Deep Surrogates for Structural Models

Hui Chen[1] Antoine Didisheim[2] Simon Scheidegger[3]

[1]MIT
[2]University of Melbourne
[2]University of Lausanne

August 26, 2023

Outline

A Motivating Example

Bates Model

$$\frac{dS_t}{S_{t^-}} = (r - d - \lambda m)\,dt + \sqrt{v_t}\,dW_1 + dZ_t,$$
$$dv_t = \kappa(\theta - v_t)\,dt + \sigma\sqrt{v}\,dW_2, \quad \mathrm{E}[W_{1,t}W_{2,t}] = \rho\,dt,$$

where Z is a pure jump process with intensity λ and double-exponential log jump size J:

$$\omega(J) = p\frac{1}{\nu_u}e^{-\frac{1}{\nu_u}J}1_{\{J>0\}} + (1-p)\frac{1}{\nu_d}e^{\frac{1}{\nu_d}J}1_{\{J<0\}}, \text{ with } p \geq 0.$$

Pricing a European call option:

$$C(S_0, v_0, T) = \mathrm{E}_0\left[e^{-rT}(S_T - K)^+\right]$$

A Motivating Example

- Bates model:
  - ↪ Extension of the Black-Scholes model with stochastic volatility and double-exponential jumps.
  - ↪ 4 observable states ($S_t, K, T-t, r_t$), 1 hidden state (v_t), and 8 parameters ($\kappa, \theta, \sigma, \rho, \lambda, p, \nu_u, \nu_d$)
  - ↪ Solution technique: Fourier inversion of conditional characteristic function.

A Motivating Example

- Bates model:
  - ↪ Extension of the Black-Scholes model with stochastic volatility and double-exponential jumps.
  - ↪ 4 observable states $(S_t, K, T-t, r_t)$, 1 hidden state (v_t), and 8 parameters $(\kappa, \theta, \sigma, \rho, \lambda, p, \nu_u, \nu_d)$
  - ↪ Solution technique: Fourier inversion of conditional characteristic function.

- Potential applications:
  - ↪ Can option-implied jump risk measure predict the returns for individual stocks?
  - ↪ What is the impact of parameter instability for option pricing? Market making?
  - ↪ How to evaluate the model's out-of-sample (pricing or hedging) performance?
  - ↪ How well does the model match the empirical distribution of option returns?
  - ↪ How to compute value-at-risk for an option portfolio?

How costly is it?

- SPX: 4000 option prices on a typical day
- Horse race:
  - ↪ Modern FFT implementation vs. deep surrogate
  - ↪ Single core vs. GPU

	FFT	Deep Surrogate	Deep Surrogate + GPU
pricing, 1-day	10s		
estimation, 1-day	180s per iter		
estimation, 1-year	125h		

A standard solution in science and engineering: Look-up tables

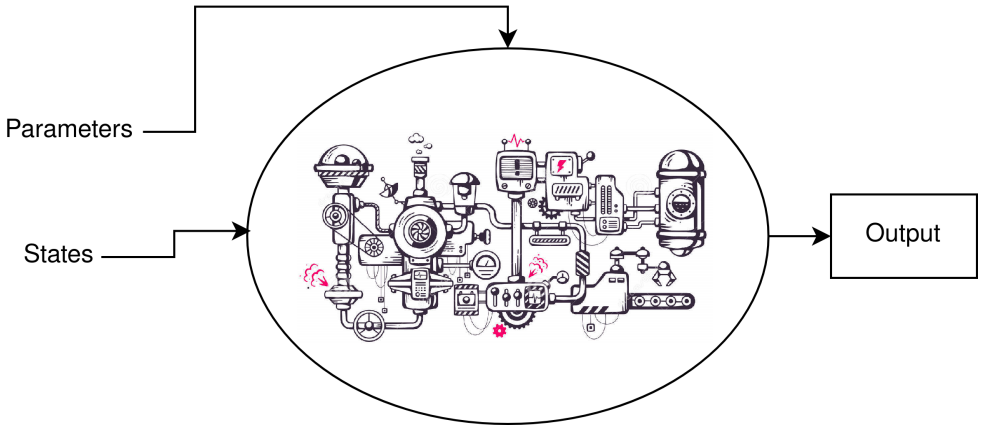
$$\Pr(z \leq z_1) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z_1} e^{-\frac{1}{2}z^2} dz$$

A standard solution in science and engineering: Look-up tables

$$\Pr(z \le z_1) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z_1} e^{-\frac{1}{2}z^2} dz$$

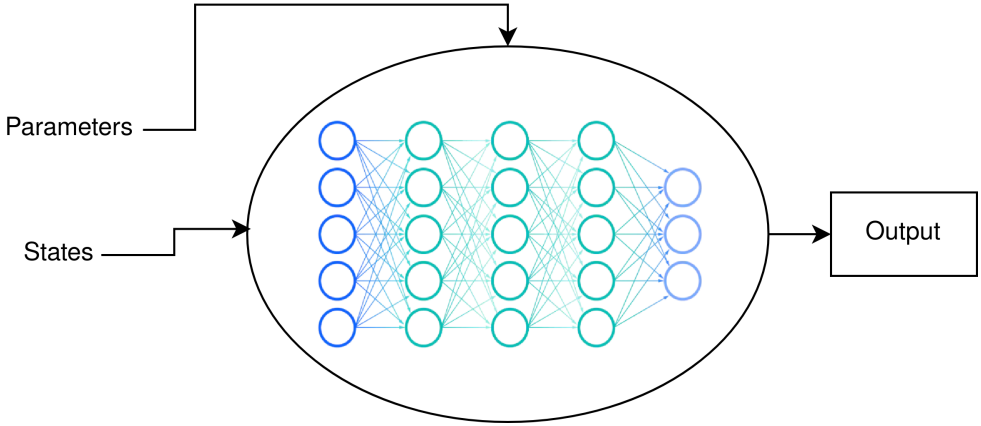
z_1	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857

Deep Surrogate: A 21st-Century “Look-up Table”



$$f(\underbrace{\Omega_t, H_t \mid \theta}_{\mathbf{x}_t}) = y_t$$

Deep Surrogate: A 21st-Century “Look-up Table”



$$\phi(\mathbf{x}_t|\theta_{NN}) = y_t$$

How costly is it?

- SPX: 4000 option prices on a typical day
- Horse race:
  - ↪ Modern FFT implementation vs. deep surrogate
  - ↪ Single core vs. GPU

	FFT	Deep Surrogate	Deep Surrogate + GPU
pricing, 1-day	10s	0.6s	
estimation, 1-day	180s per iter	3.2s per iter	
estimation, 1-year	125h	2.2h	

How costly is it?

- SPX: 4000 option prices on a typical day
- Horse race:
  - ↪ Modern FFT implementation vs. deep surrogate
  - ↪ Single core vs. GPU

	FFT	Deep Surrogate	Deep Surrogate + GPU
pricing, 1-day	10s	0.6s	0.06s
estimation, 1-day	180s per iter	3.2s per iter	0.3s per iter
estimation, 1-year	125h	2.2h	.2h

Outline

Problem formulation

- Let an economic model be represented by a set of restrictions between observable states $s \in \mathbb{R}^n$, hidden states $h \in \mathbb{R}^h$, and endogenous quantities $y \in \mathbb{R}^k$, with parameters $\theta \in \mathbb{R}^p$,

$$y = F(s, h|\theta)$$

- By treating θ as pseudo-states, we can focus on the augmented state $x \equiv (s, h, \theta) \in \mathbb{R}^d$ with $d = n + h + p$, and

$$y = f(x) = F(s, h|\theta)$$

- By imposing a hierarchical prior on θ, we can derive a model-implied joint probability distribution of x and y, $\mathbb{P}_{(\mathcal{X},\mathcal{Y})}$.

Deep Surrogate

- Ideally, for a given a loss function $\mathscr{L}$, we search in the set of all possible functions to mimize the expected loss of approximation:

$$\hat{f} = \arg\min \left\{ E\left[\mathscr{L}(f(x), y)\right] : f \in \mathscr{Y}^{\mathscr{X}} \right\}$$

  - ↪ But we already know $f \Rightarrow \hat{f} = f$

Deep Surrogate

- Ideally, for a given a loss function $\mathscr{L}$, we search in the set of all possible functions to mimize the expected loss of approximation:

$$\hat{f} = \arg\min \left\{ E\left[\mathscr{L}(f(x), y)\right] : f \in \mathscr{Y}^{\mathscr{X}} \right\}$$

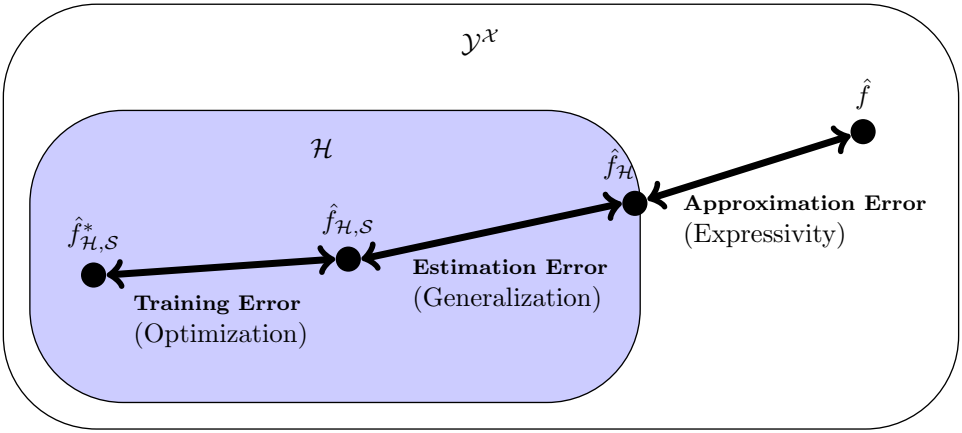
  ↪ But we already know $f \Rightarrow \hat{f} = f$

- Goal of deep surrogate: Generate a training set $S = ((x_i, y_i))_{i=1}^{m}$ of iid samples drawn from $\mathscr{X} \times \mathscr{Y}$; find a function from the set $\mathscr{H}$ of deep neural networks to minimize the empirical loss of approximation:

$$\hat{f}_{\mathscr{H},S} = \arg\min \left\{ \frac{1}{m} \sum_{i=1}^{m} \mathscr{L}(f(x_i), y_i) : f \in \mathscr{H} \right\}$$

  ↪ A neural network is characterized by activation function σ, architecture $a = (a_0 = d, a_1, \ldots, a_{L-1}, a_L = k)$, and parameterization $\theta_{NN} = \left((W_l, b_l)\right)_{l=1}^{L}$

Sources of approximation errors



Deep Surrogate – Expressivity

Universal approximation theorem

For every $f \in C$ and every $\epsilon > 0$, there exists a shallow neural network $\Phi_{f,\epsilon}$ with width N such that $\|\Phi_{f,\epsilon} - f\|_\infty \leq \epsilon$.

- Funahashi (1989), Hornik, Stinchcombe, and White (1989), Cybenko (1989)

UAT also applies to the "dual problem:" NNs with fixed width and unrestricted depth

- Hanin and Sellke (2017), Lu et al. (2017), Hanin (2019)
- Requires width $\geq d$

Deep Surrogate – Curse of dimensionality

- Approximation for general class of smooth functions typically require the number of parameters $P(\mathbf{a}_{d,\epsilon})$ of the networks grows exponentially in d, as does the size of the training sample.
  - ↪ For reference, approximating a d-dimensional function on a Cartesian grid requires $\mathcal{O}(n^d)$ evaluations.

- With additional regularity conditions and/or special structure, one can train an accurate surrogate with lower complexity and fewer observations.

- Berner et al. (2020): A deep neural network can overcome the curse of dimensionality when approximating the solution of a d-dim Kolmogorov equation with affine drift and diffusion.

$$\max\{m_{d,\epsilon}, P(\mathbf{a}_{d,\epsilon})\} \leq \text{poly}(d, 1/\epsilon)$$

  - ↪ Solution to the PDE can be written as expectation over terminal condition (Feynman-Kac).
  - ↪ Terminal values can be readily approximated by network with certain activation function.

Additional guidance form learning theory

- Activation: ReLU struggles with the approximation of certain functions (e.g., $x \to x^2$) relative to activation functions that are C^2.
  - ↪ Rolnick and Tegmark (2018)

- Depth: Advantage of deep neural networks over shallow ones.
  - ↪ Far more complex shallow networks might be needed to obtain the same approximation accuracy as a deep network.
  - ↪ Yarotsky (2017), Petersen and Voigtlaender (2018), Rolnick and Tegmark (2018)

Why Surrogates?

Deep surrogate is different from standard ML:

- Compared to other methods, deep neural networks are more hungry for data.
- The cost of producing a large training sample should be an important consideration.
- Unlike in standard ML, we know the true model ⇒ unlimited data (only limited by computational resources); essentially no errors.
- Double descent: Use a large number of epochs
  - ↪ Stephenson and Lee (2021); Nakkiran et al., (2021)

Why Surrogates?

Deep surrogate is different from standard ML:

- Compared to other methods, deep neural networks are more hungry for data.
- The cost of producing a large training sample should be an important consideration.
- Unlike in standard ML, we know the true model ⇒ unlimited data (only limited by computational resources); essentially no errors.
- Double descent: Use a large number of epochs
  - ↪ Stephenson and Lee (2021); Nakkiran et al., (2021)

Once trained, the deep surrogate

- is highly accurate;
- is cheaper to use by orders of magnitude; makes the gradients readily available;
- is easy to store (for 10^6 parameters - 20 MB vs. $\sim 10^6$GB when using Cartesian grid).

Why Surrogates?

Deep surrogate is different from standard ML:

- Compared to other methods, deep neural networks are more hungry for data.
- The cost of producing a large training sample should be an important consideration.
- Unlike in standard ML, we know the true model ⇒ unlimited data (only limited by computational resources); essentially no errors.
- Double descent: Use a large number of epochs
  - ↪ Stephenson and Lee (2021); Nakkiran et al., (2021)

Once trained, the deep surrogate

- is highly accurate;
- is cheaper to use by orders of magnitude; makes the gradients readily available;
- is easy to store (for 10^6 parameters - 20 MB vs. $\sim 10^6$GB when using Cartesian grid).

Pay the cost upfront; use it for free later.

- High quality surrogates can be shared with and build on by a community.
- Deep surrogates for workhorse quantitative economic and financial models.

Comparison of approximation methods

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

Application: Option pricing

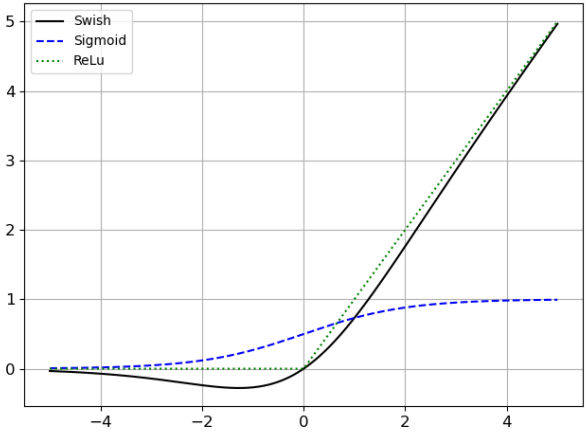
- We build deep surrogates for the Heston model (HM) and Bates model (BDJM), and analyze their empirical performances in several new dimensions.
- Key findings:
  1. Out of sample, RF outperforms parametric counterparts for short horizons (< 7 days) but underperforms at long horizons.
  2. Structural models are much better than RF in hedging performance.
  3. Significant evidence of time-varying parameter instability for HM and BDJM.
  4. Parameter instability linked to market liquidity.
  5. Tail risk estimated through BDJM is informative of future market returns.

Outline

Application

- Often times, $f(\cdot)$ can be computationally costly to evaluate.
- Objective of a surrogate: Replace $f(\cdot)$ by a surrogate model $\widehat{f}(\cdot)$, which
  - ↪ predicts the output of $f(\cdot)$ with high accuracy;
  - ↪ is cheap to evaluate.
- How do we know when we are done? Examine approximation errors out of sample.
- Active learning: Sample more from regions where approximation errors are large $\Rightarrow$ retrain $\Phi(\cdot)$.
- Theory-guided sampling: Sample strategically depending on the shape of f.

Swish: Continuous derivatives



$$Sigmoid: \frac{1}{1+e^{-x}}, \quad ReLu: max(x,0), \quad Swish: \frac{x}{1+\exp(-\gamma x)},$$

Building the surrogate

- Suppose the set of admissible inputs $\mathcal{X}$ is a hypercube, $[\underline{x}, \overline{x}]^d$.

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

- Draw x_i uniformly from $[\underline{x}, \overline{x}]^d$ and compute $y_i = f(x_i)$.
- For a given training sample of size m, we train the neural network $\phi(x|\theta_{NN})$ by minimizing the average ℓ_1-norm of the difference between the actual outcome and DNN prediction,

$$\theta^*_{NN} = \arg\min_{\theta_{NN}} \frac{1}{m} \sum_{i=1}^{m} |\phi(x_i|\theta_{NN}) - y_i|.$$

Building the surrogate

- To check the quality of the surrogate model, we generate a new random sample $\left(\mathbf{x}_j^o, \mathbf{y}_j^o\right)$ of size J and compute the validation error.
- A convergence criterion could be

$$\sup_j \|\phi(x_j^o|\theta_{NN}^*) - y_j^o\|_1 \leq \varepsilon \tag{†}$$

  for some $\varepsilon > 0$.
- If (†) is not satisfied, retrain ϕ with (potentially) a deeper layer and more training data (especially from regions where approximation error is large).

Architecture

HM (9-d)

- 6 fully connected layers of 400 neurones with *swish* activations.
- 806,001 trainable parameters.
- Custom input layer.
- Surrogate's training size: 10^8.

BDJM (13-d)

- 7 fully connected layers of 400 neurones with *swish* activations.
- 967,201 trainable parameters.
- Custom input layer.
- Surrogate's training size: 10^9.

Structural estimation using deep surrogate

- Once we have the deep surrogate, it is easy to use it to for structural estimation.
- For example, suppose we have a set of moment conditions

$$\mathrm{E}[f(\Omega_t|\theta)] = 0$$

- In finite sample, we have

$$g_n(\theta) = \frac{1}{n}\sum_{i=1}^{n} f(\Omega_t, \theta)$$

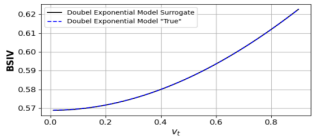
- GMM estimator:

$$\hat{\theta}_{GMM} = \arg\min_{\theta} g_n'(\theta)\, W\, g_n(\theta)$$

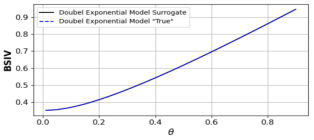
- We can use the deep surrogate $\phi()$ to replace $f()$. Moreover, through back-propagation and automatic differentiation the surrogate provides the gradients of $\phi(\cdot)$ analytically, which translate the GMM FOCs into a system of nonlinear equations.

Outline

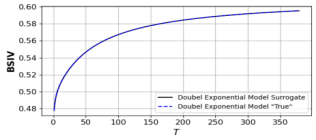
Accuracy of the deep surrogate: IV



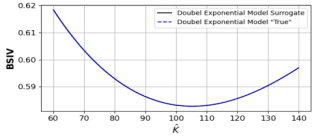
(a)



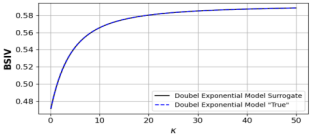
(b)



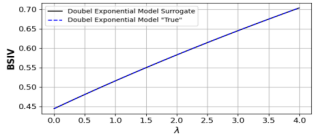
(c)



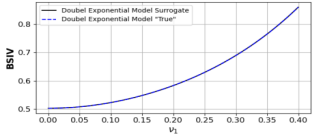
(d)



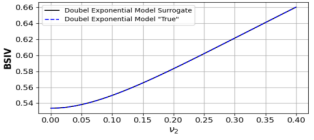
(e)



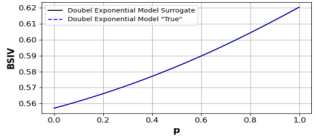
(f)



(g)

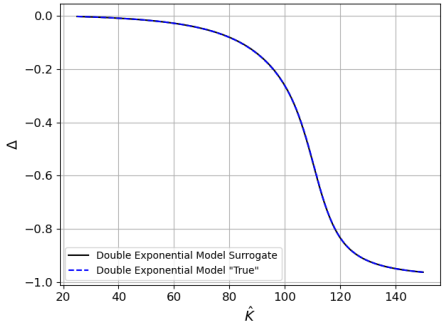


(h)

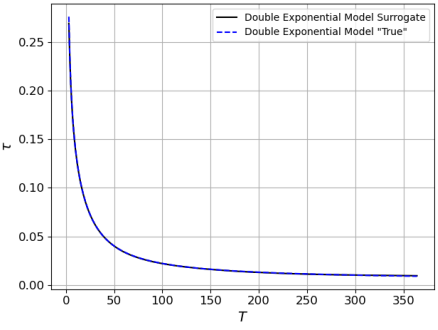


(i)

Accuracy of the deep surrogate: Greeks



(a) Delta

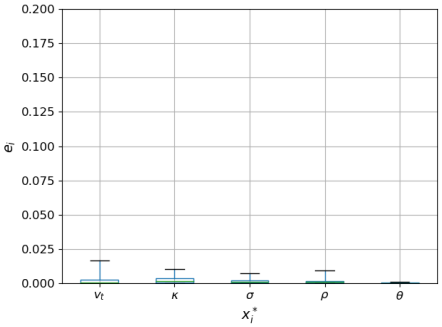


(b) Theta

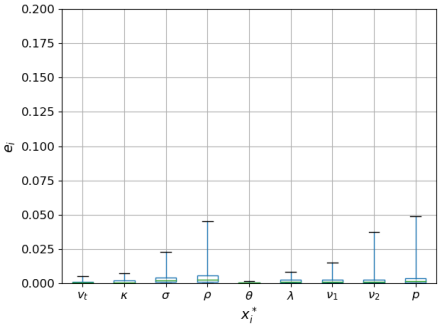
Identifying Parameters

To test our surrogate we,

- simulate N=1000 options with same parameters and hidden states, but randomly selected maturity and moneyness;
- use surrogate to estimate the parameters and hidden state;
- measure performance using relative estimation error: $e_i = \frac{|x_i^{true} - x_i^*|}{|\bar{x}_i - \underline{x}_i|}$.



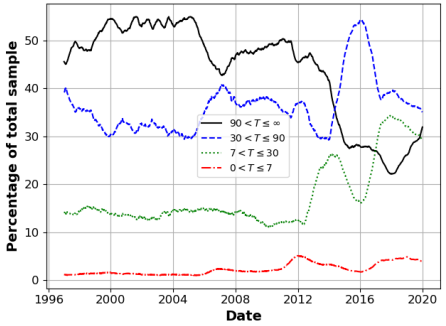
(c) HM



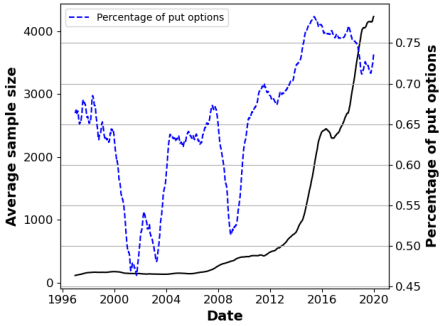
(d) BDJM

Data

- S&P500 index options in the years between 2001 and 2019.
- Remove options with maturity above 250 days.
- Remove options with extreme moneyness ($0.1 \leq \Delta \leq 0.9$).

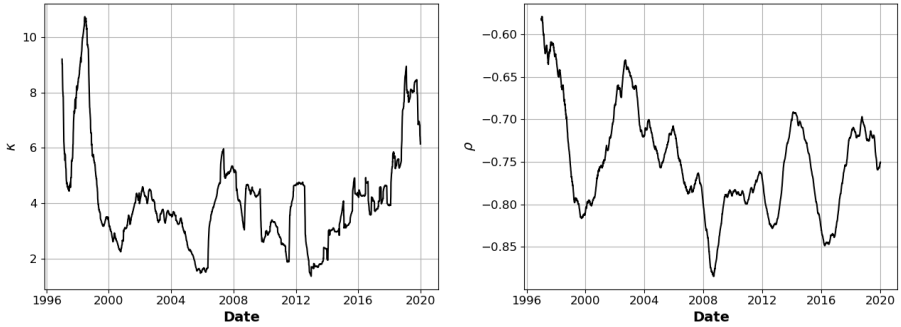


(e) Maturities in sample



(f) Number of options

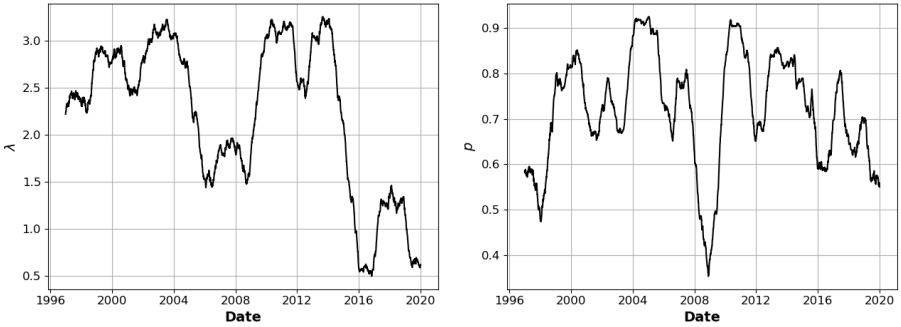
Parameter Estimates



(g) κ (h) ρ

Figure: The figures above show the daily estimation parameters of the BDJM. We smoothed the values with a rolling mean on the last 252 days.

Parameter Estimates

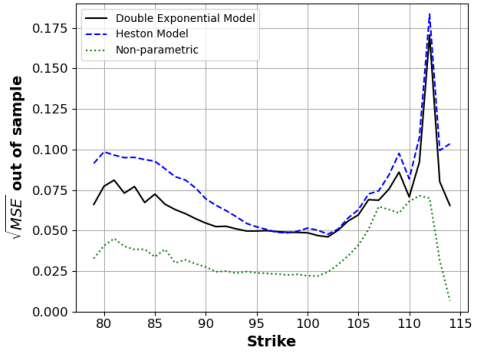


(a) λ (b) p

Figure: The figures above show the daily estimation parameters of the BDJM. We smoothed the values with a rolling mean on the last 252 days.

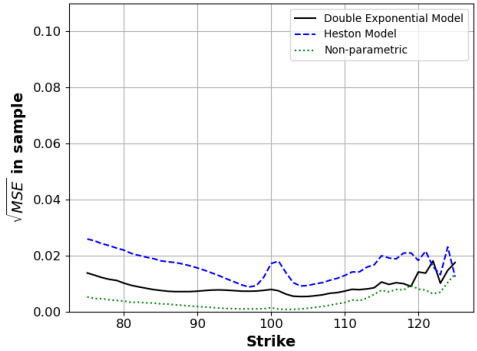
Forecasting the volatility surface I

(a) $0 < T \leq 7$, in-sample

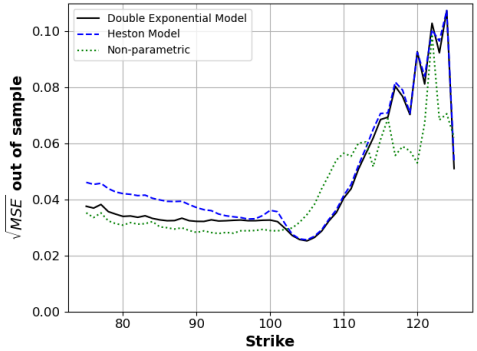


(b) $0 < T \leq 7$,out-of-sample

Forecasting the volatility surface II

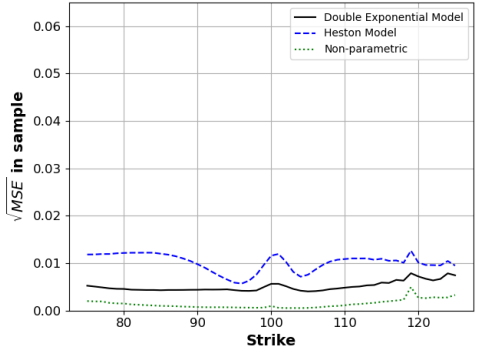


(c) $7 < T \leq 30$, in-sample

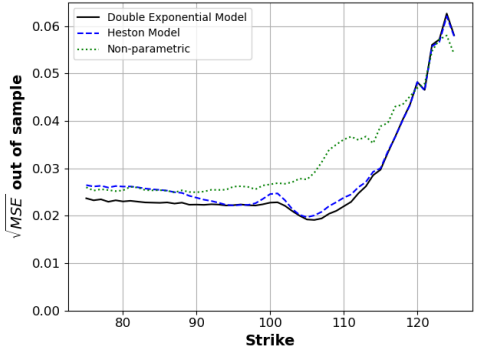


(d) $7 < T \leq 30$, out-of-sample

Forecasting the volatility surface III

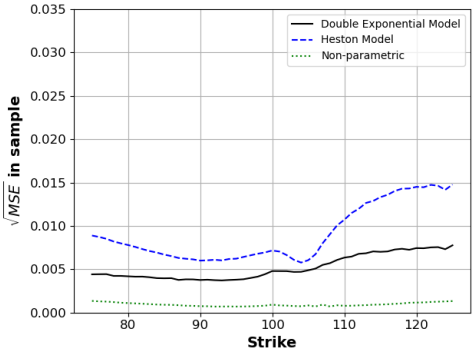


(e) $30 < T \leq 90$, in-sample

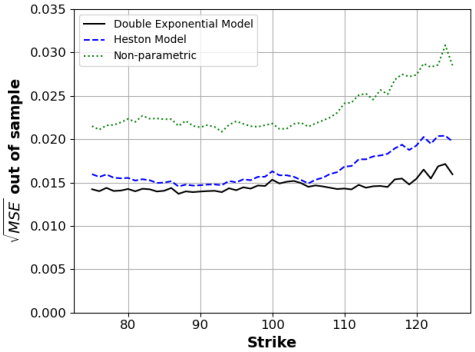


(f) $30 < T \leq 90$, out-of-sample

Forecasting the volatility surface IV



(g) $90 < T \leq \infty$, in-sample



(h) $90 < T \leq \infty$, out-of-sample

Hedging Performance

Table: Absolute replication error: $\left|\frac{(p_{i,t}-p_{i,t-1})-\delta_{i,t}^{(m)}(S_t-S_{t-1})}{p_{t-1}}\right|$

	BSM	RF	HM	BDJM
mean	0.19380	0.20133	0.15501	0.14175
std	0.42104	0.32173	0.39458	0.28051
5%	0.00423	0.00754	0.00601	0.00595
50%	0.08686	0.12512	0.07958	0.07914
95%	0.65096	0.59113	0.48294	0.44869

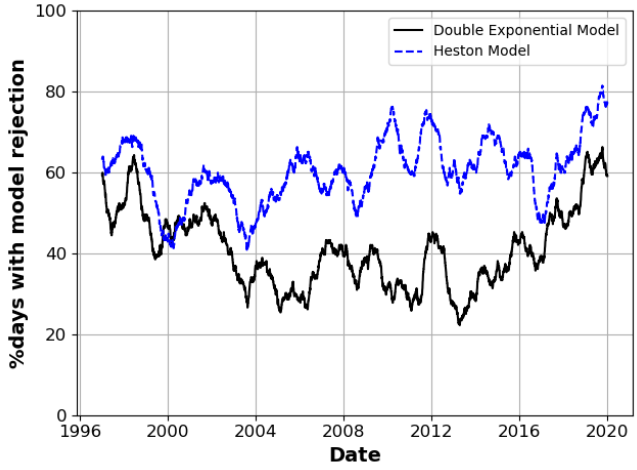
Parameter Stability

- We can test whether the models' parameters are stable across time.
- We use the test developed by Andersen, Fusari, and Todorov (2015):

$$(\Theta_t - \Theta_{t+1})' \left(\widehat{\text{Avar}}(\Theta_t) + \widehat{\text{Avar}}(\Theta_{t+1})\right)^{-1} (\Theta_t - \Theta_{t+1}) \xrightarrow{\mathscr{L}-s} \chi^2(q)$$

  - Θ_t, and Θ_{t+1} denote the estimated parameters in the time periods t and $t+1$, respectively.
  - $\widehat{\text{Avar}}(\Theta_t)$ and $\widehat{\text{Avar}}(\Theta_{t+1})$ denote consistent estimates of the asymptotic variances of these parameters.

Parameter Stability



(i) $\alpha = 1\%$

Parameter Stability and Market Liquidity

- Parameter instability increases the difficulty for hedging, which raises inventory risks for market makers.
- We examine whether parameter instability translates directly into larger bid-ask spreads. To do so, we estimate:

$$\frac{Ask_{t,i} - Bid_{t,i}}{Bid_{t,i}} = \alpha + \beta_1 \log \chi^2(q)_{i,t,BDJM} + \beta X_{i,t} + \epsilon_{i,t}$$

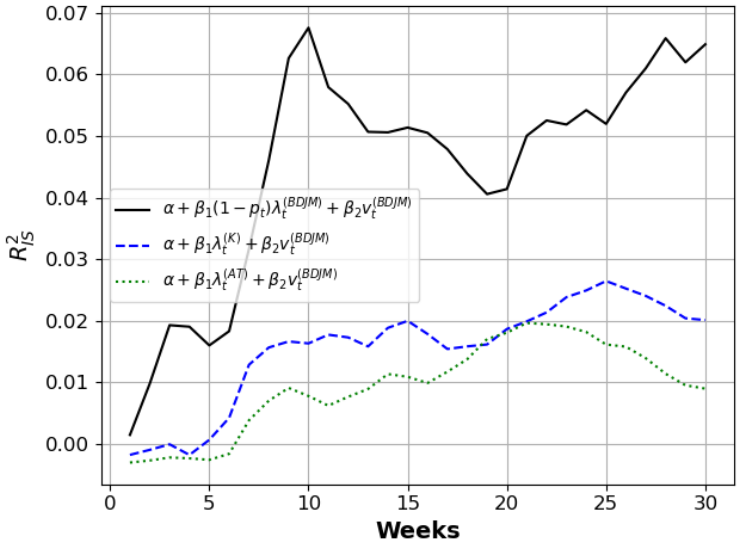
Parameter Stability and Market Liquidity

$\log \chi^2(q)_{BDJM}$		0.0012***		0.001***
		(0.0)		(0.0)
$\log \chi^2(q)_{HM}$			0.0012***	0.001***
			(0.0)	(0.0)
C_i	-0.1466***	-0.1462***	-0.1462***	-0.1459***
	(0.0004)	(0.0004)	(0.0004)	(0.0004)
$C_i * T_{i,t}$	0.0002***	0.0002***	0.0002***	0.0002***
	(0.0)	(0.0)	(0.0)	(0.0)
$C_i * K_{i,t}$	0.0001***	0.0001***	0.0001***	0.0001***
	(0.0)	(0.0)	(0.0)	(0.0)
$T_{i,t}$	-0.0004***	-0.0004***	-0.0004***	-0.0004***
	(0.0)	(0.0)	(0.0)	(0.0)
$JUMP_t$	-0.0089***	-0.0065***	-0.0076***	-0.0059***
	(0.0002)	(0.0002)	(0.0002)	(0.0002)
one	0.4073***	0.4076***	0.4056***	0.4062***
	(0.0003)	(0.0003)	(0.0003)	(0.0003)
$K_{i,t}$	-0.0001***	-0.0001***	-0.0001***	-0.0001***
	(0.0)	(0.0)	(0.0)	(0.0)
VIX_t	-0.0021***	-0.0023***	-0.0022***	-0.0023***
	(0.0)	(0.0)	(0.0)	(0.0)
$Volume_{i,t}$	-0.0***	-0.0***	-0.0***	-0.0***
	(0.0)	(0.0)	(0.0)	(0.0)
Observations	4,948,103	4,948,103	4,948,103	4,948,103
R^2	0.2176	0.218	0.218	0.2183

Market risk premium and tail risks I

- Kelly and Jiang (2014) showed that tail risks estimated through a power law on the cross-section of returns have strong predictive power for aggregate market returns.
- Andersen et al. (2020) further showed that similar results with tail risks estimated from index options prices.
- BDJM does estimate tail risks through parameters $\nu_d, \lambda, p, \nu, \rho$

Market risk premium and tail risks II



Outline

Conclusion

- Surrogate models are functions which replicate a structural-model at an exponentially lower computational cost.
- We treat states and parameters as pseudo-states and train a single neural network to build deep-surrogates of complex option pricing models.
- Thanks to those surrogate we can
  - ↪ compare the performance of these models with simple non-parametric and highlight shapes of the implied volatility surface where parametric models underperform;
  - ↪ assess parameter stability;
  - ↪ construct model-implied tail risk measures;
  - ↪ characterize conditional distribution of option returns.

Deep Surrogate Library

Git Repository

https://github.com/DeepSurrogate/

- Please give it a try!
- Next steps: Expand the offering of models in option pricing and other fields.
- Further explore the economic consequences of parameter instability and information content of tail risk measures.