Journal of Computational Finance **26**(3), 33–72
DOI: 10.21314/JCF.2022.028

Copyright Infopro Digital Limited 2023. All rights reserved. You may share using our article tools. This article may be printed for the sole use of the Authorised User (named subscriber), as outlined in our terms and conditions. https://www.infopro-insight.com/termsconditions/insight-subscriptions

Research Paper

Estimating risks of European option books using neural stochastic differential equation market models

Samuel N. Cohen, Christoph Reisinger and Sheng Wang

Mathematical Institute, Andrew Wiles Building, Radcliffe Observatory Quarter, Woodstock Road, Oxford OX2 6GG, UK; emails: samuel.cohen@maths.ox.ac.uk, christoph.reisinger@maths.ox.ac.uk, sheng.wang@maths.ox.ac.uk

(Received March 18, 2022; revised October 19, 2022; accepted December 22, 2022)

ABSTRACT

In this paper we examine the capacity of arbitrage-free neural stochastic differential equation market models to produce realistic scenarios for the joint dynamics of multiple European options on a single underlying. We subsequently demonstrate their use as a risk simulation engine for option portfolios. Through backtesting analysis we show that our models are more computationally efficient and accurate for evaluating the value-at-risk of option portfolios than standard filtered historical simulation approaches, with better coverage and less procyclicality.

Keywords: market models; European options; risk measures; market simulators; no-arbitrage; stochastic differential equation (SDE).

1 INTRODUCTION

Managing option trade books usually requires modeling and simulating dynamics of liquid vanilla options. A desirable model, and therefore trajectories simulated from the model, should preserve statistical properties of the real data while respecting

Corresponding author: S. Wang

Print ISSN 1460-1559 | Online ISSN 1755-2850
© 2023 Infopro Digital Risk (IP) Limited

underlying financial constraints. Previous work (see, for example, Cohen *et al* 2021a) considered building arbitrage-free neural stochastic differential equation (SDE) market models under the real-world measure, and they established corresponding model-estimation algorithms that take as input time series price data of finitely many liquid options. The model consists of an SDE system for states representing the underlying and a small number of factors chosen for a statistically accurate dynamic representation of the option surface and the minimization of dynamic and static arbitrage. The coefficients of the SDE are calibrated under hard linear inequality constraints on the joint option price processes; these constraints are dictated by static arbitrage relationships. For practical applications, Cohen *et al* (2021a) consider only synthetic data simulated from a Heston stochastic local volatility model, and they do not demonstrate performance on any concrete tasks. This paper validates the modeling approach of Cohen *et al* (2021a) using real-world data, and it explores the model's capability as a realistic option market simulator. In particular, we will see that these methods yield computationally efficient and accurate models for evaluating value-at-risk, with less procyclicality and better performance than traditional filtered historical simulation (Barone-Adesi *et al* 2002).

Essentially, we investigate the feasibility of using a neural-SDE market model as a generative model for time series of option prices. Simulating synthetic data from a generative model is an important measure to ensure data privacy (Bellovin *et al* 2019), and it can be useful for boosting deep-learning performance – for instance, by providing an environment to train further models with reinforcement learning (Buehler *et al* 2019; Ritter and Kolm 2019). Synthetic data generators are probabilistic models that simulate observations similar to historical data, in the sense of capturing specific features of the time series. Recent developments of generative models for financial time series focus on generative adversarial networks (GANs) (Henry-Labordere 2019; Koshiyama *et al* 2021; Ni *et al* 2020; Takahashi *et al* 2019; Wiese *et al* 2019) and variational autoencoders (Buehler *et al* 2020; Wiese *et al* 2021) and are typically not driven by fundamental financial modeling principles. We highlight the work of Wiese *et al* (2021) and Chataigner *et al* (2020), who take similar approaches to ours, although they use different representations of arbitrage conditions.

Building generative models for multiple option price time series, with the same underlying asset and different strikes and maturities, is by nature a high-dimensional problem, in which complex arbitrage constraints need to be respected among different dimensions. Wiese *et al* (2019, 2021) first transform option prices into an equivalent representation in the form of discrete local volatilities (Wissel 2008) with simple static arbitrage constraints; they then apply principal component analysis to reduce the dimension and formulate a typical GAN problem. Compared with GAN approaches, our neural-SDE market models largely reduce the black-box nature of

neural network models (see the discussion by Cohen *et al* (2021b)) while still retaining their computational advantages. Specifically, our neural-SDE market models give drift and diffusion estimates of each risk factor modeled, which helps us to understand the expected return and volatility of investing in each risk factor.

Here, we verify that these neural-SDE market models produce price paths similar to historical data for European equity index markets, in that they display volatility clustering (in the returns on the underlying index, and this is replicated for an option portfolio similar to the Chicago Board Options Exchange's Volatility Index (VIX)) and they match marginal and joint densities for the underlying and option portfolios. We then focus on risk simulation for option portfolios, which is especially convenient because the models are built under the real-world measure. In particular, we assess how well the models evaluate risks for a comprehensive set of option portfolios that represent typical risk profiles, such as (calendar) spreads, risk reversals and strangles. In fact, these portfolios can be viewed as statistical features of option price surfaces. We therefore verify that the neural-SDE market models are capable of simulating data similar to historical data, in the sense that they capture these highly structured while financially meaningful features.

Evaluating market risk for portfolios of nonlinear derivatives such as options is challenging from both an accuracy perspective and a speed perspective. We focus on value-at-risk (VaR), a risk measure promoted by JP Morgan RiskMetrics in the mid-1990s (JP Morgan–Reuters 1996) and quickly mandated by international regulators such as the US Securities and Exchange Commission and the Basel Committee on Banking Supervision, as the measure for disclosing portfolio risk. Mathematically, the key is to establish a loss distribution for the evaluated portfolio. As our models generate samples from the full loss distribution of the portfolio, our approach could also be used to compute estimates of other risk measures, such as expected shortfall. We focus on VaR because of its widespread use in the regulation of financial markets, and because of the existence of standard backtesting approaches, which allow us to verify the performance of our method against standard criteria.

Parametric approaches to VaR estimation, such as the delta method (JP Morgan–Reuters 1996) and delta–gamma method (Britten-Jones and Schaefer 1999; Fallon 1996), give closed-form approximations of the loss distribution that are fast but rely heavily on distributional assumptions on the risk factors. In addition, El-Jahel *et al* (1999) have shown that these methods give biased VaR estimates due to poor local (linear and quadratic) approximations of the loss.

Conversely, nonparametric methods based on Monte Carlo estimation of the loss distribution are unbiased and consistent, and they have flexible distributional assumptions, but they can present a substantial computational burden. First, a large number of samples of the risk factors (known as "risk scenarios") are needed, as VaR considers only the (upper) tail of the loss distribution. Second, for each risk scenario,

the portfolio value needs to be reevaluated, and since this value is generally not available in closed form, computationally costly numerical procedures are needed. If Monte Carlo simulation is used for pricing the portfolio, it results in a two-level (or nested) simulation scheme. Variance-reduction techniques can be used in both outer and inner levels to accelerate simulations; specifically, for the outer level, which estimates quantiles, there is a vast literature discussing the use of control variates (Hesterberg and Nelson 1998; Hsu and Nelson 1990), antithetic variates (Avramidis and Wilson 1998) and, more popularly, importance sampling or stratified sampling (Glasserman *et al* 2000, 2002; Sun and Hong 2010). Further nonparametric VaR estimation methods include historical simulation (Butler and Schachter 1997) and volatility-filtered historical simulation (Barone-Adesi and Giannopoulos 2001; Barone-Adesi *et al* 1999, 2002; Gurrola-Perez and Murphy 2015), which sample risk scenarios based on empirical distributions of risk factors. Nevertheless, regardless of how risk scenarios are simulated, the valuation of option portfolios often requires reevaluating option prices on the basis of risk factors (for multiple simulation trajectories and time points), which is computationally expensive.

In our market models, option prices are linear in the risk factors, resulting in a negligible computational cost for reevaluating an option portfolio for each simulated risk scenario. Despite the simplicity of the linear factor representation, it is sufficiently rich to achieve satisfactory statistical accuracy. As will be detailed below, we simulate risk factors from a flexible, offline-trained neural-SDE with historical innovations.[1] Through out-of-sample backtesting, we show that our model yields more statistically and economically appealing VaR estimates for various option portfolios than traditional filtered historical simulation approaches (see, for example, Barone-Adesi *et al* 2002).

The paper is organized as follows. First, we review the construction and estimation methodology of arbitrage-free neural-SDE market models in Section 2. Then, we explain the data set used and elaborate how to apply the models to real-world data in Section 3, in which the in-sample and out-of-sample performance of the models is assessed extensively. Given the trained models, we estimate VaR over various risk horizons and confidence levels for a varied set of option portfolios and backtest the out-of-sample VaR performance in Section 4. Finally, Section 5 summarizes our observations and addresses a few further extensions.

[1] We carried out neural-SDE model calibration to real-world data, risk factor simulation and VaR backtesting using PYTHON for various option portfolios in the neuralSDE-marketmodel GitHub repository. URL: https://github.com/vicaws/neuralSDE-marketmodel.

2 ARBITRAGE-FREE NEURAL-SDE MARKET MODELS

Here, we briefly summarize our neural-SDE approach to constructing market models. Interested readers may refer to Cohen *et al* (2021a) for a discussion of model choices, theoretical strengths and limitations, and other implementational details.

2.1 Factor-based market models

We consider a financial market with a stock and a collection of European call options on the stock.[2] We denote by $C_t(T, K)$ the market price at time t of a European call option with expiry T and strike K. Over time, the range of liquid options in (T, K)-coordinates tends to change stochastically. In contrast, there is a usually stable range of times-to-expiry and moneynesses for which options are actively quoted and for which price data are thus readily available. It is therefore empirically advantageous to build models in terms of relative coordinates (τ, m), where $\tau = T - t$ is time-to-expiry and m denotes moneyness. Specifically, we model the normalized call price surface $(\tau, m) \mapsto \tilde{c}_t(\tau, m)$ with the definition

$$\tilde{c}_t(\tau, m) = \frac{C_t(t + \tau, \mathrm{e}^m F_t(t + \tau))}{D_t(t + \tau) F_t(t + \tau)}, \tag{2.1}$$

where $D_t(T)$ denotes the market discount factor for time T (assuming a deterministic interest rate) and $F_t(T)$ denotes the model-independent, arbitrage-free futures price for delivery of the stock at T.

We fix a constant compact set $\mathcal{R}_{\mathrm{liq}} \subset \{(\tau, m) \in \mathbb{R}^2\}$, which represents the range of times-to-expiry and moneynesses for which options are liquid. For each τ and m we assume a time-independent linear representation of $\tilde{c}_t(\tau, m)$ by latent factors $\xi \in \mathbb{R}^d$, given by

$$\tilde{c}_t(\tau, m) = G_0(\tau, m) + \sum_{i=1}^{d} G_i(\tau, m)\xi_{it}, \tag{2.2}$$

where $G_i \in C^{1,2}(\mathcal{R}_{\mathrm{liq}})$ for $i = 0, \ldots, d$, and we refer to it as a "price basis function". We then model the joint dynamics of the stock price S and the factors ξ by a $(d + 1)$-dimensional diffusion process solving the following pair of time-homogenous SDEs:

$$\frac{\mathrm{d}S_t}{S_t} = \alpha(\xi_t)\,\mathrm{d}t + \gamma(\xi_t)\,\mathrm{d}W_{0,t}, \qquad S_0 = s_0 \in \mathbb{R}, \tag{2.3a}$$

$$\mathrm{d}\xi_t = \mu(\xi_t)\,\mathrm{d}t + \sigma(\xi_t)\,\mathrm{d}W_t, \qquad \xi_0 = \zeta_0 \in \mathbb{R}^d, \tag{2.3b}$$

[2] Although we refer to the asset as a "stock" throughout the paper, our methods are equally applicable to other assets such as equity indexes, currencies or commodities.

where $W_0 \in \mathbb{R}$ and $W = [W_1 \cdots W_d]^{\mathrm{T}} \in \mathbb{R}^d$ are standard independent Brownian motions under the real-world measure $\mathbb{P}$. We use α_t, γ_t, μ_t and σ_t to denote $\alpha(\xi_t)$, $\gamma(\xi_t)$, $\mu(\xi_t)$ and $\sigma(\xi_t)$, respectively. We denote by $L^p_{\mathrm{loc}}(\mathbb{R}^d)$ the space of all $\mathbb{R}^d$-valued, progressively measurable and locally p-integrable (in t, $\mathbb{P}$-almost-surely) processes, and we assume that $\alpha \in L^1_{\mathrm{loc}}(\mathbb{R})$, $\mu \in L^1_{\mathrm{loc}}(\mathbb{R}^d)$, $\gamma \in L^2_{\mathrm{loc}}(\mathbb{R})$ and $\sigma \in L^2_{\mathrm{loc}}(\mathbb{R}^d)$.

REMARK 2.1 We assume, in the model of ξ (2.3b), that the drift and diffusion coefficients are independent of S, in the sense that volatility surfaces (represented by ξ) are unaffected by the underlying price level. Nevertheless, it is possible, and simple, to add S as an additional argument and train the model with neural nets (as done in Cohen *et al* (2021a)).

REMARK 2.2 For simplicity we assume that independent Brownian motions drive the randomness in S and ξ; we could easily add a correlation structure and estimate the resulting larger covariance matrix, albeit with a greater risk of overfitting. In Appendix C.2 (available online) we discuss performance when estimating models for S and ξ jointly with a full $\mathbb{R}^{(d+1)\times(d+1)}$ covariance matrix. With the current model setup (2.3), to take into account nonnegligible correlations between ξ and log returns of S for realistic forward simulations, we randomly sample innovations from the historical residuals of the fitted models (2.3), rather than from standard Gaussians; see the discussion in Section 3.6.

2.2 Arbitrage considerations

According to the first fundamental theorem of asset pricing (Harrison and Kreps 1979), a model admits no arbitrage if there exists a measure $\mathbb{Q} \sim \mathbb{P}$ such that the discounted price processes for all tradable assets are martingales. (Formally, this can be weakened to local or σ-martingales, depending also on the precise form of no-arbitrage assumed.) For this no-arbitrage condition to hold, we derive a Heath–Jarrow–Morton-type (HJM-type) drift restriction (see also Heath *et al* 1992) by analyzing the existence of a process for the market price of risk $\psi_t = [\psi_{1,t} \cdots \psi_{d,t}]^{\mathrm{T}}$. Considering a finite set of option specifications $\mathcal{L}_{\mathrm{liq}} = \{(\tau_j, m_j)\}_{j=1,\ldots,N} \subset \mathcal{R}_{\mathrm{liq}}$, we look for a solution ψ_t to

$$\boldsymbol{G}^{\mathrm{T}}\sigma_t\psi_t = \boldsymbol{G}^{\mathrm{T}}\mu_t - z_t, \quad \text{where } z_t = \left(-\frac{\partial}{\partial\tau} - \frac{1}{2}\gamma_t^2\frac{\partial}{\partial m} + \frac{1}{2}\gamma_t^2\frac{\partial^2}{\partial m^2}\right)\boldsymbol{c}_t, \quad (2.4)$$

with $\boldsymbol{c}_t = [\tilde{c}_t(\tau_1, m_1) \cdots \tilde{c}_t(\tau_N, m_N)]^{\mathrm{T}}$. Here, we define $\boldsymbol{G}$ as the $d \times N$ matrix with ith row $\boldsymbol{G}_i$, where $\boldsymbol{G}_i = (G_{ij})_{j=1}^N \in \mathbb{R}^N$ for $i = 0, \ldots, d$ and $G_{ij} = G_i(\tau_j, m_j)$. We call $\boldsymbol{G}^{\mathrm{T}}$ a "price basis matrix" and each of its columns a "price basis vector". The relation (2.4) leads to a mechanism that discourages dynamic arbitrage in the

factor-decoding algorithm (Algorithm 1 in Appendix A online) and to a metric for measuring the proportion of dynamic arbitrage in factor-reconstructed prices (Cohen *et al* 2021a, Section 4.3).

Nevertheless, the HJM-type drift restrictions cannot be implemented exactly in practice, because our models are built on finitely many options on a fixed liquid lattice in the (τ, m)-coordinates, which do not correspond to fixed contracts (ie, fixed T and K) over time, so solutions to (2.4) will not exist for a fixed price basis $\mathbb{G}$. Further consideration of no-arbitrage conditions is therefore needed. In particular, we can consider static, model-free arbitrage constraints on the underlying factors. Using the construction in our previous work (Cohen *et al* 2020), we write static arbitrage constraints in the form $\boldsymbol{A}\boldsymbol{c}_t \geqslant \hat{\boldsymbol{b}}$, where $\boldsymbol{A} = (A_{ij}) \in \mathbb{R}^{R\times N}$ and $\hat{\boldsymbol{b}} = (\hat{b}_j) \in \mathbb{R}^R$ are a known constant matrix and vector, and R is the number of static arbitrage constraints. Given the factor representation (2.2), we have $\boldsymbol{c}_t = \boldsymbol{G}_0 + \boldsymbol{G}^{\mathrm{T}}\xi_t$. Consequently, the market model allows no static arbitrage among options on the liquid lattice $\mathcal{L}_{\mathrm{liq}}$ if ξ_t satisfies, for all t,

$$\boldsymbol{A}\boldsymbol{G}^{\mathrm{T}}\xi_t \geqslant \boldsymbol{b} := \hat{\boldsymbol{b}} - \boldsymbol{A}\boldsymbol{G}_0^{\mathrm{T}}. \tag{2.5}$$

2.3 Model inference

As the first step of model inference, we decode the factor representation (2.2) by calibrating the price basis functions $\{G_i\}_{i=0}^d$. This leads to Algorithm 1 (see Appendix A online), which extracts a smaller number of market factors from prices of finitely many options with the joint objectives of eliminating static and dynamic arbitrage in reconstructed prices and guaranteeing statistical accuracy.

To estimate the drift and diffusion of the decoded factor process ξ, as specified in (2.3b), we represent the drift and diffusion functions by neural networks, referred to as "neural-SDEs". Leveraging deep-learning algorithms, we train the neural networks by maximizing the likelihood of observing the factor paths, subject to the derived arbitrage constraints. As seen in (2.5), static arbitrage constraints are characterized by the convex polytope state space $\{y \in \mathbb{R}^d : \boldsymbol{A}\boldsymbol{G}^{\mathrm{T}}y \geqslant \boldsymbol{b}\}$ for the factors; we identify sufficient conditions on the drift and diffusion to restrict the factors to the polytope, using the classic results of Friedman and Pinsky (1973). Consequently, the neural network that is used to parameterize the drift and diffusion functions needs to be constrained. We achieve this by developing appropriate transformations for the output of the neural network, as detailed in Cohen *et al* (2021a, Section 3).

3 CALIBRATION TO MARKET DATA

We have seen decent performance from our modeling and inference approach when applied to synthetic data generated by a stochastic local volatility model (Jex *et al*

1999). The focus here is to investigate its ability to capture the dynamics of real-world data, and whether practically useful conclusions can be drawn from these methods.

3.1 Option price data description and preprocessing

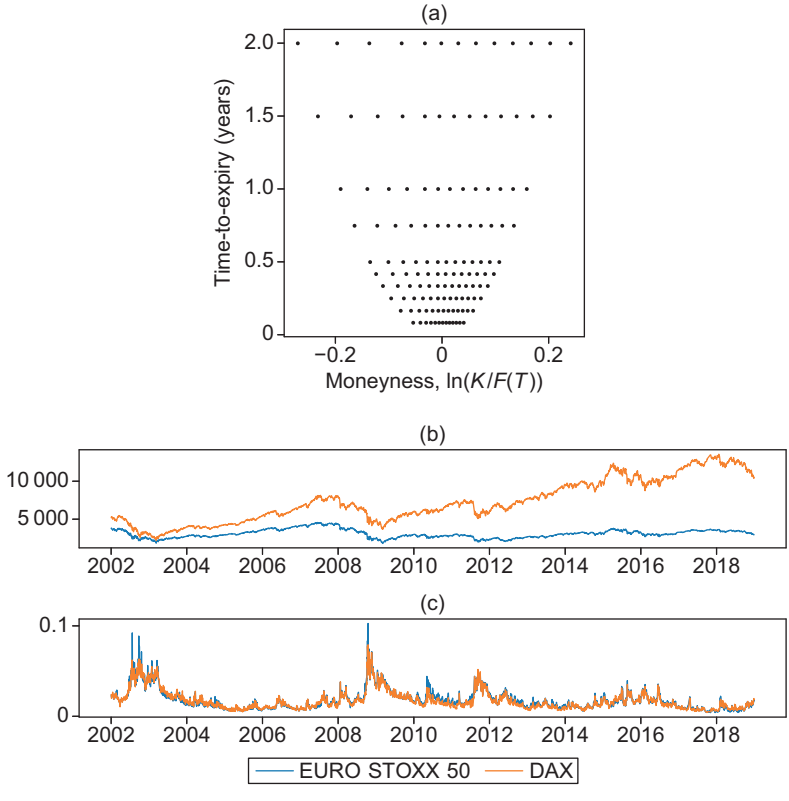
We study the two most traded vanilla European options listed on Eurex Exchange, namely, EURO STOXX 50 index options and Deutsche Atkienindex (DAX) options. OptionMetrics' IvyDB Europe offers historical daily settlement prices[3] for calls with expiries of 30, 60, 91, 122, 152, 182, 273, 365, 547 and 730 calendar days, at deltas of 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75 and 0.8 (OptionMetrics 2020). We collect daily call option prices from January 2, 2002 to December 30, 2019, as well as corresponding futures curves and euro zero curves.

We first normalize call prices by discount factors and futures prices following (2.1). However, the normalized call price surfaces have fixed delta parameters, rather than fixed moneynesses, over time. We therefore choose the historical median moneynesses (for each delta-quoted option) to define our liquid option lattice $\mathcal{L}_{\text{liq}}$, and we interpolate prices to the fixed set of moneyness parameters. We give details on the interpolation in Appendix B (available online). In addition, whenever there is static arbitrage in the reported historical option prices, we perturb the prices using the arbitrage-repair algorithm of Cohen *et al* (2020). In Appendix B we give statistics on the presence of arbitrage in raw and interpolated option price data, and we assess the impact of repairing arbitrage. We show the liquid lattice $\mathcal{L}_{\text{liq}}$ in Figure 1(a).

In parts (b) and (c) of the figure we show historical daily prices for the EURO STOXX 50, the DAX and the one-month at-the-money normalized call options written on the two indexes. Since the normalized option price has a deterministic bijective relationship with the implied volatility, we see that the two options have very similar historical volatility dynamics. It is consequently reasonable to assume a common background model for the volatility processes of the two options. We therefore concatenate the normalized call price data of the EURO STOXX 50 options and DAX options when decoding factors and learning factor dynamics. This enriches the data set available for our statistical estimation problems by assuming that our market models are universal for options written on these two different underlyings. Specifically, there is only one functional form for the drift $\mu(\cdot)$ and the diffusion $\sigma(\cdot)$ for the factor dynamics (2.3), despite different underlying options. Nevertheless, this does not restrict the underlying indexes' models to being the same; as evidenced

[3] These prices are calculated from implied volatilities interpolated using a methodology based on a kernel-smoothing algorithm, according to the OptionMetrics' IvyDB Europe reference manual (OptionMetrics 2020).

FIGURE 1 Scattergram of the liquid option lattice and plots of historical prices.



(a) Liquid lattice $\mathcal{L}_{\text{liq}}$. (b) Historical daily prices: underlying indexes. (c) Historical daily prices: one-month at-the-money call option prices (normalized).

in parts (b) and (c) of Figure 1, the DAX appears to have a more positive drift.[4] Hence, we build a universal model for the factor dynamics but separate models for each index.

Using $k \in \{\mathrm{U}, \mathrm{D}\}$ to distinguish data of EURO STOXX 50 options and DAX options, we denote the observation times as $0 = t_1 < \cdots < t_{L^k} = T^k$ and let $\delta t = t_i - t_{i-1} = T^k/(L^k - 1)$ for all $2 \leqslant i \leqslant L^k$. We split the L^k historical data samples into subsamples for training and estimation $\{t_1, \ldots, t_{E^k}\}$ and subsamples for testing $\{t_{E^k+1}, \ldots, t_{L^k}\}$. We use the training samples only for estimating a

[4] Two factors contribute to the drift difference between the two indexes: in addition to the fact that the two equity indexes have different constituent stocks, the EURO STOXX 50 is a price return index while the DAX is a total return index.

TABLE 1 Details of the option data used.

Parameter	Value
Number of training samples E^k	4296 (Jan 1, 2002 to Dec 31, 2018)
Number of testing samples $L^k - E^k$	253 (Jan 1, 2019 to Dec 31, 2019)
Number of options N	130
δt	One day*

*We use the Act/365 Fixed day count convention for converting $\delta t = 1/365$ to a year fraction.

neural-SDE market model, while the testing samples are used for testing the performance of out-of-sample risk simulation, as detailed in Section 4.4. In Table 1 we list the details of the data used.

REMARK 3.1 We focus our attention on options on a single underlying asset, but some generalizations of this are straightforward within our framework. For example, given two assets that are not perfectly correlated, and independent options on each asset, it would be easy to determine the corresponding no-arbitrage restrictions (as these will apply to each option book separately) and to then build a joint model for both assets and their options. Difficulties would arise when considering options on more than one asset (in addition to options on single assets), as the no-arbitrage conditions would be less tractable.

3.2 Factor decoding

Suppose we observe price data $\boldsymbol{C}^k = (C^k_{lj}) \in \mathbb{R}^{L^k \times N}$, with $C^k_{lj} = \tilde{c}^k_{t_l}(\tau_j, m_j)$, for $k \in \{\mathrm{U}, \mathrm{D}\}$. Given the concatenated data $\boldsymbol{C} = [\boldsymbol{C}^{\mathrm{U}}; \boldsymbol{C}^{\mathrm{D}}] \in \mathbb{R}^{L \times N}$, with $L = L^{\mathrm{U}} + L^{\mathrm{D}}$, we aim to construct factor data $\boldsymbol{\Xi} = (\Xi_{li}) \in \mathbb{R}^{L \times d}$, with $\Xi_{li} = \xi_{i,t_l}$. In particular, we represent $\boldsymbol{C}$, with residuals $\boldsymbol{\Upsilon} \in \mathbb{R}^{L \times N}$, by

$$\boldsymbol{C} = \mathbf{1}_L \otimes \boldsymbol{G}_0 + \boldsymbol{\Xi}\boldsymbol{G} + \boldsymbol{\Upsilon}, \tag{3.1}$$

where $\mathbf{1}_L$ is an L-vector of ones and $\otimes$ denotes the outer product of two vectors. We build factors to reflect the joint goals of eliminating static and dynamic arbitrage in reconstructed prices and guaranteeing statistical accuracy. In Appendix A we recall the factor-decoding algorithm that is introduced as Algorithm 1 in Cohen *et al* (2021a).

3.2.1 Vega-weighted prices

We call $\boldsymbol{C} - \boldsymbol{\Upsilon}$ the "reconstructed prices" and the Frobenius norm $\|\boldsymbol{\Upsilon}\|_{\mathrm{F}}$ the "reconstruction error". As in-the-money (ITM) and long-dated options have higher prices, the objective of minimizing $\|\boldsymbol{\Upsilon}\|_{\mathrm{F}}$ is naturally more effective in reducing relative

calibration errors for ITM and long-dated option prices than for other option prices. To avoid overfitting to ITM and long-dated option prices, practitioners often weight each option by the reciprocal of its Black–Scholes Vega. In fact, using the approximation $\Delta C \approx \text{Vega} \times \Delta\sigma^{\text{imp}}$, weighting prices by $1/\text{Vega}$ roughly corresponds to measuring reconstruction errors in implied volatilities. To follow this approach, we introduce a constant weight matrix $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_N)$ and construct factors from $\boldsymbol{C\Lambda}$ instead, as the following two optimization problems are equivalent:

$$\min_{\boldsymbol{\Xi},\boldsymbol{G}} \|(\boldsymbol{C} - \mathbf{1}_L \otimes \boldsymbol{G}_0 - \boldsymbol{\Xi}\boldsymbol{G})\boldsymbol{\Lambda}\|_{\text{F}} \Longleftrightarrow \min_{\hat{\boldsymbol{\Xi}},\hat{\boldsymbol{G}}} \|\boldsymbol{C\Lambda} - \mathbf{1}_L \otimes \hat{\boldsymbol{G}}_0 - \hat{\boldsymbol{\Xi}}\,\hat{\boldsymbol{G}}\|_{\text{F}}.$$

We now associate the weight λ with the option Vega. At time t the Black–Scholes Vega for the (τ, m)-option is

$$\mathcal{V}_t(\tau, m) = \frac{\mathrm{d}\tilde{c}_t}{\mathrm{d}\sigma_t^{\text{imp}}}(\sigma_t^{\text{imp}}; \tau, m) = \sqrt{\tau}\phi\left(-\frac{m}{\sigma_t^{\text{imp}}\sqrt{\tau}} + \tfrac{1}{2}\sigma_t^{\text{imp}}\sqrt{\tau}\right) > 0, \qquad (3.2)$$

where $\phi(\cdot)$ is the density of the standard normal distribution. This leads us to set

$$\lambda_j = \frac{1}{\bar{\mathcal{V}}_j} \text{ for each } j, \quad \text{where } \bar{\mathcal{V}}_j = \frac{1}{L}\sum_{l=1}^{L} \mathcal{V}_{t_l}(\tau_j, m_j).$$

REMARK 3.2 Weighting price data by a constant matrix does not change the factor-decoding algorithm, except that the coefficient and constant terms of the static arbitrage constraints (2.5) need to be revised. Specifically, we aim for a new factor representation, $\boldsymbol{\Lambda}\boldsymbol{c}_t = \boldsymbol{G}_0 + \boldsymbol{G}^{\text{T}}\xi_t$. The static arbitrage constraints $\boldsymbol{A}\boldsymbol{c}_t \geqslant \hat{\boldsymbol{b}}$ then yield

$$(\boldsymbol{A}\boldsymbol{\Lambda}^{-1})\boldsymbol{G}^{\text{T}}\xi_t \geqslant \boldsymbol{b}^{\text{new}} := \hat{\boldsymbol{b}} - (\boldsymbol{A}\boldsymbol{\Lambda}^{-1})\boldsymbol{G}_0.$$

Nevertheless, for notational simplicity, we write $\boldsymbol{C} \leftarrow \boldsymbol{C\Lambda}$, $\boldsymbol{A} \leftarrow \boldsymbol{A}\boldsymbol{\Lambda}^{-1}$ and $\boldsymbol{b} \leftarrow \boldsymbol{b}^{\text{new}}$ for the rest of the paper.

3.2.2 *Primary and secondary factors*

While decoding more factors can improve the representation of call prices, it also leads to higher-dimensional models, which may overparameterize the dynamics of noisy factors and lead to overfitting. To overcome these issues, we assume that some, "primary", factors are modeled by the SDE (2.3) and other, "secondary", factors are modeled as mean-reverting Ornstein–Uhlenbeck processes.

We now investigate how many primary and secondary factors are needed for a reasonable representation of the call price data. In Section 4.3 of Cohen *et al* (2021a) we define three metrics to examine the quality of factor reconstruction: the mean absolute percentage error (MAPE), the proportion of dynamic arbitrage (PDA) and the

TABLE 2 MAPE, PDA and PSAS metrics when including different combinations of factors.

Factors	MAPE	PDA	PSAS
Dynamic arbitrage	10.75	2.20	33.74
Dynamic arbitrage + Statistical accuracy	4.61	1.92	8.26
Dynamic arbitrage + Static arbitrage	4.94	1.99	1.29
Statistical accuracy	5.40	13.07	7.13
Statistical accuracy + Dynamic arbitrage	5.40	1.80	8.13
Statistical accuracy + Static arbitrage	4.82	7.83	0.30

All values are given in percent.

proportion of statically arbitrageable samples (PSAS).[5] As discussed in the previous section, we use Vega-weighted versions of these quantities in what follows. Applying the factor-decoding algorithm (Algorithm 1 in Appendix A), we show in Table 2 the three metrics for a few different factors and combinations of up to two factors.

Using only two factors (ie, one statistical accuracy factor and one static arbitrage factor (the last row in the table)), we can represent the whole collection of call prices with reasonable accuracy. Note that using a static arbitrage factor significantly reduces the number of violations of the static arbitrage constraints, as evidenced by the corresponding reduction in the PSAS. Since arbitrageable data samples are invalid inputs for the model inference and need to be thrown away, it is crucial to ensure a low PSAS. The high value of the PDA is of less concern for real data and will be reduced further by including more secondary factors.

REMARK 3.3 We will restrict our attention to this simple three-factor model (ie, two primary factors ξ, in addition to the stock index price), as it also allows us to easily demonstrate qualitative features of the model. However, see Appendix C.1 (available online) for the analysis of a market model with three primary factors. In general, the number of factors needed will depend on the market and data to be modeled, in much the same way as for traditional factor models. For the data considered,

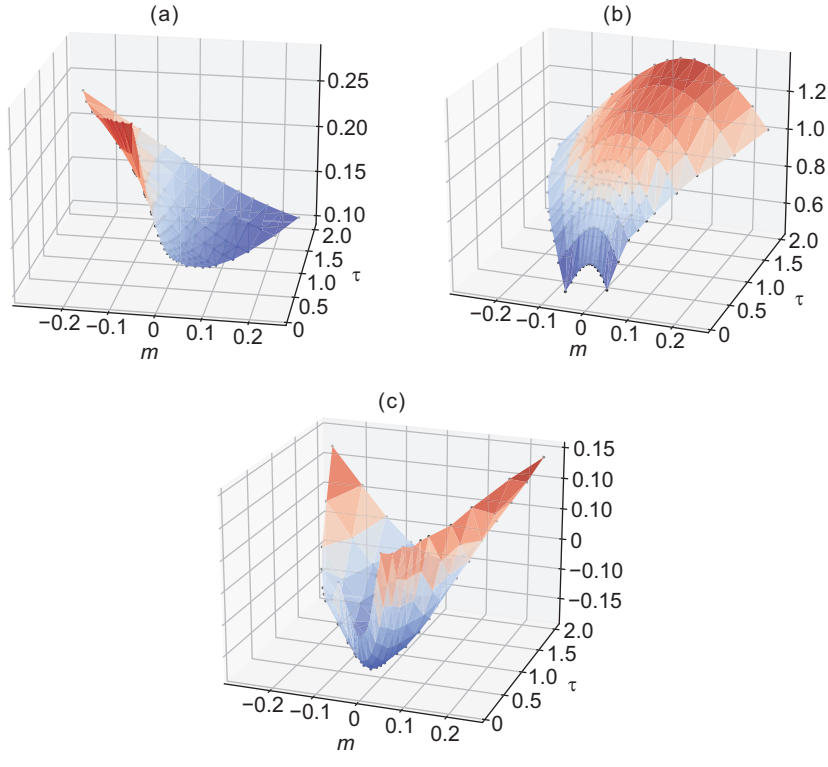
[5] We recall that the MAPE and PSAS metrics are defined by

$$\mathrm{MAPE} = \frac{1}{(L+1)N}\sum_{l=0}^{L}\sum_{j=1}^{N}\frac{|\tilde{c}_{t_l}(\tau_j,m_j)/\bar{\mathcal{V}}_j - G_{0j} - \sum_{i=1}^{d} G_{ij}\xi_{it_l}|}{\tilde{c}_{t_l}(\tau_j,m_j)/\bar{\mathcal{V}}_j},$$

$$\mathrm{PSAS} = 1 - \frac{\sum_{l=0}^{L}\mathbf{1}_{\{\boldsymbol{A}\boldsymbol{\Lambda}^{-1}\boldsymbol{G}^{\mathrm{T}}\xi_{t_l}\geq\boldsymbol{b}\}}}{L+1}.$$

The formula for the PDA is somewhat involved; we refer readers to Section 4.3 of Cohen *et al* (2021a).

FIGURE 2 Price basis functions of the normalized call price surface.



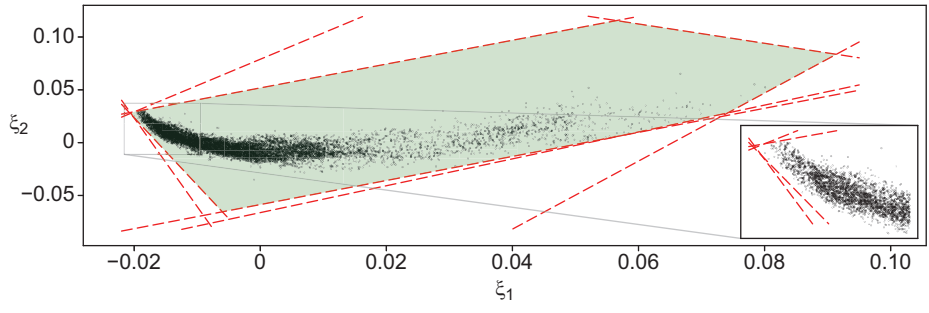
(a) $G_0(\tau, m)$. (b) $G_1(\tau, m)$. (c) $G_2(\tau, m)$.

we find that a small number of factors is sufficient to capture much of the behavior of interest.

We plot the price basis functions of these two factors (denoted G_1 and G_2) and of the constant term of $\tilde{c}$ (denoted G_0) in Figure 2. The points in the liquid lattice (in (τ, m)-coordinates) are also shown on this plot. Here, the real-valued functions G_1 and G_2 are obtained by interpolating the price basis vectors $\boldsymbol{G}_1$ and $\boldsymbol{G}_2$, and G_0 is obtained by interpolating the normalized call prices averaged over time.

Given these factors, we use the linear programming method (Caron *et al* 1989) to eliminate redundant constraints in the system $\boldsymbol{A}\boldsymbol{G}^{\mathrm{T}}\xi \geqslant \boldsymbol{b}$, which is the projection of the original no-arbitrage constraints, constructed in price space, to the $\mathbb{R}^2$ factor space. This results in only eight constraints, which we indicate as red dashed lines in Figure 3. The convex polygonal domain bounded by these constraints (the light-green

FIGURE 3 Trajectory (black dots) of the primary $\mathbb{R}^2$ factors and the corresponding static arbitrage constraints (red dashed lines) projected to the $\mathbb{R}^2$ factor space.

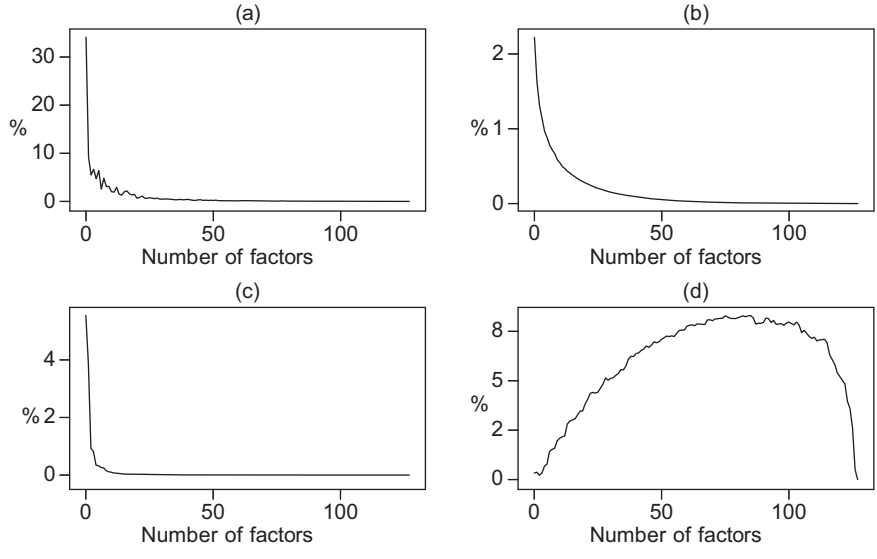


area) is the statically arbitrage-free zone for the factors; that is, provided the factor process remains in this region, we are guaranteed to have no static arbitrage in the reconstructed call prices on our liquid lattice $\mathcal{L}_{\text{liq}}$. Also shown in Figure 3 is the cloud of factor realizations decoded from the market data. We observe that it has a qualitatively similar shape to that in Cohen *et al* (2021a) for simulated prices from a Heston stochastic local volatility model.

To further reduce the factor reconstruction error, we include a few more, secondary factors, whose dynamics are modeled as mean-reverting Ornstein–Uhlenbeck (OU) processes.[6] We take the residuals from the reconstruction of call prices with two primary factors and consecutively decode the secondary factors from the residuals (using principal component analysis), with the objective of maximizing statistical accuracy. In Figure 4 we show how various metrics change with the inclusion of more secondary factors. In part (a) the factor magnitude is defined as the ratio of the min–max range of the corresponding factor to that of the first primary factor, provided that the price basis vectors have a unit norm. It measures each secondary factor's importance in reconstructing option prices relative to the first primary factor. While the MAPE and PDA decrease with inclusion of more secondary factors, the PSAS is not always reduced. For example, inclusion of 13 secondary factors results in an MAPE of 0.48%, a PDA of 0.06% and a PSAS of 2.42%.

[6] An Ornstein–Uhlenbeck model is chosen as it is a simple mean-reverting process, which agrees with what we observe in the decoded paths of option prices.

FIGURE 4 Price reconstruction metrics with the inclusion of more secondary factors.



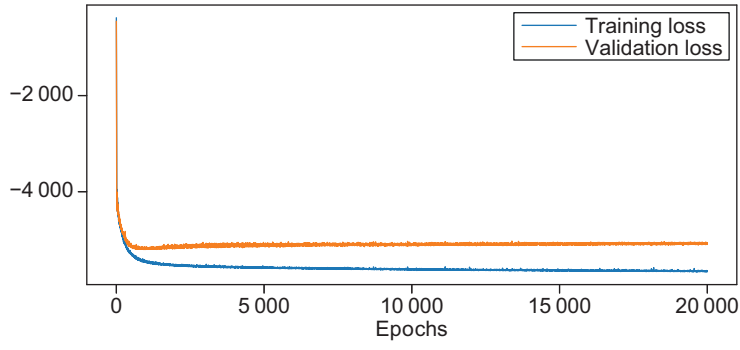
(a) Factor magnitude. (b) MAPE. (c) PDA. (d) PSAS.

REMARK 3.4 The PSAS initially increases with more secondary factors but eventually drops to zero. This is because adding secondary factors leads to small perturbations of the price surface, potentially violating monotonicity and convexity and thus resulting in (small) arbitrages in reconstructed prices. In other words, the OU model for the secondary factors does not necessarily rule out static arbitrage, particularly because their drift and diffusion do not satisfy Friedman and Pinsky's condition. This is less important when choosing the number of secondary factors than it is for primary factors, for several reasons. First, we no longer need to remove arbitrageable data points to estimate these factors' dynamics. Moreover, the magnitudes of secondary factors are much smaller than those of primary ones (Figure 4(a)). In addition, we could always apply the arbitrage-repair algorithm (Cohen *et al* 2020) to perturb our simulated secondary factors in order to restore monotonicity and convexity.

3.3 Factor dynamics estimation

The estimation of the SDE model for primary factors (2.3 b) with static arbitrage constraints has been formulated as an unconstrained deep-learning problem in (Cohen

FIGURE 5 Evolution of training losses and validation losses for the model of the primary factors ξ.



et al 2021a, Section 3). In this study we use the network architecture

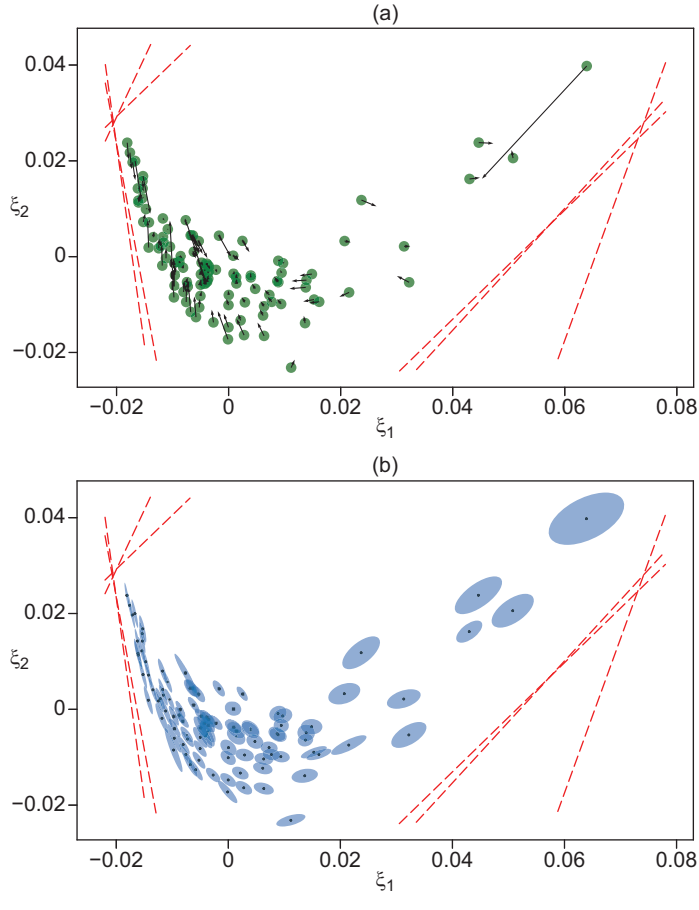
$$\phi^{\xi,\theta} = \mathcal{F}_2 \circ \mathcal{A}_{\mathrm{ReLU}} \circ \mathcal{F}_{256} \circ \mathcal{A}_{\mathrm{ReLU}} \circ \mathcal{F}_{256} \circ \mathcal{A}_{\mathrm{ReLU}} \circ \mathcal{F}_{256}, \tag{3.3}$$

with weight and bias parameters θ, where $\mathcal{F}_x$ is a fully connected layer, or affine transformation, with x units, and $\mathcal{A}_{\mathrm{xxx}}$ is an activation function. Each layer $\mathcal{F}$ is parametric, but we omit the parameters for notational simplicity. In addition, to mitigate overfitting problems, we train both networks with 50% sparsity, meaning that the 50% smallest weights are pruned to zero. We implement and train our model using the standard tools within the `TensorFlow` environment (Abadi *et al* 2015). In Figure 5 we show the evolution of training losses and validation losses over epochs during the training of the model.[7] The loss value rapidly declines for the first few epochs and then gradually converges.

Next, we sample a few factor data points and visualize their drift and diffusion coefficients in Figure 6. As observed in the trajectory of the factor data in Figure 3, the data set is distributed around a lower-dimensional manifold. For samples on the periphery of the data set, the learned drifts tend to point inward to the manifold, while the principal direction of their diffusions tends to align with the closest boundary. This broadly agrees with the phenomena seen in synthetic data in Cohen *et al* (2021a).

[7] The first 90% of the training samples are used for training and the last 10% are reserved as validation data.

FIGURE 6 (a) Drift vectors (arrows) and (b) diffusion matrixes (ellipses, representing the principal components of the diffusion) for some randomly selected factor data points.



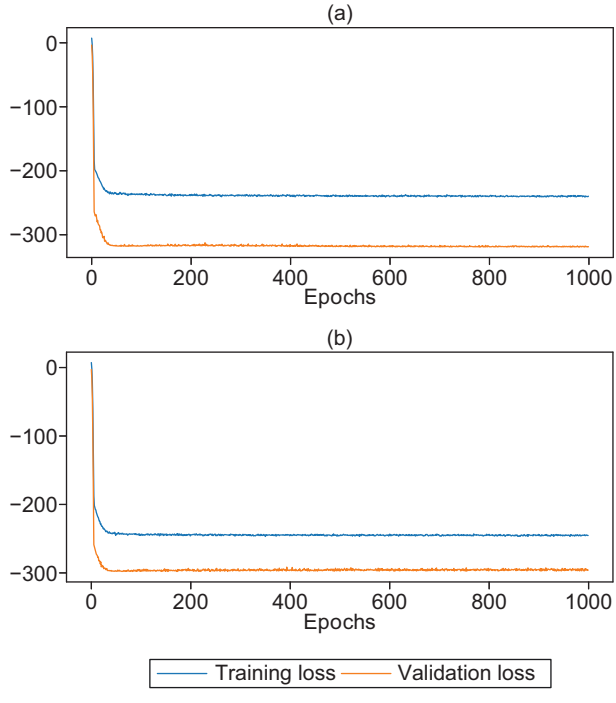
For each of the secondary factors, we discretize the OU model using the Euler–Maruyama scheme, which yields a first-order autoregressive model with normal noises, and we then fit it to the factor time series data using maximum likelihood estimation.

3.4 Underlying index dynamics estimation

Let $X_t = \ln S_t$; then, the model for S (2.3 a) implies, by Itô's lemma,

$$\mathrm{d}X_t = r(\xi_t)\,\mathrm{d}t + \gamma(\xi_t)\,\mathrm{d}W_{0,t}, \quad \text{where } r(\xi_t) = \alpha(\xi_t) - \tfrac{1}{2}\gamma^2(\xi_t).$$

FIGURE 7 Evolution of training losses and validation losses for the models of S.



(a) EURO STOXX 50. (b) DAX.

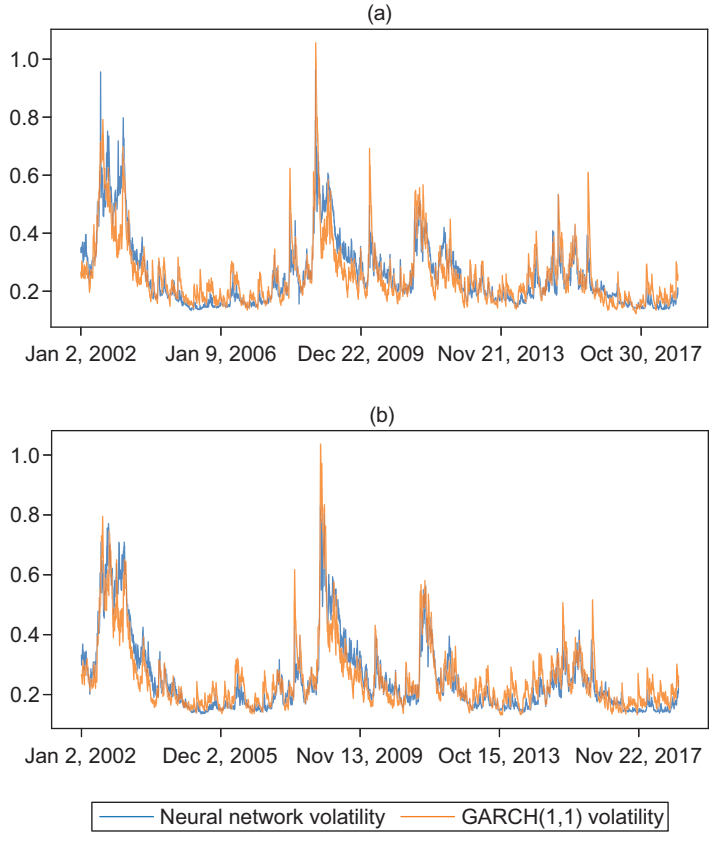
Since drift estimation is well known to be noisy, we assume a constant drift for X_t, ie, $r(\xi_t) \equiv r$. With this assumption, an unbiased estimate for r is

$$\hat{r} = \frac{1}{\delta t}\left(\frac{1}{L-1}\sum_{l=1}^{L-1} \ln\left(\frac{S_{t_{l+1}}}{S_{t_l}}\right)\right) = \frac{1}{T}\ln\left(\frac{S_T}{S_0}\right). \tag{3.4}$$

Hence, for each underlying equity index, we first estimate a constant drift using historical log returns and then train a neural-SDE model with only its diffusion represented by a neural network. Specifically, for our training samples (from January 1, 2002 to December 31, 2018), $\hat{r}^{\mathrm{U}} = -0.027$, $\hat{r}^{\mathrm{D}} = 0.071$. Compared with the model for the primary factors, we use a smaller network $\phi^{S,\theta}: \mathbb{R}^2 \to \mathbb{R}$ for the neural-SDE model of S:

$$\phi^{S,\theta} = \mathcal{F}_2 \circ \mathcal{A}_{\mathrm{ReLU}} \circ \mathcal{F}_{128} \circ \mathcal{A}_{\mathrm{ReLU}} \circ \mathcal{F}_{128} \circ \mathcal{A}_{\mathrm{ReLU}} \circ \mathcal{F}_{128}. \tag{3.5}$$

FIGURE 8 Comparison of the volatility estimates for the options' underlying indexes.

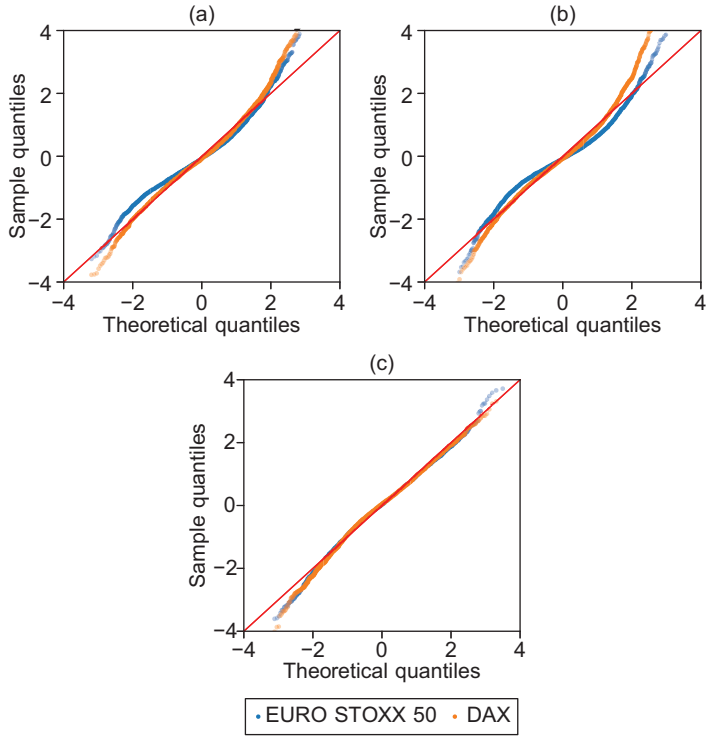


(a) EURO STOXX 50. (b) DAX.

In Figure 7 we show the evolution of training losses and validation losses over epochs during the training of the models for both the EURO STOXX 50 and the DAX. The loss value quickly drops during the first 10 epochs then slowly declines and converges within 50 epochs.

To examine how well the diffusion of S is modeled by the neural networks, we use a generalized autoregressive conditional heteroscedasticity (GARCH) model (specifically, GARCH$(1, 1)$) to estimate empirical volatilities and compare them with the in-sample volatilities computed from the trained neural networks. As evidenced in Figure 8, the neural network models produce similar volatility dynamics to the GARCH models.

FIGURE 9 Q–Q plots of historical in-sample model residuals.



(a) Residuals of ξ_1. (b) Residuals of ξ_2. (c) Residuals of $\ln(S)$.

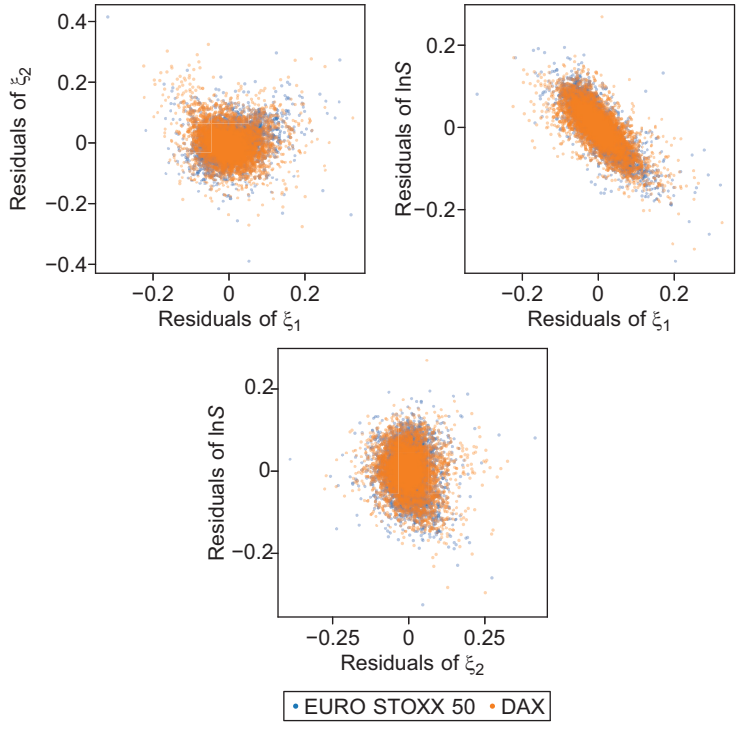
3.5 Analysis of historical residuals

Historical residuals of the trained neural-SDE models (2.3) at time t_l, for $l = 1, \ldots, L-1$, are computed by

$$\hat{Z}^{\xi}_{t_l} = \sigma^{-1}(\xi_{t_l})(\xi_{t_{l+1}} - \xi_{t_l} - \mu(\xi_{t_l})\,\delta t), \qquad \hat{Z}^{S}_{t_l} = \gamma^{-1}(\xi_{t_l})\left(\ln\frac{S_{t_{l+1}}}{S_{t_l}} - \hat{r}\,\delta t\right).$$

Our approximate likelihood corresponds to an assumption that (Z^S_t, Z^{ξ}_t) are independent standard normal white noises. Therefore, it is worth checking whether the estimated residuals $(\hat{Z}^S_t, \hat{Z}^{\xi}_t)$ appear to be independent standard normals within our training data. In Figure 9 we show the normal quantile–quantile (Q–Q) plots of the empirical marginal distributions of the residuals. For the primary factors, the residuals show mildly fat tails. For the underlying stock indexes, the residuals appear slightly left-skewed, implying higher-than-expected probabilities for large losses.

FIGURE 10 Scatterplots of historical in-sample model residuals.



To investigate dependence between the historical residuals for each factor, we show in Figure 10 the scatterplots of each pair of residuals. The residuals among risk factors ξ_i look fairly uncorrelated, indicating that the trained neural nets have well captured the factor covariances over time. However, there is evident negative correlation between the residuals of $\ln S$ and ξ_1, coinciding with the commonly seen leverage effect (Black 1976). In fact, we could capture the leverage effect using the neural-SDE model if a full covariance matrix for $(\ln S, \xi)$ were included in the model, which would consequently produce uncorrelated residuals (see Appendix C.2 online).

3.6 Out-of-sample simulation of factors and the implied volatility surfaces

We now assess the trained model's ability to simulate time series data that are like the real input data. We take the trained model and simulate sample paths using a tamed

Euler scheme (see Hutzenthaler *et al* 2012),[8] which is given by

$$\left.\begin{aligned} \xi_{t+\delta t} &= \xi_t + \frac{\mu(\xi_t)}{1+|\mu(\xi_t)|\sqrt{\delta t}}\delta t + \frac{\sigma(\xi_t)}{1+\|\sigma(\xi_t)\|\sqrt{\delta t}} Z_t^{\xi}, \\ \ln\frac{S_{t+\delta t}}{S_t} &= \hat{r}\,\delta t + \frac{\gamma(\xi_t)}{1+|\gamma(\xi_t)|\sqrt{\delta t}} Z_t^{S}, \end{aligned}\right\} \tag{3.6}$$

where the values of μ, σ and γ are approximated by the trained neural networks. Rather than generating the innovations (Z_t^S, Z_t^{ξ}) from normal distributions, we randomly sample them from historical residuals of the trained models. There are two major advantages of drawing innovations from historical residuals rather than normal distributions. First, historical residuals have fatter tails than normal ones (see Figure 9). Second, the joint historical residuals implicitly preserve the (potentially higher-order) dependence of (S, ξ), which the covariance terms fail to capture (see Figure 10).

We show the time series of ξ in parts (a)–(d) of Figure 11. We see that the dependence structure between ξ_1 and ξ_2 is captured well by the scatter plot in part (e). In addition, the simulated factors remain within the no-arbitrage region, owing to the hard constraints imposed on the drift and diffusion functions. We also simulate secondary factors from their fitted OU processes and check the similarity in their distributions (Figure 12).

The trained model is also capable of simulating a variety of realistic patterns of the implied volatility surface. This is demonstrated in Figure 13, in which we pick a few implied volatility surface patterns observed in historical data and find those that are closest (in the sense of minimizing the ℓ^2 distance for ξ) in the simulated sample path.

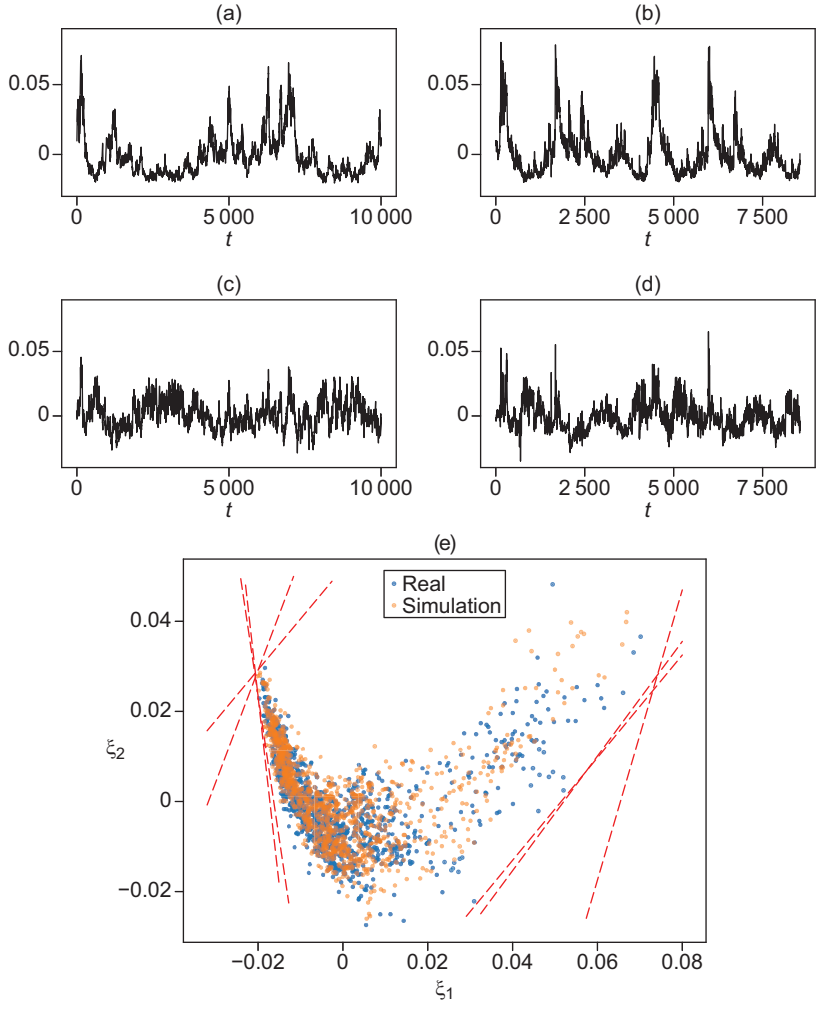
3.7 VIX-like volatility index simulation

We follow the VIX calculation methodology in Chicago Board Options Exchange (2019) to compute a volatility index from the real and simulated option prices. Specifically, the VIX is a linear combination of out-of-the-money call and put option prices, which can be further written as a linear combination of call prices only, provided that put–call parity holds under no-arbitrage.

In Figure 14 we show the real historical joint distribution of the volatility index and the log return of S in comparison with the distribution from one simulation using the trained market model. For the simulated volatility index, both its marginal distribution and the joint distribution with the log return of S look reasonably similar

[8] We include the taming method simply to ensure stability of simulations if our neural networks were to produce unusually large values for drifts or volatilities.

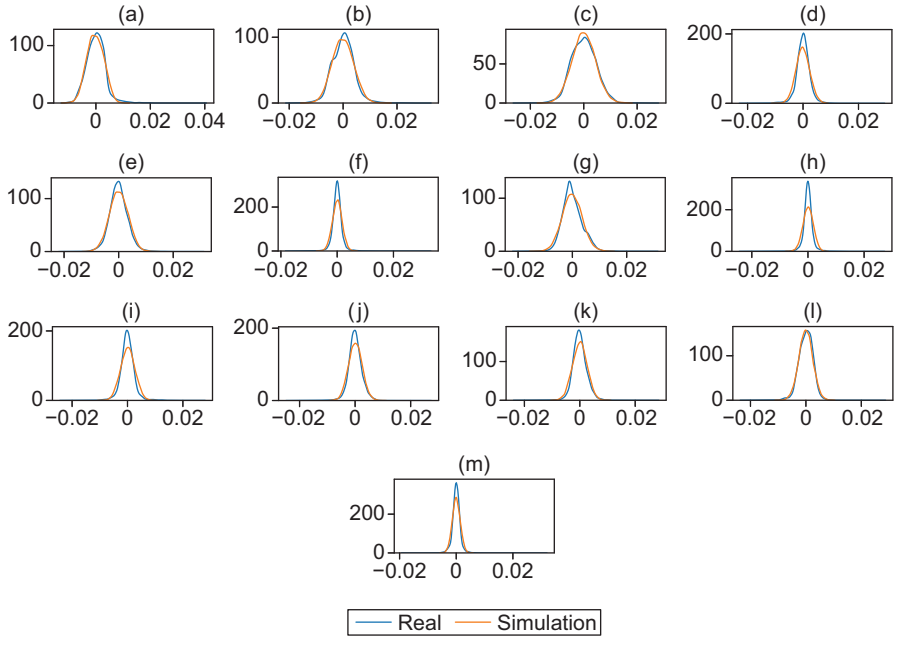
FIGURE 11 Simulation of primary factors from the trained neural-SDE model compared with the real data.



(a) Simulated ξ_{1t}. (b) Real ξ_{1t}. (c) Simulated ξ_{2t}. (d) Real ξ_{2t}. (e) Trajectory of ξ (primary).

to those of the real data. This demonstrates that our model captures the dependence structure between the volatility index and the underlying S. Looking at the simulated time series of the volatility index and the log return of S, we see several occurrences of volatility clustering in the return series, which always coincide with high values for the volatility index.

FIGURE 12 Distribution of OU-simulated and real secondary factors.



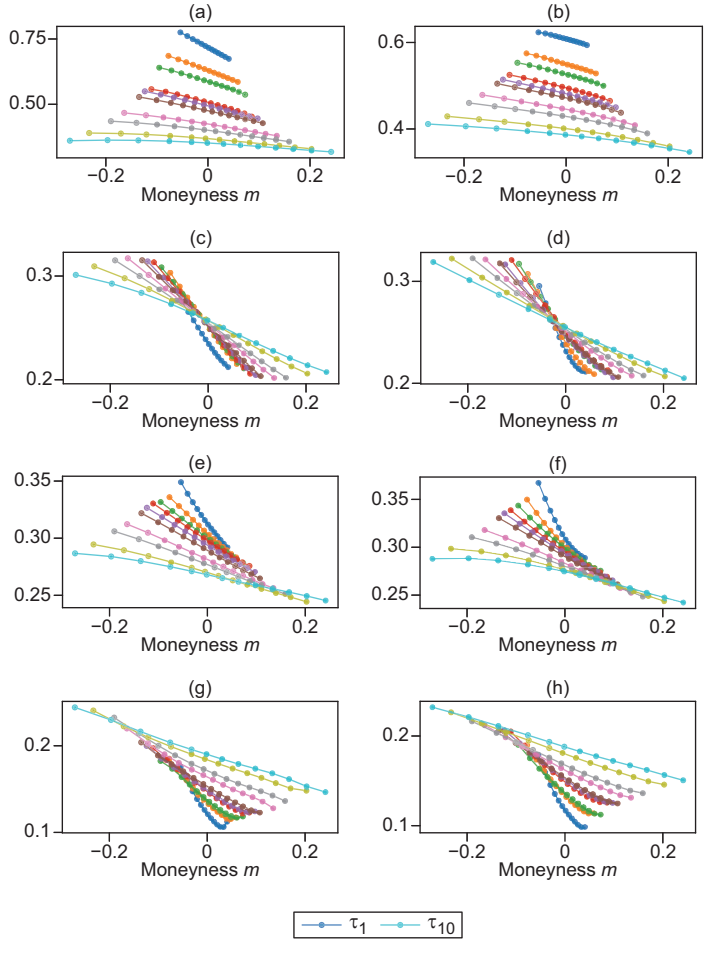
(a) ξ_3. (b) ξ_4. (c) ξ_5. (d) ξ_6. (e) ξ_7. (f) ξ_8. (g) ξ_9. (h) ξ_{10}. (i) ξ_{11}. (j) ξ_{12}. (k) ξ_{13}. (l) ξ_{14}. (m) ξ_{15}.

4 RISK MANAGEMENT

As our market models are defined and trained under the real-world measure, a natural application is simulating risk scenarios for option portfolios. In this section we explain how this is done and compare our approach's performance with that of the filtered historical simulation (FHS) method (Gurrola-Perez and Murphy 2015), which is a standard approach widely used for managing the risk of futures and options by mainstream central counterparties (for example, CME Group,[9] Eurex Clearing (2022) and LCH[10]).

[9] URL: www.cmegroup.com/clearing/risk-management/futures-and-options-margin-model.html.
[10] URL: www.lch.com/risk-management/risk-management-ltd.

FIGURE 13 Examples of real and simulated implied volatility surfaces.



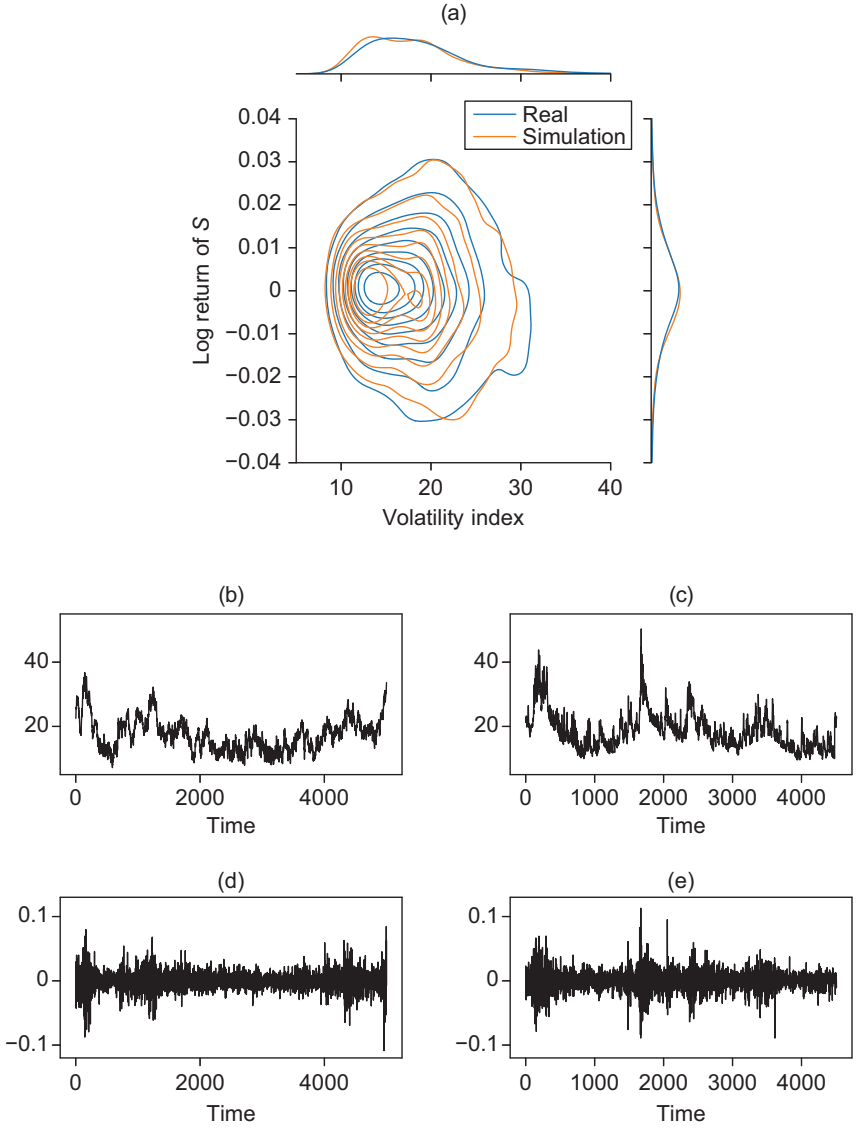
(a) EURO STOXX 50, October 17, 2008 (real). (b) Observed simulation surfaces similar to (a). (c) EURO STOXX 50, July 29, 2010 (real). (d) Observed simulation surfaces similar to (c). (e) DAX, May 29, 2002 (real). (f) Observed simulation surfaces similar to (e). (g) DAX, July 18, 2017 (real). (h) Observed simulation surfaces similar to (g). Times-to-expiry range from $\tau_1 = 30$ days to $\tau_{10} = 730$ days.

4.1 Measuring risk

Consider a portfolio Π of N call options at time t with market value

$$\Pi_t = \sum_{j=1}^{N} w_j C_t(T_j, K_j),$$

FIGURE 14 Simulation of the VIX-like volatility index and the log return of S (EURO STOXX 50) compared with the real data.



(a) Joint distribution of the log return of S and the volatility index. (b) Volatility index (simulation). (c) Volatility index (real). (d) Log return of S (simulation). (e) Log return of S (real).

where $w_j \neq 0$ denotes the weight of the jth call option in the portfolio. The profit and loss (PnL) of the portfolio at time $t + \Delta t$, for a given time horizon Δt, is then

$$\begin{aligned}\mathrm{PnL}_{\Delta t}(\Pi_t) = \Pi_{t+\Delta t} - \Pi_t &= \sum_{j=1}^{N} w_j (C_{t+\Delta t}(T_j, K_j) - C_t(T_j, K_j)) \\ &= \sum_{j=1}^{N} w_j \mathrm{PnL}_{\Delta t}(C_t(T_j, K_j)).\end{aligned}$$

For a chosen risk measure $\varrho(\cdot)$, time t and horizon Δt, we evaluate the risk of the portfolio, $\varrho(\mathrm{PnL}_{\Delta t}(\Pi_t))$, in three main steps.

STEP 1 Identify risk factors Θ and a value function V such that $\Pi_t \approx V(t, \Theta_t)$.

STEP 2 Simulate $\Theta^{(m)}_{t+\Delta t} \mid \Theta_t$ based on some dynamic model for Θ_t for $m = 1, \ldots, M$, and then compute $\Pi^{(m)}_{t+\Delta t} \approx V(t + \Delta t, \Theta^{(m)}_{t+\Delta t})$. Each $\Pi^{(m)}_{t+\Delta t}$ is called a "risk scenario".

STEP 3 For the mth risk scenario, calculate the scenario PnL as $V(t + \Delta t, \Theta^{(m)}_{t+\Delta t}) - \Pi_t$, which is assumed to be a realization of $\mathrm{PnL}_{\Delta t}(\Pi_t) \mid \Theta_t$. The M scenario PnLs therefore give an estimate of the empirical distribution of $\mathrm{PnL}_{\Delta t}(\Pi_t) \mid \Theta_t$ (ie, the distribution on which the risk measure ϱ will be evaluated).[11]

4.2 Risk simulation using the market models

Our market models characterize option risk in terms of $\Theta = (S, \xi)$, for which the joint dynamics are given as the SDE system (2.3). After an offline training of the SDE model, we discretize it and simulate the factors, which are subsequently translated to arbitrage-free option prices through simple linear transformations.

More specifically, suppose we want to simulate prices for the (T, K)-call-option at time $t + \Delta t$. Then, the simulation consists of the following steps.

STEP 1 (Decode factors at time t) We collect prices of call options on the liquid lattice $\mathcal{L}_{\mathrm{liq}}$ and normalize the prices using the time-t zero curve and futures curve. Given the price basis $\boldsymbol{G}^{\mathrm{T}}$, we solve for

$$\xi_t = (\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}})^{-1}\boldsymbol{G}(\boldsymbol{c}_t - \boldsymbol{G}_0^{\mathrm{T}}).$$

[11] In general, the conditional distribution of the PnL is not available in closed form, otherwise an analytical formula for the risk measure could be derived without simulation.

STEP 2 (Simulate factors at time $t + \Delta t$) Given the SDE model (2.3), for which the coefficients are estimated by neural networks, we simulate $S^{(m)}_{t+\Delta t}$ and $\xi^{(m)}_{t+\Delta t}$ using the tamed Euler scheme (3.6) iteratively over $\Delta t/\delta t$ time steps for $m = 1, \ldots, M$.

STEP 3 (Compute option prices at time $t + \Delta t$) Under the mth simulated scenario, we compute the normalized prices for the call options on $\mathcal{L}_{\text{liq}}$ as

$$\boldsymbol{c}^{(m)}_{t+\Delta t} = \boldsymbol{G}_0^{\mathrm{T}} + \boldsymbol{G}^{\mathrm{T}} \xi^{(m)}_{t+\Delta t}.$$

Let $h(\tau, m; \boldsymbol{c})$ be some $C^{1,2}$ function that interpolates the data $(\mathcal{L}_{\text{liq}}, \boldsymbol{c})$. Then,

$$C^{(m)}_{t+\Delta t}(T, K) = S^{(m)}_{t+\Delta t} h\left(\tau_t - \Delta t, m_t + \ln \frac{S_t}{S^{(m)}_{t+\Delta t}}; \boldsymbol{c}^{(m)}_{t+\Delta t}\right),$$

where $\tau_t = T - t$ and $m_t = \ln(K/F_t(T))$.

REMARK 4.1 Our model is written not in terms of prices C but in terms of normalized prices $\tilde{c}$. This transformation (2.1) involves discounting for both interest rates and dividends. While interest rates and dividends could be included as risk factors, they have only marginal impact on equity index options, especially when the longest expiry we consider is two years. We will therefore restrict our attention to the underlying risk and volatility risk and thus evaluate the risk of the revised option value $C_t = S_t \tilde{c}_t$.

4.3 FHS with Heston models

The historical simulation approach simply uses historical changes of a collection of risk factors $\Theta = (x_1, \ldots, x_d)$ to construct risk scenarios (Barone-Adesi *et al* 1999). In addition, historical changes of risk factors may be filtered (or scaled) based on current volatility, so that the simulated risk scenarios are more responsive to recent market conditions.

Specifically, suppose $C_t(T, K) = U(T - t, K; \Theta_t)$ and we want to simulate prices at $t + \Delta t$. To do so, there are two steps.

STEP 1 Collect historical data of Δt-returns r_j for the jth risk factor x_j, for $j = 1, \ldots, d$, over the last M time steps: $r_j(t_m)$ for $m = 1, \ldots, M$. Let $\sigma_j(t)$ be some realized volatility estimate for the jth risk factor at t. We then define

$$r^{(m)}_j(t + \Delta t) = r_j(t_m) \frac{\sigma_j(t)}{\sigma_j(t_m)} \tag{4.1}$$

TABLE 3 Number of tested portfolios of various types.

Delta-exposed			Delta-hedged				
Outright	Delta spread	Risk reversal	Delta butterfly	Delta-hedged option	Delta-neutral strangle	Calendar spread	Volatility index
140	420	60	20	60	60	90	2

as the risk factor return under the mth scenario, which leads to the corresponding risk factor scenario $x_j^{(m)}(t + \Delta t)$ by applying the return to $x_j(t)$.[12]

STEP 2 Revalue the call option $C^{(m)}_{t+\Delta t}(T, K) = U(T - t - \Delta t, K; \Theta^{(m)}_{t+\Delta t})$.

4.3.1 Risk factors

We use the parameters of calibrated Heston (1993) models as risk factors. Specifically, we calibrate the Heston parameters $\Theta = (S_0, \nu_0, \theta, k, \sigma, \rho)$ to historical Vega-weighted option prices as detailed in Appendix D (available online), yielding historical observations of Θ_t, which are then used to simulate historical scenarios for option portfolios.

4.4 VaR backtesting analysis

In order to assess the scenarios produced by risk simulation engines for a comprehensive set of risk profiles, we backtest various option portfolios ranging from outright options to heavily hedged portfolios. The list of tested portfolios is given in Table 3, with a detailed description in Appendix E (available online). We consider both long and short positions in each portfolio tested.

We split the historical data into training and testing samples according to Table 1. We use only the training samples for estimating the neural-SDE market model, which will then be used to calculate risks. As a measure of the risk of a portfolio Π, we consider the VaR at a given confidence level α for a given time horizon Δt, defined

[12] We use the log return if a risk factor is positive by definition, in which case

$$x_j^{(m)}(t + \Delta t) = x_j(t) \exp(r_j^{(m)}(t + \Delta t)),$$

and the absolute return otherwise, in which case

$$x_j^{(m)}(t + \Delta t) = x_j(t) + r_j^{(m)}(t + \Delta t).$$

as

$$\mathrm{VaR}_{\alpha,\Delta t}(\Pi_t) = -\sup\{l \in \mathbb{R}\colon \mathbb{P}(\mathrm{PnL}_{\Delta t}(\Pi_t) < l) \leqslant 1-\alpha\}.$$

Given M risk simulation scenarios, the VaR is estimated by

$$\widehat{\mathrm{VaR}}_{\alpha,\Delta t}(\Pi_t) = -\sup\left\{l \in \mathbb{R}\colon \frac{1}{M}\sum_{m=1}^{M} \mathbf{1}_{\{\mathrm{PnL}_{\Delta t}^{(m)}(\Pi_t)<l\}} \leqslant 1-\alpha\right\}.$$

For each testing sample t we compute an ex ante VaR forecast $\widehat{\mathrm{VaR}}_{\alpha,\Delta t}(\Pi_t)$ and observe an ex post realized $\mathrm{PnL}_{\Delta t}(\Pi_t)$. We can therefore define the VaR breach indicator

$$\beta(\Pi_t) = \mathbf{1}_{\{\mathrm{PnL}_{\Delta t}(\Pi_t) < -\widehat{\mathrm{VaR}}_{\alpha,\Delta t}(\Pi_t)\}}.$$

The coverage ratio over the L testing periods is then computed as

$$1 - \frac{1}{L-E}\sum_{l=E+1}^{L} \beta(\Pi_{t_l}).$$

Prevailing VaR model backtesting methodologies mostly focus on the following two categories.

Coverage tests. These assess whether the empirical frequency of VaR breaches is consistent with the quantile of loss that a VaR measure is intended to reflect. We use the widely cited Kupiec (1995) proportion of failures (PF) coverage test, which assumes the null hypothesis

$$(\mathrm{H}_0)\ \ \mathbb{E}[\beta(\Pi_t) \mid \Theta_t] = 1-\alpha$$

and which models the VaR breach events in each testing sample as independent Bernoulli trials. In addition, we also consider the Basel Committee on Banking Supervision's (1996) traffic light approach: based on the number of breaches experienced for the 1-day $\mathrm{VaR}_{0.99}$ results during a 250-day testing window, the VaR measure is categorized as falling into the green zone (four breaches or fewer), yellow zone (between five and nine) or red zone (ten or more).

Independence tests. These assess whether VaR breach events appear to be independent of each other. We use the Christoffersen (1998) test, which considers unusually frequent consecutive breaches. Combining the test statistics of Kupiec's PF test and Christoffersen's independence test yields a conditional coverage mixed test (Christoffersen 1998).

We compute the VaR for horizons Δt of one, two, five and ten days, with two days and five days being typical margin periods of risk that European regulators require for risk monitoring of exchange-traded and over-the-counter derivatives, respectively.

TABLE 4 Summary statistics of coverage and independence tests for one-day VaRs computed by the neural-SDE market model and the FHS approach.

	$\text{VaR}_{0.99}$		$\text{VaR}_{0.95}$	
	Neural-SDE	**FHS**	**Neural-SDE**	**FHS**
Coverage ratio median	0.9921	0.9881	0.9484	0.9563
Coverage ratio mean	0.9887	0.9742	0.9528	0.9321
Kupiec PF (two-sided) (%)	6.92	25.23	32.51	33.69
Kupiec PF (one-sided) (%)	6.92	25.23	9.39	18.08
Christoffersen independence (%)	0.70	11.03	6.10	14.79
Conditional coverage (%)	3.53	24.88	29.23	34.39
Basel Committee traffic light (%)	69.1	62.4	N/A	N/A
	29.7	25.9		
	0.5	10.8		

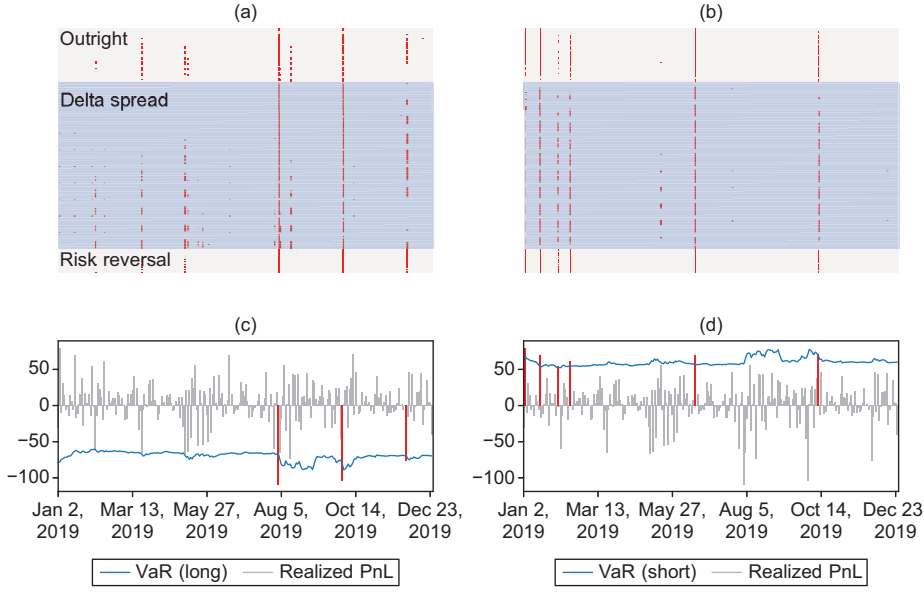
For the four statistical tests – the Kupiec PF test (two-sided), the Kupiec PF test (one-sided), the Christoffersen independence test and the conditional coverage mixed test – the table lists the percentage of the 852 tested option portfolios that fail each test at the 0.05 significance level.

REMARK 4.2 We focus on estimating the VaR of an option portfolio rather than on other measures of risk. The reason for this is simply that we can then use the established backtesting methodologies described above to assess the quality of our model. However, given that our risk estimates are based on simulated PnL values, it would be straightforward to modify our approach to compute any other risk metric that depends only on the distribution of PnL (eg, expected shortfall).

We report the backtesting coverage and independence statistics for one-day VaRs of EURO STOXX 50 options at 0.99 and 0.95 confidence levels in Table 4. We compute a coverage ratio for each of the tested option portfolios, and we calculate its sample median and mean across the portfolios. The neural-SDE VaR model gives a higher mean coverage ratio across option portfolios than the FHS model, and it leads to far fewer portfolios failing the coverage and independence tests. At the 0.95 confidence level, more portfolios fail the two-sided Kupiec PF coverage test than the one-sided version, indicating that both VaR models tend to overestimate risks but the neural-SDE approach is less prone to underestimating risks (as indicated by the one-sided test failure rate of 9.39% versus 18.08%).[13]

[13] Note that, since we reject the null hypotheses in all the coverage and independence tests at the 0.05 significance level, a perfect VaR model is expected to fail for 5% of portfolios.

FIGURE 15 Times series of one-day neural-SDE $\text{VaR}_{0.99}$ breaches (red dots) for delta-exposed option portfolios compared with that for positions in the underlying EURO STOXX 50 index.

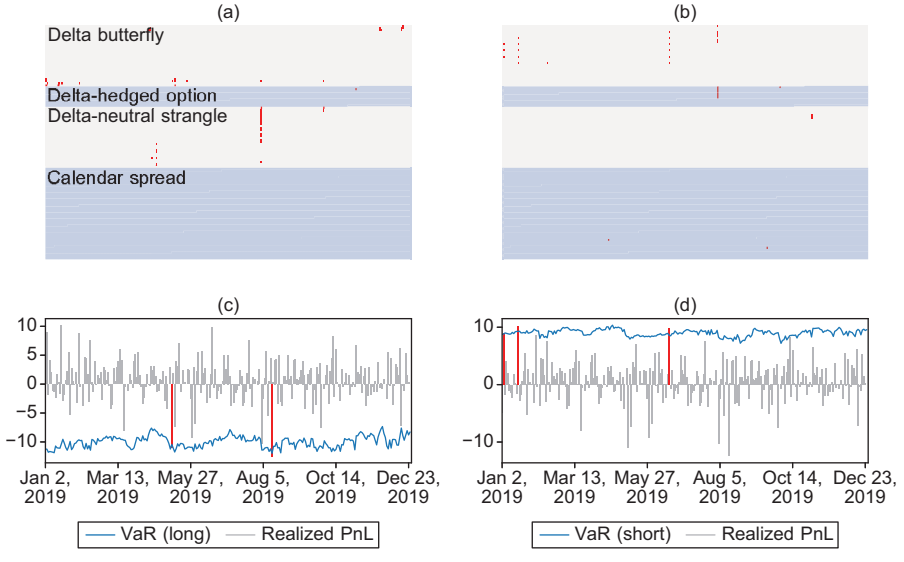


(a) Long delta. (b) Short delta. (c) Long S. (d) Short S.

To investigate VaR coverage performance across different option portfolios, we group portfolios with different primary Greek exposures (ie, delta or Vega) and examine whether their VaR breaches are synchronized during the testing window. In Figure 15 we show the time series of one-day neural-SDE $\text{VaR}_{0.99}$ breaches for primarily delta-exposed option portfolios and we compare them with positions in the underlying index. We see a few points in time when VaR breaches happen simultaneously for multiple option portfolios, and these points correspond to synchronized VaR breaches or near-breaches in holding S.

In Figure 16 we compare VaR breaches for delta-neutral and Vega-exposed portfolios with those of the volatility index portfolio. There are few breaches observed, suggesting that the VaR model is overconservative for these option types. We see some synchronized VaR breaches for Vega-short positions in delta butterflies and the volatility index. We perform a similar analysis for FHS VaRs; as shown in Figure 30 in Appendix F (available online), FHS VaRs severely underestimate risks for most delta-hedged portfolios.

FIGURE 16 Times series of one-day neural-SDE $VaR_{0.99}$ breaches (red dots) for delta-neutral and Vega-exposed option portfolios compared with that for positions in the volatility index portfolio.

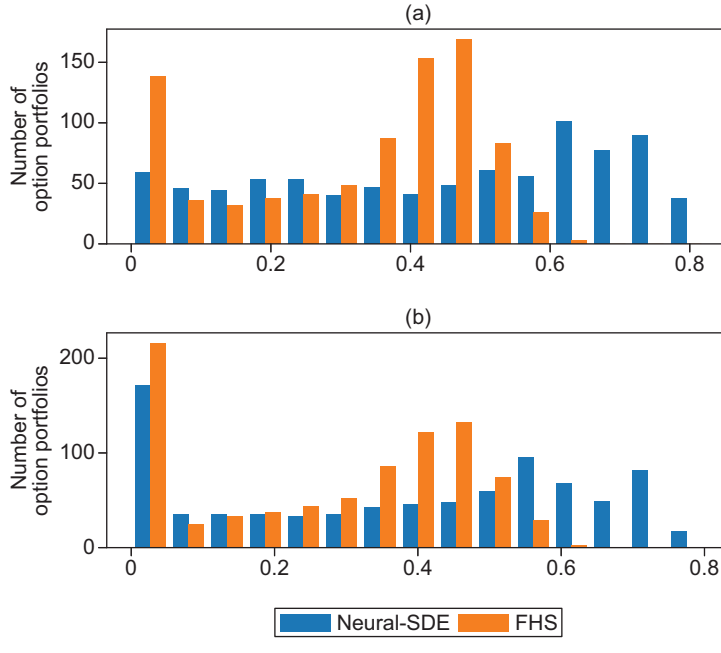


(a) Long Vega. (b) Short Vega. (c) Long volatility index. (d) Short volatility index.

The dynamics of the neural-SDE VaRs shown in Figures 15 and 16 appear to be relatively stable during the testing period, ie, the year 2019. Nevertheless, this does not imply that the neural-SDE VaRs are insensitive to changes in market conditions, as 2019 was simply an uneventful year for global stock markets. To show that the neural-SDE VaRs are adaptive to market changes, we give in-sample neural-SDE VaR backtesting results in Appendix G (available online).

While coverage and independence tests assess whether a VaR calculation approach achieves desirable statistical properties, regulators would also like to avoid overly procyclical VaR models. Measures of procyclicality quantify variations in VaR values over the testing period. Most common risk models are procyclical because for the same portfolio they estimate higher risk in times of market stress and lower risk in calm markets. While procyclicality might indicate that a model is responsive to contemporary market volatility, it may also be an artifact of a poorly calibrated model. Excessive procyclicality can cause liquidity stress, with parties posting margin having to find additional liquid assets, often at the times when it is most difficult for them to do so. To measure procyclicality we use the trough-to-peak ratio proposed by the

FIGURE 17 Distribution of trough-to-peak ratios (for one-day VaRs) over different tested option portfolios.



(a) One-day $\text{VaR}_{0.99}$. (b) One-day $\text{VaR}_{0.95}$.

Bank of England (Murphy *et al* 2014).[14] After calculating the trough-to-peak ratio for each tested portfolio, we get a distribution of the ratio across our evaluated portfolios. In Figure 17 we compare the trough-to-peak distributions computed from the neural-SDE and FHS approaches. The neural-SDE approach tends to produce less procyclical one-day VaRs at both 0.99 and 0.95 confidence levels. In particular, for one-day $\text{VaR}_{0.99}$ there is a substantial reduction in the number of portfolios with a very low trough-to-peak ratio (corresponding to a high level of procyclicality) and an increase in the number of ratios above 0.6. Consistent results for other risk horizons and confidence levels can be seen in Appendix F (online).

[14] Murphy *et al* (2014) use the peak-to-trough ratio (ie, the ratio of the maximum risk of a constant portfolio to the minimum risk over a fixed observation period). Here, we take its reciprocal because for some heavily hedged portfolios the minimum risk can be very close to zero, making the peak-to-trough ratio explode.

TABLE 5 Summary statistics of coverage tests for ten-day VaRs computed by the neural-SDE market model and the FHS approach.

	$\text{VaR}_{0.99}$		$\text{VaR}_{0.95}$	
	Neural-SDE	**FHS**	**Neural-SDE**	**FHS**
Coverage ratio median	1.0000	0.9918	0.9712	0.9637
Coverage ratio mean	0.9963	0.9800	0.9584	0.9464
Kupiec PF (two-sided) (%)	4.10	17.92	51.52	37.12
Kupiec PF (one-sided) (%)	4.10	17.92	10.07	14.05

TABLE 6 Average and standard deviation of the elapsed time for simulating one risk scenario.

	Simulation and revaluation		**Arbitrage repair**	
	Neural-SDE	**FHS (Heston)**	**Neural-SDE**	**FHS (Heston)**
Expected time (ms)	5.0	147.3	14.3	N/A
Standard deviation (ms)	0.3	58.7	7.1	N/A

More backtesting results for two-day, five-day and ten-day VaRs as well as for DAX options are reported in Appendix F (online), but we also briefly summarize the backtesting results for ten-day VaRs of EURO STOXX 50 options in Table 5.[15] As with the one-day VaRs, the neural-SDE approach leads to fewer portfolios failing the one-sided coverage test, the independence test and the conditional coverage test (the null hypotheses are rejected at the 0.05 significance level for all tests). This indicates that the neural-SDE approach has consistently superior performance for long-horizon 0.99-confidence VaR calculation. For 0.95-confidence VaRs, the neural-SDE approach is more conservative than the FHS approach, and the results are more mixed than in the one-day case.

4.5 Computation time comparison

We carried out the backtesting analysis on a 2.4 GHz eight-core Intel Core i9-9980HK central processing unit with 32 GB random-access memory; the average elapsed time for simulating one risk scenario per backtested day is summarized in Table 6.

[15] Given that we consider breaches in overlapping ten-day periods, the Christoffersen independence test is not applicable.

In the FHS approach, evaluating option portfolios with simulated risk factors (ie, the Heston parameters) is very time-consuming. We perform this by calling the efficiently implemented MATLAB function `optByHestonNI` (MathWorks 2022), which gives the option price from the Heston model by using numerical integration. In contrast, the neural-SDE approach evaluates option portfolios through a simple linear combination of simulated risk factors. This is why the time for simulation and revaluation is reduced from 147.3 ms to 5.0 ms. The neural-SDE approach does have the additional step of detecting and repairing arbitrage in the simulated option prices, but this is computationally cheap due to its formulation in Cohen *et al* (2020) as an efficient linear programming problem.

5 CONCLUSIONS AND FURTHER EXTENSIONS

We investigated neural-SDE market models using historical price data of EURO STOXX 50 and DAX index options, the two most traded equity options listed on Eurex. We assessed the in-sample goodness-of-fit by investigating the neural network training loss convergence, the post-fitting historical residuals and the comparison between the volatility estimates for $\ln S$ and the benchmark GARCH$(1, 1)$ estimates. We then verified that our trained model can forward-simulate long trajectories of option price data (or implied volatility data) that have similar marginal and joint distributions to historical data. By backtesting the option portfolio VaR generated from the trained neural-SDE market model in an out-of-sample setting, we found that the model beats the traditional filtered historical approaches, with statistically more efficient coverage, less procyclicality and less computational effort – and this is consistently true for VaRs computed with various risk horizons and confidence levels.

While the neural-SDE market model has shown success in simulation and risk management, there remain directions in which the current framework could be extended to a wider range of applications. For example, a simple and straightforward extension would be to examine other risk measures, such as expected shortfall. Further, in order to model option types other than European vanillas, it would be necessary to derive arbitrage constraints for those options, or to modify the approach accordingly.

DECLARATION OF INTEREST

The authors report no conflicts of interest. The authors alone are responsible for the content and writing of the paper.

ACKNOWLEDGEMENTS

This publication is based on work supported by the Engineering and Physical Sciences Research Council (EPSRC) Centre for Doctoral Training in Industrially Focused Mathematical Modeling (grant EP/L015803/1) in collaboration with CME Group. Samuel Cohen and Christoph Reisinger acknowledge the support of the Oxford-Man Institute for Quantitative Finance. Samuel Cohen also acknowledges the support of the Alan Turing Institute and the UK Research and Innovation Prosperity Partnership Scheme (Framework for Responsible Adoption of Artificial Intelligence in the Financial Services Industry (FAIR)) under EPSRC grant EP/V056883/1.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I. J., Harp, A., Irving, G., Isard, M., Jia, Y., Józefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D. G., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P. A., Vanhoucke, V., Vasudevan, V., Viégas, F. B., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: large-scale machine learning on heterogeneous systems. Preprint (arXiv:1603.04467).

Avramidis, A. N., and Wilson, J. R. (1998). Correlation-induction techniques for estimating quantiles in simulation experiments. *Operations Research* **46**(4), 574–591 (https://doi.org/10.1287/opre.46.4.574).

Barone-Adesi, G., and Giannopoulos, K. (2001). Non-parametric VaR techniques: myths and realities. *Economic Notes* **30**(2), 167–181 (https://doi.org/10.1111/j.0391-5026.2001.00052.x).

Barone-Adesi, G., Giannopoulos, K., and Vosper, L. (1999). VaR without correlations for nonlinear portfolios. *Journal of Futures Markets* **19**(5), 583–602 (https://doi.org/bn24q9).

Barone-Adesi, G., Giannopoulos, K., and Vosper, L. (2002). Backtesting derivative portfolios with filtered historical simulation (FHS). *European Financial Management* **8**(1), 31–58 (https://doi.org/10.1111/1468-036X.00175).

Basel Committee on Banking Supervision (1996). Supervisory framework for the use of "backtesting" in conjunction with the internal models approach to market risk capital requirements. Standards Document, January, Bank for International Settlements. URL: www.bis.org/publ/bcbs22.htm.

Bellovin, S. M., Dutta, P. K., and Reitinger, N. (2019). Privacy and synthetic datasets. *Stanford Technology Law Review* **22**(1), 1–52 (https://doi.org/10.31228/osf.io/bfqh3).

Black, F. (1976). Studies of stock market volatility changes. In *Proceedings of the 1976 Meeting of the Business and Economic Statistics Section, American Statistical Association*, pp. 177–181. American Statistical Association, Alexandria, VA.

Britten-Jones, M., and Schaefer, S. M. (1999). Non-linear value-at-risk. *Review of Finance* **2**(2), 161–187 (https://doi.org/10.1023/A:1009779322802).

Buehler, H., Gonon, L., Teichmann, J., and Wood, B. (2019). Deep hedging. *Quantitative Finance* **19**(8), 1271–1291 (https://doi.org/10.1080/14697688.2019.1571683).

Buehler, H., Horvath, B., Lyons, T., Arribas, I. P., and Wood, B. (2020). A data-driven market simulator for small data environments. Working Paper, Social Science Research Network (https://doi.org/10.2139/ssrn.3632431).

Butler, J., and Schachter, B. (1997). Estimating value-at-risk with a precision measure by combining kernel estimation with historical simulation. *Review of Derivatives Research* **1**, 371–390.

Caron, R. J., McDonald, J. F., and Ponic, C. M. (1989). A degenerate extreme point strategy for the classification of linear constraints as redundant or necessary. *Journal of Optimization Theory and Applications* **62**(2), 225–237 (https://doi.org/10.1007/BF00941055).

Chataigner, M., Crépey, S., and Dixon, M. (2020). Deep local volatility. *Risks* **8**(3), Paper 82 (https://doi.org/10.3390/risks8030082).

Chicago Board Options Exchange (2019). CBOE volatility index. White Paper, CBOE. URL: https://cdn.cboe.com/resources/vix/VIX_Methodology.pdf.

Christoffersen, P. (1998). Evaluating interval forecasts. *International Economic Review* **39**(4), 841–862 (https://doi.org/10.2307/2527341).

Cohen, S. N., Reisinger, C., and Wang, S. (2020). Detecting and repairing arbitrage in traded option prices. *Applied Mathematical Finance* **27**(5), 345–373 (https://doi.org/10.1080/1350486X.2020.1846573).

Cohen, S. N., Reisinger, C., and Wang, S. (2021a). Arbitrage-free neural-SDE market models. Working Paper, Social Science Research Network (https://doi.org/10.2139/ssrn.3851998).

Cohen, S. N., Snow, D., and Szpruch, L. (2021b). Black-box model risk in finance. Working Paper, Social Science Research Network (https://doi.org/10.2139/ssrn.3782412).

El-Jahel, L., Perraudin, W., and Sellin, P. (1999). Value at risk for derivatives. *Journal of Derivatives* **6**(3), 7–26 (https://doi.org/10.3905/jod.1999.319116).

Eurex Clearing (2022). Eurex Clearing Prisma: portfolio-based risk management. Report, August, Eurex. URL: https://bit.ly/3vX6FAB.

Fallon, W. (1996). Calculating value-at-risk. Working Paper 96-49, Wharton School, University of Pennsylvania, Philadelphia, PA.

Friedman, A., and Pinsky, M. A. (1973). Asymptotic stability and spiraling properties for solutions of stochastic equations. *Transactions of the American Mathematical Society* **186**, 331–358 (https://doi.org/10.1090/S0002-9947-1973-0329031-5).

Glasserman, P., Heidelberger, P., and Shahabuddin, P. (2000). Variance reduction techniques for estimating value-at-risk. *Management Science* **46**(10), 1349–1364 (https://doi.org/10.1287/mnsc.46.10.1349.12274).

Glasserman, P., Heidelberger, P., and Shahabuddin, P. (2002). Portfolio value-at-risk with heavy-tailed risk factors. *Mathematical Finance* **12**(3), 239–269 (https://doi.org/10.1111/1467-9965.00141).

Gurrola-Perez, P., and Murphy, D. (2015). Filtered historical simulation value-at-risk models and their competitors. Working Paper 525, Bank of England (https://doi.org/10.2139/ssrn.2574769).

Harrison, J. M., and Kreps, D. (1979). Martingales and arbitrage in multiperiod securities markets. *Journal of Economic Theory* **20**(3), 381–408 (https://doi.org/10.1016/0022-0531(79)90043-7).

Heath, D., Jarrow, R., and Morton, A. (1992). Bond pricing and the term structure of interest rates: a new methodology for contingent claims valuation. *Econometrica* **60**(1), 77–105 (https://doi.org/10.2307/2951677).

Henry-Labordere, P. (2019). Generative models for financial data. Working Paper, Social Science Research Network (https://doi.org/10.2139/ssrn.3408007).

Hesterberg, T. C., and Nelson, B. L. (1998). Control variates for probability and quantile estimation. *Management Science* **44**(9), 1295–1312 (https://doi.org/10.1287/mnsc.44.9.1295).

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies* **6**(2), 327–343 (https://doi.org/10.1093/rfs/6.2.327).

Hsu, J. C., and Nelson, B. L. (1990). Control variates for quantile estimation. *Management Science* **36**(7), 8350–8385 (https://doi.org/10.1287/mnsc.36.7.835).

Hutzenthaler, M., Jentzen, A., and Kloeden, P. E. (2012). Strong convergence of an explicit numerical method for SDEs with nonglobally Lipschitz continuous coefficients. *Annals of Applied Probability* **22**(4), 1611–1641 (https://doi.org/10.1214/11-AAP803).

Jex, M., Henderson, R., and Wang, D. (1999). Pricing exotics under the smile. *Risk* **12**(11), 72–75. URL: www.risk.net/infrastructure/1530288/pricing-exotics-under-smile.

JP Morgan–Reuters (1996). RiskMetrics: Technical Document, 4th edn. Reuters Limited and Morgan Guaranty Trust Company of New York. URL: www.msci.com/documents/10199/5915b101-4206-4ba0-aee2-3449d5c7e95a.

Koshiyama, A., Firoozye, N., and Treleaven, P. (2021). Generative adversarial networks for financial trading strategies fine-tuning and combination. *Quantitative Finance* **21**(5), 797–813 (https://doi.org/10.1080/14697688.2020.1790635).

Kupiec, P. H. (1995). Techniques for verifying the accuracy of risk measurement models. *Journal of Derivatives* **3**(2), 73–84 (https://doi.org/10.3905/jod.1995.407942).

MathWorks (2022). optByHestonNI. Software Package, Version R2022b. URL: https://uk.mathworks.com/help/fininst/optbyhestonni.html.

Murphy, D., Vasios, M., and Vause, N. (2014). An investigation into the procyclicality of risk-based initial margin models. Financial Stability Paper 29, Bank of England (https://doi.org/10.2139/ssrn.2437916).

Ni, H., Szpruch, L., Wiese, M., Liao, S., and Xiao, B. (2020). Conditional Sig-Wasserstein GANs for time series generation. Working Paper, Social Science Research Network (https://doi.org/10.2139/ssrn.3623086).

OptionMetrics (2020). IvyDB Europe reference manual. Version 2.4. URL: https://whr.tn/3ZtQ05f.

Ritter, G., and Kolm, P. N. (2019). Dynamic replication and hedging: a reinforcement learning approach. *Journal of Financial Data Science* **1**(1), 159–171 (https://doi.org/10.3905/jfds.2019.1.1.159).

Sun, L., and Hong, L. J. (2010). Asymptotic representations for importance-sampling estimators of value-at-risk and conditional value-at-risk. *Operations Research Letters* **38**(4), 246–251 (https://doi.org/10.1016/j.orl.2010.02.007).

Takahashi, S., Chen, Y., and Tanaka-Ishii, K. (2019). Modeling financial time-series with generative adversarial networks. *Physica* A **527**, Paper 121261 (https://doi.org/10.1016/j.physa.2019.121261).

Wiese, M., Bai, L., Wood, B., and Buehler, H. (2019). Deep hedging: learning to simulate equity option markets. Working Paper, Social Science Research Network (https://doi.org/10.2139/ssrn.3470756).

Wiese, M., Wood, B., Pachoud, A., Korn, R., Buehler, H., Murray, P., and Bai, L. (2021). Multi-asset spot and option market simulation. Working Paper, Social Science Research Network (https://doi.org/10.2139/ssrn.3980817).

Wissel, J. S. (2008). Arbitrage-free market models for liquid options. PhD Thesis, ETH Zürich.