A Variational Autoencoder Approach to Conditional Generation of Possible Future Volatility Surfaces

Jacky Chen, John Hull, Zissis Poulos, Haris Rasul, Andreas Veneris, Yuntao Wu

January 18, 2025

Jacky Chen (jackyjc.chen@rotman.utoronto.ca) is an adjunct professor at the Joseph L. Rotman School of Management, University of Toronto, 105 St. George Street, Toronto, ON, Canada M5S3E6. Tel: 416 543 5375

John Hull (john.hull@rotman.utoronto.ca) is a professor at the Joseph L. Rotman School of Management, University of Toronto, 105 St. George Street, Toronto, ON, Canada M5S3E6. Tel: 416 978 8615

Zissis Poulos (zpoulos@yorku.ca) is an assistant professor at the School of Information Technology, York University, 1 University Blvd., Markham, ON, Canada L6G0H2. Tel: 416 978 5703

Haris Rasul (haris.rasul@mail.utoronto.ca) is an undegraduate research assistant at Rotman School of Management, FinHub Financial Innovation Lab, University of Toronto. 105 St. George Street, Toronto, ON, Canada M5S3E6. Tel: 647 294 1422

Andreas Veneris (veneris@eecg.utoronto.ca) is a professor at the Edward S. Rogers Sr. Department of Electrical & Computer Engineering, cross-appointed with the Department of Computer Science at the University of Toronto, 10 Kings College Road, Toronto, ON, Canada M5S3G8. Tel: 416 946 3062

Yuntao Wu (Corresponding Author) (winstonyt.wu@mail.utoronto.ca) is a PhD student at the Edward S. Rogers Sr. Department of Electrical & Computer Engineering, University of Toronto, and a research assistant at FinHub Financial Innovation Lab, 10 Kings College Road, Toronto, ON, Canada M5S3G8. Tel: 604 561 6665

A Variational Autoencoder Approach to Conditional Generation of Possible Future Volatility Surfaces

January 18, 2025

Abstract

We develop a novel method to generate future implied volatility surfaces using historical sequences of surfaces and conditioning variables such as historical returns. The proposed architecture, based on a conditional variational autoencoder (CVAE) and a long short-term memory network (LSTM), encodes historical data and represents sequences of observations. This architecture can generate a wide spectrum of future surfaces conditional on any historical data set. Applied to S&P500 data, the model predicts future return polarity with 60% accuracy.

THREE KEY TAKEAWAYS:

1. We propose an architecture combining Conditional Variational Autoencoders (CVAE) and Long Short-Term Memory (LSTM) networks to generate possible future implied volatility surfaces using historical data.

2. The proposed model achieves approximately 60% accuracy in forecasting the polarity of S&P500 returns over a five-day horizon, demonstrating practical relevance in financial forecasting.

3. The proposed model effectively condenses the principal components of implied volatility surfaces while preserving critical information, ensuring robust predictive performance and applicability in real-world financial analytics.

Keywords: Implied volatility surfaces; variational autoencoders; long short-term memory; deep learning; simulation modeling

JEL Classification: C45, G10

If the assumptions underlying the model developed by Black and Scholes (1973) and Merton (1973) held, a single volatility could be used to price all options dependent on a particular asset and that volatility would not change through time. In reality, the volatility that must be used in conjunction with the Black-Scholes-Merton model to price an option on an asset at any given time depends on the option's strike price and time to maturity. The volatility as a function of these parameters is referred to as the implied volatility surface (IVS). The IVS changes through time. A key question for a trader responsible for options dependent on a particular asset is therefore: "By how much can the IVS change?" Every possible change gives rise to a gain or loss on the trader's portfolio. The trader must be satisfied that, with the hedges contemplated, the risks are acceptable.

There have been many attempts to model the dynamics of IVSs. Examples are Cont and Da Fonseca (2002), Cont, Da Fonseca, and Durrleman (2002), Carmona, Ma, and Nadtochiy (2017), Cuchiero, Khosrawi, and Teichmann (2020), Cohen, Reisinger, and Wang (2020), Bloch and Böök (2021), Shang and Kearney (2022), Cont and Vuletić (2023), Francois et al. (2022), and Choudhary, Jaimungal, and Bergeron (2023). Some of these jointly model the asset price and the volatility surface while others model only the volatility surface.

In this research, we use a variational autoencoder (VAE) to model a wide range of potential scenarios for next-day IVS. This is, as far as we know, a different approach from that taken by other researchers. A VAE is an attractive tool because it requires no assumptions about the nature of the model. It uses historical data to produce a multivariate normal distribution for a number of latent variables that relate the next-day IVS to recent asset price returns and recent IVSs. By sampling from the latent variables alternative future scenarios for the IVS are generated. These sampled scenarios collectively form a distribution, representing a wide spectrum of potential future outcomes. We apply the methodology to options on the S&P500.

We define the Implied Volatility Surface (IVS) using 25 points, combining five levels of moneyness with five different times to maturity. According to the assumptions made by Variational Autoencoder (VAE) models, the actual IVS should be a random sample from the IVSs generated by the VAE. We test this hypothesis by comparing the original IVS on a given date with the IVS generated by the VAE model. The model proposed in this paper demonstrates both viability and explainability in modeling potential next-day volatility surfaces based on historical data. By examining the generated surfaces with maximum likelihood, we find that they accurately capture the original surfaces at most grid points, with higher than 90% R^2. Furthermore, the proposed model architecture effectively distinguishes the evolution of IVSs and asset prices during recession and non-recession periods. Quantitative analysis reveals that the principal components of the surfaces produced by our model possess predictive power regarding SPX price movements and volatility. Our generated surfaces achieve an approximately 60% accuracy in forecasting whether the SPX price will rise or fall over the next five days.

METHODOLOGY

Variational Auto-Encoders (VAEs)

The basic VAE architecture proposed by Kingma and Welling (2013) is an unsupervised generative model consisting of two parts: an encoder and a decoder.

The encoder infers from observed data, x, the latent variables, z, in the form of a probability distribution $q_\phi(z|x)$ with parameters, ϕ. We choose $q_\phi(z|x)$ to match a multivariate normal dis-

tribution, $p(z) = \mathcal{N}(0, I)$, where I is the identity matrix. We can ensure the matching of the two probability distributions by minimizing the Kullback-Leibler (KL) divergence:

$$\min_{\phi} D_{KL}(q_\phi(z|x)||p(z)). \tag{1}$$

The decoder takes the latent variables, z, from the encoder to reproduce the observed data, x. Thus, a suitable objective function for the decoder part is to maximize the marginal log-likelihood of the observed data, x, in expectation over the distribution of the latent variables:

$$\max_{\theta} \mathbb{E}_{q_\phi(z|x)}(\log p_\theta(x|z)), \tag{2}$$

where θ denotes the parameters of the decoder.

The objective function of a VAE model is therefore:

$$\max_{\theta,\phi} \mathbb{E}_{q_\phi(z|x)}(\log p_\theta(x|z)) - D_{KL}(q_\phi(z|x)||p(z)). \tag{3}$$

Higgins et al. (2017) proposed a regularization factor on the KL divergence:

$$\max_{\theta,\phi} \mathbb{E}_{q_\phi(z|x)}(\log p_\theta(x|z)) - \beta D_{KL}(q_\phi(z|x)||p(z)). \tag{4}$$

Changing β changes the focus of training. A small value for β means that we focus on the accuracy of matching the generated data with observed data. A large β means that we focus more on enforcing structure in the space of hidden variables. This causes the decoder to smoothly interpolate between samples of hidden variables to obtain synthetic data that do not have extreme and unrealistic differences compared to observed data. In practice, an appropriate value of β can be found with hyperparameter tuning.

Typically, the decoder produces the mean value $\mu_{x|z}$, and we assume a unitary covariance. Therefore $\log p_\theta(x|z) = -\frac{1}{2}\|x - \mu_{x|z}\|_2^2$. Finally, the expectation can be approximated by averaging, so the maximum likelihood can be replaced by minimizing a mean squared error, which is the reconstruction loss.

Since we assume the prior $p(z) = \mathcal{N}(0, I)$, the KL divergence can also be reduced to an exact expression:

$$D_{KL}(q_\phi(z|x)||p(z)) = \frac{1}{2}\sum_i \left(-1 - \log\sigma_i^2 + \sigma_i^2 + \mu_i^2\right), \tag{5}$$

where $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ for each individual latent variable z_i in the vector z. The objective function can then be rewritten as:

$$\begin{aligned} \max\ & -\frac{1}{2}\|x - \mu_{x|z}\|_2^2 - \frac{\beta}{2}\sum_i \left(-1 - \log\sigma_i^2 + \sigma_i^2 + \mu_i^2\right), \text{ or} \\ \min\ & \frac{1}{2}\|x - \mu_{x|z}\|_2^2 - \frac{\beta}{2}\sum_i \left(1 + \log\sigma_i^2 - \sigma_i^2 - \mu_i^2\right). \end{aligned} \tag{6}$$

The basic VAE architecture generates new data samples unconditionally. Sohn, Lee, and Yan (2015) introduced a Conditional VAE (CVAE), which generates data based on a given condition c. In this setting, the encoder maps inputs to latent factors according to the distribution $q_\phi(z|x,c)$

using both the observed data x and the condition c. The decoder then generates x by $p_\theta(x|z,c)$, providing control over the generated output. For instance, Ning et al. (2023) use CVAE to generate parameters for Stochastic Differential Equations (SDEs) that characterize the behavior of IVSs conditional on observable market states, such as spot rates or indices.

VAEs have also been applied to the task of timeseries generation (see Desai et al. (2021)). In this application, one-dimensional convolutional layers are employed by the encoder to extract time-dependent features and construct the latent space, while transposed convolutional layers are used to reconstruct the timeseries. Trend blocks and seasonality blocks can be incorporated to improve the interpretability of such models. Beyond VAEs, other generative models like Generative Adversarial Networks (GANs) have also been applied to generate price scenarios with a particular focus on tail risk (see Cont et al. (2023)).

Long Short-Term Memory (LSTM)

The asset price and IVS changes we work with can be viewed as timeseries data, where today's values can depend on those of the previous day. Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) networks as originally proposed by Hochreiter and Schmidhuber (1997), effectively incorporate this time evolution and have been shown to be useful in timeseries generation (Siami-Namini, Tavakoli, and Siami Namin (2018)). Within each recurrent unit of the LSTM, a long-term cell state c_t and a short-term hidden state h_t are maintained through self-recurrence by four interacting neural network layers:

1) **Forget Gate Layer**: Compares h_{t-1} and the current input x_t, and decides which elements in cell state c_{t-1} to retain and which to forget (f_t).

2) **Input Gate Layer**: Determines which cell units to update (i_t) and creates new candidate values $\tilde{c}_t$.

3) **Update Layer**: Updates the cell state c_t using f_t, i_t and $\tilde{c}_t$.

4) **Output Layer**: Calculates o_t based on x_t and h_{t-1}, and updates the short-term memory h_t.

Let $\sigma(\cdot)$ represent the sigmoid function. The LSTM model can be represented by the following equations:

$$\begin{aligned}
\text{Forget: } f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \\
\text{Input: } i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \\
\tilde{c}_t &= \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \\
\text{Update: } c_t &= f_t \times c_{t-1} + i_t \times \tilde{c}_t, \\
\text{Output: } o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \\
h_t &= o_t \times \tanh(c_t),
\end{aligned} \tag{7}$$

where $\times$ denotes the Hadamard product (element-wise product), and $W \cdot [h,x] = W_h h + W_x x$ represents the tensor product of the weights with the concatenated vector $[h,x]$. W_x is the weight matrix associated with the current inputs x_t informing the LSTM computation for future possible outcomes. W_h is the weight matrix associated with past information, generating future outcomes. These weights are learned and optimized during the training process to minimize the loss function.

There are also several variants of LSTM, though Greff et al. (2017) shows that these variants perform similarly. VAE and LSTM, or other memory models, have been combined to perform tasks in natural language generation (Bowman et al. (2016)) and anomaly detection in timeseries (Lin et al. (2020)).

Proposed Model

Let x_t represent the observed IVS on day t, constructed as outlined in the next Section. To model price changes, we use the log-returns $r_t \equiv \text{Ret}_t = \log\left(\frac{p_t}{p_{t-1}}\right)$, where p_t denotes the asset price on day t. Let y_t be conditioning variables that can impact IVS dynamics, which include underlying asset returns r_t. In the subsequent text, the subscript t may be omitted referring to general operations that do not depend on specific days.

To use a VAE to model changes in asset prices and IVSs, we consider the problem: Given the context information, $\mathbf{x}_c = (...,x_{t-3},x_{t-2},x_{t-1})$ and $\mathbf{y}_c = (...,y_{t-3},y_{t-2},y_{t-1})$, representing the historical timeseries of observed IVSs and conditioning variables before day t respectively, can we generate distributions of possible IVSs and asset returns for day t and beyond, *i.e.* $\mathbf{x}_n = (x_t,x_{t+1},x_{t+2},...)$ and $\mathbf{r}_n = (r_t,r_{t+1},r_{t+2},...)$?

The basic VAE architecture learns an unconditional probability distribution derived from the complete set of observations. This architecture is limited to generating random samples without any conditions, making it challenging to control the generated data. The more frequently observed data points are more likely to be generated. However, it is impossible to determine the specific day for which the IVSs are generated. CVAEs address this by setting the historical timeseries as conditions. However, without controlling time steps, a look-ahead bias may occur when generating new IVSs. For example, a surface generated on day t could also use the latent values for day $t+1$, $t+2$ and beyond in the standard CVAE formulation. Additionally, as the length of historical context and the length of future values to generate increase, more IVSs and conditioning variables need to be encoded. This increases the number of parameters, as there are no recurrent units in the CVAE, and inputs of each day require different parameters for encoding. Finally, CVAEs using simple multilayer perceptron (MLP) or convolutional neural network (CNN) can only accept fixed-size input, making it challenging to generate surfaces based on variable-length contexts without disrupting the structures.

To address these challenges, we propose a new architecture that combines the strengths of CVAEs and LSTM, making it suited for our task. A high-level overview of the model is provided below, while detailed mathematical formulations can be found in appendix. The architecture uses a CVAE as its backbone and consists of three main components:

1) **Encoder**: The encoder processes the historical context and the values to be generated ($\mathbf{x}_c,\mathbf{y}_c$, $\mathbf{x}_n,\mathbf{y}_n$), forming a distribution $\mathcal{N}(\mu_{t,i},\sigma^2_{t,i})$ for each latent variable on each day. The latent representation $\mathbf{z} = (...,z_{t-1},z_t,z_{t+1},...)$ is then sampled from this distribution. During generation steps, the latent variables for the historical context are computed as $\mu_{t,i}$ by the encoder, and the latent variables for the days to generate are individually sampled from the standard normal $\mathcal{N}(0,I)$.

2) **Context Encoder**: The context encoder generates an efficient representation of the historical context $(\mathbf{x}_c,\mathbf{y}_c)$, denoted $\zeta = (...,\zeta_{t-3},\zeta_{t-2},\zeta_{t-1})$;

3) **Decoder**: The decoder uses ζ as a condition and the sampled latent variables $\mathbf{z}$ to reconstruct the surfaces and returns, $\hat{\mathbf{x}}_n$, $\hat{\mathbf{r}}_n$.

The LSTM is integral to each of these three components. With a limited number of parameters and no look-ahead bias, it efficiently captures the time dependencies of the inputs when generating embeddings in the encoder and context encoder. It also generates outputs based on previous day conditions in the decoder, accommodating inputs and outputs of any length. Furthermore, incorporating LSTM allows us to separate the tasks of capturing temporal and spatial features.

We define the objective function for each day t as follows:

$$L(\phi,\theta,x_t,y_t) = \frac{1}{HW}\|x_t-\hat{x}_t\|_2^2 + \alpha\|r_t-\hat{r}_t\|_2^2 - \frac{\beta}{2}\sum_{i=1}^{L}\left(1+\log\sigma_{t,i}^2-\sigma_{t,i}^2-\mu_{t,i}^2\right), \tag{8}$$

where H and W are the dimensions of the IVSs, and L is number of latent variables. The term $\frac{1}{HW}\|x_t-\hat{x}_t\|_2^2$ represents the mean squared error (MSE) of points on the generated surface $\hat{x}_t$ and $\|r_t-\hat{r}_t\|_2^2$ is the MSE of the generated underlying asset returns $\hat{r}_t$[1]. α is the regularization factor for the error on returns. It is included in our loss function to account for underlying asset dynamics, aiming to improve the generation of IVSs that align with observed market trends. Increasing α reduces the reconstruction loss on conditioning variables (specifically returns), but may slightly increase the loss on IVSs. However, an optimal α could potentially minimize the overall loss. The choice of α depends on the relative scale of returns and IVSs. β is the regularization factor for the KL divergence, as detailed in methodology section. This objective function is computed for each generated day within a batch and averaged across all days in the batch.

The proposed architecture can be extended to generate implied volatilities of options whose moneyness and time to maturity do not match the fixed grid points that are used for training. The extension involves using a point-wise decoder and two additional conditioning variables (*i.e.*, moneyness and time to maturity). The point-wise decoder takes as input ζ, the sampled latent variables $\mathbf{z}$, the desired moneyness and time to maturity values, and generates a single implied volatility corresponding to the latter, instead of generating the entire grid. The point-wise decoder modification was proposed by Bergeron et al. (2022) in the context of time-agnostic IVS generation. Employing a point-wise decoder eliminates the need to perform interpolation after generating implied volatilities. An additional benefit of the VAE architecture is that the latent variables can be calibrated using observed market data. Specifically, when there are many observed implied volatilities available on a particular day, the encoder can be used to infer the latent variables that best explain the observed data. When data is sparse, the decoder can be used in isolation to infer the latent variables by minimizing the reconstruction error with a quasi-Newton optimizer, such as Limited-memory BFGS. The efficacy of this calibration strategy is demonstrated by Bergeron et al. (2022).

S&P500 IMPLIED VOLATILITY SURFACE ESTIMATION

S&P500 Call Option data from January 2000 to February 2023 were downloaded from OptionMetrics[2]. Initially, all options without a reported implied volatility were excluded from the dataset. Additionally, only options with a positive open interest, moneyness (defined as the strike

1. We also experimented with generating the complete conditioning variable vector y_t and computing the loss as $\frac{1}{E}\|y_t-\hat{y}_t\|_2^2$, but it introduced bias towards specific surface properties and did not improve performance.
2. The access is gained through https://wrds-www.wharton.upenn.edu/pages/about/data-vendors/optionmetrics/

price divided by the index level, K/S) ranging between 0.7 and 1.3, and a time to maturity from 1 month up to 2 years were retained. We use a 5×5 grid for the implied volatility surface, defined by time-to-maturity (ttm) values of 1, 3, 6, 12, and 24 months and moneyness values of 0.7, 0.85, 1, 1.15, and 1.3. It is worth noting that this 5×5 implied volatility surface is not directly observable from the available options on any given trading day due to variations in available strike prices and expiration dates. Therefore, we employ the bivariate linear interpolation method outlined in Cao et al. (2022) to determine the implied volatilities for standard reference points. For each option with moneyness $M = K/S$ and ttm T, we define $M_l, M_u \in \{0.7, 0.85, 1, 1.15, 1.3\}$ as the standard moneyness values closest to M such that $M_l \leq M \leq M_u$, and define $T_l, T_u \in \{1, 3, 6, 12, 24\}$ as the standard ttm values closest to T such that $T_l \leq T \leq T_u$. Then, the implied volatility for the (M, T) option is determined by a bivariate linear interpolation between the implied volatilities of the $(M_l, T_l), (M_l, T_u), (M_u, T_l), (M_u, T_u)$ options. The best-fit implied volatilites for standard moneyness and ttm are those that minimize the squared differences between the interpolated and reported implied volatilities. Occasionally, fitting for extreme moneyness/maturity combinations (*e.g.* one-month implied volatility for moneyness levels of 0.7, 0.85, 1.15 and 1.3, and three-month implied volatility for moneyness levels of 0.7 and 1.3) fails due to unreliable or nonexistent S&P500 implied volatilities. This results in either zero or extremely high implied volatility. For interpolated implied volatility values ≤ 0.01 or ≥ 1.0, we find the option on the same day with the closest time to maturity and moneyness, and use its implied volatility as the estimate.

We use $x[\text{ttm}, \text{Moneyness}]$ to index a single point on the surface, with ttm in the order of years. For example, $x[0.25, 1]$ means the implied volatility when ttm= 3 months and Moneyness $\frac{K}{S} = 1$.

RESULTS

Conditioning Variables Definition

One of the conditioning variables we use is the daily log return of the asset price. Additionally, we define two other conditioning variables: skew and slope. The skew is defined as:

$$\text{skew} = \frac{x[1, 0.85] + x[1, 1.15]}{2} - x[1, 1]. \tag{9}$$

The slope is defined as:

$$\text{slope} = x[2, 1] - x[0.25, 1]. \tag{10}$$

These features are utilized in both training and evaluation phases. Skew measures the asymmetry of the implied volatility around the at-the-money point, while slope measures the at-the-money volatility term structure. Both indicators may reflect potential upward or downward trends in IVs.

Training and Model Performance

We use data from 2000-01-03 to 2015-11-27 (4000 days) for training, data from 2015-11-30 to 2019-11-18 (1000 days) for validation, and data from 2019-11-19 to 2023-02-24 (822 days) for testing. The model is defined and trained using PyTorch 2.3.1[3] on an RTX 3080 Linux machine[4]. It uses 3 convolution layers with an output size of 5 for surface encoding and identity encoding for the

3. The code is compatible with any PyTorch version, but performance may vary.
4. https://www.runpod.io/console/pods

conditioning variables. The latent and context embedding dimensions are set to 5. We use a 2-layer LSTM with 100 hidden units as memory and apply a 0.2 dropout rate. The KL divergence weight is set to $\beta = 10^{-5}$. Training, validation, and testing were conducted on sequences of variable lengths between 4 and 10, with the context length being one less than the sequence length. The validation set is used to select the optimal model during 500 epochs of training. The batch size is 64 for training and 16 for validation and testing. AdamW (Kingma and Ba (2017), Loshchilov and Hutter (2019)) is used as the optimizer with a learning rate of 10^{-5}. All random seeds are reset to 0 before dataset preparation and before training. We use 64-bit floating points in calculations for better precision. A variety of feature configurations were tested, and comparisons of two representative models, along with relevant evaluation metrics, are reported in Exhibit 1.

In the tables and subsequent sections, we refer to the model trained only using IVSs without any conditioning variables as "No EX". The model trained with conditioning variables, returns, skew, and slope, where the loss is computed on IVSs and returns only, is referred to as "EX Loss".

Most of the experiments are performed on the "MLE surfaces", denoted $\hat{x}_{MLE,t}$, with a 5-day (trading week) context $(\mathbf{x}_c, \mathbf{y}_c) = (x_{t-5}, y_{t-5}, ..., x_{t-1}, y_{t-1})$. To generate MLE surfaces, we first compute the latent variable z. For the context, we compute $z_{t-5}, z_{t-4}, ..., z_{t-1}$ as the mean values $\mu_{t-5}, \mu_{t-4}, ..., \mu_{t-1}$ of context $(\mathbf{x}_c, \mathbf{y}_c)$ encoded by the encoder. For the day to generate, we use $z_t = 0$ as the maximum likelihood in the prior distribution. We also embed the context $(\mathbf{x}_c, \mathbf{y}_c)$ into ζ using the context encoder. Then we generate the MLE surfaces using the decoder with $z = [z_{t-5}, ..., z_{t-1}, z_t]$ and ζ. To compute a wide range of possible scenarios, such as in the case of arbitrage testing in appendix, we set $z_t \sim \mathcal{N}(0, I)$ to be a random sample from the prior distribution, and then generate the surfaces. In both cases, the dropout rate of the LSTM layers is set to 0. The generation starts from 2000-01-10 and ends on 2023-02-15, covering a total of 5810 days.

Panel A of Exhibit 1 reports the training statistics of the two representative models on the validation and test sets. Panels B and C report the RMSE between the MLE surfaces and the observed surfaces across all 5810 days. Grid points with high RMSEs typically correspond to those with unreliable or nonexistent S&P500 implied volatilities, requiring approximation with the nearest available values. For better-behaved grids, the RMSEs are generally around 1% to 2%. Additionally, the EX Loss model exhibits generally lower RMSEs compared to the No EX model, indicating that the conditioning variables improve the model's performance.

To further evaluate the goodness of fit of the MLE surfaces, we perform the following regression on all 25 grid points:

$$x_t\,[\text{ttm}, \text{moneyness}] = \alpha + \beta_1 \hat{x}_{MLE,t}\,[\text{ttm}, \text{moneyness}] + \varepsilon_t. \tag{11}$$

The MLE surfaces, $\hat{x}_{MLE,t}$, should represent the actual level of each individual grid point on the surface. This accuracy is achieved during the training process, where two key objectives are minimized: the mean squared error between the generated surfaces and the observed surfaces, and the distance between the distribution of latent variables and the standard normal distribution. This dual minimization ensures that the model's most probable generations closely align with the actual data. Exhibit 2 reports the regression results for the No EX model, whereas Exhibit 3 reports the results for the EX Loss model. The coefficients β_1 are around 1 for most of the surface grid points, with intercepts around 0 and R^2 values around 90%, suggesting that the observed next-day surfaces are well approximated by the MLE surfaces. Comparing Exhibit 2 and Exhibit 3, we see that the EX Loss model produces regression coefficients closer to 1, intercepts closer to 0, and higher R^2 values. This indicates that the MLE surfaces generated by the EX Loss model align more closely with

the observed surfaces, consistent with the observation that the EX Loss model produces better-fit surfaces based on RMSE analysis.

Forecasting Recession and Volatility Dynamics

First, we examine the context encoding process of observed S&P500 surface data. We use a 5-day context (x_{t-5}, y_{t-5}), (x_{t-4}, y_{t-4}), ..., (x_{t-1}, y_{t-1}) to generate the 5×5 encoding $(\zeta_{t-5}, \zeta_{t-4}, \zeta_{t-3}, \zeta_{t-2}, \zeta_{t-1})$. These ζ values are then concatenated into a one-dimensional vector $\chi = [\zeta_{t-5}, \zeta_{t-4}, \zeta_{t-3}, \zeta_{t-2}, \zeta_{t-1}] \in \mathbb{R}^{25}$, where the first 5 values are from ζ_{t-5}, and the last 5 values are from ζ_{t-1}. This χ value is calculated for 5810 days from January 10, 2000, to February 15, 2023. We then apply PCA on $[\chi_1, \chi_2, ..., \chi_{5810}]$ to extract the first and second principal components (FPC and SPC) for each day. For the No EX model, the FPC explains 84.5% of the variance, while the SPC accounts for 6%. In the EX Loss model, FPC explains 80% of the variance, and SPC explains 7%.

In the first qualitative analysis, we categorize the days according to whether they were in an NBER recession period and then plot the principal components. The NBER recession periods within our analysis are April 1, 2001, to November 30, 2001; January 1, 2008, to June 30, 2009; and March 1, 2020, to April 30, 2020[5]. Exhibit 8 shows the principal components extracted from both the original surfaces and the generated MLE surfaces. Panel A presents the results for the No EX model, while Panel B shows the results for the EX Loss model. The majority of the stable market dates, represented by dots, cluster near the origin in the bottom left corner. In contrast, the principal components for the NBER recession dates, indicated by triangles, are scattered towards the top right corner. This noticeable separation indicates the model's ability to capture distinct volatility characteristics within varying market regimes and periods of recession. Also, this behavior is consistent in both the No EX and EX Loss models.

To provide a quantitative estimate of the model's performance in predicting future recession periods, we use a probit regression model. This model predicts whether a period falls within an NBER recession:

$$P(NBER_{t:t+30}) = \Phi(\alpha + \beta_0 \text{FPC}_t + \beta_1 \text{SPC}_t + \varepsilon_t). \tag{12}$$

We define $NBER_{t:t+30}$ as an indicator variable that is 1 if any of day t to day $t+30$ is in an NBER recession period, and 0 otherwise. We use the FPC_t and SPC_t values extracted from encoded IVSs on days $t-5, ..., t-1$ as the predictor variables. Φ is the standard normal cumulative distribution function. Columns (1) and (2) in Panels A and B of Exhibit 4 report the estimated coefficients and standard errors for the probit regression model. Column (1) shows the results for the encoding of the original surfaces, while Column (2) presents the results for the encoding of the generated MLE surfaces. The estimated coefficients and R^2 values are close, with similar significance levels, indicating that the generated MLE surfaces effectively capture the important features of the original surfaces. Comparing Panels A and B, we observe that the MLE surfaces generated by the EX Loss models provide a higher R^2 (36% compared to 34%) and a more significant second principal component. Panels A and B in Exhibit 9 plot the estimated probability of the probit regression model. The red and blue plots closely align, validating the model's ability to accurately extract and reflect important features from the original data. The SPX price is also plotted in the graph. Since the IVSs are based on options for S&P500 indices, the estimated probability of recession is

5. Monthly periods from FRED (2023): https://fred.stlouisfed.org/series/USREC

not only high during actual NBER recessions but also high when SPX prices drop. This leads to our next experiments on future return volatility prediction and classification.

We define log return $\text{Ret}_t = \log\left(\frac{p_t}{p_{t-1}}\right)$ as before, and define Vol5d_t as the standard deviation of 5-day returns Ret_t, Ret_{t+1},..., and Ret_{t+4}, which captures the volatility of 5-day returns. We use the FPC_t and SPC_t values extracted from encoded IVSs on days $t-5,\ldots,t-1$ as the predictor variables to estimate the future return volatility Vol5d_t using linear regression:

$$\text{Vol5d}_t = \alpha + \beta_0 \text{FPC}_t + \beta_1 \text{SPC}_t + \varepsilon_t. \tag{13}$$

The results are reported in Columns (3) and (4) in Panels A and B of Exhibit 4. Both the principal components extracted from encoded original and generated MLE surfaces predict Vol5d_t with high statistical significance. The R^2 values are around 40%. The EX Loss model provides a slightly higher R^2 in extracting the features from the original surfaces (R^2 increases from 43% to 46%), but the difference for the generated MLE surfaces is not that significant.

Autoencoders and VAEs typically perform lossy compression when the latent dimension is smaller than the original input data dimension. In our case, the latent dimension is 5, while the input data dimension is 25, indicating potential information loss. To examine the extent of information loss and potential additional information captured by our VAE model, we conduct the same experiment on the original IVSs and evaluate the incremental effect of encoding. PCA is applied to the original IVSs from day $t-5,\ldots,t-1$, encompassing a total of 125 points. The first principal component (PC) explains 54.3% of the variance, while the second PC explains 8.6%. A total of 10 PCs are required to explain 86% of the variance. The VAE effectively encodes most of the variance into the first two PCs. We then perform return volatility forecasting using the following configuration:

$$\text{Vol5d}_t = \alpha + \sum_{i=1}^{k} \gamma_i \text{PC}_{i,t} + \beta_0 \text{FPC}_t + \beta_1 \text{SPC}_t + \varepsilon_t. \tag{14}$$

The results are summarized in Exhibit 5. Columns (1), (3), and (5) report the incremental effects of FPCs and SPCs extracted from the encoded IVSs in regressions that include the top two PCs ($\text{PC}_{1,t}$ and $\text{PC}_{2,t}$) derived from the original IVSs. The FPCs and SPCs from the encoded IVSs, generated by both the No EX and EX Loss models, remain significant in these regressions and enhance R^2 by 3.2% and 6.1%, respectively. Similarly, the FPCs and SPCs derived from the encodings of generated MLE surfaces yield comparable improvements in R^2, suggesting minimal information loss during the generation process. Columns (2), (4), and (6) show the incremental effects of FPCs and SPCs in regressions that incorporate the top ten PCs extracted from the original IVSs. The coefficients and statistical significance for the top two PCs are reported. Comparing Columns (1) and (2), it is evident that adding eight additional PCs increases R^2 while having minimal impact on the coefficients for $\text{PC}_{1,t}$ and $\text{PC}_{2,t}$. When FPCs and SPCs from the encoded IVSs are included, R^2 improves slightly by approximately $1 \sim 2\%$. This smaller improvement reflects the fact that the top ten PCs already explain most of the variance in the original IVSs, leaving limited room for further enhancement by a similar lossy compression method. However, in Column (4) of Panel B, the significance of FPCs and SPCs from the encoded IVSs surpasses that of the first two PCs extracted from the original IVSs. Additionally, in Column (6) of both panels, the FPCs and SPCs from the encoded MLE surfaces remain significant. These results indicate that the FPCs and SPCs from the encoded surfaces produced by our VAE model capture additional information not explained by the

top PCs extracted directly from the IVSs. This added explanatory power is relevant for predicting S&P500 volatility.

We also estimate recession probabilities using the following model:

$$P(NBER_{t:t+30}) = \Phi\left(\alpha + \sum_{i=1}^{k} \gamma_i \text{PC}_{i,t} + \beta_0 \text{FPC}_t + \beta_1 \text{SPC}_t + \varepsilon_t\right). \tag{15}$$

The estimated recession probabilities, based on the first two PCs and the top ten PCs, are presented in Panel C of Exhibit 9. The recession probabilities derived from the FPCs and SPCs of the encoded IVSs align well with broader trends. Specifically, they show higher recession probabilities during periods of SPX price declines and NBER-defined recessions. This qualitative alignment confirms that our VAE model successfully captures significant and explanatory features within the IVSs, enabling it to effectively identify trends associated with underlying asset price movements and recession periods.

Classification of Future Returns

We now use these principal components to classify return movements. We define $\text{Ret}_{t_1:t_2}$ as an indicator variable that is 1 if the cumulative log return over the period $[t_1, t_2]$, $\log\left(\frac{p_{t_2}}{p_{t_1}}\right)$, is positive, and -1 if the return is negative. The FPC_t and SPC_t values extracted from encoded IVSs on days $t-5, \ldots, t-1$ are used as the predictor variables for classifying 1-day and 5-day log returns, $\text{Ret}_{t-1:t}$, $\text{Ret}_{t:t+1}$, $\text{Ret}_{t-1:t+4}$ and $\text{Ret}_{t:t+5}$. For classification, we employ the MLP classifier from the scikit-learn library with default settings[6] and set the random seed to 0 to ensure reproducibility.

First, we conduct common in-sample and out-of-sample tests by splitting the generated surfaces into training (2000–2015) and testing (2016–2023) sets. The classifier is trained on the training set and evaluated on both the training and testing sets. Exhibit 6 reports the accuracy, precision, recall, F1-score, and AUC, with precision, recall, and F1-scores weighted by the number of samples in each class. The AUC is calculated based on the indicator being 1 (positive return). Both the No EX model (Panel A) and the EX Loss model (Panel B) achieve approximately 60% accuracy on the testing set for predicting future 5-day log returns, with an AUC of 0.51. Predictions for 1-day log returns show slightly lower accuracy but marginally better AUC scores. Notably, the EX Loss model demonstrates slightly higher testing accuracy, suggesting that the inclusion of conditioning variables enhances predictive performance.

Next, we assess the practical applicability of generated MLE surfaces for training downstream classification models. Classification models for $\text{Ret}_{t_1:t_2}$ are trained using the principal components computed from the encoded MLE surfaces over the period 2000–2023 and evaluated on the PCs derived from the encoded original surfaces for the same timeframe. Results in Exhibit 7 show that models trained on generated data achieve comparable performance on the original data, with accuracy, precision, recall, F1-scores, and AUC scores around 57% for 5-day log returns. These findings indicate that the generated MLE surfaces effectively encapsulate the essential features of historical data for classifying future return movements. Consequently, they serve as a viable pre-training step for predictive and classification tasks in real-world scenarios.

6. 2-Layer MLP with 100 hidden units, ReLU activation, and Adam Optimizer with a constant 1e-3 learning rate. The strength of the L2 regularization is 1e-4

CONCLUSIONS

In this paper we propose a new model that incorporates the historical context and additional features to generate distributions of IV surfaces on a single day. The same method can be easily extended to generate distributions on multiple days, based on a history of any length. The performance is consistent with a SABR dataset which is a well-defined simplified model for asset pricing and volatility. The results for SABR are recorded in appendix.

We show that our VAE model with conditioning variables can capture the historical variations, and the surfaces generated by our model can be used to predict future realized stock returns and volatilities. The generated dataset could potentially be applied to pre-training of classifications on timeseries data.

Potential future work to expand on this analysis includes:

1) Simultaneously generating surfaces and returns for multiple days, and examining the propagation of errors,

2) Investigating the influence of historical return and implied volatility values on each point of the generated surfaces, and providing a more rigorous interpretation of temporal dependencies,

3) Generalizing the model so that it works on existing datasets to compare its performance with CVAE or a conventional timeseries VAE, and

4) Replacing the standard normal distribution prior with a prior that more accurately represents the distribution of IVSs. This adjustment allows for more precise modeling of the data.

Exhibit 1: Training Statistics

This table presents the basic statistics of the No EX and EX Loss models. Panel A details the validation and test losses during the training process, broken down into surface reconstruction loss (RE Surface), return reconstruction loss (RE Return), and KL divergence loss (KL). Panels B and C report the RMSE between MLE surfaces and the observed surfaces for the models.

Panel A. Training Statistics

	Validation				Test			
	Loss	RE Surface	RE Return	KL	Loss	RE Surface	RE Return	KL
No EX	1.24e-03	1.20e-03	0	3.78e+00	1.81e-03	1.77e-03	0	4.06e+00
EX Loss	1.30e-03	1.20e-03	5.59e-05	4.49e+00	1.99e-03	1.80e-03	1.43e-04	5.03e+00

Panel B. No EX RMSE

	K/S=0.7	K/S=0.85	K/S=1	K/S=1.15	K/S=1.3
1 month	0.1770	0.0538	0.0308	0.0574	0.1013
3 month	0.1130	0.0296	0.0221	0.0281	0.0751
6 month	0.0430	0.0229	0.0168	0.0132	0.0581
1 year	0.0279	0.0167	0.0134	0.0120	0.0162
2 year	0.0431	0.0165	0.0129	0.0141	0.0221

Panel C. EX Loss RMSE

	K/S=0.7	K/S=0.85	K/S=1	K/S=1.15	K/S=1.3
1 month	0.1847	0.0466	0.0273	0.0600	0.0974
3 month	0.0845	0.0243	0.0185	0.0268	0.0728
6 month	0.0368	0.0171	0.0129	0.0115	0.0618
1 year	0.0247	0.0117	0.0091	0.0091	0.0154
2 year	0.0423	0.0131	0.0104	0.0125	0.0210

Exhibit 2: Regression for IVSs with MLE Surfaces (No EX)

This table reports the results of following regression:

$$x_t\,[\text{ttm}, \text{moneyness}] = \alpha + \beta_1 \hat{x}_{MLE,t}\,[\text{ttm}, \text{moneyness}] + \varepsilon_t.$$

The regression is performed on all 25 points of the surfaces generated by the No EX model. The columns represent the moneyness grids, while the rows represent the time-to-maturity grids. Panel A reports the coefficient β_1 for each grid point, and Panel B shows the intercepts α. Asterisks (*, **, ***) denote the associated p-values below the 10%, 5%, and 1% levels, respectively. The standard errors are estimated using Heteroskedasticity Consistent (HC3) Robust Standard Errors. Panel C presents the R^2 of each regression.

Panel A. Coefficients

	K/S=0.7	K/S=0.85	K/S=1	K/S=1.15	K/S=1.3
1 month	1.6313***	1.3792***	1.1515***	0.9804***	1.5459***
3 month	1.2731***	1.3178***	1.1207***	0.9953***	1.2172***
6 month	1.6908***	1.2797***	1.1581***	1.0664***	0.9779***
1 year	1.3994***	1.2462***	1.1657***	1.1282***	1.0431***
2 year	1.3575***	1.2183***	1.1314***	1.1003***	1.0980***

Panel B. Intercepts

	K/S=0.7	K/S=0.85	K/S=1	K/S=1.15	K/S=1.3
1 month	-0.0767***	-0.1193***	-0.0234***	0.0015	-0.1442***
3 month	-0.1414***	-0.0822***	-0.0183***	0.0086***	-0.0500***
6 month	-0.2065***	-0.0671***	-0.0259***	-0.0078***	0.0070**
1 year	-0.1149***	-0.0554***	-0.0312***	-0.0227***	-0.0029***
2 year	-0.0889***	-0.0535***	-0.0288***	-0.0199***	-0.0169***

Panel C. R^2

	K/S=0.7	K/S=0.85	K/S=1	K/S=1.15	K/S=1.3
1 month	0.408	0.597	0.870	0.553	0.188
3 month	0.113	0.840	0.904	0.852	0.323
6 month	0.692	0.877	0.936	0.953	0.388
1 year	0.811	0.920	0.944	0.957	0.871
2 year	0.554	0.885	0.919	0.901	0.779

Exhibit 3: Regression for IVSs with MLE Surfaces (EX Loss)

This table reports the results of following regression:

$$x_t\,[\text{ttm}, \text{moneyness}] = \alpha + \beta_1 \hat{x}_{MLE,t}\,[\text{ttm}, \text{moneyness}] + \varepsilon_t.$$

The regression is performed on all 25 points of the surfaces generated by the EX Loss model. The columns represent the moneyness grids, while the rows represent the time-to-maturity grids. Panel A reports the coefficient β_1 for each grid point, and Panel B shows the intercepts α. Asterisks (*, **, ***) denote the associated p-values below the 10%, 5%, and 1% levels, respectively. The standard errors are estimated using Heteroskedasticity Consistent (HC3) Robust Standard Errors. Panel C presents the R^2 of each regression.

Panel A. Coefficients

	K/S=0.7	K/S=0.85	K/S=1	K/S=1.15	K/S=1.3
1 month	0.9881***	1.3163***	1.1636***	1.1317***	1.4942***
3 month	1.3668***	1.1428***	1.1044***	1.0302***	1.4216***
6 month	1.2272***	1.0860***	1.0808***	1.0454***	1.2162***
1 year	1.0756***	1.0144***	1.0088***	1.0331***	1.0017***
2 year	1.0565***	0.9966***	0.9860***	0.9987***	0.9881***

Panel B. Intercepts

	K/S=0.7	K/S=0.85	K/S=1	K/S=1.15	K/S=1.3
1 month	0.0839***	-0.0970***	-0.0252***	-0.0071***	-0.1134***
3 month	-0.1509***	-0.0381***	-0.0173***	0.0019**	-0.0896***
6 month	-0.0727***	-0.0200***	-0.0144***	-0.0060***	-0.0292***
1 year	-0.0245***	-0.0028***	-0.0018***	-0.0074***	0.0001
2 year	-0.0152***	-0.0017	0.0008	-0.0036***	-0.0010

Panel C. R^2

	K/S=0.7	K/S=0.85	K/S=1	K/S=1.15	K/S=1.3
1 month	0.328	0.703	0.904	0.528	0.265
3 month	0.503	0.872	0.933	0.861	0.395
6 month	0.716	0.913	0.955	0.963	0.324
1 year	0.805	0.942	0.965	0.970	0.878
2 year	0.534	0.910	0.940	0.922	0.798

Exhibit 4: Recession Probability and Volatility Estimation

This table reports the results of following regression:

$$P(NBER_{t:t+30}) = \Phi(\alpha + \beta_0 \text{FPC}_t + \beta_1 \text{SPC}_t + \varepsilon_t),$$
$$\text{Vol5d}_t = \alpha + \beta_0 \text{FPC}_t + \beta_1 \text{SPC}_t + \varepsilon_t,$$

where $NBER_{t:t+30}$ is 1 if a NBER recession period is in the next 30 days, Vol5d_t is the standard deviation over the 5-day log returns $\text{Ret}_t, \text{Ret}_{t+1}, ..., \text{Ret}_{t+4}$, Φ is the standard normal cumulative distribution function, and FPC_t and SPC_t are the first and second principal components of encoded historical IVSs, respectively. Asterisks (*, **, ***) denote the associated p-values below the 10%, 5%, and 1% levels, respectively. The standard errors are estimated using Newey-West standard errors, with maximum lags set to 30.

Panel A. No EX

	NBER		Vol5d	
	Encoded Original	Encoded MLE	Encoded Original	Encoded MLE
	(1)	(2)	(3)	(4)
FPC	0.264***	0.276***	0.0014***	0.0014***
	(0.041)	(0.042)	(0.00)	(0.00)
SPC	0.108	0.017	0.0005	0.0013***
	(0.076)	(0.118)	(0.00)	(0.00)
Intercept	-1.522***	-1.545***	0.0090***	0.0092***
	(0.149)	(0.161)	(0.00)	(0.00)
R^2	0.337	0.338	0.430	0.400
N	5810	5805	5810	5805

Panel B. EX Loss

	NBER		Vol5d	
	Encoded Original	Encoded MLE	Encoded Original	Encoded MLE
	(1)	(2)	(3)	(4)
FPC	0.227***	0.267***	0.0014***	0.0014***
	(0.033)	(0.039)	(0.00)	(0.00)
SPC	0.460***	0.683***	0.0012***	0.0018***
	(0.101)	(0.159)	(0.00)	(0.00)
Intercept	-1.525***	-1.498***	0.0090***	0.0091***
	(0.138)	(0.136)	(0.00)	(0.00)
R^2	0.353	0.361	0.466	0.404
N	5810	5805	5810	5805

Exhibit 5: Incremental Effect of Encoded Surfaces for Volatility Estimation

This table reports the results of following regression:

$$\text{Vol5d}_t = \alpha + \sum_{i=1}^{k} \gamma_i \text{PC}_{i,t} + \beta_0 \text{FPC}_t + \beta_1 \text{SPC}_t + \varepsilon_t,$$

where Vol5d_t is the standard deviation over the 5-day log returns $\text{Ret}_t, \text{Ret}_{t+1}, ..., \text{Ret}_{t+4}$, $\text{PC}_{i,t}$ are the ith principal components of original historical IVSs, and FPC_t and SPC_t are the first and second principal components of encoded historical IVSs, respectively. Columns (1), (3), and (5) show incremental effects with 2 PCs of original IVSs, while Columns (2), (4), and (6) use 10 PCs. Coefficients for $\text{PC}_{1,t}$ and $\text{PC}_{2,t}$ are reported. Asterisks (*, **, ***) denote the associated p-values below the 10%, 5%, and 1% levels, respectively. The standard errors are estimated using Newey-West standard errors, with maximum lags set to 30.

Panel A. No EX Vol5d

	Original		Encoded Original		Encoded MLE	
	2PCs	10PCs	2PCs	10PCs	2PCs	10PCs
	(1)	(2)	(3)	(4)	(5)	(6)
PC_1	0.0068***	0.0068***	0.0030***	0.0064***	0.0056***	0.0101***
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
PC_2	-0.0061***	-0.0061***	0.0011	-0.0042**	-0.0037**	-0.0107***
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
FPC			0.0009***	0.0002	0.0003	-0.0008*
			(0.00)	(0.00)	(0.00)	(0.00)
SPC			0.0011***	0.0009***	0.0013***	0.0006**
			(0.00)	(0.00)	(0.00)	(0.00)
Intercept	0.0090***	0.0090***	0.0090***	0.0090***	0.0090***	0.0089***
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
R^2	0.427	0.507	0.459	0.516	0.439	0.516
N	5810	5810	5810	5810	5805	5805

Panel B. EX Loss Vol5d

	Original		Encoded Original		Encoded MLE	
	2PCs	10PCs	2PCs	10PCs	2PCs	10PCs
	(1)	(2)	(3)	(4)	(5)	(6)
PC_1	0.0068***	0.0068***	0.0066***	0.0034	0.0157***	0.0170***
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
PC_2	-0.0061***	-0.0061***	-0.0003	0.0003	-0.0124***	-0.0158***
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
FPC			0.0002	0.0008*	-0.0018**	-0.0022***
			(0.00)	(0.00)	(0.00)	(0.00)
SPC			0.0022***	0.0013***	0.0025***	0.0012***
			(0.00)	(0.00)	(0.00)	(0.00)
Intercept	0.0090***	0.0090***	0.0090***	0.0090***	0.0089***	0.0088***
	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)	(0.00)
R^2	0.427	0.507	0.488	0.519	0.489	0.529
N	5810	5810	5810	5810	5805	5805

Exhibit 6: Return Classification (IS/OOS)

In this table, we classify the indicator variables $\text{Ret}_{t-1:t}$, $\text{Ret}_{t:t+1}$, $\text{Ret}_{t-1:t+4}$ and $\text{Ret}_{t:t+5}$ using FPC_t and SPC_t, the first and second principle components of encoded historical IVSs. Ret_{t_1,t_2} is 1 if $\log\left(\frac{p_{t_2}}{p_{t_1}}\right)$ is positive and -1 if negative. We report the accuracy, weighted precision, recall, F1-scores, and AUC scores, with weights based on the number of observations in each class. We train on generated data from 2000 to 2015 and evaluate on generated data from 2016 to 2023.

Panel A. No EX

		Accuracy	Precision (Weighted)	Recall (Weighted)	F1-score (Weighted)	AUC
$\text{Ret}_{t-1:t}$	train (2000-2015)	0.545	0.549	0.545	0.472	0.537
	test (2016-2023)	0.539	0.519	0.539	0.439	0.515
$\text{Ret}_{t:t+1}$	train (2000-2015)	0.535	0.558	0.535	0.394	0.529
	test (2016-2023)	0.543	0.542	0.543	0.405	0.530
$\text{Ret}_{t-1:t+4}$	train (2000-2015)	0.560	0.554	0.560	0.504	0.561
	test (2016-2023)	0.586	0.525	0.586	0.511	0.487
$\text{Ret}_{t:t+5}$	train (2000-2015)	0.553	0.542	0.553	0.494	0.556
	test (2016-2023)	0.584	0.509	0.584	0.497	0.501

Panel B. EX Loss

		Accuracy	Precision (Weighted)	Recall (Weighted)	F1-score (Weighted)	AUC
$\text{Ret}_{t-1:t}$	train (2000-2015)	0.533	0.527	0.533	0.415	0.533
	test (2016-2023)	0.542	0.580	0.542	0.390	0.522
$\text{Ret}_{t:t+1}$	train (2000-2015)	0.536	0.544	0.536	0.415	0.531
	test (2016-2023)	0.542	0.542	0.542	0.393	0.518
$\text{Ret}_{t-1:t+4}$	train (2000-2015)	0.571	0.574	0.571	0.510	0.573
	test (2016-2023)	0.603	0.534	0.603	0.492	0.501
$\text{Ret}_{t:t+5}$	train (2000-2015)	0.559	0.550	0.559	0.518	0.558
	test (2016-2023)	0.604	0.537	0.604	0.490	0.512

Exhibit 7: Return Classification (Generated/Original)

In this table, we classify the indicator variables $\text{Ret}_{t-1:t}$, $\text{Ret}_{t:t+1}$, $\text{Ret}_{t-1:t+4}$ and $\text{Ret}_{t:t+5}$ using FPC_t and SPC_t, the first and second principle components. Ret_{t_1,t_2} is 1 if $\log\left(\frac{p_{t_2}}{p_{t_1}}\right)$ is positive and -1 if negative. We report the accuracy, weighted precision, recall, F1-scores, and AUC scores, with weights based on the number of observations in each class. We train on generated data, and evaluate on original data.

Panel A. No EX

		Accuracy	Precision (Weighted)	Recall (Weighted)	F1-score (Weighted)	AUC
$\text{Ret}_{t-1:t}$	train (Encoded MLE)	0.541	0.555	0.541	0.420	0.530
	test (Encoded Original)	0.530	0.509	0.530	0.426	0.511
$\text{Ret}_{t:t+1}$	train (Encoded MLE)	0.534	0.522	0.534	0.392	0.528
	test (Encoded Original)	0.534	0.524	0.534	0.391	0.522
$\text{Ret}_{t-1:t+4}$	train (Encoded MLE)	0.567	0.547	0.567	0.426	0.544
	test (Encoded Original)	0.569	0.554	0.569	0.439	0.544
$\text{Ret}_{t:t+5}$	train (Encoded MLE)	0.565	0.533	0.565	0.454	0.547
	test (Encoded Original)	0.572	0.560	0.572	0.463	0.550

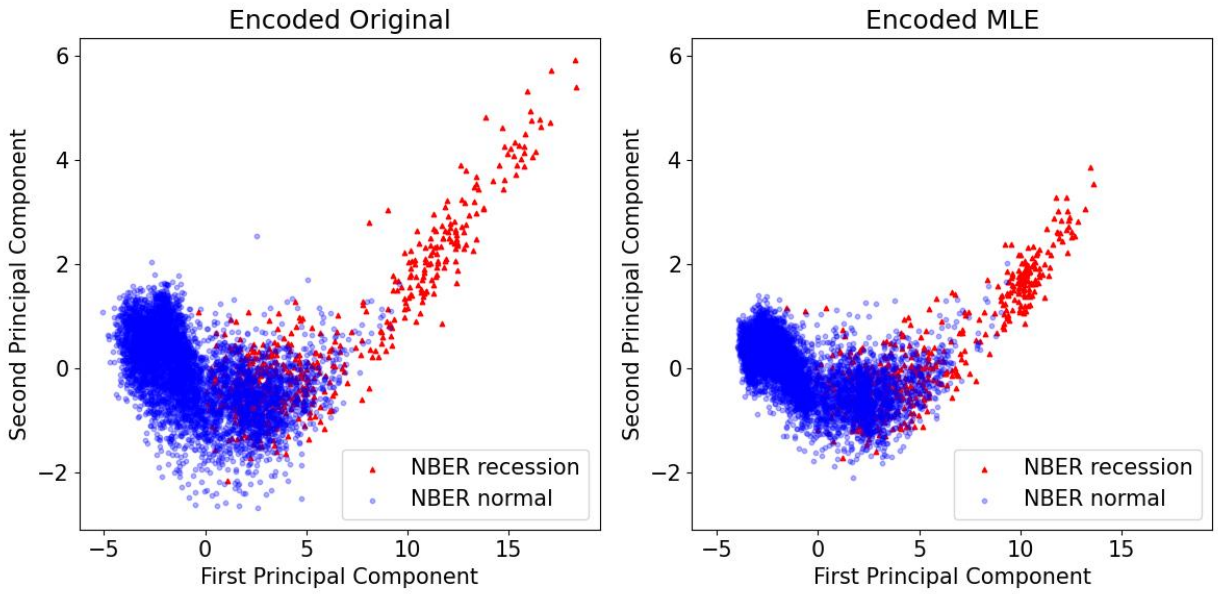
Panel B. EX Loss

		Accuracy	Precision (Weighted)	Recall (Weighted)	F1-score (Weighted)	AUC
$\text{Ret}_{t-1:t}$	train (Encoded MLE)	0.535	0.525	0.535	0.406	0.534
	test (Encoded Original)	0.536	0.532	0.536	0.402	0.523
$\text{Ret}_{t:t+1}$	train (Encoded MLE)	0.537	0.530	0.537	0.430	0.529
	test (Encoded Original)	0.532	0.514	0.532	0.423	0.520
$\text{Ret}_{t-1:t+4}$	train (Encoded MLE)	0.582	0.580	0.582	0.497	0.555
	test (Encoded Original)	0.572	0.557	0.572	0.479	0.539
$\text{Ret}_{t:t+5}$	train (Encoded MLE)	0.573	0.556	0.573	0.508	0.547
	test (Encoded Original)	0.570	0.550	0.570	0.511	0.539

Exhibit 8: PCA of Context Embeddings

The left figure shows the first and second principal components of the context embedding for the original surfaces, while the right figure shows the same for the generated MLE surfaces. The horizontal axis represents the first principal component, and the vertical axis represents the second. Triangles denote data points during NBER recessions, and circles denote data points during normal years. The principal components are computed using a 5-day context from days $t-5$ to $t-1$. Panel A displays the results for No EX model, and Panel B displays the EX Loss model.

A. No EX



B. EX Loss

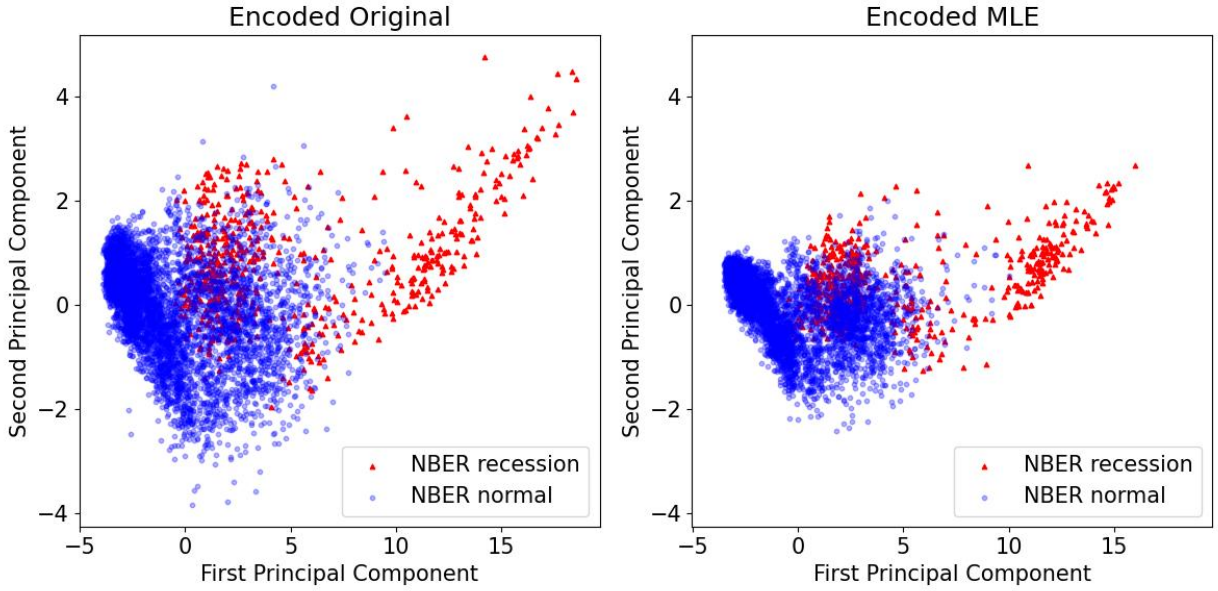
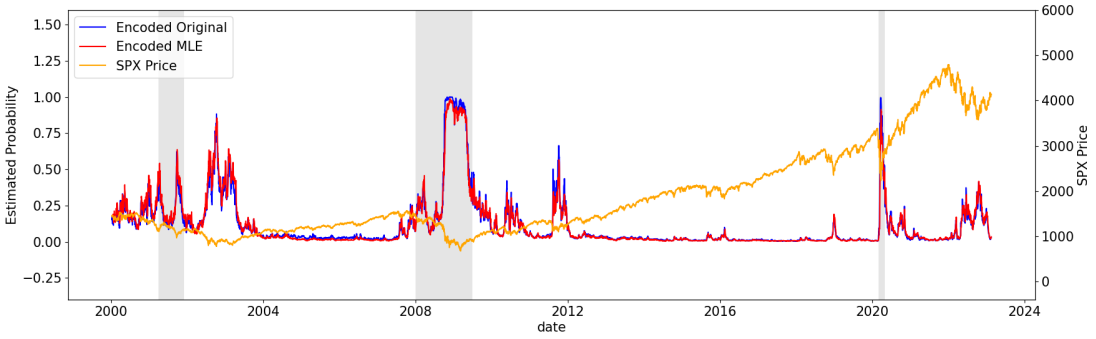


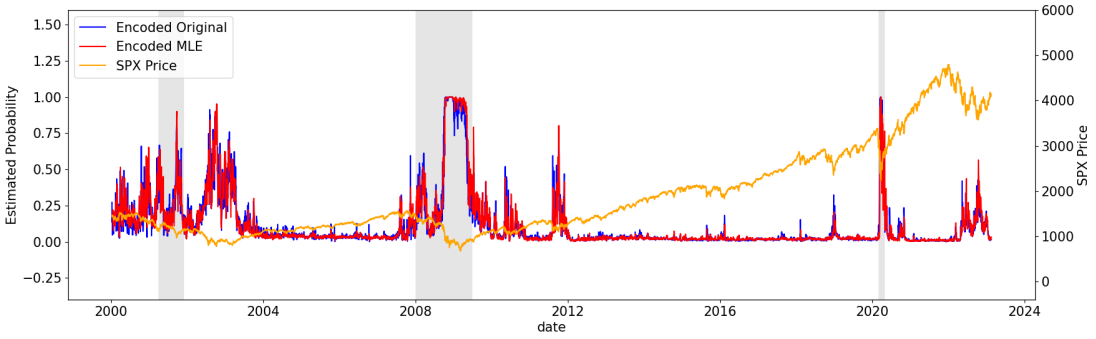
Exhibit 9: Recession Probability Estimation

Panels A and B show the estimation for encoded surfaces by No EX model and EX Loss model, and Panel C shows the estimation for original IVSs, respectively. In each figure, gray bars represent NBER recession periods, and the yellow curve represents the S&P500 Price. In Panels A and B, the blue plot shows recession probability estimated from the principal components of encoded original IVSs, and the red plot shows recession probability estimated from the principal components of encoded MLE IVSs. In Panel C, the blue plot shows recession probability estimated by the first two principal components, and the red plot shows recession probability estimated by ten principal components.

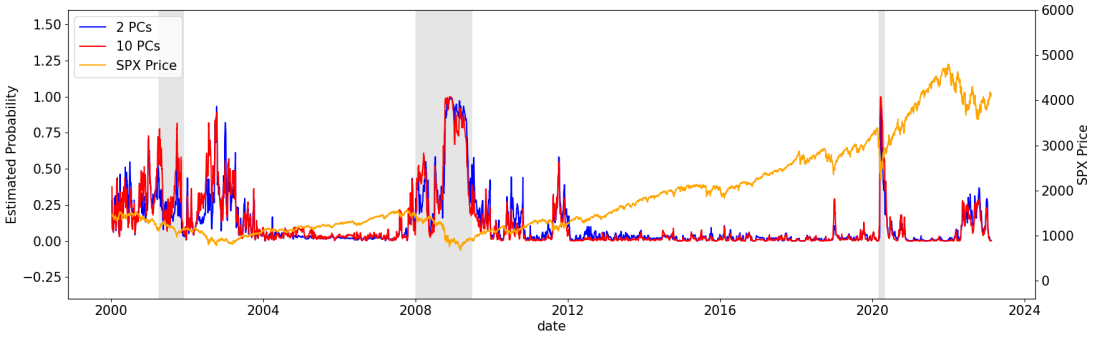
A. No EX



B. EX Loss



C. Original IVSs



Acknowledgment: This research was carried out at Rotman School of Management, FinHub Financial Innovation Lab, University of Toronto. 105 St. George Street, Toronto, ON, M5S 3E6, Canada.

References

Bergeron, Maxime, Nicholas Fung, John Hull, Zissis Poulos, and Andreas Veneris. 2022. "Variational Autoencoders: A Hands-Off Approach to Volatility." *The Journal of Financial Data Science* 4 (2): 125–138.

Black, Fischer, and Myron Scholes. 1973. "The pricing of options and corporate liabilities." *Journal of political economy* 81 (3): 637–654.

Bloch, Daniel Alexandre, and Arthur Böök. 2021. *Deep Learning Based Dynamic Implied Volatility Surface.* SSRN Scholarly Paper. Rochester, NY. https://doi.org/10.2139/ssrn.3952842. https://ssrn.com/abstract=3952842.

Bowman, Samuel R., Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. 2016. "Generating Sentences from a Continuous Space," arXiv: 1511.06349 [cs.LG]. http://arxiv.org/abs/1511.06349.

Cao, Jay, Jacky Chen, John Hull, and Zissis Poulos. 2022. "Deep Learning for Exotic Option Valuation." *The Journal of Financial Data Science* 4 (1): 41–53.

Carmona, Rene, Yi Ma, and Sergey Nadtochiy. 2017. "Simulation of Implied Volatility Surfaces via Tangent Levy Models." *SIAM Journal on Financial Mathematics* 8:171–213.

Choudhary, Vedant, Sebastian Jaimungal, and Maxime Bergeron. 2023. "FuNVol: A Multi-Asset Implied Volatility Market Simulator using Functional Principal Components and Neural SDEs," arXiv: 2303.00859 [q-fin.CP]. https://arxiv.org/abs/2303.00859.

Cohen, Sam N., Christoff Reisinger, and Sheng Wang. 2020. "Detecting and Repairing Arbitrage in Traded Option Prices." *Applied Mathematical Finance* 27:345–373.

Cont, Rama, Mihai Cucuringu, Renyuan Xu, and Chao Zhang. 2023. "Tail-GAN: Learning to Simulate Tail Risk Scenarios," arXiv: 2203.01664 [q-fin.RM]. https://arxiv.org/abs/2203.01664.

Cont, Rama, and José Da Fonseca. 2002. "Dynamics of Implied Volatility Surfaces." *Quantitative Finance* 2 (1): 45–60.

Cont, Rama, José Da Fonseca, and Valdo Durrleman. 2002. "Stochastic models of implied volatility surfaces." *Economic Notes* 31:361–377.

Cont, Rama, and Milena Vuletić. 2023. "Simulation of Arbitrage-Free Implied Volatility Surfaces." *Applied Mathematical Finance* 30:94–121.

Cuchiero, Christa, Walid Khosrawi, and Josef Teichmann. 2020. "A generative adversarial network approach to calibration of local stochastic volatility models." *Risks* 8(4):101. https://doi.org/10.3390/risks8040101.

Desai, Abhyuday, Cynthia Freeman, Zuhui Wang, and Ian Beaver. 2021. "TimeVAE: A Variational Auto-Encoder for Multivariate Time Series Generation," arXiv: 2111.08095 [cs.LG]. http://arxiv.org/abs/2111.08095.

Francois, Pascal, Rémi Galarneau-Vincent, Genevieve Gauthier, and Frédéric Godin. 2022. "Venturing into Uncharted Territory: An Extensible Parametric Implied Volatility Surface Model." *Journal of Futures Markets* 42:10:1912–1940.

FRED. 2023. *NBER based Recession Indicators for the United States from the Period following the Peak through the Trough [USREC].* Retrieved from FRED, Federal Reserve Bank of St. Louis. Accessed on July, 31, 2023. https://fred.stlouisfed.org/series/USREC.

Gatheral, Jim, and Antoine Jacquier. 2014. "Arbitrage-free SVI volatility surfaces." *Quantitative Finance* 14:1:59–71.

Greff, Klaus, Rupesh K. Srivastava, Jan Koutnik, Bas R. Steunebrink, and Jurgen Schmidhuber. 2017. "LSTM: A Search Space Odyssey." *IEEE Transactions on Neural Networks and Learning Systems* 28 (10): 2222–2232. https://doi.org/10.1109/tnnls.2016.2582924. https://doi.org/10.1109/tnnls.2016.2582924.

Hagan, P., D. Kumar, A. Lesniewski, and D. Woodward. 2002. "Managing smile risk." *Wilmott,* 84–108.

Higgins, Irina, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework." In *International Conference on Learning Representations.* https://openreview.net/forum?id=Sy2fzU9gl.

Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* (Cambridge, MA, USA) 9 (8): 1735–1780. ISSN: 0899-7667. https://doi.org/10.1162/neco.1997.9.8.1735. https://doi.org/10.1162/neco.1997.9.8.1735.

Kingma, Diederik P, and Max Welling. 2013. "Auto-Encoding Variational Bayes," arXiv: 1312.6114 [stat.ML]. http://arxiv.org/abs/1312.6114.

Kingma, Diederik P., and Jimmy Ba. 2017. "Adam: A Method for Stochastic Optimization," arXiv: 1412.6980 [cs.LG]. https://arxiv.org/abs/1412.6980.

Lin, Shuyu, Ronald Clark, Robert Birke, Sandro Schönborn, Niki Trigoni, and Stephen Roberts. 2020. "Anomaly Detection for Time Series Using VAE-LSTM Hybrid Model." In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* 4322–4326. https://doi.org/10.1109/ICASSP40776.2020.9053558.

Loshchilov, Ilya, and Frank Hutter. 2019. "Decoupled Weight Decay Regularization." In *International Conference on Learning Representations.* https://openreview.net/forum?id=Bkg6RiCqY7.

Merton, Robert C. 1973. "Theory of Rational Option Pricing." *The Bell Journal of Economics and Management Science* 4 (1): 141–183.

Ning, Brian (Xin), Sebastian Jaimungal, Xiaorong Zhang, and Maxime Bergeron. 2023. "Arbitrage-Free Implied Volatility Surface Generation with Variational Autoencoders." *SIAM Journal on Financial Mathematics* 14 (4): 1004–1027. https://doi.org/10.1137/21M1443546. eprint: https://doi.org/10.1137/21M1443546. https://doi.org/10.1137/21M1443546.

Shang, Han Lin, and Fearghal Kearney. 2022. "Dynamic functional time-series forecasts of foreign exchange implied volatility surfaces." *Journal of Banking & Finance* 125:106–120.

Siami-Namini, Sima, Neda Tavakoli, and Akbar Siami Namin. 2018. "A Comparison of ARIMA and LSTM in Forecasting Time Series." In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA),* 1394–1401. https://doi.org/10.1109/ICMLA.2018.00227.

Sohn, Kihyuk, Honglak Lee, and Xinchen Yan. 2015. "Learning Structured Output Representation using Deep Conditional Generative Models." *Advances in Neural Information Processing Systems* 28:3483–3491. http://papers.nips.cc/paper/5775-learning-structured-output-representation-using-deep-conditional-generative-models.

APPENDIX: MODEL DETAILS

Let C be the context length (the number of relevant historical days), N be the number of days we want to generate, $T = C + N$ be the overall sequence length, and L be the number of latent variables. Let $x_t \in \mathbb{R}^{H \times W}$ be the observed IVS on day t, where H represents time-to-maturity grid size, W represents moneyness (K/S) grid size. Let p_t be the asset price on day t. We use log-return $r_t = \log\left(\frac{p_t}{p_{t-1}}\right)$ to model price changes. Let $y_t \in \mathbb{R}^E$ be the conditioning variables of interest, where E is the number of conditioning variables. One such conditioning variable can be r_t. Let $\mathbf{x}_c = (x_{t-C}, ..., x_{t-1}) \in \mathbb{R}^{C \times H \times W}$ be the historical IVSs from day $t-C$ to $t-1$, $\mathbf{x}_n = (x_t, ..., x_{t+N-1}) \in \mathbb{R}^{N \times H \times W}$ be the IVSs we want to generate from day t to day $t+N-1$. Let $\mathbf{y}_c = (y_{t-C}, ..., y_{t-1}) \in \mathbb{R}^{C \times E}$ be the conditioning variables from day $t-C$ to $t-1$, and $\mathbf{y}_n = (y_t, ..., y_{t+N-1}) \in \mathbb{R}^{N \times E}$ be the conditioning variables from day t to $t+N-1$. If a, b are two constants, then $[a,b]$ is the usual interval notation. If $a = (a_1, ..., a_n)$, $b = (b_1, ..., b_m)$ are two vectors/tensors, then $[a,b] = (a_1, ..., a_n, b_1, ..., b_m)$ is the vector/tensor concatenation.

The model consists of three parts:

1) **Encoder**: Takes $\mathbf{x} = [\mathbf{x}_c, \mathbf{x}_n] \in \mathbb{R}^{T \times H \times W}$ and $\mathbf{y} = [\mathbf{y}_c, \mathbf{y}_n] \in \mathbb{R}^{T \times E}$ and builds a distribution $\mathcal{N}(\mu_{t,i}, \sigma_{t,i})$ for each day t and each latent variable $z_{t,i}$. The latents $\mathbf{z} = (z_{t-C}, ..., z_{t+N-1}) \in \mathbb{R}^{T \times L}$ are then sampled individually from the corresponding distributions. $\forall i \in [t-C, t+N-1]$, the encoder generates the daily latent vector z_i by $q_\phi(z_i | \mathbf{x}_{[t-C,...,i]}, \mathbf{y}_{[t-C,...,i]}, i)$, where $\mathbf{x}_{[t-C,...,i]} = (x_{t-C}, ..., x_i)$.

2) **Context Encoder**: Takes $\mathbf{x}_c$ and $\mathbf{y}_c$ and generates a compressed representation of the context $\zeta = (\zeta_{t-C}, ..., \zeta_{t-1}) \in \mathbb{R}^{C \times L}$.[7] $\forall i \in [t-C, t-1]$, the context encoder generates an embedding $\zeta_i = \text{Embed}(\mathbf{x}_{c,[t-C,...,i]}, \mathbf{y}_{c,[t-C,...,i]})$, where $\mathbf{x}_{c,[t-C,...,i]} = (x_{t-C}, ..., x_i)$.

3) **Decoder**: Takes $\mathbf{z}$ from the encoder and ζ from the context encoder and reconstructs $\hat{\mathbf{x}}_n = (\hat{x}_t, ..., \hat{x}_{t+N-1})$, $\hat{\mathbf{y}}_n = (\hat{y}_t, ..., \hat{y}_{t+N-1})$. Each $\hat{x}_i$, $\hat{y}_i$, $i \in [t, t+N-1]$ is reconstructed by sampling from $p_\theta(x_i, y_i | \zeta, \mathbf{z}_{[t-C,...,i]}, i)$, where $\mathbf{z}_{[t-C,...,i]} = (z_{t-c}, ..., z_i)$.

Each component uses convolutional neural networks (CNNs) or multi-layer perceptrons (MLPs) to encode/decode the surface structures on each day, and LSTM to capture time dependencies. Graphical illustrations of each part and the complete model are shown in Exhibit 10, 11, 12, 13. All convolutional/transpose convolutional/linear layers within the CNN/TCNN (transpose convolutional neural network)/MLP blocks are followed by ReLU activation ($\max(x, 0)$), unless an identity function (Id) or single linear layer is used in place of MLPs, or the layer is the last layer of the TCNN/MLP in the decoder. The TCNN/MLP blocks used in the decoder are symmetric to the CNN/MLP blocks used in the encoder, ensuring matching input and output shapes as well as encoding and decoding processes. For all convolutional and transpose convolutional layers, the kernel size is fixed to 3×3 with a stride of 1 and equal size padding, so the grid size remains unchanged after multiple convolutions. Embedding function f_1, f_2, g_1, g_2 are performed by these blocks in the encoder and context encoder.

7. We can configure the context embedding size to values other than L, but we use L as embedding size here to be consistent with the main article.

Exhibit 10: Encoder

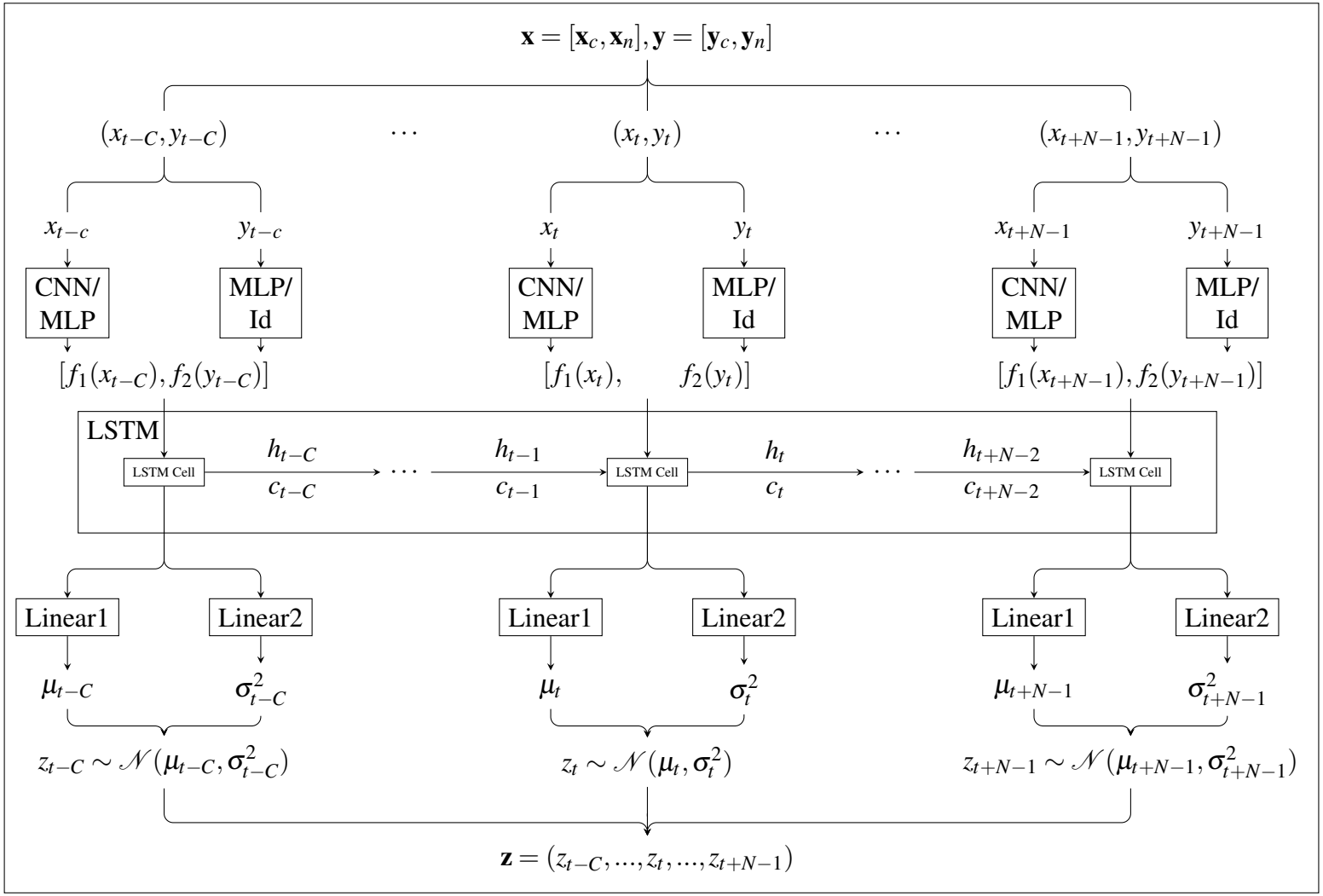


Exhibit 11: Context Encoder

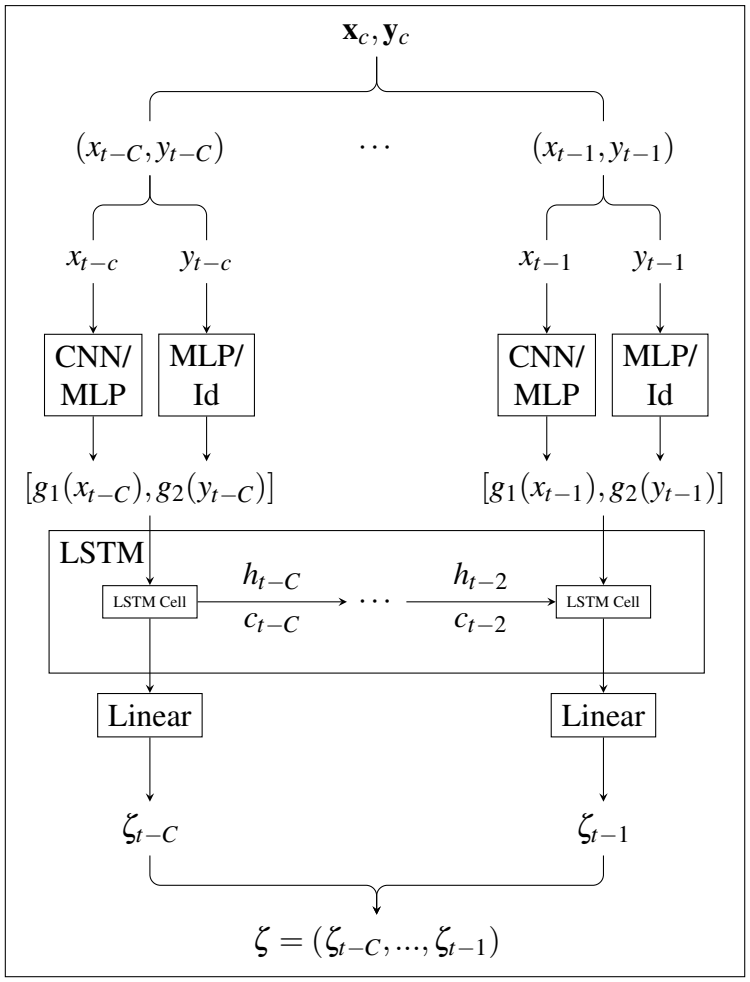


Exhibit 12: Decoder

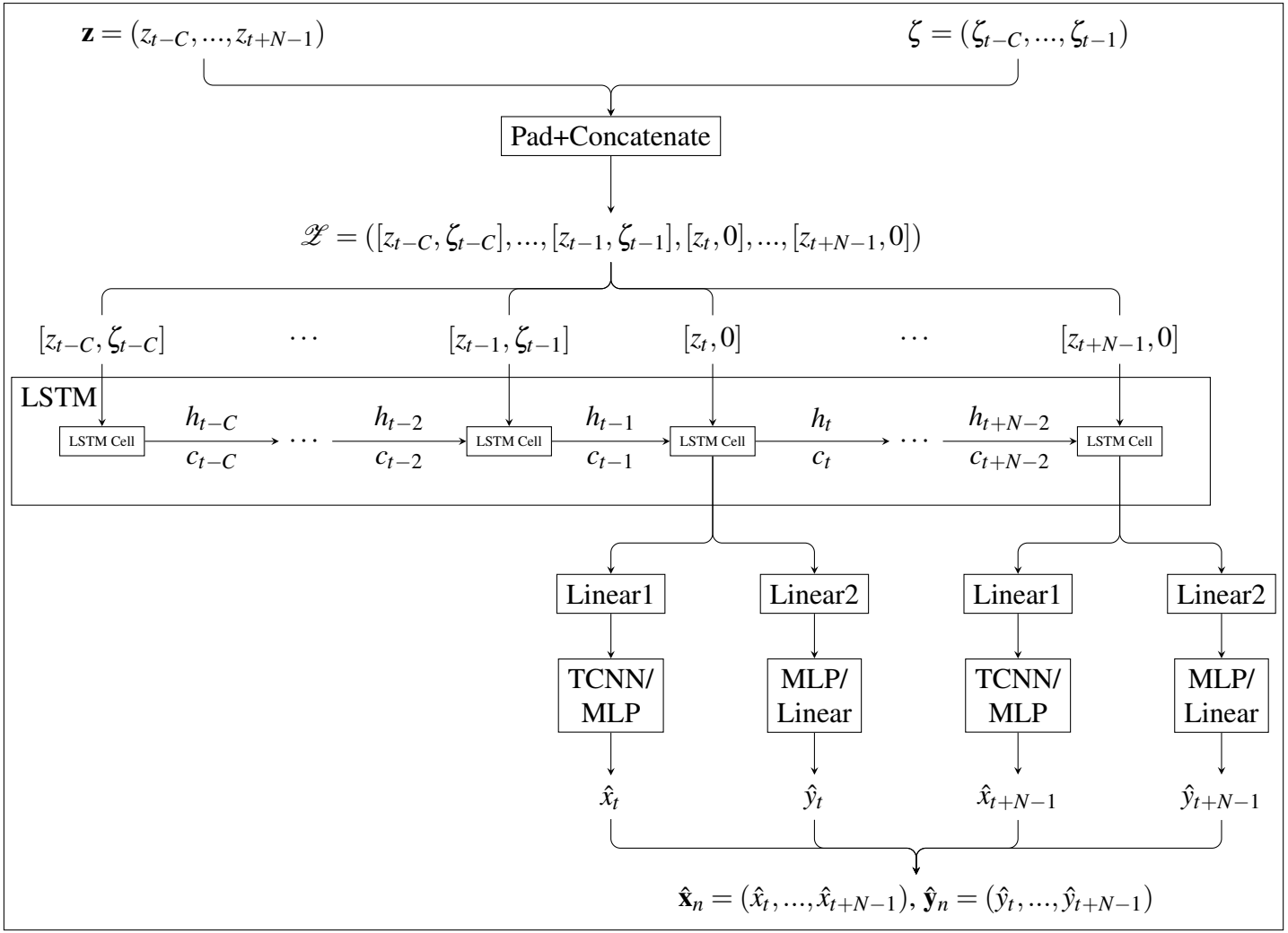
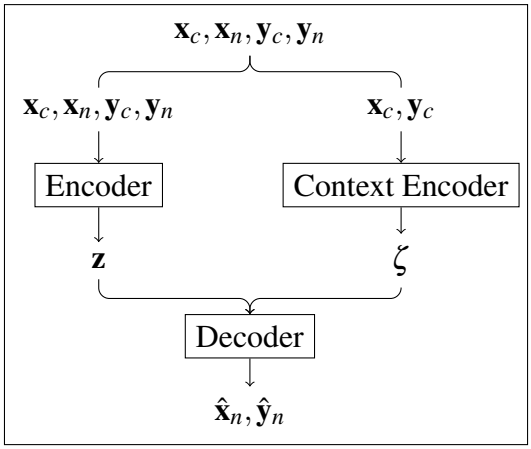


Exhibit 13: Overall Architecture



APPENDIX: ARBITRAGE CONDITIONS

Following Gatheral and Jacquier (2014) and Bergeron et al. (2022), we specify static arbitrage conditions as follows: let $M = \frac{K}{S}$ be the moneyness, t be the time to maturity. Define $w(t,M) = t \cdot x(t,M)$ as the total implied variance surface, where x is the IVS on a single day. An IVS is free of *calendar arbitrage* if

$$\frac{\partial w}{\partial t} \geq 0. \tag{16}$$

Let $w' = \frac{\partial w}{\partial M}$ and $w'' = \frac{\partial^2 w}{\partial M^2}$. The IVS is free of *butterfly arbitrage* if

$$\left(1 - \frac{Mw'}{2w}\right)^2 - \frac{w'}{4}\left(\frac{1}{w} + \frac{1}{4}\right) + \frac{w''}{2} \geq 0. \tag{17}$$

An IVS is said to be free of static arbitrage if the conditions in Equation 16 and 17 are satisfied.

We test the arbitrage conditions on the realized IVSs constructed by calibration on the available options. In the calibrated surfaces, there are 2585 surfaces with at least one calendar arbitrage opportunities and 595 surfaces with at least two calendar arbitrage opportunities. In comparison, there are 2182 days where more than 5% of generated surfaces with no conditioning variables have at least one calendar arbitrage opportunity, and there are only 389 days where more than 5% of generated surfaces have more than one calendar arbitrage opportunity, marking a significant reduction compared to the observed surfaces. Also, when conditioning variables are added, there are 1768 days where more than 5% of generated surfaces have one calendar arbitrage opportunity, and no surfaces have more than one calendar arbitrage opportunity. Hence, with additional conditioning variables regulating the IVS structure, we can reduce the possibility of calendar arbitrage in the generated surfaces. Nonetheless, they fall short of avoiding butterfly arbitrage.

It is possible to incorporate the calendar arbitrage condition into the calibration of IVSs on the available options we have, and the learned model can further reduce the calendar arbitrage opportunity with the better calibrated data. However, incorporating the butterfly arbitrage condition into the calibration may lead to realized market volatility being captured less well. To further reduce the possibility of static arbitrage in the generated surfaces, we could extend our loss function in Equation (8) similarly to Bergeron et al. (2022).

APPENDIX: CONTROLLED SABR ENVIRONMENT

The histogram and latent variable tests were performed under a controlled environment in order to study the the model's capacity to understand latent features and patterns in path trajectories of price returns and volatility surface evolution. The SABR model developed by Hagan et al. (2002), was used to generate over 10,000 paths of evolving implied volatility surfaces given the same fixed grid of moneyness and time to maturity using General Brownian Motion for underlying asset price evolution. We define our SABR model with the following properties: A Skew parameter of β = 1 , initial risk free rate of $r_f = 0$, volatility of volatility of $volvol = 0.3$, correlation parameter of ρ = -0.7, and an initial underlying asset price of S_0 = 10. On the SABR data set, the model reported a reconstruction loss of 4.56×10^{-5} of surfaces, reconstruction loss of 3.97×10^{-5} on returns, and KL loss of 0.59 in validation; in testing, losses of 4.65×10^{-5} and 4.00×10^{-5} were reported for surface and returns reconstruction respectively with the same KL loss.

The model configuration used on S&P500 data yielded more balanced outcomes for this dataset in terms of volatility level and returns. Additionally, it produces a higher frequency of outcomes wherein realized additional features such as skew and slope are similar to the mean of daily generated normal distributions. This is evident through a greater number of outcomes falling within the Q2 and Q3 buckets, coupled with smaller daily z-scores, in our histogram tests as illustrated in Exhibit 14 and Exhibit 15.

The SABR dataset has been valuable for enhancing the explainability our VAE model's latent space, as well as the influence of significant additional features like skew and slope. We investigated the effect of altering latent variables while fixing the others on two model variants. It is observed that including different features obtain distinct outcomes illustrated in Exhibit 16. The set of graphs in Panel A shows how the generated surfaces changes when we manipulate each latent value for a baseline model trained only on surface context, excluding any other conditioning variables. The second set of graphs in Panel B shows the impacts of latent value manipulation for a model that trains losses on returns and includes the other conditioning variables: skew and slope. When changing a latent value of the VAE model with skew and slope context information, there is reduced variation in the surface level and shape relative to the realized curve, in contrast to the model that excludes these features. Note this figure illustrates changes in the first latent variable for ttm $= 1$ only, however this trend holds for all 5 latent factors in the model and all moneyness and time to maturity slices.

Exhibit 14: Histogram Plots of SABR Surfaces

This Figure shows the histogram plots for the best VAE model that includes skew and slope information. The left picture shows the distribution and the right picture shows the z-scores for the levels.

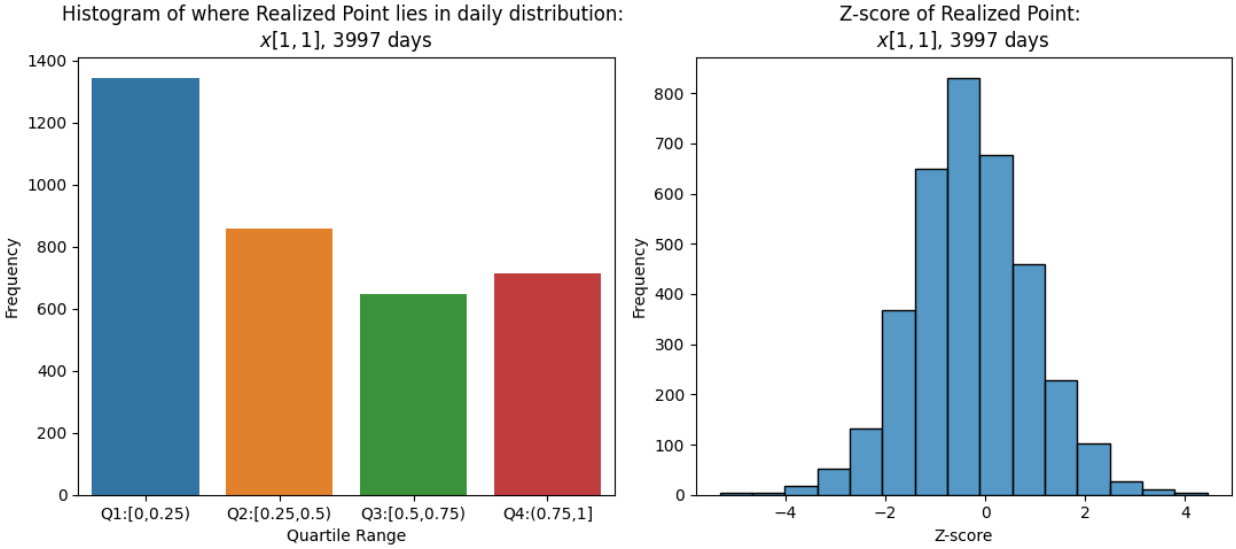


Exhibit 15: Histogram Plots of SABR Surface Structural Properties

This Figure shows the histogram plots for the best VAE model that includes skew and slope information. The top row shows the distribution and z-scores for the skews. The second row shows the distribution and z-scores for the slopes.

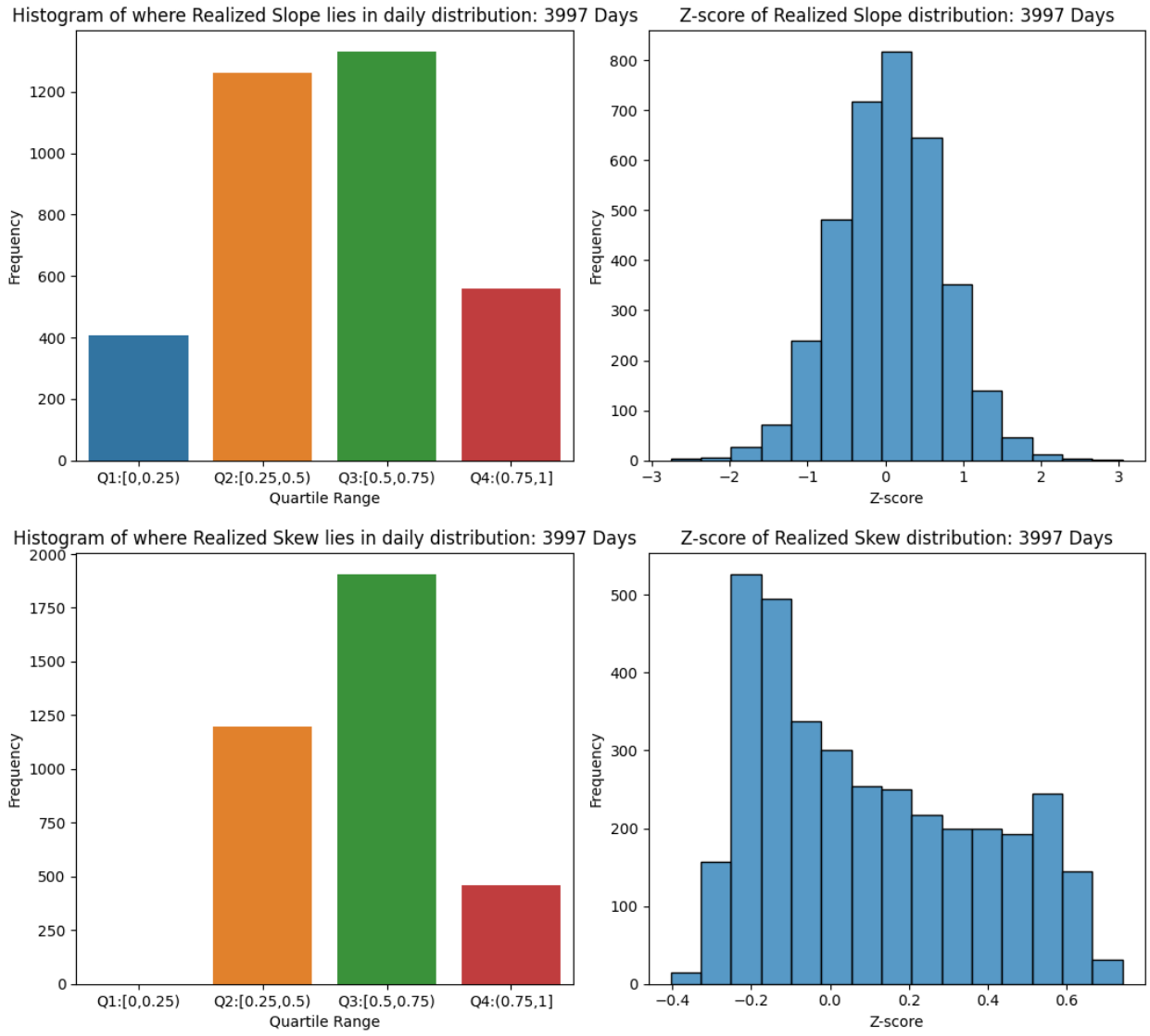
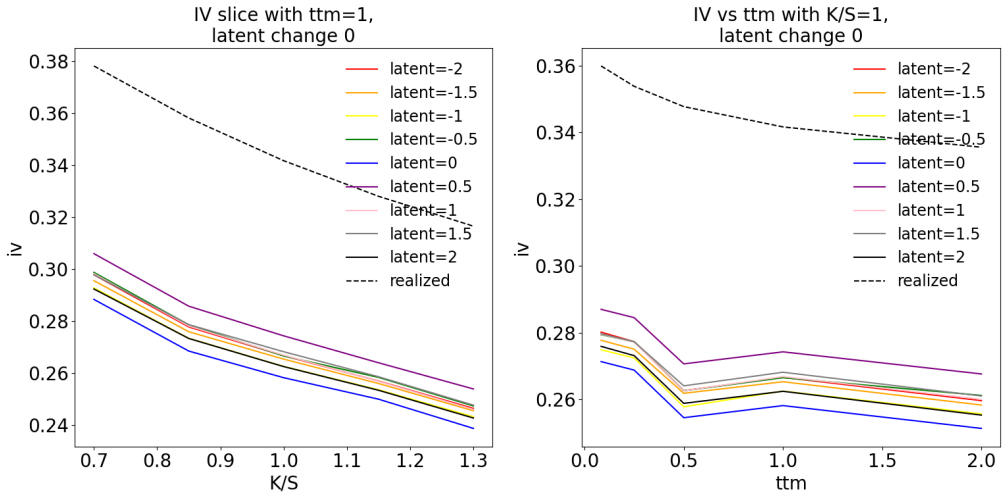
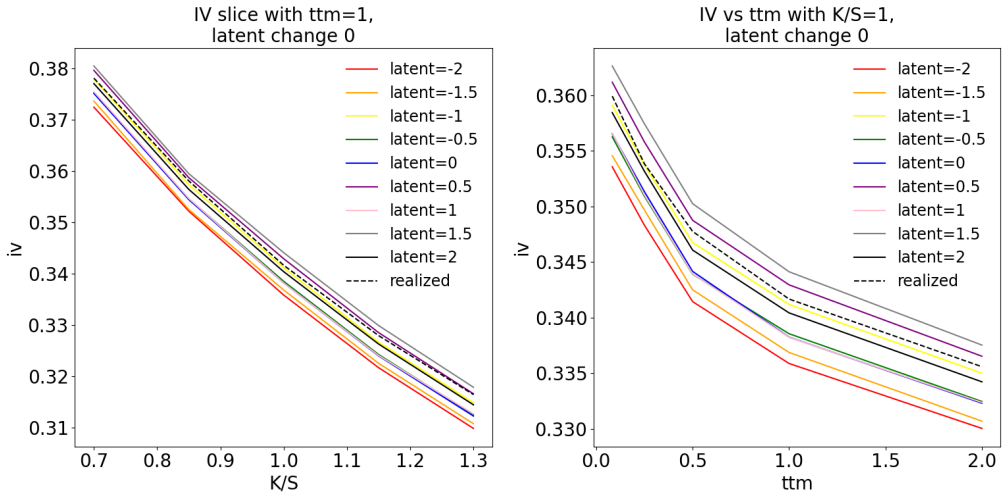


Exhibit 16: SABR Latent Manipulation

A. VAE SABR Model with Surface Context Only



B. VAE SABR Model with Return, Slope, Skew, and Surface Context



APPENDIX: ADDITIONAL TABLES

Exhibit 17: Incremental Effect of Encoded Surfaces for Recession Probability Estimation

This table replicates Exhibit 5 for recession probability estimation with the following configuration:

$$P(NBER_{t:t+30}) = \Phi\left(\alpha + \sum_{i=1}^{k} \gamma_i \text{PC}_{i,t} + \beta_0 \text{FPC}_t + \beta_1 \text{SPC}_t + \varepsilon_t\right),$$

where $NBER_{t:t+30}$ is 1 if a NBER recession period is in the next 30 days.

Panel A. No EX NBER

	Original		Encoded Original		Encoded MLE	
	2PCs	10PCs	2PCs	10PCs	2PCs	10PCs
	(1)	(2)	(3)	(4)	(5)	(6)
PC_1	0.864***	0.869***	-0.288	0.923	0.008	0.914
	(0.142)	(0.161)	(0.672)	(0.797)	(0.485)	(0.611)
PC_2	-2.942***	-3.213***	-1.128	-3.303**	-1.495	-3.295***
	(0.783)	(0.728)	(1.032)	(1.481)	(0.968)	(1.219)
FPC			0.269	-0.013	0.215*	-0.009
			(0.164)	(0.180)	(0.126)	(0.144)
SPC			0.003	-0.011	-0.011	-0.125
			(0.112)	(0.083)	(0.127)	(0.105)
Intercept	-1.610***	-1.708***	-1.626***	-1.711***	-1.613***	-1.732***
	(0.184)	(0.205)	(0.212)	(0.204)	(0.201)	(0.208)
R^2	0.352	0.392	0.366	0.392	0.366	0.394
N	5810	5810	5810	5810	5805	5805

Panel B. EX Loss NBER

	Original		Encoded Original		Encoded MLE	
	2PCs	10PCs	2PCs	10PCs	2PCs	10PCs
	(1)	(2)	(3)	(4)	(5)	(6)
PC_1	0.864***	0.869***	1.653*	1.678	-0.161	-1.004
	(0.142)	(0.161)	(0.896)	(1.189)	(0.614)	(0.777)
PC_2	-2.942***	-3.213***	-2.823***	-3.457**	-0.991	-0.558
	(0.783)	(0.728)	(1.044)	(1.455)	(0.986)	(1.198)
FPC			-0.127	-0.145	0.257*	0.429**
			(0.183)	(0.238)	(0.141)	(0.178)
SPC			0.370***	0.186*	0.512***	0.251**
			(0.130)	(0.103)	(0.198)	(0.124)
Intercept	-1.610***	-1.708***	-1.626***	-1.698***	-1.566***	-1.655***
	(0.184)	(0.205)	(0.175)	(0.205)	(0.169)	(0.198)
R^2	0.352	0.392	0.380	0.395	0.377	0.403
N	5810	5810	5810	5810	5805	5805