is measured using the Implied Volatility Root Mean Squared Error (*IVRMSE*) (4.7). Table 7 shows the results of the *IVRMSE* on the balanced test set using 1 to 6 factors. As these results are based on the balanced test set, they show the ability of the models to exactly replicate the input data, as there is no interpolation needed to reconstruct the balanced test set.

First, PCA performs very well on the balanced test set. Because the training of PCA is deterministic, there is no way to end up in a local minimum which makes its performance very robust. Moreover, as the factors are orthogonal to each other, adding extra factors rarely damages the performance on the balanced test set as this makes reproducing the original balanced IVS easier. We can see this back in the results, as the *IVRMSE* consistently decreases for a higher number of factors. This leads to PCA achieving the lowest *IVRMSE* of all methods for the 3- to 6-factor models, with the overall lowest *IVRMSE* of 0.60% for the 6-factor PCA model.

Next, $IPCA_B$ and $IPCA_U$ both have the highest *IVRMSE* for most numbers of factors. This is not entirely surprising as the factors behave unusually with large spikes, which could mean that there is something wrong with the model. Although $IPCA_B$ and $IPCA_U$ perform worst, they do not entirely fail to recreate the IVS with a *IVRMSE* of under 4.00% for all numbers of factors. Lastly, as expected, $IPCA_B$ outperforms $IPCA_U$ on the balanced test set, as $IPCA_B$ is trained on the balanced training set.

Furthermore, AE_1 performs particularly well for low numbers of factors, outperforming PCA for 1- and 2-factor models, and achieving the overall lowest *IVRMSE* for the 1-factor

Table 7: *IVRMSE* for the balanced test set (%).

	Number of factors (M)					
Method	1	2	3	4	5	6
PCA	2.73	1.93	**0.94**	**0.78**	**0.70**	**0.60**
IPCA_B	3.40	2.85	2.10	2.05	2.03	2.01
IPCA_U	3.65	3.28	2.55	2.13	2.03	2.01
AE_1	**2.64**	1.85	1.10	0.85	0.79	0.85
AE_2	6.60	**1.47**	1.24	0.87	0.83	0.87

Note: **Bold** values indicate the lowest *IVRMSE* for each M.

model of 2.64%. For higher numbers of factors, AE_1 still performs well, although it is outperformed by PCA.

Lastly, AE_2 has a very high *IVRMSE* for the 1-factor model of 6.60%, which is an outlier compared to the other numbers of factors. Therefore, this result could be caused by reaching a local minimum in training. This understates the danger of AEs and Neural Networks in general, as there always is a chance of ending up in a disadvantageous local minimum during training, which can substantially influence the performance of the model. For the other numbers of factors, AE_2 has a similar *IVRMSE* to AE_1, having the overall lowest *IVRMSE* for 2-factor models with 1.47%.

Table 8 displays the *IVRMSE* for the unbalanced test set. The difference with Table 7 is that the results in Table 8 are obtained using the original (unbalanced) test set. This implies that the test set does not consist of a steady grid, meaning that most methods need to apply some sort of interpolation. Also, these results come closer to real-life applications because almost none of the options lay exactly on the balanced grid.

As PCA does not have a built-in mechanism for interpolation, we use the same method used for constructing the balanced dataset, a Gaussian kernel smoother with the same hyperparameter, $b = 10^{-4}$ (3.2, 3.3).

Table 8 shows that the performance of PCA is, among all methods, most negatively impacted by having to perform interpolation. This most likely has to do with the absence of a built-in method to perform interpolation for PCA, which makes it dependent on a non-trained method to do so. For the 1- and 2-factor models, PCA performs worst out of all methods, and for higher numbers of factors, it outperforms both IPCA methods in terms of

IVRMSE, but still falls behind both AE methods by some margin. Similar to the PCA results for the balanced test set, the *IVRMSE* again decreases consistently for higher numbers of factors, with the lowest *IVRMSE* being 2.77% for the 6-factor PCA model.

Table 8: *IVRMSE* for the unbalanced test set (%).

	Number of factors (M)					
Method	1	2	3	4	5	6
PCA	4.35	3.71	2.89	2.81	2.78	2.77
IPCA_B	4.01	3.61	3.13	3.04	2.88	2.82
IPCA_U	3.77	3.46	2.92	2.90	2.85	2.82
AE_1	**3.53**	2.76	2.20	2.94	2.30	2.46
AE_2	8.18	**2.44**	**2.05**	**1.78**	**1.76**	**2.31**

Note: **Bold** values indicate the lowest *IVRMSE* for each M.

Next, IPCA again performs poorly compared to the other methods, especially for higher numbers of factors. IPCA_U does outperform IPCA_B in the unbalanced setting, which is to be expected as IPCA_U is trained on the unbalanced training set. Remarkably, IPCA_U has a higher *IVRMSE* on the unbalanced test set than for the balanced test set. This suggests that the unbalanced test set may be intrinsically harder to model than the balanced test set. This may be caused by the inclusion of more extreme data points in the unbalanced test set, these are somewhat flattened in the balanced test set due to the interpolation. Lastly, IPCA_B and IPCA_U seem to converge to a similar model for a higher number of factors as the *IVRMSE* for both the balanced (2.01%) and the unbalanced (2.82%) test set are equal between the 6-factor models of the two methods.

Lastly, AE clearly outperforms both IPCA and PCA for the unbalanced test set. Table 8 shows that for all numbers of factors, one of the AE methods returns a model which has the lowest *IVRMSE*. Again, the *IVRMSE* of the 1-factor AE_2 model is remarkably bad with 8.18%, which is not strange as this is the same model which achieved a 6.60% *IVRMSE* for the balanced test set, but now tested on the unbalanced test set. Hence, AE_1 outperforms AE_2 (and the other methods) for the 1-factor model, with a *IVRMSE* of 3.53%. However, for the 2- to 6-factor models, AE_2 is the dominant method. This is in contrast to the results for the balanced test set, where AE_1 generally slightly outperforms AE_2. The lowest *IVRMSE* for the unbalanced test set is obtained by the 5-factor AE_2 model, with 1.76%

5.3 Forecasting performance

After the interpretability and modeling performance, the forecasting performance is the last performance measure. As discussed, the forecasts are made indirect for PCA and AE_I, hybrid for IPCA, and direct for AE_D. Moreover, the forecasts are made 1-day, 1-week, and 1-moth ahead ($h = 1, 5, 21$ respectively).

Table 9: Forecasting *IVRMSE* for the balanced test set (%).

	Number of factors (M)					
Method	1	2	3	4	5	6
			Horizon = 1			
PCA	3.15	2.56	**1.96**	**1.91**	**1.88**	**1.86**
$IPCA_B$	3.96	5.93	> 99	> 99	> 99	> 99
$IPCA_U$	4.60	4.00	67.13	> 99	> 99	> 99
$AE_{1,I}$	**3.10**	**2.54**	2.08	1.96	1.94	1.96
$AE_{2,I}$	6.60	2.75	2.17	1.97	1.94	1.96
$AE_{1,D}$	3.20	3.02	2.73	2.65	3.27	3.23
$AE_{2,D}$	6.62	2.84	2.74	2.58	2.76	2.55
			Horizon = 5			
PCA	3.95	**3.59**	**3.21**	**3.19**	**3.20**	**3.17**
$IPCA_B$	5.26	4.51	> 99	> 99	> 99	> 99
$IPCA_U$	6.50	4.76	45.93	> 99	> 99	> 99
$AE_{1,I}$	**3.91**	3.67	3.31	3.25	3.26	3.27
$AE_{2,I}$	6.60	3.97	3.44	3.30	3.25	3.27
$AE_{1,D}$	4.48	4.22	4.77	4.73	4.32	4.55
$AE_{2,D}$	6.64	4.23	3.93	4.43	4.45	4.43
			Horizon = 21			
PCA	**5.94**	**5.84**	**5.73**	**5.69**	**5.71**	**5.67**
$IPCA_B$	10.88	6.64	11.48	> 99	> 99	> 99
$IPCA_U$	11.88	6.92	96.91	> 99	> 99	> 99
$AE_{1,I}$	5.95	6.10	5.76	5.76	5.73	5.74
$AE_{2,I}$	6.60	6.48	6.00	5.79	5.72	5.72
$AE_{1,D}$	6.59	6.36	6.63	6.47	6.53	6.86
$AE_{2,D}$	6.53	6.94	6.15	6.63	6.59	6.29

Note: **Bold** values indicate the lowest *IVRMSE* for each M and horizon.

Table 9 shows the forecasting *IVRMSE* of all models on the balanced test set. First of all, similar to the modeling performance, PCA generally outperforms the other methods on the balanced test set with both $AE_{1,I}$ and $AE_{2,I}$ following closely. This relationship between PCA and AE_I seems to be relatively consistent over all horizons. Moreover, the *IVRMSE*

of $\text{AE}_{1,I}$ and $\text{AE}_{2,I}$ are very similar across different numbers of factors (except $M = 1$) and horizons. For all horizons, the lowest (best) *IVRMSE* is achieved by PCA, with 1.86%, 3.17%, and 5.67% for $h = 1, 5, 21$ respectively.

Second, both IPCA_B and IPCA_U achieve a relatively low *IVRMSE* for the 1- and 2-factor models for the 1-day and 1-week ahead forecasts, with IPCA_B generally outperforming IPCA_U. The extreme increase of the *IVRMSE* for higher numbers of factors is likely to be caused by the unpredictable behavior of one of the factors in the 3-factor model of both IPCA_B and IPCA_U, as discussed in Section 5.1.

Lastly, the direct forecasts of $\text{AE}_{1,D}$ and $\text{AE}_{2,D}$ can not compete with the indirect forecasts. Although the *IVRMSE* for low numbers of factors is somewhat similar to those of the indirect forecasts, the difference becomes more clear for higher numbers of factors. It can not be said whether $\text{AE}_{1,D}$ outperforms $\text{AE}_{2,D}$ or vice versa. Table 9 shows that both methods achieve the lower *IVRMSE* approximately an equal amount of times. However, $\text{AE}_{2,D}$ has a slight edge as it achieves the lowest overall *IVRMSE* between the two methods for each horizon.

Table 10 shows the forecasting *IVRMSE* of all models on the unbalanced test set. As discussed in Section 5.2, testing on the unbalanced test set gives a better representation of the performance of the models in real-life applications. Using the unbalanced test set instead of the balanced test set seems to affect the relative performance among the models in a similar manner as for the modeling performance. That is, the performance of PCA is most negatively affected, which leads to AE_I being the dominant forecasting method. Still, PCA outperforms both AE_D and IPCA which means that it is still the second best method after AE_I.

Remarkably, for all horizons, $\text{AE}_{1,I}$ achieves a substantially lower *IVRMSE* than $\text{AE}_{2,I}$ for 1-3 factor models, and $\text{AE}_{2,I}$ achieves a substantially lower *IVRMSE* for 4-6 factor models. Therefore, it can not be said which of these methods is superior. However, as $\text{AE}_{2,I}$ achieves the overall lowest *IVRMSE* for each horizon (although with a slight margins), an edge can be given to $\text{AE}_{2,I}$ over $\text{AE}_{1,I}$. The lowest *IVRMSE* is achieved using the 4-factor $\text{AE}_{2,I}$ model for each horizon, and is equal to 2.99%, 4.56%, and 7.47% for $h = 1, 5, 21$ respectively.

Lastly, the results for both IPCA_B and IPCA_U are similar to those for the balanced test

Table 10: Forecasting *IVRMSE* for the unbalanced test set (%).

Method	Number of factors (M) 1	2	3	4	5	6
Horizon = 1						
PCA	4.80	4.29	3.68	3.66	3.64	3.62
IPCA_B	4.67	7.35	> 99	> 99	> 99	> 99
IPCA_U	4.58	4.27	78.41	> 99	> 99	> 99
$\text{AE}_{1,I}$	**4.12**	**3.58**	**3.13**	3.93	3.53	3.56
$\text{AE}_{2,I}$	8.18	3.73	3.19	**2.99**	**3.00**	**3.36**
$\text{AE}_{1,D}$	4.25	4.15	3.88	3.82	4.77	4.56
$\text{AE}_{2,D}$	8.16	4.03	3.93	3.73	4.11	3.70
Horizon = 5						
PCA	5.72	5.37	4.97	4.99	5.00	4.96
IPCA_B	5.99	5.81	> 99	> 99	> 99	> 99
IPCA_U	6.67	5.62	48.57	> 99	> 99	> 99
$\text{AE}_{1,I}$	**5.19**	**4.96**	**4.57**	5.37	5.18	5.12
$\text{AE}_{2,I}$	8.18	5.15	4.74	**4.56**	**4.58**	**4.87**
$\text{AE}_{1,D}$	5.79	5.54	6.15	6.40	5.68	6.13
$\text{AE}_{2,D}$	8.23	5.64	5.30	6.02	6.10	6.00
Horizon = 21						
PCA	8.02	7.90	7.84	7.81	7.83	7.80
IPCA_B	11.86	8.34	12.64	> 99	> 99	> 99
IPCA_U	12.76	8.47	> 99	> 99	> 99	> 99
$\text{AE}_{1,I}$	**7.66**	**7.85**	**7.48**	8.19	8.07	7.97
$\text{AE}_{2,I}$	8.18	8.17	7.78	**7.47**	**7.48**	**7.67**
$\text{AE}_{1,D}$	8.37	8.09	8.47	8.20	8.38	8.81
$\text{AE}_{2,D}$	8.05	8.83	7.95	8.52	8.43	8.12

Note: **Bold** values indicate the lowest *IVRMSE* for each M and horizon.

set. Again, for models with 3 factors or more, the *IVRMSE* is extremely high, likely caused by the unusual and unpredictable behavior of (some of) the factors.

Because the factors may have a connection with some macroeconomic variables, we also attempt to forecast the factors using a Neural Network (NN) instead of a VAR model for the indirect and hybrid forecasting methods. As input for the NN, we use the covariates discussed in Section 3.2, complemented with the lagged value of the factors. Table E.1 in Appendix E shows the *IVRMSE* of these forecasts on the unbalanced test set. Clearly, these results are inferior to those obtained with the VAR model, especially for larger horizons. Therefore, we conclude that a VAR model is more suitable for forecasting the factors than a

NN with the current selection of covariates.

6 Discussion

6.1 Conclusion

In this research, we compare the performance of different methods to construct factor models describing the Implied Volatility Surface (IVS). The used methods are Principal Component Analysis (PCA), Instrumented PCA (IPCA), and Autoencoders (AE). The performance of the methods is split up into three parts: Interpretability, modeling performance, and forecasting performance. The modeling performance and forecasting performance are both measured using the Implied Volatility Root Mean Squared Error (*IVRMSE*).

Firstly, comparing the interpretability of the 3-factor models of each method leads to the conclusion that the factors of both PCA and AE_1 are easily interpretable using some characteristics of the IVS (level, skew, and term structure). This is somewhat surprising for AE_1 as this is a NN where more complicated nonlinear combinations can be made between the data and the factors. Moreover, AE_2 still contains a factor that mimics the level of the IVS almost perfectly, but the other factors are less similar to the other characteristics of the IVS. It seems like the interpretability suffers from more hidden layers in the AE. This is not surprising, as increasingly more complicated connections between the IVS data and the factors can be made with more hidden layers. Lastly, for both $IPCA_B$ and $IPCA_U$, one of the factors behaves in an unexpected way by fluctuating around 0 most of the time, with occasionally a large spike. This leads to the conclusion that either IPCA is not suitable for modeling the IVS, or the chosen covariates are not of sufficient quality for IPCA to converge properly.

Secondly, the modeling performance is measured on the balanced and unbalanced test set using the *IVRMSE*. For the balanced test set, PCA has the lowest (best) *IVRMSE* for most numbers of factors, and the lowest overall *IVRMSE* of 0.60%. AE_1 and AE_2 closely follow PCA, having a lower *IVRMSE* for the 1- and 2-factor models. $IPCA_B$ and $IPCA_U$ substantially fall behind the other methods, which was to be expected considering the aforementioned unexpected behavior of the factor. Testing on the unbalanced test set gives a

better representation of the performance of the methods in real-life applications. This is because the methods need to perform some interpolation instead of exactly replicating the input grid (balanced test set). For the unbalanced test set, both AE_1 and AE_2 are clearly the dominant methods, with the lowest *IVRMSE* of 1.76%, and with AE_2 slightly outperforming AE_1. The *IVRMSE* of PCA is affected the most by switching to the unbalanced test set. This is not surprising, as it is the only method without a built-in interpolation method. The *IVRMSE* of both $IPCA_B$ and $IPCA_U$ is again substantially worse than that of AE. However, it has similar *IVRMSE* to PCA for the unbalanced test set.

Lastly, the forecasting performance is measured on the balanced and unbalanced test set using the forecasting *IVRMSE*. Similar to the modeling performance, PCA is the dominant method for the balanced test set with $AE_{1,I}$ and $AE_{2,I}$ following closely (all indirect forecasts). Moreover, for the unbalanced test set, the *IVRMSE* of PCA increases substantially more than for the rest of the methods, which results in AE_I being the dominant method. Furthermore, the direct forecasts of AE_D perform decently well, but are outperformed by both AE_I and PCA for the balanced and unbalanced test set. For both AE_I and AE_D, the difference in forecasting *IVRMSE* between the models with 1 and 2 hidden layers is not substantial. Lastly, the *IVRMSE* of IPCA increases extremely for the models with 3 or more factors. This is likely to be caused by the aforementioned unpredictable behavior of (some of) the factors of IPCA.

All in all, AE_1 seems to be the only method that competes for first place in all three performance compartments. Together with PCA, it is most easily interpretable, and together with AE_2, it achieves the best modeling and forecasting performance, measured on the unbalanced test set. Therefore, regardless of the focus lying on the interpretability of the factors or minimizing the *IVRMSE*, AE_1 is a viable method to use.

6.2 Limitations and research recommendations

We used IPCA to account for the possibility of the IVS being non-static, and AE to account for the nonlinearities in the data. However, we did not incorporate a method which accounts for both of these possible shortcomings of PCA at once. It may be an interesting avenue for future research to investigate the performance of such a method.

Finally, the current implementation of IPCA does not give results as expected. This already becomes clear by viewing the behavior of the factors of the 3-factor model, with one factor fluctuating close to 0, occasionally having large spikes. The modeling and forecasting performance of IPCA is also worse than the others. Therefore, one can conclude that either IPCA does not work properly for modeling the IVS, or the current set of covariates is insufficient. If the latter is the case, it may be worth further looking into which covariates should be added to make IPCA perform better.

Bibliography

Almeida, C., Fan, J., Freire, G., and Tang, F. (2022). Can a machine correct option pricing models? *Available at SSRN 3835108*.

Almeida, C., Freire, G., Azevedo, R., and Ardison, K. (2021). Nonparametric option pricing with generalized entropic estimators. *Available at SSRN 2535790*.

Andersen, T. G., Fusari, N., and Todorov, V. (2015a). Parametric inference and dynamic state recovery from option panels. *Econometrica*, 83(3):1081–1145.

Andersen, T. G., Fusari, N., and Todorov, V. (2015b). The risk premia embedded in index options. *Journal of Financial Economics*, 117(3):558–584.

Aruoba, S. B., Diebold, F. X., and Scotti, C. (2009). Real-time measurement of business conditions. *Journal of Business & Economic Statistics*, 27(4):417–427.

Avellaneda, M., Healy, B., Papanicolaou, A., and Papanicolaou, G. (2020). Pca for implied volatility surfaces. *The Journal of Financial Data Science*, 2(2):85–109.

Badshah, I. (2009). Modeling the dynamics of implied volatility surfaces. *Available at SSRN 1347981*.

Baker, S. R., Bloom, N., and Davis, S. J. (2016). Measuring economic policy uncertainty. *The quarterly journal of economics*, 131(4):1593–1636.

Bates, D. S. (2000). Post-'87 crash fears in the s&p 500 futures option market. *Journal of econometrics*, 94(1-2):181–238.

Bergeron, M., Fung, N., Hull, J., Poulos, Z., and Veneris, A. (2022). Variational autoencoders: A hands-off approach to volatility. *The Journal of Financial Data Science*, 4(2):125–138.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654.

Brenner, M. and Subrahmanyan, M. G. (1988). A simple formula to compute the implied

standard deviation. *Financial Analysts Journal*, 44(5):80–83.

Büchner, M. and Kelly, B. (2022). A factor model for option returns. *Journal of Financial Economics*, 143(3):1140–1161.

Cao, J., Chen, J., and Hull, J. (2020). A neural network approach to understanding implied volatility movements. *Quantitative Finance*, 20(9):1405–1413.

Cont, R. and Da Fonseca, J. (2002). Dynamics of implied volatility surfaces. *Quantitative finance*, 2(1):45.

Dickey, D. A. and Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, 74(366a):427–431.

Duffie, D., Pan, J., and Singleton, K. (2000). Transform analysis and asset pricing for affine jump-diffusions. *Econometrica*, 68(6):1343–1376.

Fung, N. J. C.-k. (2021). *Variational Autoencoders for Volatility Surfaces*. PhD thesis, University of Toronto (Canada).

Gao, S., Zhang, Y., Jia, K., Lu, J., and Zhang, Y. (2015). Single sample face recognition via learning deep supervised autoencoders. *IEEE transactions on information forensics and security*, 10(10):2108–2118.

Gu, S., Kelly, B., and Xiu, D. (2021). Autoencoder asset pricing models. *Journal of Econometrics*, 222(1):429–450.

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The review of financial studies*, 6(2):327–343.

Kelly, B. T., Pruitt, S., and Su, Y. (2019). Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics*, 134(3):501–524.

Kelly, B. T., Pruitt, S., and Su, Y. (2020). Instrumented principal component analysis. *Available at SSRN 2983919*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. Retrieved from https://arxiv.org/abs/1412.6980 on February 11, 2022.

Li, J., Luong, M.-T., and Jurafsky, D. (2015). A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*.

Merton, R. C. (1976). Option pricing when underlying stock returns are discontinuous. *Journal of financial economics*, 3(1-2):125–144.

Mixon, S. (2002). Factors explaining movements in the implied volatility surface. *Journal of Futures Markets: Futures, Options, and Other Derivative Products*, 22(10):915–937.

Pearson, K. (1901). LIII. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Rubinstein, M. (1985). Nonparametric tests of alternative option pricing models using all reported trades and quotes on the 30 most active cboe option classes from august 23, 1976 through august 31, 1978. *The Journal of Finance*, 40(2):455–480.

Sircar, K. R., Papanicolaou, G. C., et al. (1999). Stochastic volatility, smile & asymptotics. *Applied Mathematical Finance*, 6:107–145.

Skiadopoulos, G., Hodges, S., and Clewlow, L. (2000). The dynamics of the s&p 500 implied volatility surface. *Review of derivatives research*, 3(3):263–282.

Appendix A Newton-Raphson method

We initialize the algorithm with an educated guess of the IV. Then the Newton-Raphson method iteratively updates the IV using the partial derivative of the option price with respect to the IV (Vega). Because of the put-call parity, the option price of any put option can be converted to the option price of the call option which forms a pair with the aforementioned put option. After the conversion, the put-call pair should both have the same IV. For convenience, we convert all options to call options before using the Newton-Raphson method such that the Vega is calculated in the same manner for all options. We use a maximum of 100 iterations ($J = 100$) and error margin of 10^{-4} ($\delta = 10^{-4}$) for the Newton-Raphson algorithm.

Algorithm 1 Newton-Raphson method for estimating the IV

Require: Price of call option C, stock price S, strike price K, tenor τ, risk-free rate r, dividend yield rate q, maximum number of iterations J, error margin δ.

1: Initialize $IV_0 = \sqrt{\frac{2\pi}{\tau}}\frac{C}{S}$ (following Brenner and Subrahmanyan (1988)).
2: **for** $j = 0, 1, \ldots, J$ **do**
3: Calculate the implied call price (IC): $IC_j = BS(C, S, K, \tau, r, q, IV_j)$.
4: **if** $|IC_j - C| < \delta$ **then**
5: $IV = IV_j$,
6: STOP.
7: **end if**
8: Update the IV using the Vega ($\mathcal{V}$):

$$d_{1,j} = \frac{\log(S/K) + (r - q + IV_j^2/2)\,\tau}{IV_j\sqrt{\tau}}. \tag{A.1}$$

$$\mathcal{V}_j = e^{-q\tau} S\sqrt{\tau}\,\phi(d_{1,j}), \tag{A.2}$$

where $\phi(\cdot)$ is the probability density function (pdf) of a Standard Normal distribution.

$$IV_{j+1} = IV_j - \frac{IC_j - C}{\mathcal{V}_j}. \tag{A.3}$$

9: **end for**

Appendix B 3D plot full sample

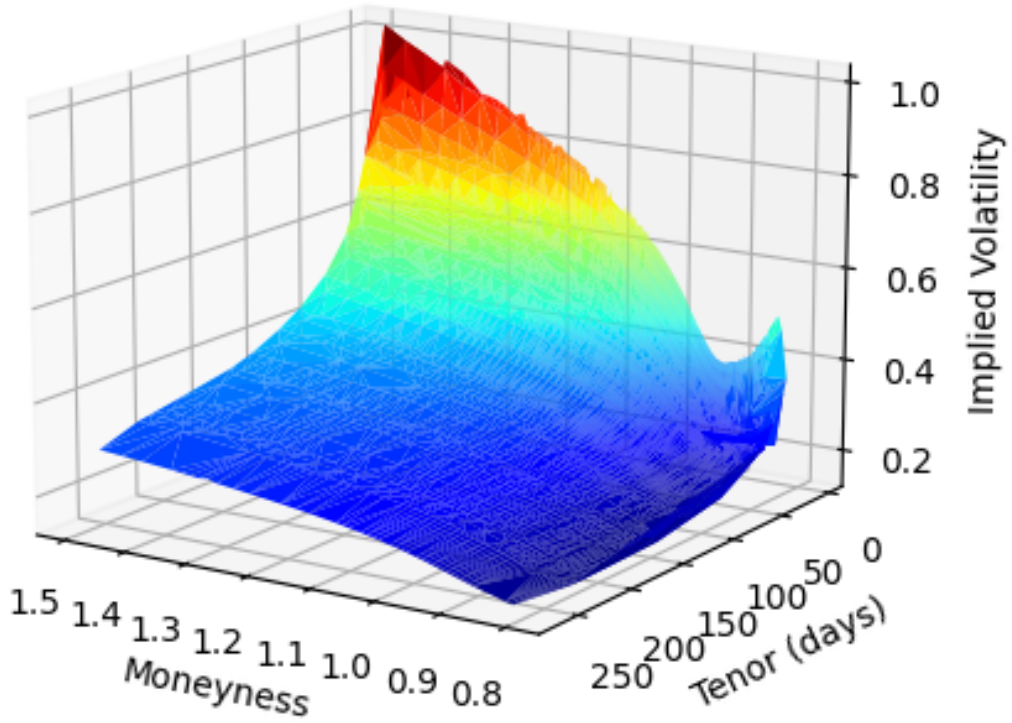
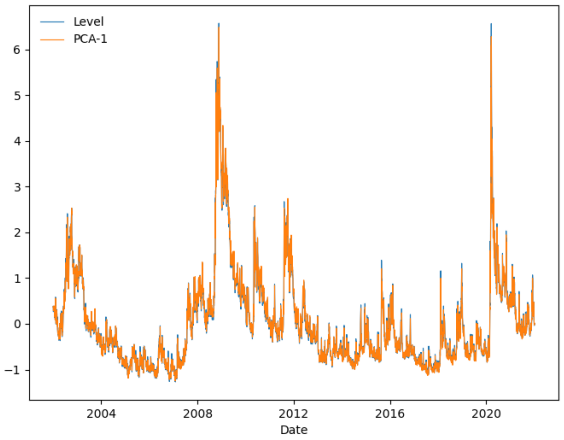
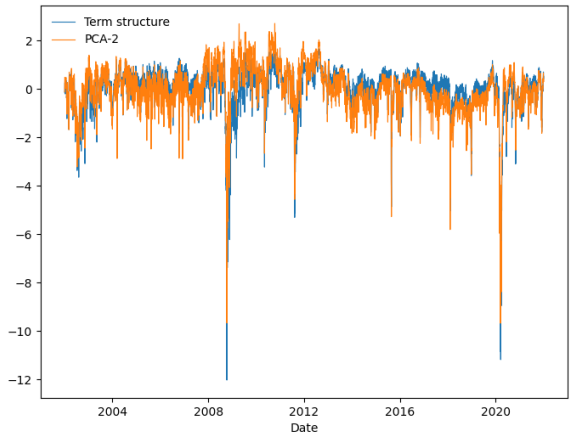


Figure B.1: Plot of the full IVS on February 28, 2020 (unbalanced).

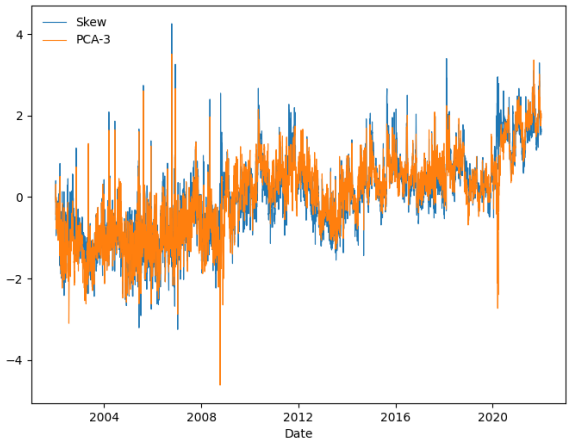
Appendix C Factor plots



(a) PCA-1 and IVS level.

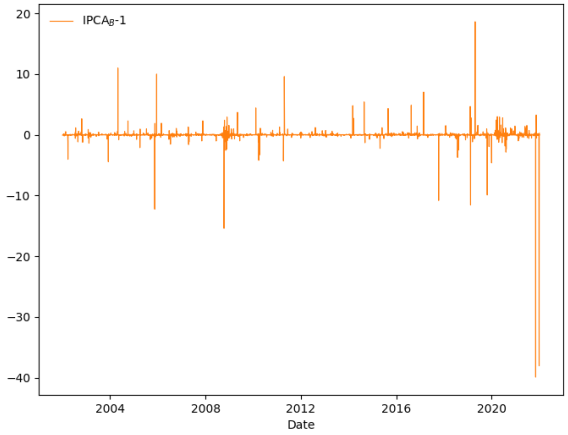


(b) PCA-2 and IVS term structure.

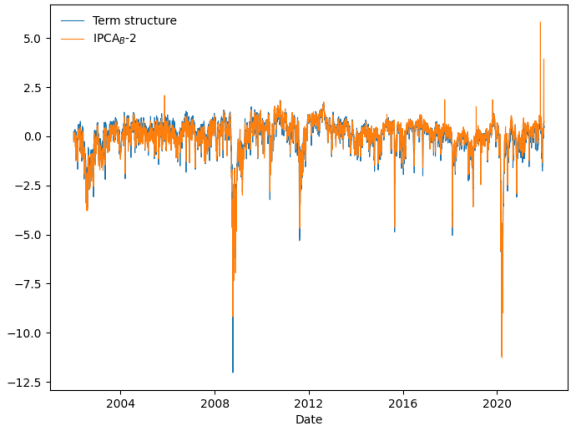


(c) PCA-3 and IVS skew.

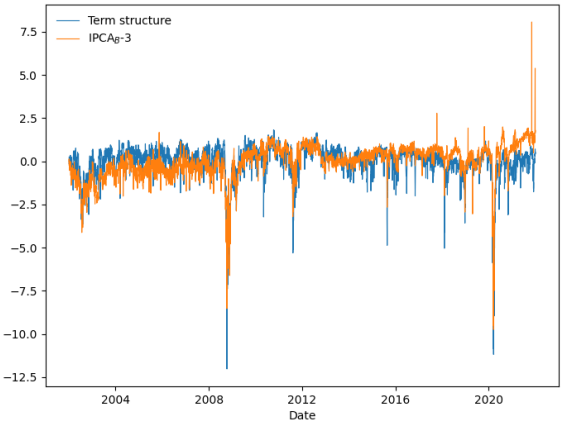
Figure C.1: PCA factors of the 3-factor model with the highest correlated IVS characteristic over the entire sample (both standardized).



(a) IPCA$_B$-1 without IVS characteristic.

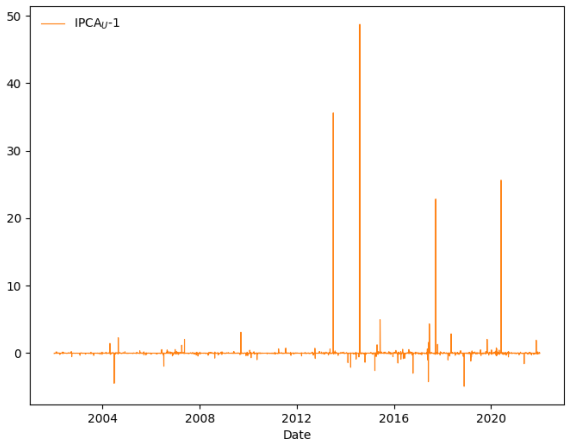


(b) IPCA$_B$-2 and IVS term structure.

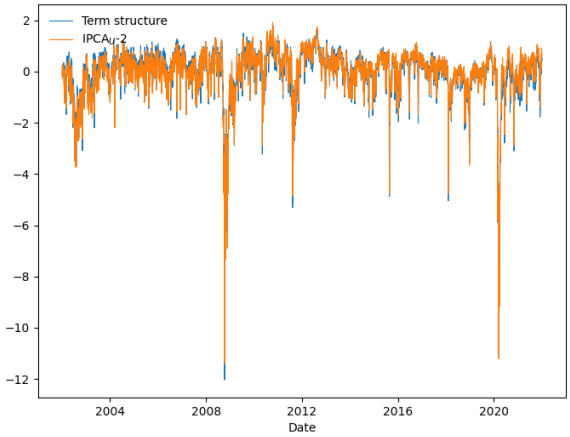


(c) IPCA$_B$-3 and IVS term structure.

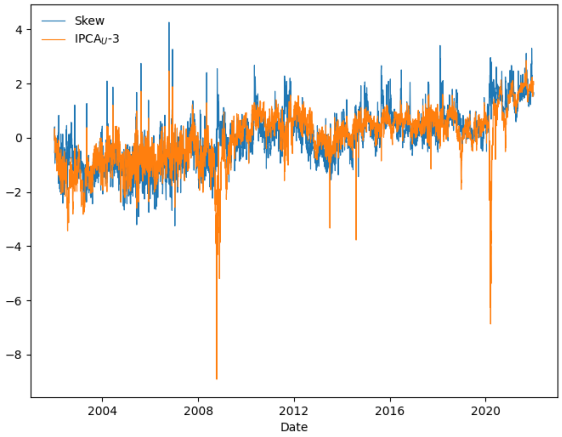
Figure C.2: IPCA$_B$ factors of the 3-factor model with the highest correlated IVS characteristic over the entire sample (both standardized).



(a) IPCA$_U$-1 without IVS characteristic.

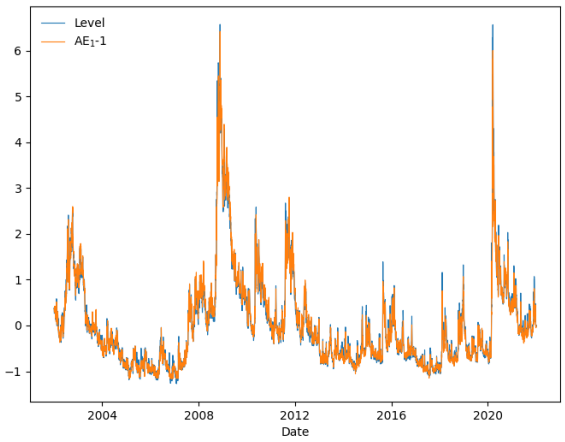


(b) IPCA$_U$-2 and IVS term structure.

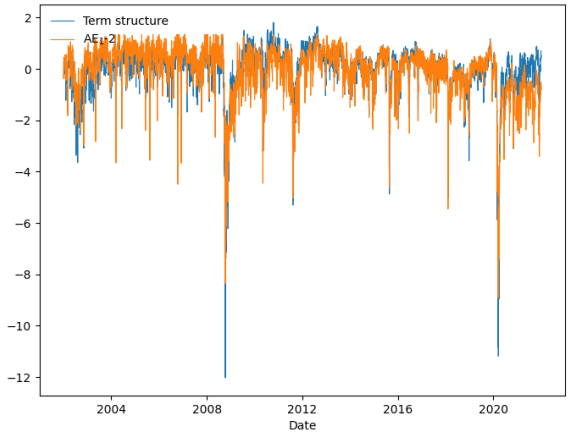


(c) IPCA$_U$-3 and IVS skew.

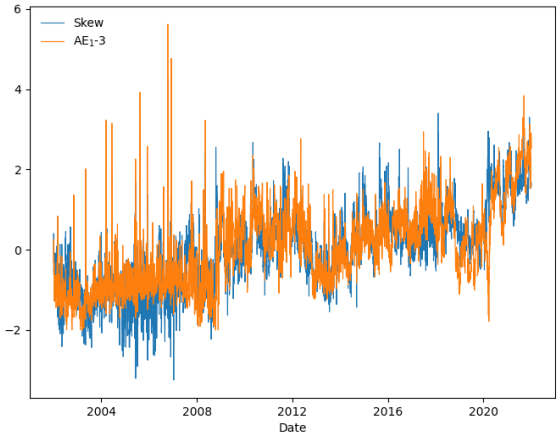
Figure C.3: IPCA$_U$ factors of the 3-factor model with the highest correlated IVS characteristic over the entire sample (both standardized).



(a) AE_1-1 and IVS level.

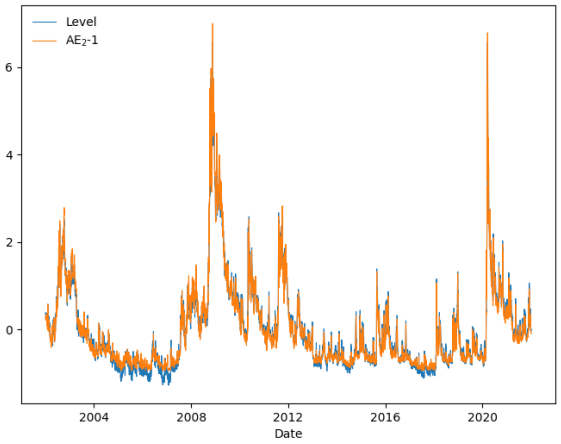


(b) AE_1-2 and IVS term structure.

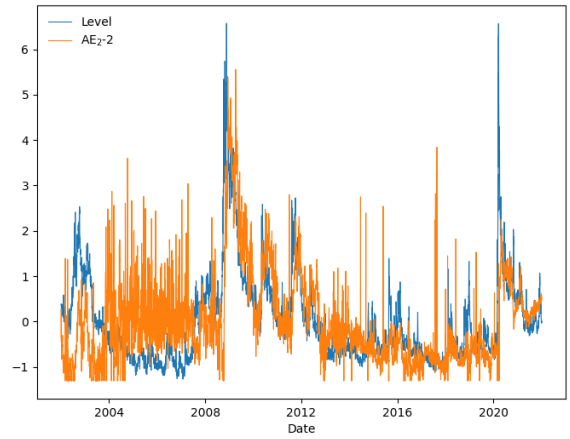


(c) AE_1-3 and IVS skew.

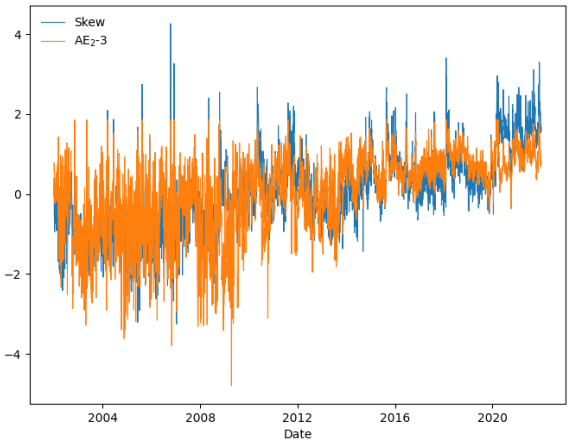
Figure C.4: AE_1 factors of the 3-factor model with the highest correlated IVS characteristic over the entire sample (both standardized).



(a) AE_2-1 and IVS level.



(b) AE_2-2 and IVS level.



(c) AE_2-3 and IVS skew.

Figure C.5: AE_2 factors of the 3-factor model with the highest correlated IVS characteristic over the entire sample (both standardized).

Appendix D Augmented Dickey-Fuller test

Table D.1: ADF test statistics of factors and characteristics of the IVS.

Series	Test statistic	p-value
Level	-4.15	0.001
Term structure	-8.38	0.000
Skew	-3.64	0.005
PCA-1	-4.01	0.001
PCA-2	-7.99	0.000
PCA-3	-3.14	0.023
IPCA_B-1	-70.22	0.000
IPCA_B-2	-7.90	0.000
IPCA_B-3	-6.13	0.000
IPCA_U-1	-70.95	0.000
IPCA_U-2	-8.04	0.000
IPCA_U-3	-5.40	0.000
AE_1-1	-3.87	0.002
AE_1-2	-6.95	0.000
AE_1-3	-3.34	0.013
AE_2-1	-4.12	0.001
AE_2-2	-3.94	0.002
AE_2-3	-5.07	0.000

Appendix E Additional forecasting results

Table E.1: Forecasting *IVRMSE* for the unbalanced test set (%) using a NN to forecast the factors.

Method	Number of factors (M) 1	2	3	4	5	6
Horizon = 1						
PCA	5.69	6.75	5.68	5.14	5.28	5.45
IPCA_B	5.02	6.86	> 99	> 99	> 99	> 99
IPCA_U	5.04	4.36	> 99	> 99	> 99	> 99
$\text{AE}_{1,I}$	7.29	6.42	6.44	11.32	5.18	4.47
$\text{AE}_{2,I}$	8.16	51.90	6.23	4.87	4.66	5.50
Horizon = 5						
PCA	11.98	9.90	14.42	12.32	10.43	11.41
IPCA_B	6.01	6.54	> 99	> 99	> 99	> 99
IPCA_U	8.71	6.55	> 99	> 99	> 99	> 99
$\text{AE}_{1,I}$	16.49	29.36	16.03	16.78	13.94	8.41
$\text{AE}_{2,I}$	8.16	39.62	15.16	16.25	12.73	15.28
Horizon = 21						
PCA	33.67	36.70	35.46	43.17	37.43	24.62
IPCA_B	13.66	12.24	> 99	> 99	> 99	> 99
IPCA_U	32.20	12.75	> 99	> 99	> 99	> 99
$\text{AE}_{1,I}$	33.61	43.64	30.79	58.45	45.67	40.61
$\text{AE}_{2,I}$	8.17	53.52	36.02	65.63	35.44	54.96

Notes: The NN used for the indirect forecast has 2 hidden layers with a width of 64 and ReLu activation functions on all layers but the output layer, which has a linear activation function.