Deep Surrogates for Finance: With an Application to Option Pricing

Hui Chen Antoine Didisheim Simon Scheidegger*

October 27, 2023

Abstract

We introduce "deep surrogates" – high-precision approximations of structural models based on deep neural networks, which speed up model evaluation and estimation by orders of magnitude and allow for various compute-intensive applications that were previously infeasible. As an application, we build a deep surrogate for a high-dimensional workhorse option pricing model. The surrogate enables us to re-estimate the model at high frequency to construct an option-implied tail risk measure, which is highly predictive of future market crashes. It also enables us to systematically examine the model's out-of-sample performances, which reveals the tradeoffs between structural and reduced-form approaches for option pricing. Moreover, we construct a measure for the degree of parameter instability and connect it to option market illiquidity in the data. Finally, we use the surrogate to construct conditional distributions of option returns, which is useful for risk management and provides a new way to test the model.

Keywords: surrogate, deep neural network, tail risk index, parameter instability, illiquidity, distribution of option return

*Chen: MIT Sloan and NBER. Email: huichen@mit.edu. Didisheim: Department of Finance, HEC Lausanne, University of Lausanne and Swiss Finance Institute. Email: antoine.didisheim@unil. Scheidegger: Department of Economics, HEC Lausanne, University of Lausanne and Enterprise for Society. Email: simon.scheidegger@unil. The trained surrogate for this paper is available at: https://github.com/DeepSurrogate/OptionPricing.

1 Introduction

Driven by advances in theory and the availability of "big data," contemporary models in economics are becoming increasingly complex. State-of-the-art structural models often impose a substantial roadblock to researchers as they increasingly suffer from the curse of dimensionality, that is, the computational burden grows exponentially with every additional degree of freedom. The problem becomes even more challenging for analyses that require a large number of model evaluations, for example, when performing out-of-sample analysis of a DSGE model or identifying the heterogeneity in a large cross section of households through a life-cycle model.

To tackle these challenges, we introduce deep surrogates, a new framework to evaluate and estimate structural models. Conceptually, our method re-purposes the concept of surrogate models commonly applied in physics and engineering for the context of economic models. By taking advantage of the strong approximating power of deep neural networks (DNN), one can create high-precision, cheap-to-evaluate surrogates for high-dimensional models. The surrogate takes the same input from the original model, including both the state variables and model parameters, and produces the same output at a significantly lower computational cost. Simply put, the surrogate is a high-dimensional look-up table. Once built, it is fast to use, easy to share and build on, and avoids the cost and uncertainty of individual replications.

To build a deep surrogate for a structural model, we first generate a sufficiently representative training sample by evaluating the original model and collecting the relevant output at different locations in the input space, and then train a neural network to minimize the fitting errors in the training sample. Generating a training sample of sufficient size to ensure approximation accuracy across the input space is the main bottleneck for building the surrogate. To improve the efficiency of surrogate building, we propose an iterative procedure that alternates between surrogate training and validation. Both the training sample size and network complexity will grow in the process until the desired level of approximation accuracy is reached.

Deep neural networks are uniquely suitable for our task due to two theoretical reasons. First, the strong approximation power of DNNs is formally established by the universal

approximation theorem (see e.g., Hornik, Stinchcombe, and White, 1989; Hanin, 2019). Second, DNNs have also been shown to be able to achieve high accuracy with a relatively simple network structure (as measured by the total number of trainable parameters) and modest training sample size.[1] Intuitively, provided that the target function is sufficiently smooth and is from a bounded domain, and the training examples are reasonably well scattered across the domain, a neural network is capable of "learning" its features with a relatively small number of examples. For example, for a popular class of affine option pricing models, the training sample needed is shown to only grow polynomially in the dimension of the input (see Grohs et al., 2023; Berner, Grohs, and Jentzen, 2020).[2] This is in clear contrast to traditional grid-based approximation methods, where the required grid points grow exponentially in input dimension. In that case, even when a single function evaluation is relatively inexpensive, approximation of a high-dimensional function in this way can quickly become infeasible.

We apply the deep surrogate methodology to study a workhorse option pricing model, a double-exponential jump-diffusion model with stochastic volatility. Since it is a direct extension of Bates (1996),[3] we refer to it as the "Bates model" for simplicity. The model features 13 states and parameters. Using the iterative training procedure, during which the depth of the network and the size of the training sample grow gradually, we achieve the desired approximation accuracy with a network of 7 hidden layers and nearly 10^6 parameters. The final training sample size is 10^9, which is highly sparse considering the dimensionality of the input space ($d = 13$). We then verify through simulation that the deep surrogate not only prices options and produces the greeks accurately but can also accurately identify the structural parameters in GMM estimations. Moreover, it helps reduce the average time for estimating the model parameters by orders of magnitude.

The deep surrogate enables us to carry out a variety of analyses that would have incurred prohibitive computation costs otherwise. By re-estimating the model parameters daily, we are

[1] See Gühring, Raslan, and Kutyniok (2020) for a recent survey on the large body of work of approximation results for deep neural networks.

[2] Neural networks have also been shown to alleviate or break the curse of dimensionality for compositional or polynomial functions (Bach, 2017; Petersen and Voigtlaender, 2018), functions in the Korobov spaces (Montanelli and Du, 2019), and generalized bandlimited functions (Montanelli, Yang, and Du, 2021).

[3] Bates (1996) considers log-normally distributed jumps.

able to i) construct a high-frequency option-implied tail risk index for the stock market, ii) quantify the time variation in the degree of parameter instability and study its implications for option market liquidity, and iii) systematically examine the out-of-sample performances of the Bates model. Finally, the surrogate enables us to efficiently characterize the model-implied conditional distributions of option returns, which can be tested in the data.

First, through the daily re-estimation of the Bates model, we recover unique information about the stock market as conveyed by the options market. These include the time variation in the correlation between changes in stock price and instantaneous variance, also known as the "leverage effect," as well as the asymmetry between upward and downward jump risks in the market. We construct a daily tail risk index using the estimated parameters, $TailRisk$, which measures the risk-neutral expected loss due to a negative jump in the market index over the next week. We show that $TailRisk$ has strong predictive power for actual tail events in the market. For example, whether defining the tail event as "a cumulative loss 10% or more in the S&P 500 over the next 5 days" or as "a daily drop of 5% or more in the S&P 500 for any of the next 5 days," the Pseudo-R^2 coefficient in a logistic regression almost doubles with $TailRisk$ compared to with the non-parametric option-based left-tail probability measure from Bollerslev, Todorov, and Xu (2015).

Next, we examine parameter stability for the Bates model. Using the statistical test proposed by Andersen, Fusari, and Todorov (2015), we construct a parameter instability measure using the GMM estimators for model parameters at daily frequency. The measure reveals the severity of parameter instability for the Bates model: we reject the hypothesis that the parameter values are the same from two adjacent days 41.6% of the time (at the 1% significance level).

Furthermore, we examine the potential economic repercussions of parameter instability within option pricing models for option market liquidity. When these models become more unstable, the ability of option market makers to accurately hedge the risks of their inventories diminishes, which could in turn diminish their risk appetite and reduce market liquidity. Consistent with this prediction, we find a positive and statistically significant relation between the degree of parameter instability and relative bid-ask spreads for the SPX options. The coefficient for parameter instability remains significantly positive after controlling for contract

fixed effect, contract level abnormal volume, time-to-maturity, and moneyness.

Third, we examine the out-of-sample performance of the Bates model and compare it against a reduced-form benchmark, which models the implied volatility surface using random forests (see, e.g., Breiman, 2001), which we refer to as the "RF model." This comparison is important as it helps us better understand the relative advantages of structural vs. reduced-form approach for option pricing. Conceptually, structural models could help guard against over-fitting by imposing restrictions between the dynamics of option prices and the underlying states, but they likely suffer from misspecification due to a lack of flexibility as well as abstractions from various factors due to tractability concerns. However, it can be difficult to thoroughly examine the out-of-sample performance of a structural model, because one needs to frequently update the parameters (typically by re-estimating them in a moving or expanding window) to capture potential parameter changes while avoiding any look-ahead bias. The deep surrogate substantially alleviates this problem.

Our empirical results confirm an interesting tradeoff between the structural and reduced-form approach for option pricing. The RF model outperforms the Bates model in terms of out-of-sample pricing accuracy for short-dated options (with time-to-maturity of 7 days or less), particularly for deep out-of-the-money puts and calls, but the Bates model outperforms for longer-dated options (with time-to-maturity above 90 days). This variation in performance can be attributed to the different risk exposures and potential of model misspecification for short and long-dated options. Short-dated options are likely more exposed to liquidity and demand factors, while the Bates model is better suited to capture intertemporal risk changes for longer-dated options. Furthermore, we document a strong correlation in pricing errors between the Bates and RF model in the time series. This result suggests that the degree of parameter instability for the Bates model, which is a key driver for its out-of-sample performance, likely reflects a more general notion of model instability for option pricing.

Finally, we apply the surrogate technology to construct the conditional distribution of option returns. Following the methodology of Israelov and Kelly (2017), we estimate a vector autoregressive for the relevant states, construct a distribution of future states via bootstrap, and then use the surrogate to efficiently compute option prices along the different paths. We confront the model-implied conditional return distribution, in particular its quantile forecasts,

with the data. The test helps reveal clear advantages of the Bates model over the Heston model (which does not feature jumps) in matching the return distribution.

It is worth noting that, while we use a relatively large neural network and train it for a large number of epochs, we do not use network shrinkage, dropout, or early stopping (see, e.g., Goodfellow et al. (2016)). Instead, we rely on the recently discovered double descent phenomenon, which refers to the observation that, as model complexity increases, the out-of-sample performance first increases, then decreases, and finally increases again after the in-training sample is fully interpolated (see, e.g., Stephenson and Lee, 2021; Nakkiran et al., 2021). The fact that we know the true data-generating process and can observe the data with relatively small amount of noise (due to numerical errors from solving the economic model) makes the setting even more favorable for using complex models to build surrogates.

Related Literature

Our paper contributes to three strands of the literature: (i) methods for constructing surrogate models in general and their application to high-dimensional models in economics and finance; (ii) applications of deep learning in finance and economics; and (iii) empirical option pricing.

Model estimation, calibration, and uncertainty quantification can be daunting numerical tasks because of the need to perform sometimes hundreds of thousands of model evaluations to obtain converging estimates of the relevant parameters and converging statistics (see, e.g., Fernández-Villaverde, Rubio-Ramírez, and Schorfheide, 2016, among others). To this end, a broad strand of literature in engineering, physics (see, e.g., Tripathy and Bilionis, 2018),[4] but also in finance and economics has long tried to replace expensive model evaluations that suffer from the curse of dimensionality with cheap-to-evaluate surrogate models that mitigate the said curse. Heiss and Winschel (2008) for instance, approximate the likelihood by numerical integration on Smolyak sparse grids, whereas Scheidegger and Treccani (2018)

[4]In physics and engineering, Gaussian processes regression (see, e.g., Williams and Rasmussen, 2006; Tripathy, Bilionis, and Gonzalez, 2016; Chen, Zabaras, and Bilionis, 2015), radial basis functions (Park and Sandberg, 1991), or relevance vector machines (Bilionis and Zabaras, 2012) are often used to build surrogate models. More recently, following the rapid developments in the theory of stochastic optimization and artificial intelligence as well as the advances in computer hardware leading to the widespread availability of graphic processing units (GPUs; see, e.g., Scheidegger et al. (2018); Aldrich et al. (2011), and references therein), researchers have turned their attention towards *deep neural networks* (see, e.g., Liu et al., 2019).

apply adaptive sparse grids to approximate high-dimensional probability density functions (PDFs) in the context of American option pricing. Scheidegger and Bilionis (2019) propose a Gaussian process-based method to carry out uncertainty quantification in the context of discrete-time dynamic stochastic models. Kaji, Manresa, and Pouliot (2020) propose a simulation-based estimation method for structural models in economics using a generative adversarial neural network. Norets (2012) extends the state-space by adding the model parameters as "pseudo-states" to estimate finite-horizon, dynamic discrete choice models.

Secondly, our paper is part of the emergent literature on the applications of deep learning to economics and finance, where shallow and deep neural networks are used, for instance, in the context of asset pricing and econometric tasks (see, e.g., Hutchinson, Lo, and Poggio (1994); Chen and White (1999); Farrell, Liang, and Misra (2021); Chen, Pelger, and Zhu (2023)), and to solve a broad range of high-dimensional dynamic stochastic models in discrete and continuous-time settings, but do not directly deal with estimation.[5] We relate to this strand of literature in that we apply neural networks to tackle problems in finance. In particular, and to the best of our knowledge, we are the first to show that very deep neural networks combined with the *Swish* activation function (Ramachandran, Zoph, and Le, 2017) and double descent phenomenon (see, e.g., Stephenson and Lee, 2021; Nakkiran et al., 2021) are necessary to construct high-dimensional surrogate models to estimate dynamic models in finance. Indeed, very deep models are necessary to replicate complex structural models, and *Swish* activation functions are necessary to obtain a smooth gradient when using the surrogate model on real data.

Thirdly, in the option pricing application, the deep surrogate allows us to re-estimate the structural parameters and hidden states at high frequency efficiently. In recent work, Gao and Pan (2023) construct an option-implied crash index by estimating the model from Pan (2002). They find that changes in their crash index negatively predicts future market returns. Christoffersen and Jacobs (2004) also compare the pricing performance of a reduced-form "Practitioner's Black-Scholes model" against the Heston model and find that the reduced-form

[5]For an incomplete list of research, see, e.g., Azinovic, Gaegauf, and Scheidegger (2022); Fernandez-Villaverde, Hurtado, and Nuno (2019); Villa and Valaitis (2019); Maliar, Maliar, and Winant (2021); Duarte (2018); Fernandez-Villaverde et al. (2020); Bretscher, Fernandez-Villaverde, and Scheidegger (2022); Han, Yang et al. (2021). Recently, building on Maliar, Maliar, and Winant (2021), Kase, Melosi, and Rottner (2022) estimate a HANK model by approximating the likelihood function via neural networks.

model outperforms the Heston model out of sample. Our finding that structural option pricing models could outperform reduced-form models echoes the finding of Schaefer and Strebulaev (2008) in the corporate bond market, which shows that structural credit models are informative about out-of-sample hedge ratios even though they might imply large pricing errors.

The remainder of this article is organized as follows. In Section 2, we present the *deep surrogate* methodology and discuss how to apply it to option pricing models. Section 3 presents the applications. In section 4, we confront our methodology in the context of real data. Section 5 concludes.

2 Methodology

In this section, we first introduce the deep surrogate methodology. Then, we compare deep neural networks with other methods in the context of surrogate models.

2.1 Deep Surrogates

Consider an economic model that maps state variables $s \in \mathbb{R}^n$ into policies (e.g., consumption or investment), equilibrium prices, or their moments, which are summarized by $y \in \mathbb{R}^k$. The model has parameters $\theta \in \mathbb{R}^p$, and we represent the mapping with the function

$$y = f(s|\theta). \tag{1}$$

By treating the parameters θ as pseudo-states, we can define the augmented states $x \equiv [s, \theta]$, with $x \in \mathbb{R}^d$, $d = n + p$, and rewrite (1) as $y = f(x)$. Following the terminology of learning theory, we denote the admissible input and target space for f by $\mathcal{X}$ and $\mathcal{Y}$, respectively.

In situations where the function f is computationally expensive to evaluate, one may wish to build a cheap-to-evaluate surrogate, $\widehat{f}$, for the original function f. A familiar example of such a surrogate is the standard normal table, which stores pre-computed values for the CDF function of the standard normal distribution on a grid. We propose to use deep neural networks (DNNs) to construct the said surrogate, or *deep surrogate*.

Treating the parameters as pseudo-states has the benefit in that we can directly build one surrogate to capture the mapping between s and y under different model parameterizations. Since doing so further increases the dimensionality of the problem, it is vital that the class of functions we use to build a surrogate is sufficiently flexible (high expressivity) to approximate the original function accurately. Moreover, the process of building the surrogate should not be prohibitively expensive, for example, with the number of function evaluations required growing exponentially with the dimensionality as in the case of the grid-based method. As we explain below, for a large class of functions commonly used in economics, deep neural networks meet both requirements.

Before explaining the procedure for constructing a deep surrogate, we briefly outline the key elements of a basic type of deep neural networks called feedforward networks. These networks consist of multiple stacked-up layers of neurons. A given layer ℓ takes a vector $x^{[\ell-1]}$ as input and produces a vector $x^{[\ell]}$ as output,

$$x^{[\ell]} = \sigma\left(W_\ell x^{[\ell-1]} + b_\ell\right), \tag{2}$$

where W_ℓ and b_ℓ are parameters for the network; $\sigma(\cdot)$ is a non-linear function applied element-wise and is commonly termed an *activation function*.[6] For a neural network with L layers, $x^{[0]}$ represents the network's inputs, and $x^{[L]}$ represents the final output. Denote the dimension of $x^{[\ell]}$ by a_ℓ. Then, one can characterize a network with activation function σ, architecture of the network $a = (a_0, \cdots, a_L)$, and parameters $\phi = ((W_\ell, b_\ell))_{\ell=1}^{L}$.

The objective of a surrogate is to take any augmented state vector $x \in \mathcal{X}$ as input and produce $\widehat{f}(x)$ that is as close to the target $y = f(x)$ as possible based on some metric. Given a loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to [0, \infty]$, we aim to find a function $\widehat{f}$ that minimizes the expected loss of approximation $E\left[\mathcal{L}(\widehat{f}(x), y)\right]$, where the expectation is over the joint distribution of x and y. In the case of deep surrogate, we look for $\widehat{f}$ within the class of neural networks $\mathcal{H}$, and replace the expectation by the empirical loss over a sample (training set) $\mathcal{S} = ((x_i, y_i))_{i=1}^{m}$

[6] Popular choices for the activation function include the sigmoid function and rectified linear unit (ReLU).

drawn from $\mathcal{X} \times \mathcal{Y}$,

$$\widehat{f}_{\mathcal{H},\mathcal{S}} = \arg\min \left\{ \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\widehat{f}(x_i), y_i) : \widehat{f} \in \mathcal{H} \right\}. \tag{3}$$

Different from typical machine learning applications, where the data-generating process is unknown, we know the true f from the economic model and thus can generate as much data as desired. However, this is usually the most compute-intensive step in building a deep surrogate, because the original function is costly to evaluate (hence the need for a surrogate) and the size of the sample required for training a neural network of sufficient accuracy is large. With this mind, we now describe the steps for training a deep surrogate.

Generating the training sample. We obtain random draws $\tilde{x}_i$ from the input space $\mathcal{X}$ as follows. Assume $\mathcal{X}$ is a hypercube. For each element $x_t^{(j)}$ in x_t, let $\underline{x}^{(j)}$ and $\bar{x}^{(j)}$ be it's lower bound and upper bound, respectively, and

$$\underline{x} = [\underline{x}^{(1)}, \underline{x}^{(2)}, \cdots, \underline{x}^{(m)}], \quad \overline{x} = [\bar{x}^{(1)}, \bar{x}^{(2)}, \cdots, \bar{x}^{(m)}]. \tag{4}$$

We then draw $\tilde{x}_i$ according to:

$$\tilde{x}_i = \underline{x} + \tilde{R}_i(\bar{x} - \underline{x}), \tag{5}$$

where $\tilde{R}_i = \text{diag}(\tilde{r}_i)$, with $\tilde{r}_i \in \mathbb{R}^m$ following an appropriate multivariate distribution. One possibility is to first impose a hierarchical prior on parameters θ, and then draw x according to the conditional distribution implied by the model. Alternatively, one can use the multivariate uniform distribution, which will overweight the states that are relatively rarely visited in the model. The resulting training sample is $\mathcal{S} = ((x_i, y_i))_{i=1}^m$.

Surrogate training. Next, we train the deep surrogate $\widehat{f}_{\mathcal{H},\mathcal{S}}$ over the sample $\mathcal{S}$. For a given architecture (the depth L and the width of each layer a_ℓ), this amounts to searching for network parameters ϕ to minimize the empirical loss in (3). For our application, we will use the ℓ_1-norm between the prediction of the surrogate and the target as the loss function, which is simply the mean-absolute error when $k = 1$.

We perform the minimization in (3) using stochastic gradient descent. In standard machine

learning applications, the stochastic gradient descent is typically stopped before convergence to avoid over-fitting. In contrast, we rely on double-descent (see, e.g., Stephenson and Lee, 2021; Nakkiran et al., 2021), a recently discovered phenomenon that is particularly relevant here thanks to two unique features of building a surrogate: (i) the target y_i is (essentially) noise-free (limited to the numerical errors when evaluating the original function f), and (ii) we can increase the size of the training sample as desired with knowledge of the true model.

Surrogate validation. After training, we evaluate the accuracy of the deep surrogate model $\widehat{f}(x|\phi^*)$. Since it is important that the approximation of the surrogate is accurate in different parts of the input space, not just on average, we propose to use the following acceptance criterion:

$$\sup_{x\in\mathcal{X}} \mathcal{L}\left(\widehat{f}(x|\phi^*), f(x)\right) \leqslant \varepsilon, \tag{6}$$

where ε is some small positive constant. This means that the approximation error of the surrogate, as measured by the loss function $\mathcal{L}$, needs to be bounded by ε globally.

To implement (6), we generate a new sample of size N as validation set, $\mathcal{V} = \left(x_j^o, y_j^o\right)_{j=1}^N$, and evaluate the condition

$$\sup_{j} \mathcal{L}\left(\widehat{f}(x_j^o|\phi^*), f(x_j^o)\right) \leqslant \varepsilon, \tag{7}$$

If Condition (7) is satisfied, our training of the deep surrogate is finished. Otherwise, we can add flexibility to $\widehat{f}$ by modifying its architecture (e.g., increasing the depth or width), and increase the size of the training sample, especially from regions of the input space where the approximation error is large, to further reduce approximation error. The latter is a form of active learning, which systematically augments the training set with observations from areas where the model performs poorly (see Ren et al., 2021, for more detaiils). We then re-train and re-validate the model, and we iterate on these steps until Condition (7) is met.

2.2 Benefits of the Deep Surrogate

Having explained the algorithm for building a deep surrogate, we now turn to the advantages that deep neural networks offer relative to conventional function approximation methods.

The deep surrogate acts as a (high-dimensional) lookup table. Once trained, the evaluation of a deep surrogate can be reduced to a succession of matrix multiplications and the application of activation functions, which is highly parallelizable and is well suited to take advantage of modern hardware such as GPUs. Moreover, thanks to the backward propagation algorithm, we obtain the gradient of a deep surrogate for little to no extra computational costs. This can be particularly useful for model estimation and sensitivity analysis (see, e.g., Chen, Dou, and Kogan, 2022). In section 3.2, we use an option pricing model to illustrate the performance improvement from the deep surrogate.

Upfront vs. recurring cost. In exchange for faster model evaluation, the deep surrogate does require a large upfront cost, including the cost of generating a sizable training sample and training the network. However, this tradeoff is worthwhile for several reasons.

First, while the one-time upfront cost for building the surrogate can be large, it leads to a massive improvement in computing speed in all future model evaluations. Thus, in cases where a large number of model evaluations is needed (e.g., when pricing a large panel of options or performing structural estimation), the benefits of a deep surrogate post-training can easily outweigh the upfront costs.

Second, the main upfront cost is generating the training and validation samples. This process is ideally suitable for parallelization, which can significantly reduce the runtime.

Third, deep surrogate models are highly portable. One only needs to store the parameters for the neural network, which is substantially smaller than the amount of data storage needed when using a grid-based method. This feature makes it easy to share a high-quality deep surrogate among many users (just like the standard normal table), rather than having individual users incurring the cost for solving the model.

Why DNNs? One can build a surrogate model using a variety of function approximators, such as Chebyshev polynomials or splines. Deep neural networks have several desirable

Table 1: **A Comparison of Approximation Methods**

This table provides a comparison of different approximation methods. "High-dimensional input" refers to problems with state space dimensions higher than four. "Capturing local features" refers to the ability to deal with strong nonlinearity, such as kinks. "Irregularly-shaped domain" refers to problems with geometries other than a hypercube. "Large amount of data" refers to the ability to deal with more than $10,000$ observations.

	Polynomials	Splines	(Adaptive) sparse grids	Gaussian processes	Deep neural networks
High-dimensional input	✓	✗	✓	✓	✓
Capturing local features	✗	✓	✓	✓	✓
Irregularly-shaped domain	✓	✗	✗	✓	✓
Large amount of data	✓	✓	✓	✗	✓

properties for our purposes. Following Azinovic, Gaegauf, and Scheidegger (2022), Table 1 offers a comparison of DNN and other standard approximation methods.

First, neural networks are universal function approximators, in that they can approximate any continuous function arbitrarily well (see, e.g., Hornik, Stinchcombe, and White, 1989; Hanin, 2019). They allow for high-dimensional input, can handle irregularly shaped domains, and can approximate functions with kinks and ridges. In contrast, polynomial-based interpolation, which is popular in economics, have difficulties resolving kinked features.[7] Adaptive sparse grids can approximate functions with local features using a large amount of high-dimensional data but are inefficient on irregular domains (see, e.g., Brumm and Scheidegger, 2017). Gaussian processes can approximate high-dimensional functions on irregularly shaped domains, but become prohibitively slow when using a large amount of data (see, e.g., Traub and Werschulz, 1998; Williams and Rasmussen, 2006), which is often necessary to capture local features.

Second, while the complexity of the network and the size of the training sample required to achieve desired level of approximation accuracy depends on the target function, DNNs can alleviate and sometimes overcome the curse of dimensionality for certain classes of

[7]To accurately approximate local features, such as kinks, it is essential that the approximating function can fit strong nonlinearities in some areas of the domain without deteriorating the approximation quality in others. Polynomial-based interpolation, which approximates a kinked function by interpolating points with global polynomials, often leads to undesired "wiggles", known as Runge's phenomenon, which occur away from the kink).

functions. For example, Grohs et al. (2023) and Berner, Grohs, and Jentzen (2020) prove that when approximating the solution to the Black-Scholes PDEs with affine drift and diffusion coefficients, both the number of trainable network parameters and the size of the training sample grow at most polynomially (instead of exponentially) in input dimension. These theoretical guarantees imply that, in high dimensional settings, one can train high-accuracy deep surrogates with substantially fewer observations than in the case of traditional methods based on Cartesian grids. Notice that the setting considered above is not limited to option pricing. In fact, it is widely applicable in economic modeling thanks to the popularity of affine processes (see e.g., Chen and Joslin, 2012).

A third benefit of deep surrogate is that its Jacobian, $\frac{\partial \widehat{f}}{\partial x'}$, is readily available through automatic differentiation, which can be very helpful for local sensitivity analysis or working with extreme estimators. For example, suppose we want to estimate the model parameters θ using GMM, with the moment conditions implied by Eq. (1) from the economic model. Assuming the system of moment restrictions is over-identified, the GMM estimator for θ with a weighting matrix W is

$$\hat{\theta}_{GMM} = \arg\min_{\theta} g_T(\theta)^T W g_T(\theta), \quad \text{with} \quad g_T(\theta) \equiv \frac{1}{T}\sum_{t=1}^{T}(f(s_t|\theta) - y_t). \tag{8}$$

When we replace $f(s|\theta)$ with a high-precision deep surrogate, $\widehat{f}(x)$, the evaluation of both the GMM objective and its gradient becomes much easier. In fact, the first-order condition to (8) implies a system of non-linear equations, which can be solved efficiently. This is in contrast to having to estimate the gradient through numerical differentiation schemes, which further adds to the large number of costly model evaluations entailed in a typical parameter search.

3 Application to Option Pricing

Next, we apply the deep surrogate technology to conduct an in-depth study of a leading option-pricing model. We choose this example for several reasons. Option pricing models are among the most widely applied quantitative models in research and practice, and they

can help us extract useful information from the options market. Moreover, as discussed in Section 2.2, there is theoretical guarnatee (e.g., by Berner, Grohs, and Jentzen, 2020) that the solutions to affine option pricing models are particularly suitable to be approximated by DNNs. In this section, we build the surrogate and analyze its performance.

3.1 The Bates Model

The example we choose to illustrate the deep surrogate methodology is an extended version of the model of Bates (1996). This is a workhorse model that features stochastic volatility and jumps in the underlying asset, and we replace the original assumption of normal distribution for the log jump size with an asymmetric double-exponential distribution, which is an important feature in the data. While the deep surrogate methodology can be readily applied to more sophisticated models (for example, Duffie, Pan, and Singleton, 2000; Bates, 2000; Pan, 2002; Andersen, Fusari, and Todorov, 2015), we choose this example for a balance between transparency and complexity.

Under the risk-neutral measure $\mathbb{Q}$, the stock price S_t and the conditional variance v_t follow the process

$$dS_t = (r - d - \lambda\bar{\nu})S_t dt + S_t\sqrt{v_t}dW_t^s + d\left(\sum_{j=1}^{N_t} S_{t^-}\left(e^{Z_j} - 1\right)\right), \tag{9a}$$

$$dv_t = \kappa(\bar{v} - v_t)dt + \sigma\sqrt{v}dW_t^v, \tag{9b}$$

where r is the instantaneous risk-free rate and d is the dividend yield. The conditional variance v_t follows a Feller process, with the speed of mean reversion κ, long-run mean $\bar{v}$, and volatility parameter σ. The two standard Brownian motions under $\mathbb{Q}$, W_t^s and W_t^v, are correlated with $\mathbb{E}^{\mathbb{Q}}[dW_t^s dW_t^v] = \rho dt$. Finally, N is a Poisson process with constant arrival intensity λ, and Z_j follows a double-exponential distribution with density

$$f_Z(z) = p\frac{1}{\nu_u}e^{-\frac{z}{\nu_u}}1_{\{z>0\}} + (1-p)\frac{1}{\nu_d}e^{\frac{z}{\nu_d}}1_{\{z<0\}}, \tag{10}$$

where $0 < \nu_u < 1$, $\nu_d > 0$, and $0 \leqslant p \leqslant 1$ is the probability of a positive jump conditional on having a jump. The average jump size is $\bar{\nu} \equiv \frac{p}{1-\nu_u} + \frac{1-p}{1+\nu_d} - 1$. The double-exponential

assumption allows for heavier tails in jumps than the normal distribution as well as more flexibility to capture the asymmetry between positive and downward jumps, features supported by the evidence in Bollerslev and Todorov (2011).

A European option with maturity T and strike price K can be priced as the expected payoff under $\mathbb{Q}$ discounted using the risk-free rate. The price of a European put option at time t is

$$P(S_t, t, v_t) \equiv E^{\mathbb{Q}}[e^{-r(T-t)}(K - S_T)^+]. \tag{11}$$

The solution can be obtained through Fourier inversion of the characteristic function for affine processes (see Duffie, Pan, and Singleton, 2000).[8] Thanks to the put-call parity, we focus on the pricing of put options.

To facilitate a comparison of the pricing errors for options with different moneyness and maturities, we map option prices to the Black-Scholes implied volatility (BSIV). Moreover, following Andersen, Fusari, and Todorov (2017), we define a normalized moneyness measure m_t as

$$m_t = \frac{\ln\left(\frac{K}{F_{t,T}}\right)}{\sqrt{T-t}\sigma_{atm}}, \tag{12}$$

where $F_{t,T}$ is the forward price of the stock for the same maturity as that of the option, while σ_{atm} is a constant volatility measure, which we set to be the average BSIV of at-the-money options from the entire sample.

When mapping the Bates model into the deep surrogate framework, there are 5 state variables, $s_t = [m_t, \tau, v_t, r_t, d_t]$, where $\tau = T - t$ is the time-to-maturity, and the instantaneous variance v_t is latent. The risk-free rate r_t and dividend yield d_t are treated as observable states; they are assumed to remain constant over the life of the option but are allowed to vary with the time-to-maturity. In addition, there are 8 parameters, $\theta = [\kappa, \bar{v}, \sigma, \rho, \lambda, \nu_u, \nu_d, p]$. Thus, the dimension of the augmented state is $d = 13$.

[8] We apply the option pricing library QuantLib (https://www.quantlib.org) to compute prices and the corresponding BSIV.

3.2 Building the Deep Surrogate

The Bates model specified above belongs to the class of affine option pricing models, where the process of the underlying asset is an affine jump-diffusion under the risk-neutral measure. Berner, Grohs, and Jentzen (2020) (Theorem 1.1) show that, when approximating the solution in such models with a deep neural network, both the number of trainable network parameters and the size of the training sample grow only polynomially in the dimension of the input d.[9]

To construct the deep surrogate for the Bates model, we need to choose the activation function and the network architecture. Intuitively, the ReLU function appears to be a good candidate because of its resemblance to the option payoff in (11). However, we choose to use the *Swish* activation function (Ramachandran, Zoph, and Le, 2017),[10] which is given by

$$\sigma(x) = \frac{x}{1 + \exp(-\gamma x)}, \tag{13}$$

where γ can be a constant or a trainable parameter. The *Swish* activation function can be viewed as a smooth version of the ReLU function, and possesses two particular qualities that make it appropriate for the surrogate application: 1) unlike the *sigmoid*, it does not suffer from the vanishing gradient problem and, as our application shows, complex structural models require surrogate networks with a deep architecture, and 2) unlike *ReLU*, the *Swish* is smooth, which means that the gradients of the trained surrogate model will also be smooth across the state-space. This latter property is desirable when the gradients of the surrogate are needed (e.g., for estimation).

Next, we go over the details for generating the training sample, network architecture design, and the surrogate training procedure we use for the option pricing models.

Training sample and DNN architecture. To generate the training sample, we first obtain random draws $\tilde{x}_i$ from the input space $\mathcal{X}$. To do so, we set the minimum and maximum values for each state variable and all parameters (see Table A.1 in Appendix A),

[9] Berner, Grohs, and Jentzen (2020) consider affine diffusions, but their results can be readily extended to the type of jump-diffusion we consider. Their results are based on using ℓ_2-norm for the loss function. We use ℓ_1-norm for the loss function to reduce the influence of the outliers.

[10] It has been shown empirically that the performance of very deep neural nets in combination with *Swish* activation functions outperform architectures that rely on ReLU (see, e.g., Tripathy and Bilionis (2018)).

where the values for these upper and lower bounds are chosen based on a combination of model restrictions and domain knowledge. For example, the intensity of the Poisson process λ in the Bates model is, by definition, positive; the correlation between volatility shocks and price shocks ρ is restricted to be negative based on the so-called leverage effect documented in the literature.

We build the training samples and surrogate model following an iterative process described in Section 2.1, starting with a network of 3 layers and an initial training sample of 10^7 observations, then gradually increasing the training sample size and depth of the deep surrogate until the criterion (7) is met (a new validation set of size 10^6 is used in each iteration). At each step of network training, we run a mini-batch stochastic gradient descent algorithm to determine the neural network's parameters.[11] With this procedure, we obtain a final training sample of size 10^9. The final feedforward network for the Bates model for instance has 7 hidden layers, 400 neurons each. This architecture yields a total of 967,201 trainable parameters.

Although the final size of our training sample might appear large, it only sparsely populates the 13-dimensional augmented state space. It is also worth noting that we build the training sample using a naive scheme by randomly drawing points from the augmented state space based on the uniform distribution. An active learning scheme (see, e.g. Krause and Guestrin, 2007; Deisenroth, Rasmussen, and Peters, 2009; Renner and Scheidegger, 2018), that is, one that samples more intensively in regions where nonlinearity is more pronounced and where initial approximation errors are large) is likely to be much more efficient.

Surrogate performance. To illustrate the magnitude of gains in computing speed, we compare the approximate computing time needed for pricing 4,000 European options (which is roughly the daily number of SPX options traded on the CBOE) with the Bates model based on three methods: direct evaluation of the pricing formula using the Fast Fourier Transform (FFT), deep surrogate using a single CPU, and deep surrogate using a GPU. We also compare the computing time for estimating the model parameters from option prices on

[11] Specifically, we run the ADAM optimization algorithm with an initial learning rate of $0.5 * 10^{-4}$ for 15 epochs, that is, we use mini-batches of size 256 until we have used the whole data set 15 times. After each epoch, we save the model and use a validation set of 10,000 points to estimate the surrogate model's performance. In the end, we use the network's parameters after the epoch with the lowest validation error.

Table 2: **Performance Improvement from Deep Surrogates**

This table compares the estimated computing time for pricing or getting the gradient of all the SPX options on an average day and daily estimation over a year using a traditional method based on Fast Fourier Transform (FFT) versus a Deep Surrogate model. We use the Bates model for illustration. We estimate the gradient of the FFT method through numerical differentiation, while the gradient of the Deep Surrogate can be obtained directly through backward propagation. To transform the daily gradient estimation into a yearly estimation of the model's parameters, we assume an average of 10 iterations of the optimization algorithm and 252 business days in a year.

	FFT	Deep Surrogate	Deep Surrogate + GPU
pricing, 1-day	10s	0.6s	0.06s
gradient, 1-day	180s	3.2s	0.3s
estimation, 1-year	125h	2.2h	0.2h

a single day (assuming 10 iterations are needed in the parameter search), as well as the time required to perform daily estimation for an entire year. The results are shown in Table 2. Although the estimates can vary depending on hardware and implementation, the advantage of the deep surrogate relative to ordinary FFT, which suffers from the curse of dimensionality, is evident, especially when we take advantage of its parallelizability on a GPU.

A small validated pricing error is a necessary but not sufficient condition to consider a DNN accurate enough to be used as a surrogate for the option pricing model. It does not imply that it can approximate the function gradients (the option "greeks") accurately, nor does it ensure that the surrogate can be used to accurately estimate model parameters. We verify these properties systematically in Section 3.3.

3.3 Accuracy of the Deep Surrogate

In this section, we examine the accuracy of the deep surrogate models. We start by investigating how accurately the surrogate model captures the sensitivity of option value to various states and parameters. Matching these sensitivities accurately is critical for hedging as well as parameter estimation. For illustration, we vary the states or parameters one at a time while fixing the rest at their respective averages and then compare the BSIVs and their partial derivatives produced by the target model versus those by the surrogate. The results are

shown in Figure A.1 and A.2 in Appendix A. These figures highlight the complexity of the Bates model – there is significant nonlinearity along multiple dimensions. The accuracy of the deep surrogate is also visible. In particular, even though we do not explicitly target the greeks, the surrogate almost perfectly replicates these features.

Next, we examine the ability of the deep surrogate to recover parameter values in structural estimation. This task is more demanding for the surrogate than matching the true option prices (BSIVs). For example, the GMM estimator depends on the Jacobian of the moment function with respect to the parameters. It is possible that a surrogate that approximates the level of target function reasonably well might approximate the gradients poorly, which would bias the estimates.[12] Thus, when one wishes to use a surrogate for structural estimation, it is important to verify the ability of the surrogate to recover the parameters accurately in addition to approximating the target function.

Our procedure is as follows. First, we randomly draw volatility v_t and parameters θ according to Eq. (5). Second, we compute the option prices according to the Bates model on 1,000 options, with the moneyness and maturity drawn from a uniform distribution. These option prices are then converted into BSIVs. Third, we use the deep surrogate function $\widehat{f}(x)$ to run GMM estimation of the hidden state v_t and parameters θ from the cross-section of option data as in (8) based on an identity weighting matrix. At the end of the estimation, we compute the estimation error for each element of θ,

$$e_i = \frac{|\hat{\theta}_i - \theta_i|}{\bar{\theta}_i - \underline{\theta}_i}, \tag{14}$$

where $\hat{\theta}_i$ is the GMM estimate for θ_i based on the surrogate; $\bar{\theta}_i$ and $\underline{\theta}_i$ represent the maximum and minimum values in the training sample of the respective surrogate model (cf. Table A.1). We measure the estimation error for the conditional variance v_t the same way.

We repeat the above procedure 1,000 times and present the distribution of estimation errors for the conditional variance and different parameters in Figure 1. Overall, the estimation errors are quite small. The parameters that are relatively more difficult to estimate include ρ, the correlation between shocks to stock prices and volatility, ν_2, the average size of negative

[12] One way to address this problem is to make the loss function dependent on the gradient when training the surrogate (i.e., making it a multi-target learning problem).

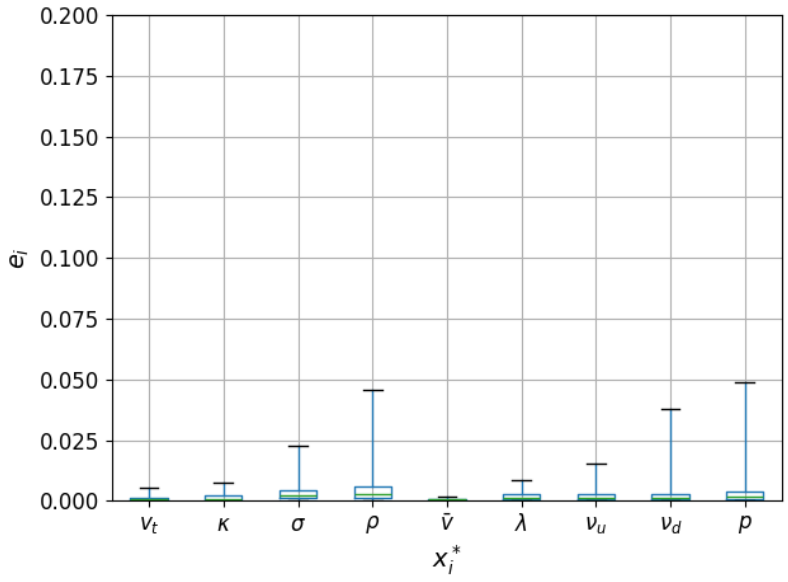


Figure 1: **Structural estimation via the deep surrogate.** In this figure, we examine the accuracy of estimating the hidden state and the parameters of the Bates model through the deep surrogate. The box-plots of the estimation errors for the hidden state v_t and the parameter are based on 1,000 simulations, each with a sample of 1,000 options. Each box shows the interquartile range, while the whiskers show the 1st and 99th percentile estimation errors across the simulations.

jumps, and p, the probability of a positive jump conditional on a jump occurring. However, the estimation errors are small even for these parameters. For example, the estimation error of the parameter ρ is lower or equal to 0.049 in 99% of our simulations.

Put together, the above results on model sensitivity and parameter estimation suggest that the deep surrogate can reliably replace the target Bates model when it comes to pricing, hedging, or parameter estimation. Unlike the original model, performing these analyses using the surrogate comes at a negligible computational cost.

We conclude this section by stressing that a deep network structure is critical to approximate the Bates model with sufficient accuracy. To do so, we train a surrogate with a single hidden layer and 400 neurons, and then apply the procedure in this section to assess the shallow network's ability to match model sensitivities and estimate parameters. In unreported results, we find that the estimation errors are significantly larger with the shallow network.

4 Empirical Analysis

In this section, we apply the deep surrogate to study the structural option pricing model introduced in Section 3.1 in real data. With the help of the deep surrogate, we conduct daily re-estimation of the model parameters. This allows us to i) construct a high-frequency option-implied tail risk index for the stock market, ii) quantify the time variation in the degree of parameter instability and study its implications for option market liquidity, and iii) systematically examine the out-of-sample performances of the Bates model. Finally, we use the surrogate to characterize the model-implied conditional distributions of option returns and confront them with the data.

We use daily data on the S&P500 index options (SPX) from OptionMetrics. Our sample includes European call and put options between 1996 and 2019. Following the literature (see, e.g., Andersen, Fusari, and Todorov, 2017), we remove options with maturity over 250 days as well as options with extreme moneyness (those with Black-Scholes delta outside the range of $[0.1, 0.9]$), since these options tend to be thinly traded. Figure A.3 in Appendix B provides a summary of our sample.

We also obtain the zero-coupon yield curves as well as the forward prices of different maturities for the S&P 500 index from OptionMetrics. In addition, we obtain LTP (left-tail probability), an option-based estimate of the (risk-neutral) probability of a 10% weekly down move for the S&P 500 index, from Bollerslev, Todorov, and Xu (2015)[13] and the VIX volatility index from the CBOE.

4.1 Tail Risk Index

On day t, for each option contract i, we calculate the maturity-specific risk-free rate r_{it} by interpolating the zero-coupon yield curve for that day. In addition, we calculate the maturity-specific dividend yield d_{it} based on the forward price of the same maturity using

[13]The LTP series is available from https://tailindex.com/index.html.

the formula

$$d_{it} = r_{it} - \ln\left(\frac{F_{t,T}}{S_t}\right) / \left(\frac{\tau_{it}}{252}\right). \tag{15}$$

We then estimate the latent state v_t and model paramters $\theta = [\kappa, \bar{v}, \sigma, \rho, \lambda, \nu_u, \nu_d, p]$ for day t by fitting the BSIVs of the cross-section of SPX options using GMM with an identity weighting matrix (see (8)). As Table 2 suggests, this task would have been prohibitively expensive in terms of computing time, but becomes feasible on a normal computer thanks to the deep surrogate.

In Figure 2, we plot the smoothed time series for the latent state v_t and the parameters based on moving averages over the past 60 days. Strikingly, all of the parameter estimates show substantial time variation, which conveys unique information from the options market. For example, the parameter ρ, which is the conditional correlation between shocks to stock price S_t and instantaneous variance v_t, provides an option-based measure of the so-called leverage effect (see e.g., Black, 1976). It ranges between -0.5 to -0.9 in most of our sample, and noticeably turned most negative during the Great Financial Crisis, reaching -0.96 in early 2008. As another example, the option market provides clear evidence that asymmetry between upward and downward jumps in the market index is needed to fit the option prices; we can see it through the fluctuations in p, the conditional probability of an upward jump, as well as ν_u and ν_d, which measure the expected size of upward and downward jumps.

Predicting tail events. We now focus specifically on the information that the option market conveys about tail risk in the market index. We define two measures of left-tail risks based on the parameter estimates on date t,

$$TailProb_t = \lambda_t \frac{n}{252}(1 - p_t) e^{-\frac{\alpha}{\nu_{d,t}}}, \tag{16}$$

$$TailRisk_t = \lambda_t \frac{n}{252}(1 - p_t)\left(1 - \frac{1}{1 + \nu_{d,t}}\right). \tag{17}$$

The first measure, $TailProb_t$, is approximately the risk-neutral probability of a negative jump of size α in the S&P 500 index over the next n days, which assumes that the parameter values will remain the same in the next n days and ignores the probability of more than one

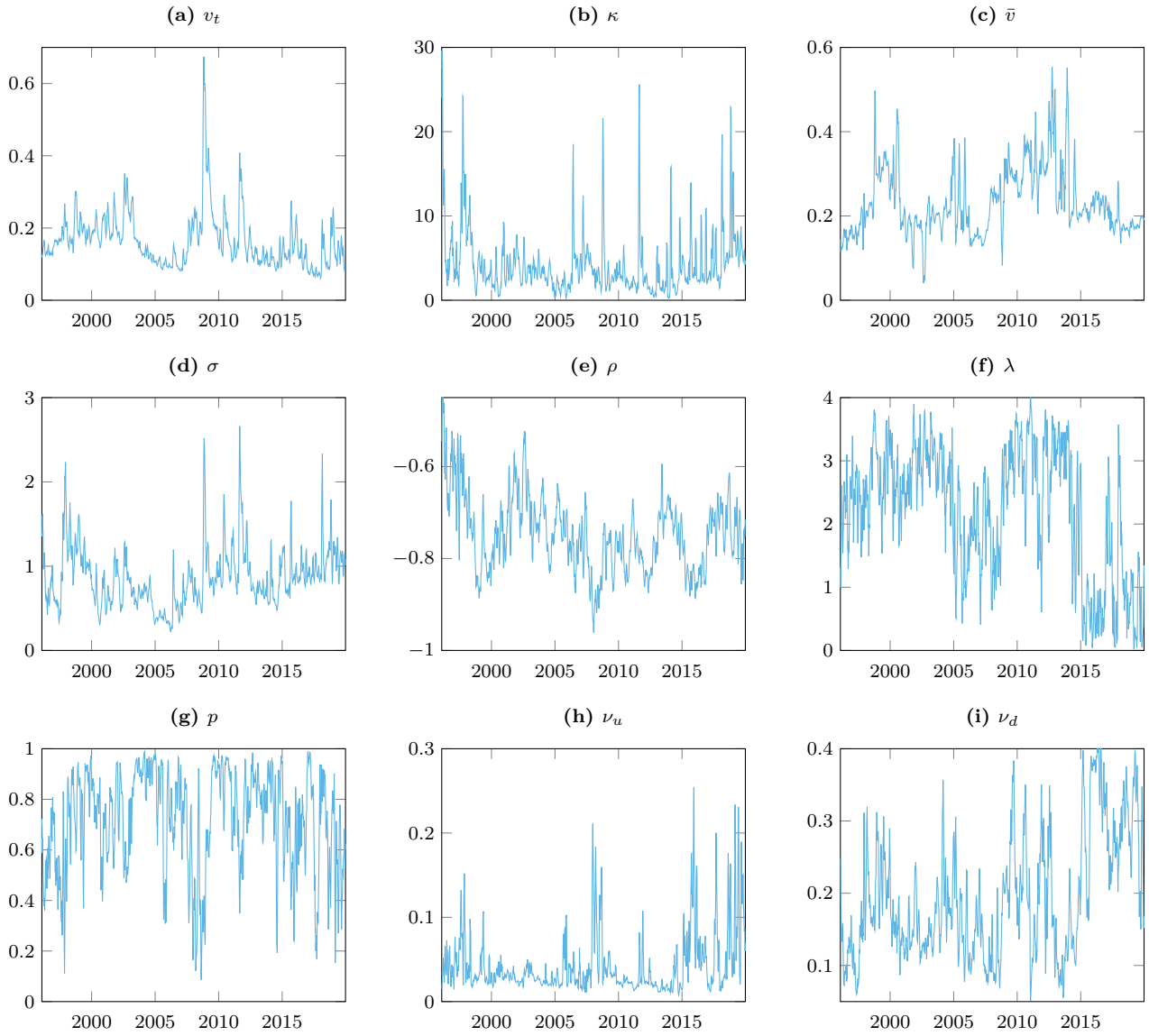


Figure 2: **Parameter estimates for the Bates model.** This figure shows the smoothed (20-day rolling means) daily estimates of the parameters and latent state of the Bates model.

jumps. The second measure, $TailRisk_t$, is approximately the risk-neutral expected loss in the index due to a negative jump over the next n days, again ignoring the probability of more than one jumps.

In Figure 3 Panel A, we compare $TailProb$ with $n = 5$ and $\alpha = 10\%$ against LTP from Bollerslev, Todorov, and Xu (2015), which is also a measure of the risk-neutral probability of a 10% down move for the S&P 500 index over a week. The measure LTP is based on a nonparametric model and is estimated exclusively with short-dated (with a tenor between 6 and 31 trading days) deep out-of-the-money SPX options. Interestingly, despite the differences

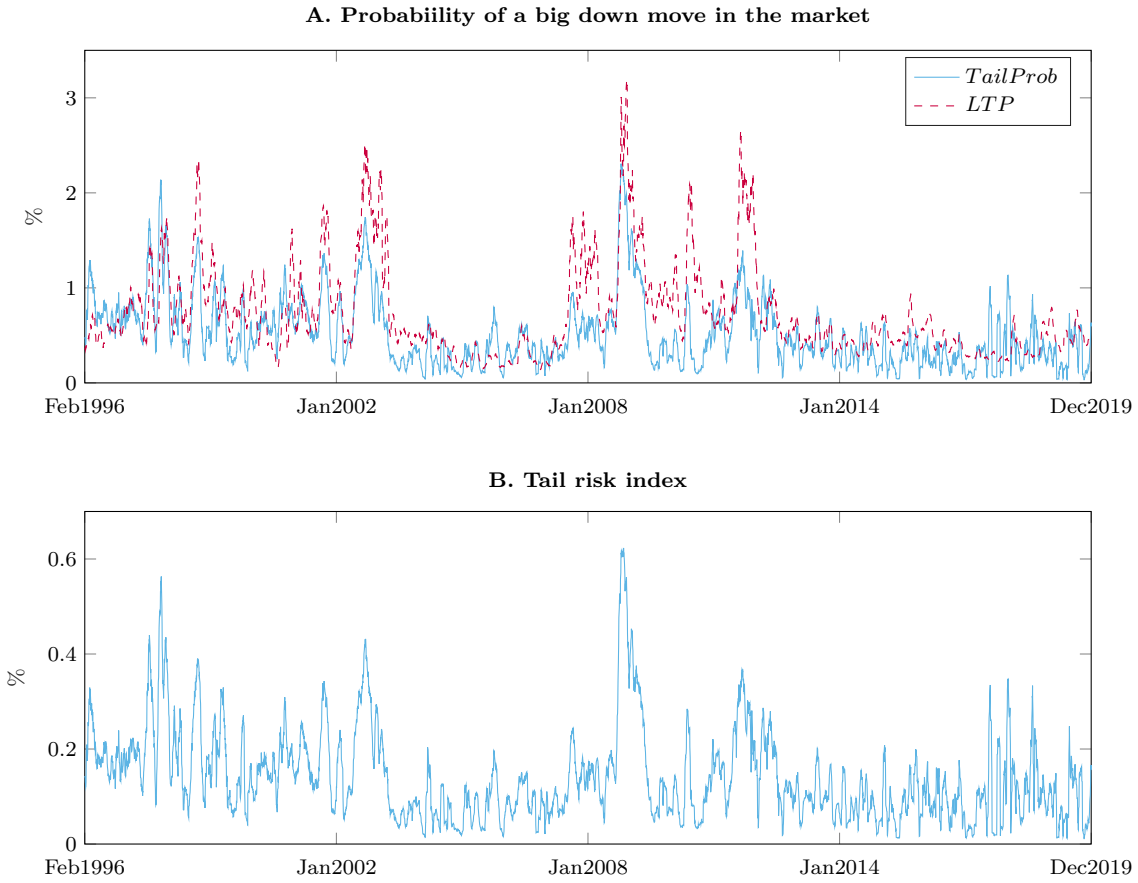


Figure 3: **Option-implied tail risk measures.** In Panel A, we compare the risk-neutral probability of a 10% negative jump in the S&P 500 index over the next 5 days based on $TailProb$ from the Bates model against that by LTP from Bollerslev, Todorov, and Xu (2015). In Panel B, we plot $TailRisk$, the risk-neutral expected loss in the S&P 500 index resulting from a negative jump over the next 5 days. Both $TailProb$ and $TailRisk$ are based on 20-day rolling means.

in methodology, the two series show similar magnitudes overall and have a correlation of 0.71 in our sample; LTP has more pronounced peaks compared to $TailProb_t$, especially before 2013. In Panel B, we plot the tail risk index $TailRisk_t$, which is also highly correlated with LTP and even more so with $TailProb$.

To discern the differences in their information content, we use the three tail risk measures to predict the probability of a left-tail event in the S&P 500 index over the next h periods, $P_t(TailEvent_{t+h})$, through a logistic regression:

$$P_t(TailEvent_{t+h}) = \frac{1}{1+\exp(-\beta_0 - \beta_1\, LTP_t - \beta_2\, TailProb_t - \beta_3\, TailRisk_t)}. \tag{18}$$

Table 3: **Predicting Tail Events**

This table presents the results of the logistic regression used to predict jumps in the S&P 500. $TailEvent_{t+h}$ is defined in two different ways. In Panel A, a tail event is defined as a cumulative return of -10% or less over the next 5 business days. In Panel B, a tail event is defined as a daily return of -5% or less for any individual days over the next 5 business days. The values in parentheses are heteroskedasticity and autocorrelation consistent (HAC) standard errors (with 13 lags); * denotes significance at the 10% level, ** at 5%, and *** at 1%. The sample period is from January 1996 to December 2019, with 5,939 observations. The adjusted McFadden Pseudo-R^2 coefficients are included.

	10% Weekly Down Move				5% Daily Drop within a Week			
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
LTP	1.468***			0.052	1.917***			-0.364
	(0.365)			(0.777)	(0.423)			(0.694)
$TailProb$		1.258***		-0.935		1.601***		-3.272***
		(0.188)		(0.812)		(0.218)		(0.716)
$TailRisk$			8.131***	12.917**			11.177***	29.903***
			(1.027)	(5.643)			(1.263)	(5.453)
Constant	-7.190***	-6.776***	-6.907***	-6.854***	-6.569***	-5.980***	-6.383***	-5.992***
	(0.606)	(0.405)	(0.407)	(0.599)	(0.677)	(0.410)	(0.461)	(0.539)
Pseudo-R^2	0.080	0.119	0.152	0.159	0.173	0.230	0.331	0.386

In order to have a reasonable number of tail events in our sample, we define a tail event in two different ways: i) a cumulative return of -10% or less in the index over the next 5 business days; ii) a daily return of -5% or less for any individual days over the next 5 business days. We define $TailProb$ accordingly, with $n = 5$ and $\alpha = 5\%$.

As Table 3 shows, individually, the three tail risk measures can all predict the two types of tail events we consider, with coefficients that are highly statistically significant. The economic magnitude is also significant. For example, a one-standard-deviation increase in $TailProb$ is associated with approximately an 80% (relative) increase in the probability of a 5% daily drop of the index in the next week; with a one-standard-deviation increase in $TailRisk$, the probability of the same event nearly triples. The pseudo-R^2 coefficient is the highest with $TailRisk$ as the predictor. When predicting 5% daily drops, the pseudo-R^2 nearly doubles, rising from 17.3% with LTP to 33.1% with $TailRisk$. When we use the three predictors jointly, the coefficient for LTP is no longer significant but remains significant for $TailRisk$, while the adjusted pseudo-R^2 only rises marginally compared to the case with having $TailRisk$ as the only predictor, suggesting again that much of the predictive

information for future tail events is contained in the measure $TailRisk$.

4.2 Parameter Instability

If a structural model is correctly specified, we should expect to see its parameter estimates from sub-samples remain stable. However, Figure 2 shows that the parameter estimates for the Bates model fluctuate significantly over time. For some of the parameters (e.g., $\bar{v}$ and σ), the dynamics appear to be relatively persistent, which means that one can address the issue by re-calibrating the parameters frequently to adapt to the latest data. However, for other parameters such as λ and p, the estimated parameter values can change drastically in a short period of time. Such type of parameter instabilities cast doubt about how reliable a model fitted with the historical data will be out of sample, which in turn has implications for option trading and risk management.

In this section, we formally test whether the Bates model parameters are stable at daily frequency. The reason for choosing the daily frequency is to inform us whether daily model recalibration could be sufficient to address the parameter instability issue. Performing the tests at lower frequencies (say monthly or annually), although less demanding computationally, is incapable of revealing the problem at higher frequencies.

We apply the statistical test developed by Andersen, Fusari, and Todorov (2015). Consider dates t and $t-1$. We define the parameter instability measure I_t on date t as

$$I_t \equiv \left(\hat{\theta}_t - \hat{\theta}_{t-1}\right)' \left(\widehat{\text{Avar}}\left(\hat{\theta}_t\right) + \widehat{\text{Avar}}\left(\hat{\theta}_{t-1}\right)\right)^{-1} \left(\hat{\theta}_t - \hat{\theta}_{t-1}\right), \tag{19}$$

where $\hat{\theta}_t$ denotes the parameter estimates on date t, and $\widehat{\text{Avar}}\left(\hat{\theta}_t\right)$ denotes consistent estimate of the asymptotic variance of $\hat{\theta}_t$ (cf. Andersen, Fusari, and Todorov (2015), equations (11)-(12)). Under the null that the option pricing model is valid for the two distinct periods, I_t is asymptotically chi-squared distributed with degrees of freedom p. Thus, it is straightforward to test whether the GMM estimates $\hat{\theta}_{GMM,t}$ on each date t are statistically different from those on date $t-1$.

Figure 4 shows the results of the daily parameter stability tests for the Bates model. On the y-axis, we show the average percentage of days over the past 60 days for which the

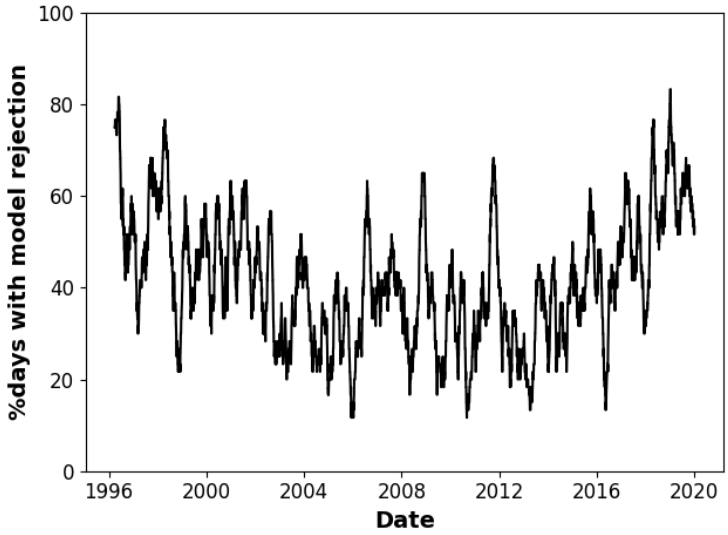


Figure 4: **Rejection rates for parameter stability tests.** This figure shows, for each day, the percentage of the last 60 days for which the parameter estimates of the Bates model are statistically significantly different (at the 1% level) on two adjacent trading days.

hypothesis was rejected at the 1% level. On average, we reject daily parameter stability at 1% level 41.6% of the time, but the rejection rate fluctuates between 20 and 80% over the sample and shows some persistence. In comparison, the rejection rate for the Heston model (Heston, 1993), which features stochastic volatility but no jumps in the price dynamics of the underlying asset, is even higher, at 60.7% on average, an indication of more severe misspecification.[14]

A potential economic consequence of parameter instability is that it becomes more difficult to manage the risks of an option portfolio. An important method for option market makers to manage inventory risks is to delta-hedge their inventory positions, where the hedge ratios are produced by some model. While market makers can use a variety of option pricing models, our parameter instability measure based on the Bates model can help identify periods of model instability more broadly, during which time the hedge ratios produced by a model that is calibrated to past data may not accurately predict future sensitivities of option values to changes in stock prices. This would elevate the risks of market making and reduce the liquidity in the market.

To examine this prediction, we run panel regressions of option bid-ask spreads on the

[14]The results for the deep surrogate for the Heston model, unreported here, are available upon request.

parameter instability measure,

$$BidAsk_{i,t} = \alpha_i + \beta_1 \ln I_t + \beta_2 BidAsk_{i,t-1} + \beta_3 VIX_t + \beta_4 LTP_t + \beta_5' X_{i,t} + \epsilon_{i,t}. \tag{20}$$

Here, $BidAsk_{i,t}$ is the relative bid-ask spread for option i at time t, defined as the closing bid-ask spread normalized by the mid-quote. Besides the log of the parameter instability measure $\ln(I_t)$, we include the contract fixed effect α_i and lagged bid-ask spread as controls. For market conditions, we control for the market volatility index VIX_t and the tail risk measure LTP_t. Finally, the contract-specific controls $X_{i,t}$ include option tenor, $\tau_{i,t}$, moneyness, $m_{i,t}$, and abnormal volume, $\widehat{Volume}_{i,t}$, which is defined as

$$\widehat{Volume}_{i,t} = \frac{Volume_{i,t}}{\frac{1}{252}\sum_{k=1}^{252} Volume_{i,t-k+1}} - 1, \tag{21}$$

where $Volume_{i,t}$ is the trading volume of contract i on date t.

The results are reported in Table 4. When we only control for lagged bid-ask spread and contract fixed effect, the coefficient on $ln(I_t)$ is negative. This is likely because the degree of parameter instability is positively correlated with market volatility and tail risk. SPX puts tend to be traded more actively when the market is volatile, which can reduce the bid-ask spreads. To isolate the effect of parameter instability from market volatility, we include both VIX_t and LTP_t as controls, after which the coefficient on $ln(I_t)$ becomes significantly positive. Controlling for abnormal volume at contract level has little effect on the coefficient for $ln(I_t)$. After further controlling for contract tenor and moneyness, both of which are related to option liquidity themselves (for example, long-dated options and deep OTM options tend to be less liquid), the coefficient for $ln(I_t)$ is smaller but remains highly significant. Overall, the results in Table 4 are consistent with the prediction that bid-ask spreads will widen when the option pricing models that market makers are using become more unstable.

Table 4: **Parameter Instability and Option Bid-Ask Spreads**

This table presents the results of panel regressions of option bid-ask spreads on the parameter instability measure. The values in parentheses are heteroskedasticity consistent standard errors clustered by contract; * denotes significance at the 10% level, ** at 5%, and *** at 1%. The sample includes all SPX put options from January 1996 to December 2019, with 3,860,843 contract-day observations.

	(1)	(2)	(3)	(4)
$\ln(I_t)$	−0.0734***	0.0509***	0.0504***	0.0129***
	(0.0017)	(0.0018)	(0.0018)	(0.0017)
$BidAsk_{i,t-1}$	0.7975***	0.7842***	0.7824***	0.5120***
	(0.0010)	(0.0010)	(0.0010)	(0.0014)
VIX_t		−28.1697***	−28.5717***	−12.8920***
		(0.2155)	(0.2168)	(0.2666)
LTP_t		1.6223***	1.5835***	0.8364***
		(0.0391)	(0.0380)	(0.0457)
$\widehat{Volume}_{i,t}$			0.0112***	0.0053***
			(0.0003)	(0.0003)
$\tau_{i,t}$				0.0092***
				(0.0003)
$m_{i,t}$				−4.3607***
				(0.0168)
Contract FE	Yes	Yes	Yes	Yes
R^2	0.7729	0.7765	0.7767	0.8153

4.3 Out-of-sample Performance

Besides risk management, parameter instability will also affect a structural model's out-of-sample performance. In this section, we examine the out-of-sample pricing and hedging performances of the Bates model based on daily re-estimation, and compare it to a reduced-form model.

To measure the out-of-sample pricing errors, we estimate the Bates model's parameters θ and hidden state v_t on date t, and then use them to predict the mid bid-ask BSIVs of all the out-of-money (OTM) SPX call and put options at time $t+h$ under the updated values for the observable states (including normalized moneyness m_{t+h}, time-to-maturity τ, risk-free rate r_{t+h}, and dividend yield d_{t+h}). For simplicity, we keep the latent state constant, $v_{t+h} = v_t$.

Reduced-form benchmark. Motivated by Christoffersen and Jacobs (2004), we use a reduced-form model as an evaluation benchmark for the Bates model. The comparison also helps shed light on the relative strengths and weaknesses of the structural option pricing model relative to reduced-form models, which are becoming increasingly popular with the advances in machine learning. Christoffersen and Jacobs (2004) considers a so-called Practitioner Black-Scholes model, which models the implied volatility surface using a polynomial of time-to-maturity and strike price. We extend this approach by replacing the polynomial with a more flexible random forest (RF) model.[15]

The random forest model's input is

$$X_{i,t} = [m_{i,t}, \tau_{i,t}, m_{i,t}\tau_{i,t}, \mathbb{1}_{m_{i,t}}], \tag{22}$$

where m is the moneyness measure defined in (12), τ is the time-to-maturity, and $\mathbb{1}_m$ is a binary indicator that equals to 1 if $m > 0$ (the option's strike is above the corresponding forward price). Although a nonlinear model like the random forest should in principle be able to recover the two features $m_{i,t}\tau_{i,t}$ and $\mathbb{1}_{m,i,t}$ from $m_{i,t}$ and $\tau_{i,t}$, we include them to speed up the algorithm training. Consistent with the way we re-estimate the parameters of the Bates model at daily frequency, we also retrain the random forest model daily to minimize the mean squared error (MSE) of the BSIVs for all SPX options each day.

In Figure 5, we plot the average implied-volatility RMSEs of SPX options at different strikes (with the underlying index level normalized to 100) for the Bates model and the RF model. Since we only use OTM options from the data, the options in our sample with strike prices less (more) than 100 are puts (calls). Panel (a) shows the in-sample pricing performances for all maturities, while panels (b) through (d) report the out-of-sample performances at a weekly horizon ($h = 5$) for different maturity buckets.

As expected, the out-of-sample RMSEs are significantly higher than the in-sample RMSEs for both models. The RF model in particular is able to match the data closely in sample thanks to its flexibility, with RMSEs less than 0.003. For reference, the average BSIV of all

[15] A random forest works by constructing a multitude of decision trees, with the algorithm's prediction defined as the mean output of each individual tree. This ensemble approach drastically reduces the risk of over-fitting. For a general introduction to random forests, see Liaw, Wiener et al. (2002), and for random forests applied to finance, see e.g., Gu, Kelly, and Xiu (2018).

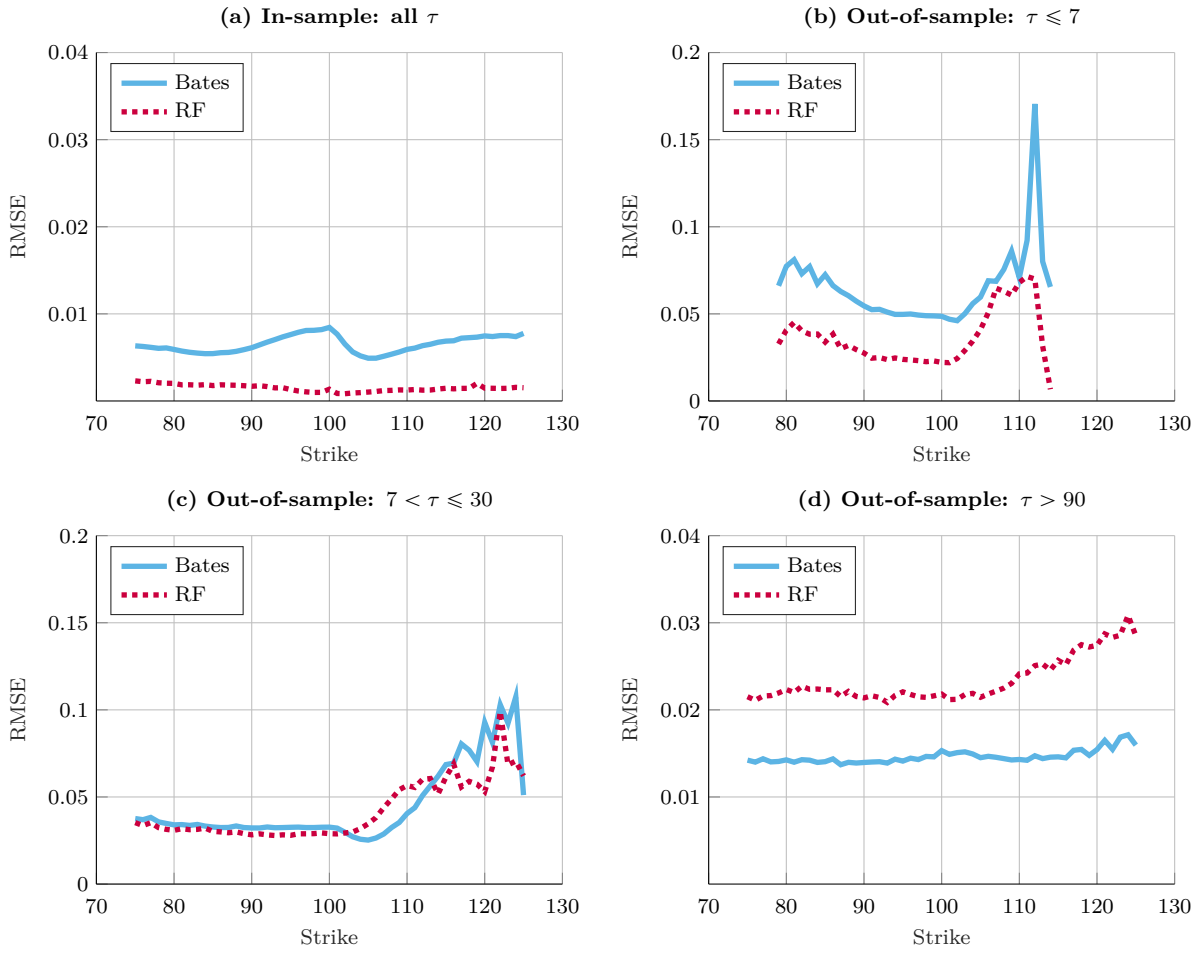


Figure 5: **Pricing performances of the Bates model and RF model across different strikes and tenors.** This figure shows the model's in- and out-of-sample performances, as measured by averages of daily RMSEs, across different strikes and maturity brackets. For out-of-sample performance, the forecasting horizon is one week ($h = 5$).

options in our sample is 0.218. For both models, the out-of-sample RMSEs are the highest for short-dated options and become smaller at longer maturities; they also tend to be higher for high strikes (deep out-of-the-money calls).

Interestingly, as Panel (a) of Figure 5 shows, the RF model produces substantially smaller out-of-sample pricing errors than the Bates model for the extremely short-dated options (with time-to-maturity of 7 days or less). The Bates model struggles the most in pricing short-dated deep OTM puts and OTM calls. For 20% OTM puts, the RMSEs for the Bates model are as high as 0.08, compared to 0.04 or lower for the RF model; for 10%+ OTM calls, the RMSEs for the Bates model are as high as 0.17, compared to 0.07 or lower for the RF model. However, the advantage of the RF model dissipates as we move to longer maturity.

For options with time-to-maturity above 30 days, the Bates model actually outperforms the RF model, which is clearly demonstrated in Panel (d).

The dependence of the relative performances of the structural Bates model and the reduced-form RF model on time-to-maturity is a nice illustration of the bias-variance tradeoff. As Andersen, Fusari, and Todorov (2017) demonstrate, short-dated options are different from longer-dated options in terms of the type of risks they are most exposed to. For example, short-dated options are not as sensitive to changes in the expected volatility of the underlying asset, but are highly sensitive to the level of spot volatility and jump intensity, as well as the distribution of the jump size. Additionally, short-dated options could also be more severely affected by liquidity and demand factors (due to investors rolling over their maturing positions as well as higher retail investor participation in the case of OTM calls). Our results on the relative performances suggest that the RF model benefits from its flexibility when pricing short-dated options, for which the Bates model is more severely misspecified; in contrast, the Bates model is better suited to capture the intertemporal changes in risks that matter more for longer-dated options, and its parsimony helps guard against over-fitting, which hurts the performance of the RF model. Our results provide systematic justification for the standard practice of filtering out extremely short-dated options in the literature.

Next, in Figure 6, we compare the out-of-sample pricing performances of the Bates model and the RF model over time. The four panels are for four different maturity buckets. Confirming the findings in Figure 5, the RF model consistently outperforms the Bates model for short-dated options (with $\tau \leqslant 7$) throughout our sample period, and the reverse is true for long-dated options (with $\tau > 90$). In the shortest maturity category (Panel (a)), the pricing errors become more volatile after the introduction of weekly SPX options in 2005, especially for the Bates model. Across maturities, the pricing errors spike up periodically, most notably in Q4 2008 (during the Great Financial Crisis), and also in Q4 2011, Q2-Q3 2015, and Q1 2018. As expected, the timings of the spikes in pricing errors for the Bates model correspond to periods of high parameter instability, as shown in Figure 4. However, the strong correlation between the pricing errors of the Bates model and RF model is evidence that the time variation in parameter instability for the Bates model reflects a much more general form of model instability.

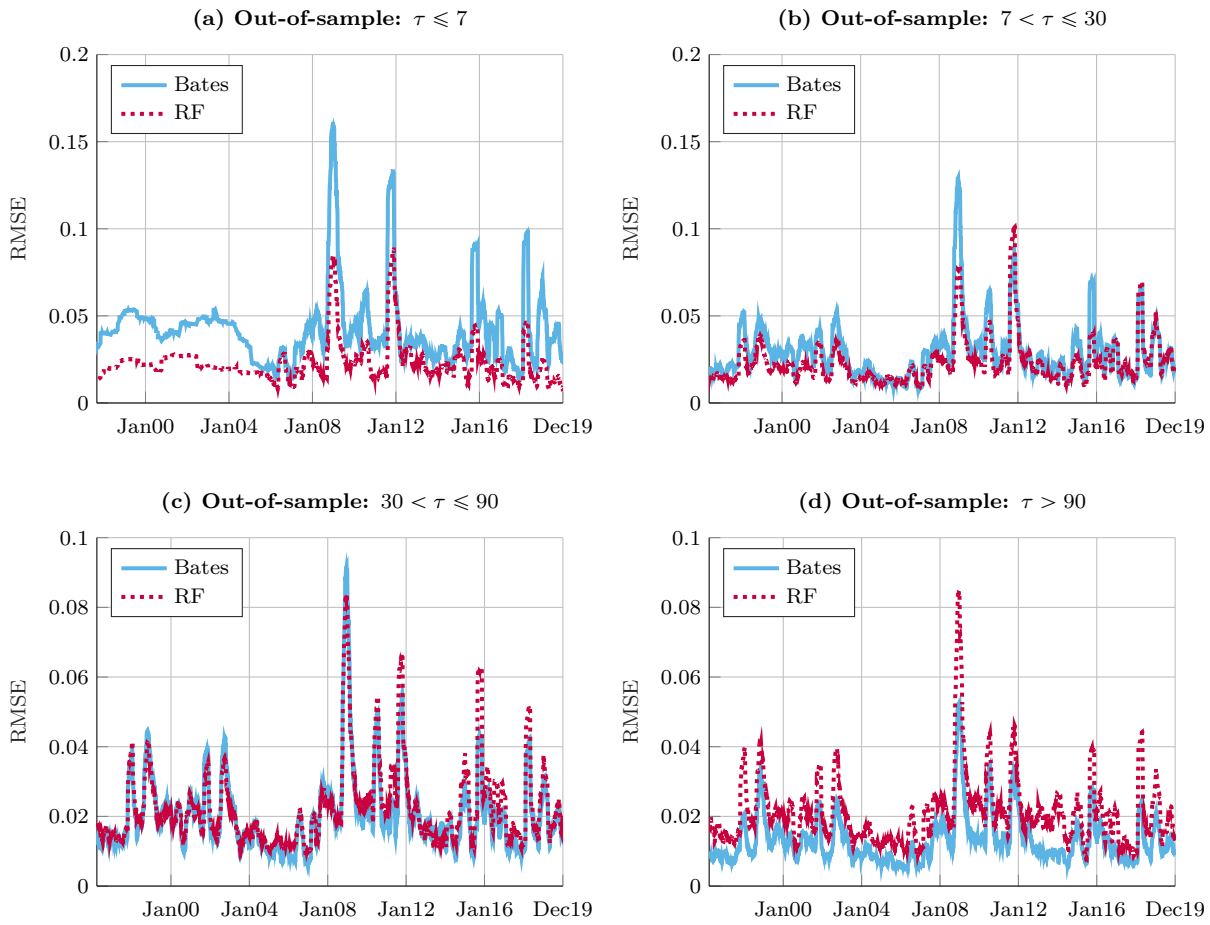


Figure 6: **Pricing performances of the Bates model and RF model over time.** This figure shows the model's out-of-sample performances, as measured by averages of daily RMSEs over 60 days, across maturity brackets. For out-of-sample performance, the forecasting horizon is one week ($h = 5$).

Finally, we also compare the out-of-sample hedging performances for the Bates model against that for the RF model the Black-Scholes model. We measure the hedging errors for contract i from date $t-1$ to t as

$$\epsilon_{i,t}^{(m)} = \frac{(p_{i,t} - p_{i,t-1}) - \delta_{i,t-1}^{(m)}(S_t - S_{t-1})}{p_{t-1}}, \quad m \in \{\text{Bates}, \text{BS}, \text{RF}\}, \tag{23}$$

where $\delta_{i,t-1}^{(m)}$ is the model-implied delta from $t-1$, S_t is the closing price of the underlying asset at time t, and $p_{i,t}$ is the mid quote of option i at time t. We standardize the error by the option price at time $t-1$.

Among the three models, the RF model produces the highest mean absolute hedging

error, likely because it is trained to match option prices but does not face any constraint on the delta it produces. The Bates model consistently produces the smallest hedging errors, even for short-dated options, where they produce larger pricing errors. This finding echoes the finding of Schaefer and Strebulaev (2008) in the corporate bond market. It shows that structural models are informative about out-of-sample hedge ratios even though they might imply large pricing errors.

4.4 Distribution of Option Returns

The last exercise we conduct with the surrogate model is to characterize the conditional distribution of option returns according to the Bates model and confront it with the data. Because option prices are nonlinear functions of the underlying state, characterizing the distribution of option returns typically requires simulating the state variables and then pricing the options under (a large number of) different realizations of the states. One would also use a similar procedure in simulated method of moment (SMM) estimations or when computing the Value-at-Risk of an option portfolio. We show that this procedure can be significantly expedited with the surrogate.

We follow Israelov and Kelly (2017) to model the dynamics of the underlying states using a vector autoregression (VAR). We denote the price of the underlying S&P500 index by S_t, and denote the first principle component of the implied volatility surface by PC_t. We then model the vector of states, which consists of S&P500 returns, the log VIX index, and the PC of the volatility surface, $X_t = \left[\frac{S_t}{S_{t-1}} - 1, \ln \text{VIX}_t, \text{PC}_t'\right]'$, as

$$X_t = \mu + \rho X_{t-1} + \Sigma_{t-1}\epsilon_t, \tag{24}$$

where the innovations ϵ_t are IID with mean zero and unit variance, and Σ_t is a diagonal matrix of GARCH$(1,1)$ volatilities. Israelov and Kelly (2017) model the volatility surface as driven by a factor model with static loadings on X_t and then price the options by interpolating the volatility surface. We also use the VAR and bootstrapped errors to forecast future changes in option moneyness and spot volatility, but price the options using the Bates model.

Specifically, we estimate the VAR in (24) on each day t using expanding windows with a

minimum of 1000 observations. We then bootstrap M residuals which are fed into Eq. (24) to construct the forecast for day $t+1$,

$$\hat{X}^b_{t+1} = \hat{\mu} + \hat{\rho} X_t + \hat{\Sigma}_t \hat{\epsilon}^b_{t+1}, \quad b = 1, \cdots, M. \tag{25}$$

Using the same procedure, we can also obtain the forecast at longer horizon, $\hat{X}^b_{t+h}$. Next, we recover the forecast for instantaneous variance v_{t+h} in the Bates model by assuming that the change in log spot volatility is expected to be the same as that of the VIX index, so that $\sqrt{v^b_{t+h}} = \frac{VIX^b_{t+h}}{VIX_t}\sqrt{v_t}$.

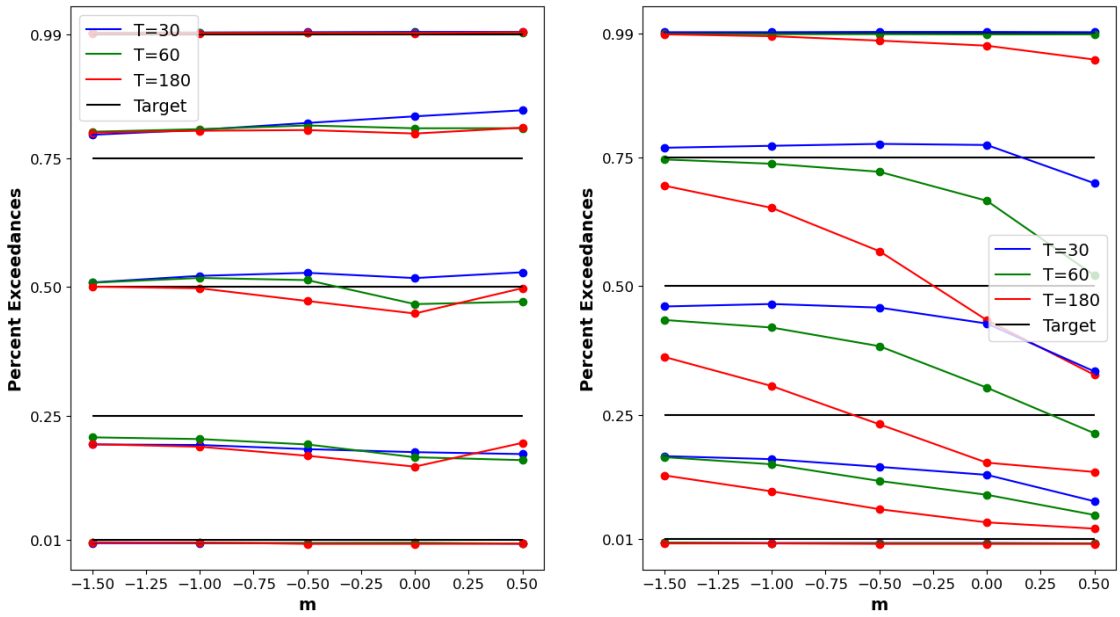
Finally, we compute the bootstrapped BSIV of an option contract i at time $t+h$ as:

$$BSIV^b_{i,t+h} = \widehat{f}\left(s^b_{i,t+h}, \theta_t \mid \phi^*\right), \tag{26}$$

where $s^b_{i,t+h} = [m^b_{i,t+h}, \tau_i, v^b_{t+h}, r_{i,t}, d_{i,t}]$ (we keep the maturity-dependent risk-free rate and dividend yield at their values at time t), θ_t contains the parameters of the model estimated at time t, and $\widehat{f}$ denotes the deep surrogate. The BSIVs can then be readily converted into option prices and returns.

Any mismatch between the model-implied conditional distribution and the data could be due to one of the following reasons: 1) the VAR model for the underlying states could be misspecified; 2) parameter instability of the structural model; and 3) other misspecification of the structural model. We capture the mismatch as follows. For any option i on date t, we generate its conditional distribution of the price h days ahead using the bootstrap method above. Following Israelov and Kelly (2017), we examine how well this model-implied conditional distribution fits the data through quantile forecasts, that is, by checking the frequency with which the actual prices fall below the conditionally forecasted quantiles. For example, if the model is correct, the actual option prices on day $t+h$ should fall below the 50th percentile, which is generated based on information on day t, 50% of the time.

For ease of illustration, we choose three target tenors, with days-to-maturity of $\tau = [30, 60, 180]$, and five target moneyness, $m = [-1.5, -1, -0.5, 0, 0.5]$. For each day, we group the options that deviate by less than 10% from the target tenor and moneyness into their corresponding category (15 groups in total). We then compute the forecasted quantiles for



(a) Bates (b) Heston

Figure 7: **Quantile forecasts vs. the data.** The figures show the performances of the conditional option return distribution from the Bates model (a) and Heston model (b) based on the percentage exceedance at various target quantiles.

each option and report the percentage exceedance, that is, the percentage of option-days in the sample for which the predicted quantile exceeds the actual prices.

Panel A of Figure 7 reports the results for the Bates model. The forecasting horizon is one day. On the y-axis, we display the target quantile (black line) and the percentage exceedance. The blue, green, and red lines represent the three different target tenors. For each tenor, we show the percentage exceedance at the 1st, 25th, 50th, 75th, and 99th percentile across the five different target moneyness. For comparison, we conduct the same analysis for the stochastic volatility model by Heston (1993),[16] which does not feature jumps in the underlying asset. The results for the Heston model are reported in Panel B.

This test based on the conditional distribution of option returns is highly effective in demonstrating the limitations of the Bates model as well as its advantages over the Heston model. For the different target quantiles (especially at the 25th, 50th, and 75th percentile), the Bates model's percentage exceedance matches the target reasonably well. For the 25th percentile, the model-predicted quantile is likely too low, while the opposite is true for the

[16] We build a separate deep surrogate for the Heston model for this exercise.

75th percentile, which could suggest more flexibility is needed in the Bates model when modeling the two sides of the jump distribution.

In contrast, across the quantiles, the percentage exceedance of the Heston model mostly falls short of the target at different maturity and moneyness. The shortfall is especially severe for longer-dated options, and for out-of-money calls. These issues are in part due to the fact that parameter instability is more severe for the Heston model, but they also reflect the Heston model's struggle to fit different parts of the volatility surface. The severe mismatch between the model-implied return distribution and the data have relevant consequences for practice. For example, the value-at-risk estimate for a portfolio of long-dated options based on the Bates model is likely to be significantly more accurate than one from the Heston model.

5 Conclusion

In this paper, we introduce *deep surrogates*: a generic framework to swiftly estimate complex structural models in economics and finance. We treat the model parameters as pseudo-state variables to create a deep neural network surrogate that replicates the target model. By alleviating the curse of dimensionality, this surrogate approach considerably reduces the computational cost for prediction and parameter estimation.

Our application, the estimation and analysis of a surrogate of the Bates option pricing model, illustrates how the *deep surrogate* methodology paves the way for a vast array of previously infeasible applications of structural models in finance and economics requiring including high-frequency re-estimation of parameters. In addition to enabling previously computationally infeasible analyses, deep surrogates can foster collaborations among academics. Indeed, trained surrogates can easily be shared by researchers, allowing others to utilize their model, thereby enhancing the accessibility and reproducibility of academic research.

Acknowledgments

We thank Lorenzo Bretscher, Xiaohong Chen, Mathieu Fournier, Bryan Kelly, Andrew Lo, Semyon Malamud, Dmitriy Muravyev, John Rust, Olivier Scaillet, Fabio Trojani, seminar participants at the University of Geneva, the University of Lausanne, MIT, and participants of the 2021 CICF, the 2021 Econometrics Society Winter Meeting, the SIAM conference 2021, the PASC 2021 conference, the Econometric Society Summer School and Conference on Dynamic Structural Econometrics 2021 at Bonn, and the Econometric Society Summer School and Conference on Deep Learning for Solving and Estimating Dynamic Models 2023 at Lausanne for comments. This work is generously supported by grants from the Swiss National Supercomputing Center (CSCS) under project ID 995, and the Swiss National Science Foundation under project IDs "Can Economic Policy Mitigate Climate-Change?," and "New methods for asset pricing with frictions."

References

Aldrich, E. M., J. Fernández-Villaverde, A. R. Gallant, and J. F. Rubio-Ramírez. 2011. Tapping the supercomputer under your desk: Solving dynamic equilibrium models with graphics processors. *Journal of Economic Dynamics and Control* 35:386–93.

Andersen, T. G., N. Fusari, and V. Todorov. 2015. Parametric inference and dynamic state recovery from option panels. *Econometrica* 83:1081–145.

———. 2017. Short-term market risks implied by weekly options. *The Journal of Finance* 72:1335–86.

Azinovic, M., L. Gaegauf, and S. Scheidegger. 2022. Deep equilibrium nets. *International Economic Review* n/a. doi:https://doi.org/10.1111/iere.12575.

Bach, F. 2017. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research* 18:1–53.

Bates, D. S. 1996. Jumps and stochastic volatility: Exchange rate processes implicit in deutsche mark options. *The Review of Financial Studies* 9:69–107.

———. 2000. Post-'87 crash fears in the S&P 500 futures option market. *Journal of Econometrics* 94:181–238.

Berner, J., P. Grohs, and A. Jentzen. 2020. Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of black–scholes partial differential equations. *SIAM Journal on Mathematics of Data Science* 2:631–57.

Bilionis, I., and N. Zabaras. 2012. Multidimensional adaptive relevance vector machines for uncertainty quantification. *SIAM Journal on Scientific Computing* 34:B881–908.

Black, F. 1976. Studies of stock price volatility changes. *In: Proceedings of the 1976 Meetings of the American Statistical Association* 171–81.

Bollerslev, T., and V. Todorov. 2011. Tails, fears, and risk premia. *The Journal of Finance* 66:2165–211. doi:10.1111/j.1540-6261.2011.01695.x.

Bollerslev, T., V. Todorov, and L. Xu. 2015. Tail risk premia and return predictability. *Journal of Financial Economics* 118:113–34.

Breiman, L. 2001. Random forests. *Machine Learning* 45:5–32.

Bretscher, L., J. Fernandez-Villaverde, and S. Scheidegger. 2022. Ricardian business cycle. *Available at SSRN 4278274* .

Brumm, J., and S. Scheidegger. 2017. Using adaptive sparse grids to solve high-dimensional dynamic models. *Econometrica* 85:1575–612. ISSN 1468-0262. doi:10.3982/ECTA12216.

Chen, H., W. W. Dou, and L. Kogan. 2022. Measuring "dark matter" in asset pricing models. *Journal of Finance*, forthcoming.

Chen, H., and S. Joslin. 2012. Generalized Transform Analysis of Affine Processes and Applications in Finance. *Review of Financial Studies* 25:2225–56.

Chen, L., M. Pelger, and J. Zhu. 2023. Deep learning in asset pricing. *Management Science* .

Chen, P., N. Zabaras, and I. Bilionis. 2015. Uncertainty propagation using infinite mixture of gaussian processes and variational bayesian inference. *Journal of computational physics* 284:291–333.

Chen, X., and H. White. 1999. Improved rates and asymptotic normality for nonparametric neural network estimators. *IEEE Transactions on Information Theory* 45:682–91.

Christoffersen, P., and K. Jacobs. 2004. The importance of the loss function in option valuation. *Journal of Financial Economics* 72:291 – 318. ISSN 0304-405X. doi:https://doi.org/10.1016/j.jfineco.2003.02.001.

Deisenroth, M. P., C. E. Rasmussen, and J. Peters. 2009. Gaussian process dynamic programming. *Neurocomputing* 72:1508–24.

Duarte, V. 2018. Machine learning for continuous-time economics Working paper.

Duffie, D., J. Pan, and K. Singleton. 2000. Transform analysis and asset pricing for affine jump-diffusions. *Econometrica* 68:1343–76. doi:10.1111/1468-0262.00164.

Farrell, M. H., T. Liang, and S. Misra. 2021. Deep neural networks for estimation and inference. *Econometrica* 89:181–213–. ISSN 0012-9682. doi:10.3982/ecta16901.

Fernandez-Villaverde, J., S. Hurtado, and G. Nuno. 2019. Financial Frictions and the Wealth Distribution. PIER Working Paper Archive 19-015, Penn Institute for Economic Research, Department of Economics, University of Pennsylvania.

Fernandez-Villaverde, J., G. Nunu, G. Sorg-Langhans, and M. Vogler. 2020. Solving high-dimensional dynamic programming problems using deep learning. Working Paper.

Fernández-Villaverde, J., J. Rubio-Ramírez, and F. Schorfheide. 2016. None. vol. 2 of *Handbook of Macroeconomics*, 527 – 724. Elsevier. doi:https://doi.org/10.1016/bs.hesmac.2016.03.006.

Gao, J., and J. Pan. 2023. Option-implied crash index. Working Paper.

Goodfellow, I., Y. Bengio, A. Courville, and Y. Bengio. 2016. *Deep learning*, vol. 1. MIT press Cambridge.

Grohs, P., F. Hornung, A. Jentzen, and P. von Wurstemberger. 2023. A proof that artificial neural networks overcome the curse of dimensionality in the numerical approximation of black–scholes partial differential equations. *Memoirs of the American Mathematical Society* 284.

Gu, S., B. Kelly, and D. Xiu. 2018. Empirical asset pricing via machine learning. Working Paper, National Bureau of Economic Research.

Gühring, I., M. Raslan, and G. Kutyniok. 2020. Expressivity of deep neural networks. *arXiv preprint arXiv:2007.04759* .

Han, J., Y. Yang, et al. 2021. Deepham: A global solution method for heterogeneous agent models with aggregate shocks. Working Paper, arXiv preprint arXiv:2112.14377.

Hanin, B. 2019. Universal function approximation by deep neural nets with bounded width and ReLU activations. *Mathematics* 7:992–.

Heiss, F., and V. Winschel. 2008. Likelihood approximation by numerical integration on sparse grids. *Journal of Econometrics* 144:62 – 80. ISSN 0304-4076. doi:https://doi.org/10.1016/j.jeconom.2007.12.004.

Heston, S. L. 1993. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The review of financial studies* 6:327–43.

Hornik, K., M. Stinchcombe, and H. White. 1989. Multilayer feedforward networks are universal approximators. *Neural networks* 2:359–66.

Hutchinson, J. M., A. W. Lo, and T. Poggio. 1994. A nonparametric approach to pricing and hedging derivative securities via learning networks. *The Journal of Finance* 49:851–89.

Israelov, R., and B. T. Kelly. 2017. Forecasting the distribution of option returns. *Available at SSRN 3033242* .

Kaji, T., E. Manresa, and G. Pouliot. 2020. An Adversarial Approach to Structural Estimation. *arXiv e-prints* arXiv:2007.06169.

Kase, H., L. Melosi, and M. Rottner. 2022. Estimating nonlinear heterogeneous agents models with neural networks. *Available at SSRN 4138711* .

Krause, A., and C. Guestrin. 2007. Nonmyopic active learning of gaussian processes: an exploration-exploitation approach. In *Proceedings of the 24th international conference on Machine learning*, 449–56.

Liaw, A., M. Wiener, et al. 2002. Classification and regression by randomforest. *R news* 2:18–22.

Liu, S., A. Borovykh, L. A. Grzelak, and C. W. Oosterlee. 2019. A neural network-based framework for financial model calibration. *Journal of Mathematics in Industry* 9:9–.

Maliar, L., S. Maliar, and P. Winant. 2021. Deep learning for solving dynamic economic models. *Journal of Monetary Economics* 122:76–101. ISSN 0304-3932. doi:https://doi.org/10.1016/j.jmoneco.2021.07.004.

Montanelli, H., and Q. Du. 2019. New error bounds for deep relu networks using sparse grids. *SIAM Journal on Mathematics of Data Science* 1:78–92.

Montanelli, H., H. Yang, and Q. Du. 2021. Deep relu networks overcome the curse of dimensionality for generalized bandlimited functions. *Journal of Computational Mathematics* 39:801–15.

Nakkiran, P., G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever. 2021. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment* 2021:124003–.

Norets, A. 2012. Estimation of dynamic discrete choice models using artificial neural network approximations. *Econometric Reviews* 31:84–106. doi:10.1080/07474938.2011.607089.

Pan, J. 2002. The jump-risk premia implicit in options: evidence from an integrated time-series study. *Journal of Financial Economics* 63:3–50.

Park, J., and I. W. Sandberg. 1991. Universal approximation using radial-basis-function networks. *Neural computation* 3:246–57.

Petersen, P., and F. Voigtlaender. 2018. Optimal approximation of piecewise smooth functions using deep relu neural networks. *Neural Networks* 108:296–330. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2018.08.019.

Ramachandran, P., B. Zoph, and Q. V. Le. 2017. Swish: a self-gated activation function. *arXiv: Neural and Evolutionary Computing* .

Ren, P., Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang. 2021. A survey of deep active learning. *ACM computing surveys (CSUR)* 54:1–40.

Renner, P., and S. Scheidegger. 2018. Machine learning for dynamic incentive problems. *Available at SSRN 3282487* .

Schaefer, S. M., and I. A. Strebulaev. 2008. Structural models of credit risk are useful: Evidence from hedge ratios on corporate bonds. *Journal of Financial Economics* 90:1 – 19. ISSN 0304-405X. doi:https://doi.org/10.1016/j.jfineco.2007.10.006.

Scheidegger, S., and I. Bilionis. 2019. Machine learning for high-dimensional dynamic stochastic economies. *Journal of Computational Science* 33:68 – 82. ISSN 1877-7503. doi:https://doi.org/10.1016/j.jocs.2019.03.004.

Scheidegger, S., D. Mikushin, F. Kubler, and O. Schenk. 2018. Rethinking large-scale economic modeling for efficiency: Optimizations for gpu and xeon phi clusters. In *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 610–9. ISSN 1530-2075. doi:10.1109/IPDPS.2018.00070.

Scheidegger, S., and A. Treccani. 2018. Pricing American Options under High-Dimensional Models with Recursive Adaptive Sparse Expectations*. *Journal of Financial Econometrics* ISSN 1479-8409. doi:10.1093/jjfinec/nby024. Nby024.

Stephenson, C., and T. Lee. 2021. When and how epochwise double descent happens. *arXiv preprint arXiv:2108.12006* .

Traub, J. F., and A. G. Werschulz. 1998. *Complexity and information*, vol. 26862. Cambridge University Press.

Tripathy, R., I. Bilionis, and M. Gonzalez. 2016. Gaussian processes with built-in dimensionality reduction: Applications to high-dimensional uncertainty propagation. *Journal of Computational Physics* 321:191–223.

Tripathy, R. K., and I. Bilionis. 2018. Deep uq: Learning deep neural network surrogate models for high dimensional uncertainty quantification. *Journal of Computational Physics* 375:565 – 588. ISSN 0021-9991. doi:https://doi.org/10.1016/j.jcp.2018.08.036.

Villa, A. T., and V. Valaitis. 2019. Machine learning projection methods for macro-finance models. *Economic Research Initiatives at Duke (ERID) Working Paper* .

Williams, C. K., and C. E. Rasmussen. 2006. *Gaussian processes for machine learning*, vol. 2. MIT press Cambridge, MA.

Appendix

A Surrogate construction

Table A.1 shows the ranges of parameters used to generate the simulated samples for training the Bates and HM surrogates, as described in Section 3.2. Figure A.1 illustrates the quality of fit of the surrogate Bates model. We vary one state or parameter at a time while fixing the others at their respective averages, and then compare the BSIVs produced by the target model with those produced by the surrogate. Figure A.2 presents the results of the same analysis for the model's partial derivatives.

Table A.1: This table presents the ranges for the surrogate model of the HM and Bates's training sample. To create the surrogate models, we generate sample options. We first draw the option parameters, states, and hidden states from uniform distributions, and then use the original model to price those options. The table below shows the min and max values of these uniform distributions. The first two columns show the minimum and maximum values of the uniform distributions used to generate the Heston Model, while the last two columns show the same values for the Bates model.

j	HM: $\underline{x}^{(j)}$	HM: $\bar{x}^{(j)}$	Bates: $\underline{x}^{(j)}$	Bates: $\bar{x}^{(j)}$
m	-9.00	5.000	-9.00	5.000
rf	0.00	0.075	0.00	0.075
dividend	0.00	0.050	0.00	0.050
v_t	0.01	0.900	0.01	0.900
T	1.00	365.000	1.00	365.000
κ	0.10	50.000	0.10	50.000
θ	0.01	0.900	0.01	0.900
σ	0.10	5.000	0.10	5.000
ρ	-	-	-1.00	-0.000
λ	-	-	0.00	4.000
ν_1	-	-	0.00	0.400
ν_2	-	-	0.00	0.400
p	-	-	0.00	1.000

Figure A.1 and Figure A.2 illustrate how well the surrogate model matches the original model. We generate simulated samples by setting the model's parameters to $\hat{K} = 100.0, r = 0.0375, T = 190, \kappa = 25.05, \bar{v} = 0.455, v_t = 0.455, \sigma = 2.55$ and varying one parameter at a time across the range defined in Table A.1. We then use Bates's model and its corresponding surrogate to price each option (Figure A.1) and obtain its derivative (Figure A.2).[17] In both

[17] We use numerical derivatives to estimate the partial derivative of the "true" Bates, and the backward

cases, the surrogate matches perfectly the original model.

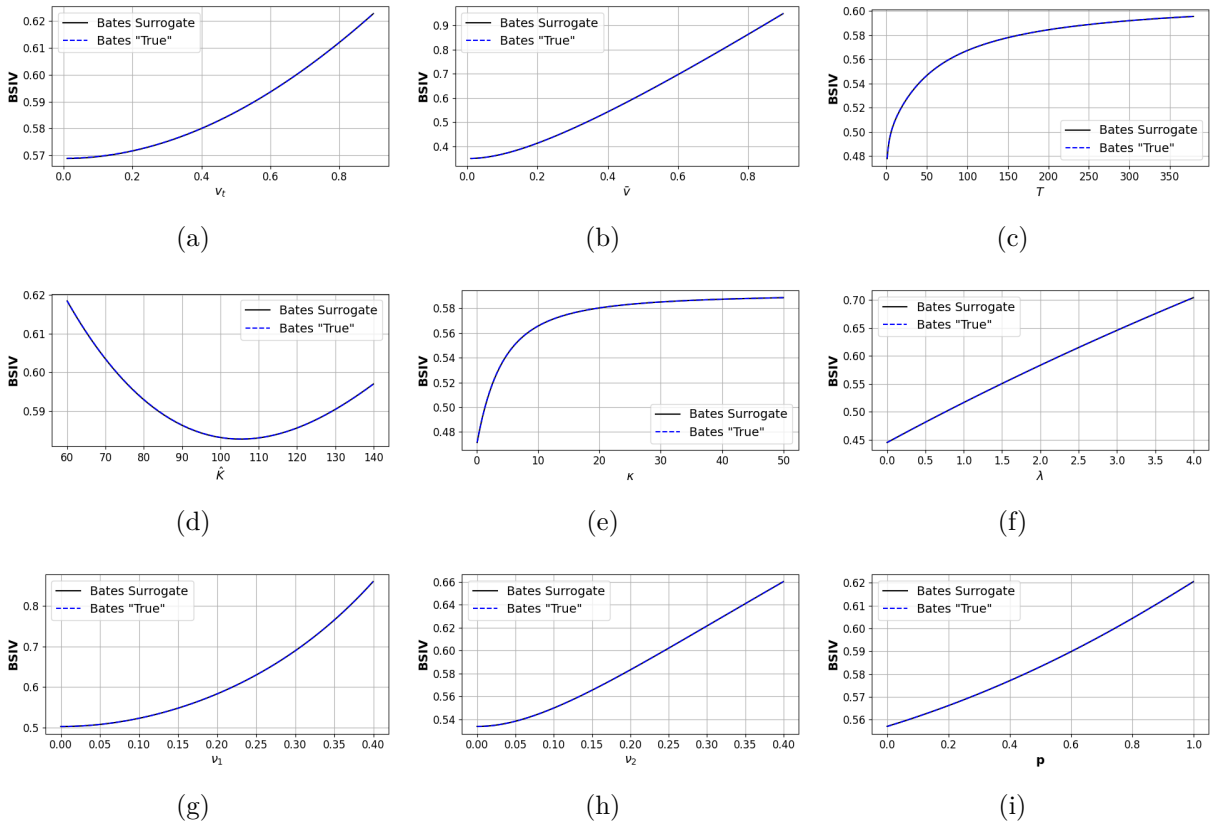


Figure A.1: The figures above show the complexity of the the Bates model's volatility surface as well as the quality of the surrogate's interpolations. To do so, the figures compare the BSIVs of the surrogate against the predictions from the "true" Bates. We populate the state space with points, where we keep all parameters and states fixed at the mid-range of possible values: $\hat{K} = 100.0, r = 0.0375, T = 190, \kappa = 25.05, \bar{v} = 0.455, v_t = 0.455, \sigma = 2.55$. On each panel, we show the BSIV predicted by the Bates and its surrogate while varying one of the parameters or states in its admissible range (cf. table A.1).

pass of the neural network to obtain the gradient estimate of the surrogate model.

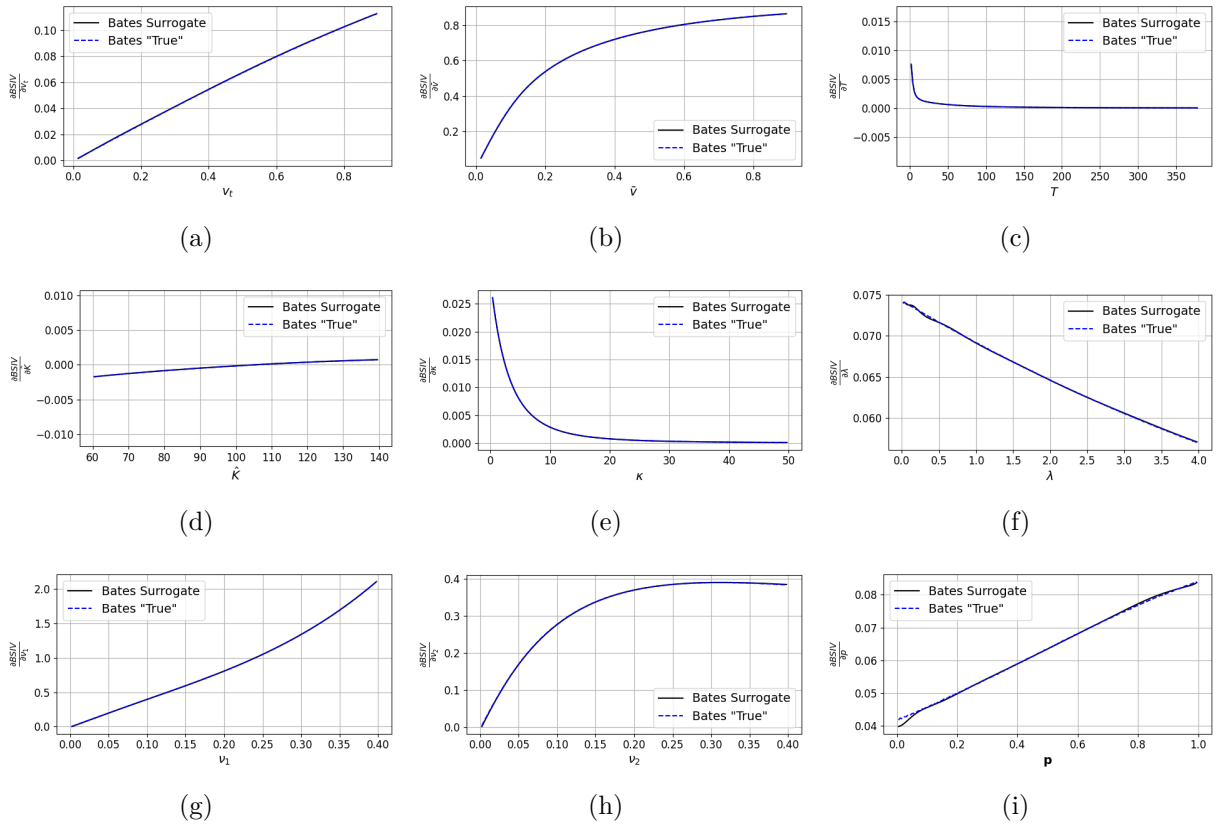
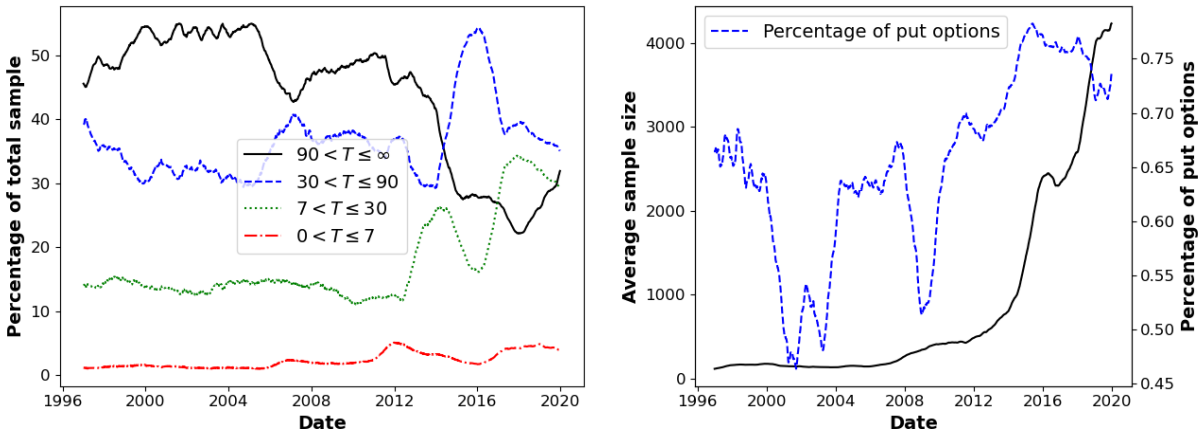


Figure A.2: The figures above show the complexity of the Bates models' volatility surface as well as the quality of the surrogate's interpolations. To do so, the figures compare the first derivative of the BSIVs of the surrogate against the predictions of the "true" Bates. We populate the state space with points, where we keep all parameters and states fixed at the mid-range of possible values: $\hat{K} = 100.0, r = 0.0375, T = 190, \kappa = 25.05, \bar{v} = 0.455, v_t = 0.455, \sigma = 2.55$. On each panel, we show the partial derivative predicted by the Bates and its surrogate while varying one of the parameters or states in its admissible range (cf. table A.1).

B Data sample

Figure A.3 provides a summary of our sample. Panel A shows the evolution of the distribution of time-to-maturity for the SPX options. Panel B shows a dramatic increase in the number of SPX options traded in the past decade (left axis). The percentage of put options in the sample (right axis) exceeds 50% most of the time and has risen above 70% in the past decade.



(a) Maturities in sample

(b) Number of options

Figure A.3: The two figures above show the evolution of the sample composition across time. Panel (a) shows the percentage of options in the sample across time per maturity brackets. Panel (b) shows the total number of options per day (left axis) and the percentage of put options in the sample (right axis). Through all plots, we smooth the results with a rolling average over the last 252 business days.