Deep Hedging: Learning to Simulate Equity Option Markets

Magnus Wiese **Lianjun Bai** **Ben Wood*** **Hans Buehler**

J.P. Morgan†

Abstract

We construct realistic equity option market simulators based on generative adversarial networks (GANs). We consider recurrent and temporal convolutional architectures, and assess the impact of state compression. Option market simulators are highly relevant because they allow us to extend the limited real-world data sets available for the training and evaluation of option trading strategies. We show that network-based generators outperform classical methods on a range of benchmark metrics, and adversarial training achieves the best performance. Our work demonstrates for the first time that GANs can be successfully applied to the task of generating multivariate financial time series.

1 Introduction

There is growing interest in applying reinforcement learning techniques to the problem of managing a portfolio of derivatives [4, 20]. This involves buying and selling not only the relevant underlying assets, but also the available exchange-traded options. In order to train an option trading model, we therefore require time-series data that includes option prices.

Unfortunately, the amount of useful real-life data available is limited; if we take a sampling interval of one day, ten years of option prices translates into only a few thousand samples. This motivates the need for a realistic simulator: we can generate much larger volumes of data for training and evaluation while preserving the key distributional properties of the real samples, thus helping to avoid overfitting.

In this article, we build and test a collection of simulators based on neural networks (NNs). We begin by transforming the option prices to an equivalent representation in the form of *discrete local volatilities* (DLVs) [5, 25] with less complicated no-arbitrage constraints; we then formalize the role of the simulator as a mapping from a state and input noise to a new set of (transformed) option prices. Next, we define a series of benchmark scores based on the key distributional features of the transformed prices. We construct a set of generative models, varying the network architecture, training method, and state compression scheme. Finally, we evaluate these models against our proposed benchmark scores, and compare with a classical baseline.

2 Financial time series simulation

There is a wide range of existing literature on the generation of synthetic time series for asset prices (see, e.g., [10]). Classical derivative pricing models (e.g., [2, 8, 14]) also require path generators, but

*Corresponding author: ben.wood@jpmchase.com

†Opinions expressed in this paper are those of the authors, and do not necessarily reflect the view of J.P. Morgan.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

these are not designed to be realistic; they describe diffusion in the risk-neutral measure $\mathbb{Q}$, rather than the real-world measure $\mathbb{P}$, and are typically limited to a small number of driving factors, for ease of computation.

Recently, generative adversarial networks (GANs) [12] have been successfully used to create realistic synthetic time series for asset prices [16, 23, 24, 26, 27]. Zhang et al. [26] and Zhou et al. [27] reported on using the objective function of GANs to predict spot prices. Koshiyama et al. [16] trained a conditional GAN to generate spot log-returns and provided a study of using generated paths to fine-tune trading strategies, and showed that the autocorrelation function (ACF) and partial autocorrelation function (PACF) could be generated accurately by using a two-layer perceptron. Takahashi et al. [23] also reported on being able generate various stylised facts [6] found in the historical series, but did not provide a detailed description of the methodology used. An unconditional approach to the generation of spot price log-returns, using Temporal Convolutional Networks (TCNs) [19], was first presented by Wiese et al. [24]; they also reproduce the relevant stylised facts, including volatility clustering and the leverage effect.

In contrast, generative models for time series of option prices are much less common: Cont [6] performs a principal component analysis (PCA) on implied volatility data; Wissel [25] provides a scheme to build a risk-neutral market model, focusing on ensuring the martingale property rather than realism. As far as we are aware, neural networks have not previously been applied to option market generation.

Our work extends the conditional modelling framework of [16] to the multivariate setting by using GANs and other calibration techniques.

3 Option prices

Our aim in this article is to simulate the prices of standard "vanilla" equity index options, of the type commonly traded on exchanges in large volumes[3]. An option is characterized by: the underlying index; the type (*call* or *put*); the strike, K; and the maturity, T. The maturity is the expiry date of the option; on that date, the option holder receives an amount $\max\big(0, s(I_T - K)\big)$, where I_T is the prevailing level of the underlying index, and $s = 1$ for a call and -1 for a put.

At any given time, not all strike/maturity/type combinations are tradable; market makers quote bid and/or offer prices for only the most relevant combinations, which broadly means those with strike closest to the current index level, and maturity closest to today. We will work with a representative grid of strikes and maturities.

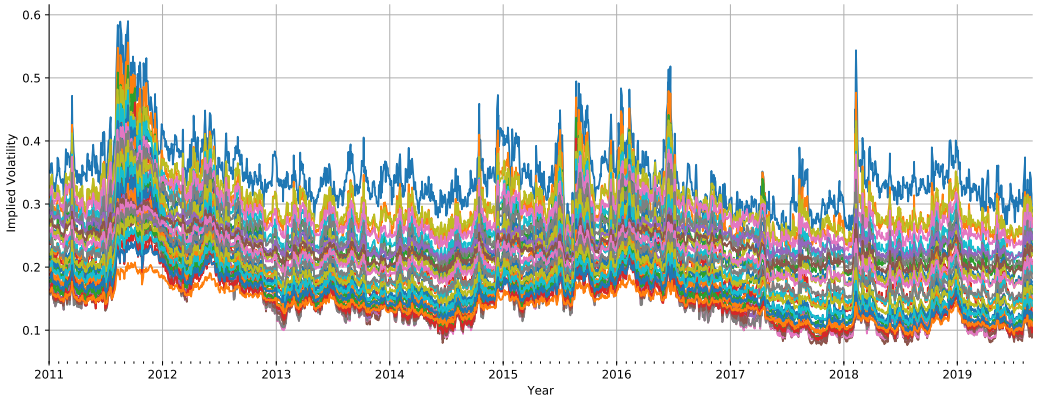

Figure 1: Implied volatilities of the EURO STOXX 50.

Market participants commonly express option prices in terms of *implied volatilities*. The implied volatility is the number one must use in the standard Black-Scholes formula [2] to obtain the option price. The other inputs to the formula are the discount factor and the index forward price. All three

[3]Eurex [9] reports an average of more than one million option contracts traded daily on EURO STOXX 50 in August 2019, corresponding to almost €650MM in premiums paid.

inputs to the price are stochastic, but the short-term price dynamics are primarily driven by changes in the implied volatility; in this paper, we focus on this contribution.

Option prices are subject to strict ordering constraints because of no-arbitrage considerations. For example, since the call option payoff is a non-increasing function of strike, the option price must also be non-increasing; violation of this rule would constitute an arbitrage opportunity[4], i.e. the possibility of making a profit while guaranteeing no loss. True arbitrage opportunities are rare, fleeting, and small; an option price simulator is more useful if it does not generate arbitrageable market states.

For this reason, it is not convenient to work with option prices directly; the ordering constraints are too complicated. The equivalent constraints on implied volatilities are even more awkward. Instead, we transform the prices to DLVs [5, 25], for which the absence of arbitrage corresponds to a simple requirement of positivity. In the absence of arbitrage, this mapping is bijective; an $N_K \times N_M$ grid of option prices is converted into an $N_K \times N_M$ grid of DLVs.

In this article we will focus on one of the largest option markets, namely options on the EURO STOXX 50 index. An overview of the distributional and dependence properties of DLVs is provided in Appendix B. The implied volatilities of the EURO STOXX 50 for the relevant sets of strikes and maturities are displayed in Figure 1.

4 Problem formulation

Throughout this paper $\mathbb{N}_0$ is the time set, $(\Omega, (\mathcal{F}_t)_{t\in\mathbb{N}_0}, \mathbb{P})$ the filtered probability space and $L^2(\mathbb{R}^N)$ denotes the space of $\mathbb{R}^N$-valued measurable functions for which the Euclidean norm $\|\cdot\|_2$ is Lebesgue integrable.

We assume a set of N_K equispaced strikes

$$\mathcal{K} = \{K_1, K_1 + \Delta K, \ldots, K_1 + (N_K - 1)\Delta K\}$$

and a set of N_M maturities

$$\mathcal{M} = \{M_1, \ldots, M_{N_M}\}$$

for which we obtain the $(N_K \cdot N_M)$-dimensional process of DLVs

$$\sigma_t := [\sigma_t(K, M)]_{(K,M)\in\mathcal{K}\times\mathcal{M}}, \text{ for } t \in \mathbb{N}_0.$$

Furthermore, we assume that the historical process $(\sigma_t, t \in \mathbb{N}_0)$ evolves through a conditional model which can be constructed by feeding in a state S_t, which we would like to condition on, and noise Z_{t+1} which drives the process. Particularly, we describe the evolution of $(\sigma_t, t \in \mathbb{N}_0)$ by a unkown mapping $g : L^2(\mathbb{R}^{N_Z}) \times L^2(\mathbb{R}^{N_S}) \to L^2(\mathbb{R}^{N_K \cdot N_M})$ which relates noise and state to the next time step, such that our process takes the form

$$\sigma_{t+1} = g(Z_{t+1}, S_t), \text{ for } t \in \mathbb{N}_0 \tag{1}$$

where $Z_{t+1} \sim \mathcal{N}(0, I)$ is N_Z-dimensional Gaussian noise and the state S_t a function of the processes history, i.e. $S_t = f(\sigma_t, \ldots, \sigma_0)$. An example of f is a projection onto the most current component, i.e. $S_t = \sigma_t$, the last L realisations, $S_t = [\sigma_t, \ldots, \sigma_{t-L}]$ or an exponentially weighted moving average.

The objective is to approximate the mapping $(Z_{t+1}, S_t) \mapsto \sigma_{t+1}$ which ideally allows us to generate more data from a given state S_t. In this paper our approach is to represent this mapping through a deep neural network

$$g : L^2\left(\mathbb{R}^{N_Z}\right) \times L^2\left(\mathbb{R}^{N_S}\right) \times \Theta \to L^2\left(\mathbb{R}^{N_K \cdot N_M}\right)$$

which is defined as a function of noise, state and its parameters $\theta \in \Theta$. For a given parameter vector $\theta \in \Theta$ the generated process $(\tilde{\sigma}_{t,\theta}, t \in \mathbb{N}_0)$ is defined as

$$\tilde{\sigma}_{t+1,\theta} = g_\theta(Z_{t+1}, \tilde{S}_{t,\theta}) \tag{2}$$

where $\tilde{S}_{t,\theta} = f(\tilde{\sigma}_{t,\theta}, \ldots, \tilde{\sigma}_{0,\theta})$. The optimal outcome is to approximate a parameter vector $\theta_{\text{ML}} \in \Theta$ such that $(\sigma_t, t \in \mathbb{N}_0)$ and $(\tilde{\sigma}_{t,\theta_{\text{ML}}}, t \in \mathbb{N}_0)$ inherit the same dynamics in terms of distributional and dependence properties.

[4] Strictly, there is an arbitrage opportunity when the bid price of the higher-strike call option exceeds the offer price of the lower-strike option.

5 Models

We now turn to the problem of modeling the process of DLV levels $(\sigma_t, t \in \mathbb{N}_0)$ and consider from now on log-DLV levels $X_t := \log(\sigma_t)$ for $t \in \mathbb{N}_0$ to ensure non-negativity of the generated time series.

The first (naive) approach to model log-DLVs is to approximate (2) directly and generate all DLVs yielding the generated time-series

$$\tilde{X}_{t+1,\theta} = g_\theta(Z_{t+1}, \tilde{S}_{t,\theta}) \tag{3}$$

where $\tilde{S}_{t,\theta} = [\tilde{X}_{t,\theta}, \ldots, \tilde{X}_{t-L,\theta}]$ includes current and past log-DLV levels for tuneable $L \in \mathbb{N}$. However, this approach suffers from having to model an arbitrarily high-dimensional process when it is necessary to generate a fine grid of DLVs yielding $N_X \gg 1$ for $N_X := N_K \cdot N_M$. Consequently, we also explore a compressed version of our generator.

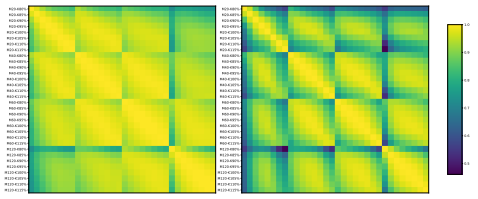

Figure 2: Cross-correlation matrix of log-DLV levels (left) and DLV log-returns (right). Labels on the y-axis indicate the maturity (M) and relative strike (K) of each row.

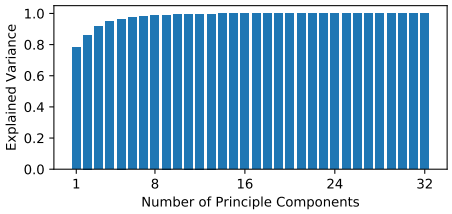

Figure 3: Cumulative sum of variance explained by the first 10 principal components.

The compressed version is motivated from the observation that log-DLV levels have high cross-correlations indicating that log-DLVs live on a lower-dimensional manifold. Figure 2 illustrates the cross-correlation matrix of log-DLVs and DLV log-returns for the set of relative strikes $\bar{\mathcal{K}} := \{0.80, 0.85, 0.90, 0.95, 1.00, 1.05, 1.10, 1.15\}$ and maturities $\bar{\mathcal{M}} := \{20, 40, 60, 120\}$. Both cross-correlation matrices show high cross-correlations between groups of different maturities as well as within a specific maturity.

We therefore apply PCA to the set of log-DLV levels $(X_t, t \in \mathbb{N}_0)$ and compress the N_X-dimensional process into an N_P-dimensional one. The compressed process is defined for $t \in \mathbb{N}_0$ as $P_t = \mathbf{W} X_t$, where $\mathbf{W} \in \mathbb{R}^{N_P \times N_X}$ denotes the compression matrix obtained through PCA. The generated process now takes the form

$$\begin{aligned} \tilde{P}_{t+1,\theta} &= g_\theta(Z_{t+1}, \tilde{S}_{t,\theta}) \\ \tilde{X}_{t+1,\theta} &= \mathbf{W}^\dagger \tilde{P}_{t+1,\theta} \end{aligned}$$

where $\tilde{S}_{t,\theta} = [\tilde{P}_{t,\theta}, \ldots, \tilde{P}_{t-L,\theta}]$ is the state of current and past compressed log-DLVs - again for tuneable $L \in \mathbb{N}$.

We also explored compressing DLV log-returns $R_t := X_t - X_{t-1},\ t \in \mathbb{N}_0$ as they are also highly correlated (see Figure 2). However, we discarded this approach since it involves taking the cumulative sum of the compressed DLV log-returns which causes a lossy compression for a small number of principle components.

Figure 3 illustrates the cumulative sum of variance explained as a function of the number of principle components for the grid $\bar{\mathcal{K}} \times \bar{\mathcal{M}}$. In section 8 we report our results for $N_P = 5$ principle components which explain $\approx 96\%$ of the variance yielding a good trade-off between dimensionality reduction and preserving information.

6 Optimization

The next step is to obtain a close approximation of θ_{ML}. Various training procedures exist ranging from conventional parametric methods such as quasi maximum likelihood estimation (qMLE) to

novel non-parametric ones such as GANs and the mean maximum discrepancy (MMD) [22]. In this paper, we explore the performance of qMLE, GANs and Wasserstein GANs (WGAN-GP) [13]. Following, we provide a brief explaination of the qMLE and GANs in the context of our explicit conditional model (3).

In qMLE we assume that the distribution of $X_{t+1}|(Z_{t+1}, S_t)$ can be described by a family of distributions $\mathcal{P} = \{\mathbb{P}_\gamma \mid \gamma \in \Gamma\}$ for some parameter space Γ. The objective is to maximize the likelihood of our generated data under our family $\mathcal{P}$ [11, Chapter 5]. Here, we assume that $X_{t+1}|(Z_{t+1}, S_t)$ follows a Gaussian distribution where we constrain our covariance matrix to be diagonal. Our parametric family thus takes the form

$$\mathcal{P} = \{\mathcal{N}(\mu, \Sigma) \mid \mu \in \mathbb{R}^{N_X}, \Sigma \in \mathbb{D}\}.$$

where $\mathbb{D} \coloneqq \{\mathrm{diag}(a_1, \ldots, a_{N_X}) \mid a_1, \ldots, a_{N_X} \in \mathbb{R}_{\geq 0}\}$ is the set of $(N_X \times N_X)$-dimensional diagional matrices with non-negative components.

A challenge in qMLE is the correct specification of $\mathcal{P}$ and the intractability of likelihood functions. GANs try to address this issue by introducing a min-max two-player game between the generator g_θ and the discriminator d_η. The discriminator aims to discriminate between real samples from the data distribution and synthetic ones generated by the generator. The objective function proposed by the original paper from Goodfellow [12] adapted to our setting is of the form

$$\mathcal{L}(\theta, \eta) = \mathbb{E}\left(\log(d_\eta([S_t, X_{t+1}]))\right) + \mathbb{E}\left(\log(1 - d_\eta([\tilde{S}_{t,\theta}, \tilde{X}_{t+1,\theta}]))\right)$$

where $\tilde{X}_{t+1,\theta}$ is defined as in (3).

A drawback of GANs is that they are notoriously hard to train which lead to the introduction of various regularization techniques to stabilize training [1, 3, 13, 17, 18, 21]. In our numerical results, we grid search over spectral normalization [18] imposed on the discriminator and generator [3] and gradient penalities proposed by Mescheder [17]. Furthermore, we also train our generative model by using WGAN-GP proposed by [13] and report on these results separately in section 8.

7 Evaluating the generated paths

In GANs the objective function cannot be used to evaluate the performance of the generator. Equally, using the likelihood of a qMLE-trained model can give a distorted image. To measure the goodness and performance of a generative model we define and introduce various metrics and scores. These scores allow us to capture whether the generator is able to generate dynamics that are similar to those found in the historical DLV series such as highly cross-correlated log-DLVs and DLV log-returns, bimodal distributions or persistence in the autocorrelation.

During training we intentionally evaluate log-DLVs instead of implied volatilities due two reasons. First, a close approximation of the generating mechanism of DLVs yields a close approximation of the generating mechanism of implied volatilities. Second, transforming DLVs to implied volatilities is costly. Once a close approximation of θ_{ML} is obtained we transform the generated DLVs to implied volatilities and compute metrics and scores for the generated implied volatilities and report on those.

We denote the historical dataset of log-DLVs by $\mathcal{D}_h = \{[x_0, \ldots, x_T]\}$ and likewise the generated dataset containing $M \in \mathbb{N}$ paths of length T through

$$\mathcal{D}_g = \left\{[\tilde{x}^{(i)}_{0,\theta}, \ldots, \tilde{x}^{(i)}_{T,\theta}]\right\}_{i=1}^{M}$$

where $[\tilde{x}^{(i)}_{0,\theta}, \ldots, \tilde{x}^{(i)}_{T,\theta}]$ denotes for any $i \in \{1, \ldots, M\}$ a time series obtained through recursive sampling from an initial state sampled from the historical dataset $\mathcal{D}_h$.

We begin by introducing a distributional metric and distributional scores, then define dependence scores and at last two scores that take into account the cross-correlation structure.

7.1 Distributional metric

Naturally, we want the unconditional distribution of the generated and historical to match closely. For this purpose, let $\mathcal{B}_h = \{B_1, \ldots, B_K\}$ be a binning such that approximately 20 elements of

the historical series $x \in \mathcal{D}_h$ fall into each bin; $\#\{t \in \{0,\dots,T\} : x_t \in B\} \approx 20$ for any $B \in \mathcal{B}_h$. With respect to the binning the empirical probability density function (epdf) of the historical $\hat{f}_h : \mathcal{B}_h \to \mathbb{R}_{\geq 0}$ and the generated $\hat{f}_g : \mathcal{B}_h \to \mathbb{R}_{\geq 0}$ can be defined. During training we track the absolute difference of the epdf

$$\sum_{B \in \mathcal{B}_h} |\hat{f}_h(B) - \hat{f}_g(B)|.$$

7.2 Distributional scores

In financial applications higher order moments such as the skewness and kurtosis are of interest as they determine the propensity to generate extremal values. We therefore define the skewness score

$$\frac{1}{N_X} \sum_{j=1}^{N_X} \left\| \text{skew}(x_{:,j}) - \text{skew}\left([\tilde{x}^{(1)}_{:,\theta,j}, \dots, \tilde{x}^{(M)}_{:,\theta,j}] \right) \right\|_2$$

where $\tilde{x}^{(i)}_{:,\theta,j}$ for $i \in \{1,\dots,M\}$ denotes the j-th dimension of the i-th generated time series and likewise the kurtosis score

$$\frac{1}{N_X} \sum_{j=1}^{N_X} \left\| \text{kurtosis}(x_{:,j}) - \text{kurtosis}\left([\tilde{x}^{(1)}_{:,\theta,j}, \dots, \tilde{x}^{(M)}_{:,\theta,j}] \right) \right\|_2 .$$

7.3 Dependence scores

Since DLVs are persistent we adopted ACF score proposed in [24]. It is defined by taking the Euclidean norm of the difference of the historical and the mean generated autocorrelation function

$$\text{ACF}_X \coloneqq \| \text{ACF}(x) - \frac{1}{|\mathcal{D}_g|} \sum_{\tilde{x}_{:,\theta} \in \mathcal{D}_g} \text{ACF}(\tilde{x}_{:,\theta}) \|_2.$$

Similarly, we define the ACF score for the log-return process $r_t = x_t - x_{t-1},\ t \in \{1,\dots,T\}$

$$\text{ACF}_R \coloneqq \| \text{ACF}(r) - \frac{1}{|\mathcal{D}_g|} \sum_{\tilde{r}_{:,\theta} \in \mathcal{D}_g} \text{ACF}(\tilde{r}_{:,\theta}) \|_2.$$

In section 8 we report the ACF_X score for the first 32 lags and the ACF_R score for the first lag.

7.4 Cross-correlation scores

To capture whether the generator generates cross-correlated log-DLVs and DLV log-returns we introduce two more scores. The cross-correlation score of log-DLVs is defined by taking the Euclidean norm of the cross-correlation matrix of log-DLVs $\|\hat{\Sigma}^X_h - \hat{\Sigma}^X_g\|_2$ where $\hat{\Sigma}^X_h, \hat{\Sigma}^X_g$ denote the cross-correlation matrix of the historical and generated respectively. Likewise, the cross-correlation score of DLV log-returns is defined $\|\hat{\Sigma}^R_h - \hat{\Sigma}^R_g\|_2$ where $\hat{\Sigma}^R_h, \hat{\Sigma}^R_g$ denote the cross-correlation matrix of the historical and generated DLV log-returns respectively.

8 Numerical results

In this section we evaluate the performance of qMLE-, GAN- and WGAN-GP-trained models for the compressed and explicit version.

8.1 Dataset

We use call option prices of the EURO STOXX 50 from 2011-01-03 to 2019-08-30 and consider the set of relative strikes and maturities

$$\bar{\mathcal{K}} \coloneqq \{0.80, 0.85, 0.90, 0.95, 1.00, 1.05, 1.10, 1.15\},$$
$$\bar{\mathcal{M}} \coloneqq \{20, 40, 60, 120\}.$$

For those option prices we compute the path of DLVs which we use for training (see Figure B.1). The total length of the time series is $\bar{T} \coloneqq 2257$.

8.2 Benchmarks

For comparison we apply the vector autoregressive model $\mathrm{VAR}(p)$ [15] and GAN-trained TCNs [24] to the same data. $\mathrm{VAR}(p)$ is a standard model for multivariate time series and assumes that X_{t+1} is an affine function of the past p observations and some Gaussian noise $Z_{t+1} \sim \mathcal{N}(0, \Sigma)$:

$$X_{t+1} = \mathbf{A}_1 X_t + \cdots + \mathbf{A}_p X_{t-p} + \mathbf{b} + Z_{t+1}.$$

TCNs model log-DLVs unconditionally. Here, the generated process takes the form

$$\tilde{X}_{t+1,\theta} = g_\theta(Z_{t+1}, \ldots, Z_{t+1-L})$$

where $(Z_t, t \in \mathbb{Z})$ is an i.i.d. Gaussian process and $L \in \mathbb{N}_0$ denotes the TCN's receptive field size (see [24]).

8.3 Training and evaluation time

We split the the historical series into a training and validation set through random sampling. The training set holds 85% of the data and is used to calibrate the parameters of the model. During training we compute the scores for $\bar{M} := 40$ generated paths of length $\bar{T}$. GAN- and WGAN-GP-trained models are evaluated every 100 generator gradient updates. qMLE-trained models are trained through early stopping and are only evaluated once after the criterion has been reached. The VAR model is also only evaluated once after the parameters are obtained through regression.

8.4 Results

Table 1 summarises the best scores obtained for each of the generative models for a fixed parameter vector. For all except the ACF_X score the explicit GAN-trained model achieves to generate the best paths.

Table 1: Scores obtained from historical implied volatilities and generated paths.

	Distributional			Dependence		Cross-Correlation	
Models	$\lvert\hat{f}_h(B) - \hat{f}_r(B)\rvert$	skew	kurtosis	ACF_X	ACF_R	$\lVert\hat{\Sigma}_h^X - \hat{\Sigma}_g^X\rVert_2$	$\lVert\hat{\Sigma}_h^R - \hat{\Sigma}_g^R\rVert_2$
GAN	**0.044**	**0.063**	**0.097**	0.186	**0.011**	**0.137**	**0.771**
GAN - PCA(5)	0.047	0.082	0.276	0.468	0.021	0.977	3.125
WGAN-GP	0.046	0.296	1.023	0.296	0.027	0.457	0.843
WGAN-GP - PCA(5)	0.055	0.115	0.139	**0.172**	0.021	0.230	1.747
qMLE	0.124	0.636	0.585	0.765	0.073	2.264	11.567
qMLE - PCA(5)	0.075	0.427	0.463	1.048	0.020	2.434	3.599
VAR(2)	0.088	0.415	0.447	0.834	0.013	1.113	2.770
VAR(2) - PCA(5)	0.223	0.328	2.302	0.914	0.025	0.477	1.982
TCN(256)[5]	0.048	0.105	0.295	0.330	0.372	0.335	1.501

The model that performs worst in terms of distributional properties is the VAR(2) - PCA(5) model. There the the fit of density and kurtosis is widely off. Notably, the ACF_X scores fit least for the VAR(2) and qMLE-trained models, independent whether they were trained on all or the compressed log-DLVs. Overall, we can also conclude from Table 1 that GAN- and WGAN-GP-calibrated models give the best fit. For TCNs we do not report on a compressed version as no good approximation could be obtained.

Although GAN-trained TCNs give a fairly good approximation in terms of distributional and cross-correlation scores the ACF_R score is far off. This makes the generated paths look very noisy. From this observation we concluded that TCNs have difficulties generating time series with high persistence from a pure i.i.d. noise process.

8.5 Explicit GAN-trained model

We presented numerical results for a wide range of models and optimization algorithms. Now, we will take a look at the properties of the explicit GAN-trained model since it performed best across most benchmark scores.

[5]The number in brackets specifies the receptive field size that was used; here 256.

As can be seen in Figure 4 the epdfs of the historical and generated log-implied volatilities match closely for most maturities and relative strikes. Arguably, the fit of the bimodal distribution for long-dated ($M = 120$ days) out of the money implied volatilies could be better. Taking a look at the historical and generated kurtosis in Figure 5 we can conclude that for most implied volatilities the approximation is accurate.

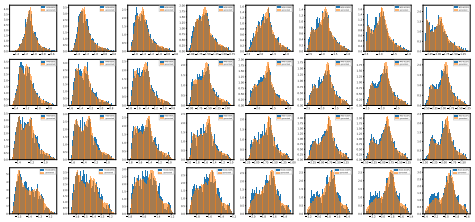

Figure 4: Empircal densities of the historical (blue) and generated (orange) log-implied volatilities.

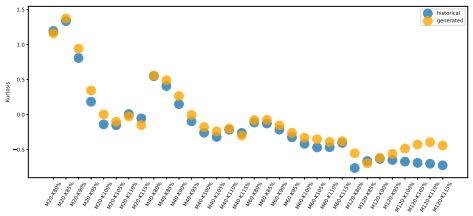

Figure 5: Kurtosis of historical (blue) and generated (orange) log-implied volatilities.

Figure 6 and Figure 7 illustrate the generated and historical cross-correlation matrices for log-implied volatilities and implied volatility log-returns. In both cases the historical is approximated accurately confirming the scores in Table 1.

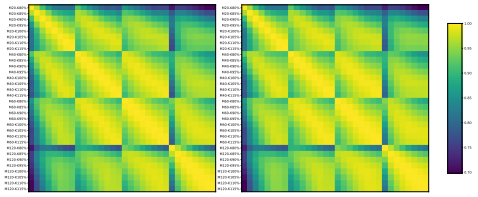

Figure 6: Cross-correlation matrix of historical (left) and generated (right) log-implied volatilities.

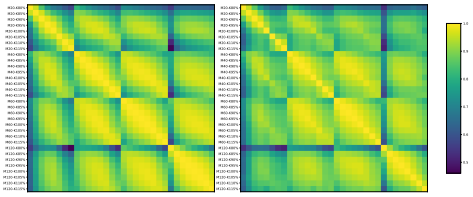

Figure 7: Cross-correlation matrix of historical (left) and generated (right) implied volatility log-returns.

Last viewing the dependence properties Figure A.1 shows that for short lags the approximation of the ACF is very good. However, for longer lags the ACF of the historical decays slower than the generated. Figure A.2 displays the ACF of implied volatility log-returns and shows that the explicit GAN-trained model is able to approximate short dependencies quiet well. However, to model longer lags a larger history is necessary.

Figure A.3 and Figure A.4 display two synthetic paths generated by the explicit GAN-trained model. Notably, the model is able to generate long-lasting periods of low volatility and periods of stress and high volatility phases. When comparing visually the synthetic paths to the historical (see Figure 1) it is difficult to discriminate them from being synthetic.

9 Conclusion

In this paper, we demonstrated that the generating mechanism of implied volatilities can be closely approximated by employing adversarial training techniques. To measure the proximity of our synthetic paths to the historical we introduced a variety of scores that capture different features of implied volatilities. In section 8 we developed a benchmark and compared the performance of GANs against a wide range of models, training algorithms and explored the effects of compressing DLVs by using PCA. There we concluded that adversarial training outperforms conventional approaches such as the VAR model and qMLE training.

Finally, our work shows for the first time that network-based models can be successfully applied to the context of generative modelling of multivariate financial time series; opening new avenues for future research and applications.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein Generative Adversarial Networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[2] Fischer Black and Myron Scholes. The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, 81:637–654, 1973.

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. *CoRR*, abs/1809.11096, 2018.

[4] Hans Buehler, Lukas Gonon, Josef Teichmann, and Ben Wood. Deep Hedging. *Quantitative Finance*, 19(8):1271–1291, 2019.

[5] Hans Buehler and Evgeny Ryskin. Discrete Local Volatility for Large Time Steps (Extended Version). *Available at SSRN 2642630*, 2017.

[6] Rama Cont and Joseé Da Fonseca. Dynamics of Implied Volatility Surfaces. *Quantitative Finance*, 2:45–60, 2002.

[7] Marcos Lopez de Prado. Tactical Investment Algorithms. 2019.

[8] Bruno Dupire. Pricing with a smile. *Risk*, 7:18–20, 1994.

[9] Eurex. Eurex Monthly Statistics. `https://www.eurexchange.com/exchange-en/market-data/statistics/monthly-statistics`, 2019. Accessed: 2019-09-23.

[10] Christian Francq and Jean-Michel Zakoïan. *GARCH Models: Structure, Statistical Inference and Financial Applications*. Wiley, 2010.

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[13] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved Training of Wasserstein GANs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5767–5777. Curran Associates, Inc., 2017.

[14] Steven Heston. A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options. *The Review of Financial Studies*, 6(2):327–343, 1993.

[15] Søren Johansen. *Likelihood-based inference in cointegrated vector autoregressive models*. Oxford University Press on Demand, 1995.

[16] Adriano Soares Koshiyama, Nick Firoozye, and Philip C. Treleaven. Generative Adversarial Networks for Financial Trading Strategies Fine-Tuning and Combination. *CoRR*, abs/1901.01751, 2019.

[17] Lars M. Mescheder. On the convergence properties of GAN training. *CoRR*, abs/1801.04406, 2018.

[18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. *CoRR*, abs/1802.05957, 2018.

[19] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[20] Gordon Ritter and Petter N Kolm. Dynamic Replication and Hedging: A Reinforcement Learning Approach. *Journal of Financial Data Science*, 1, 2019.

[21] Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised MAP Inference for Image Super-resolution. *CoRR*, abs/1610.04490, 2016.

[22] Dougal J. Sutherland, Hsiao-Yu Tung, Heiko Strathmann, Soumyajit De, Aaditya Ramdas, Alexander J. Smola, and Arthur Gretton. Generative Models and Model Criticism via Optimized Maximum Mean Discrepancy. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[23] Shuntaro Takahashi, Yu Chen, and Kumiko Tanaka-Ishii. Modeling financial time-series with generative adversarial networks. *Physica A: Statistical Mechanics and its Applications*, 527:121261, 2019.

[24] Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer. Quant GANs: Deep Generation of Financial Time Series. 2019.

[25] Johannes Wissel. Arbitrage-free market models for option prices. Technical report, FINRISK, 2007. http://www.nccr-finrisk.uzh.ch/media/pdf/wp/WP428_D1.pdf.

[26] Kang Zhang, Guoqiang Zhong, Junyu Dong, Shengke Wang, and Yong Wang. Stock Market Prediction Based on Generative Adversarial Network. *Procedia computer science*, 147:400–406, 2019.

[27] Xingyu Zhou, Zhisong Pan, Guyu Hu, Siqi Tang, and Cheng Zhao. Stock market prediction on high-frequency data using generative adversarial nets. *Mathematical Problems in Engineering*, 2018, 2018.

A Explicit GAN-trained model

A.1 Dependence properties

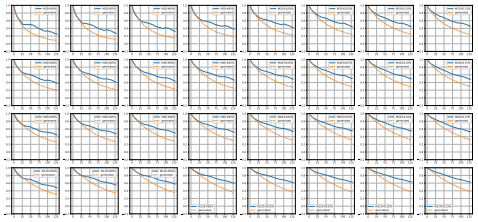

Figure A.1: ACF of historical (blue) and generated (orange) log-implied volatilities.

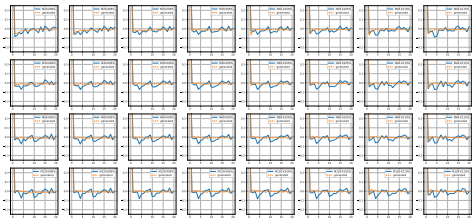

Figure A.2: ACF of implied volatility log-returns of the historical (blue) and generated (orange).

A.2 Generated paths

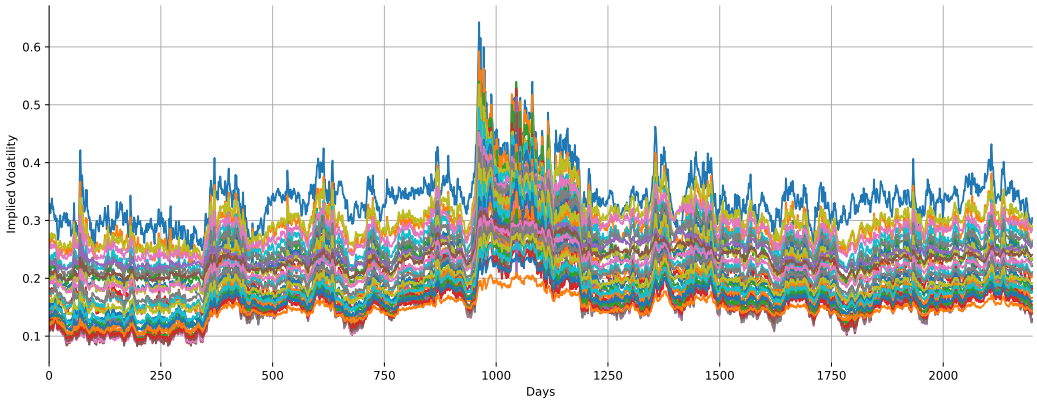

Figure A.3: Explicit GAN-generated path exhibiting phases of high and low volatility.

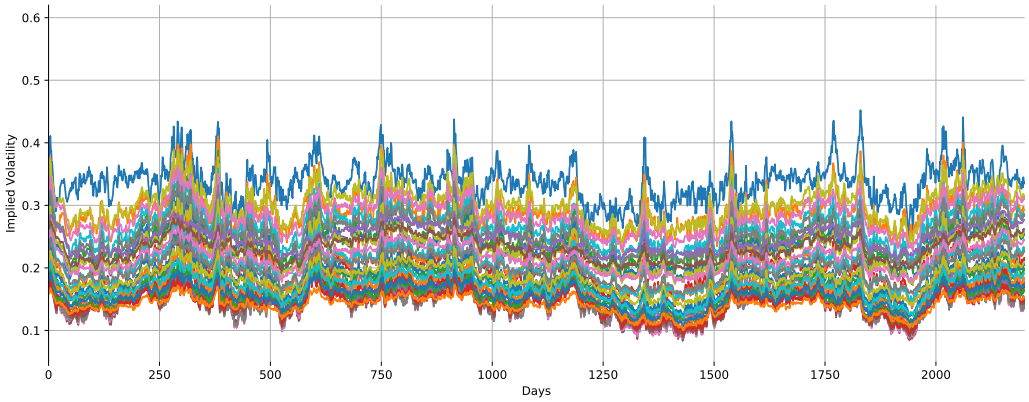

Figure A.4: Explicit GAN-generated path in a low-volatility regime.

B Discrete local volatilities

DLVs define an arbitrage-free surface that is globally closest to the surface of implied volatilities. We work with DLVs instead of implied volatilities in order to satisfy arbitrage constraints. This eliminates the issue of generating option prices that contain butterfly, calendar or spread arbitrage. Having synthetic options prices that contain arbitrage is highly undesirable. An algorithm trained on these prices could potentially learn to exploit these synthetic arbitrage opportunities which do not occure in reality; yielding an unworldly algorithm that performs well on synthetic but not real data.

In this paper, we consider the option prices of the EURO STOXX 50 from 2011-01-03 to 2019-08-30 and for the sets of relative strikes and maturities

$$\bar{\mathcal{K}} \coloneqq \{0.80, 0.85, 0.90, 0.95, 1.00, 1.05, 1.10, 1.15\},$$
$$\bar{\mathcal{M}} \coloneqq \{20, 40, 60, 120\}.$$

The historical time series of DLVs is obtained by transforming the EURO STOXX 50 implied volatilities and displayed in Figure B.1.

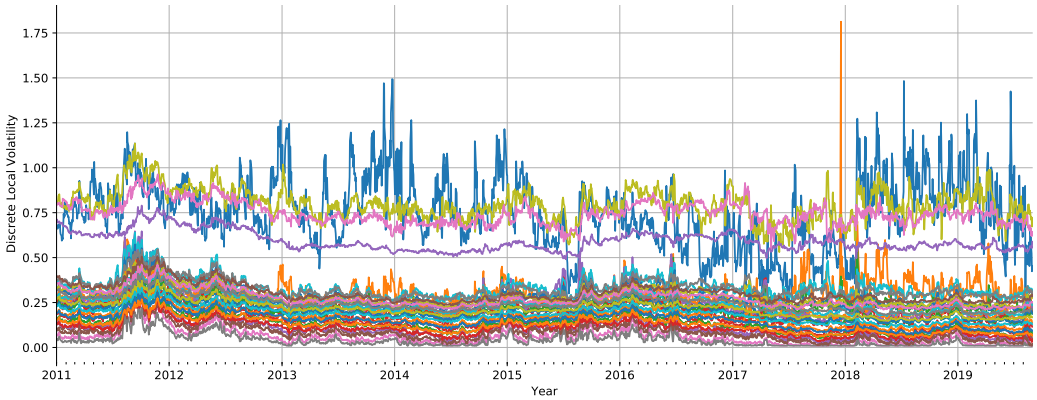

Figure B.1: Discrete local volatilities of the EURO STOXX 50 for the grid $\bar{\mathcal{K}} \times \bar{\mathcal{M}}$.

Figure B.2 shows the empirical densities of log-DLVs. Each row represents a specific maturity and columns the moneyness. Similiar to implied volatilities (see Figure 4) long-dated ($M \in \{60, 120\}$) out of the money (OTM) ($K \in \{105\%, 110\%, 115\%\}$) DLVs are characterised by bimodal distributions. The large buckets in the epdfs of the short-dated OTM DLVs ($M = 20$, $K \in \{110\%, 115\%\}$) represent the floor which is defined at 0.01.

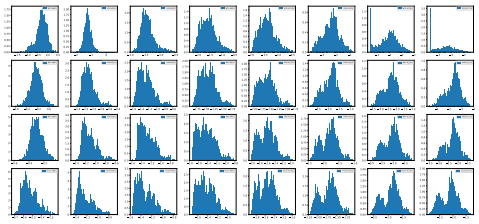

Figure B.2: Empirical densities of log-DLV levels.

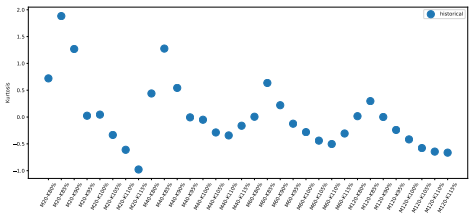

Figure B.3: Kurtosis.

The ACFs of log-DLVs and DLV log-returns are displayed in Figure B.4 and Figure B.5. Visually one can detect in Figure B.4 that the ACFs of short-dated in the money DLVs decay faster than long-dated OTM ones.

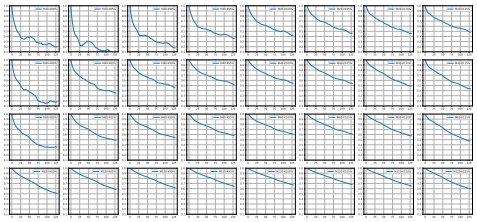

Figure B.4: ACF of log-DLVs.

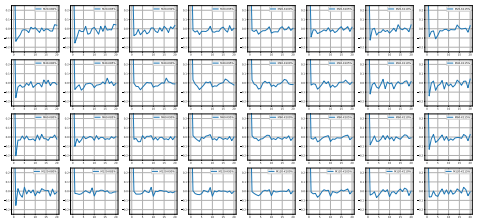

Figure B.5: ACF of DLV log-returns.

C Disclaimer

Opinions and estimates constitute our judgement as of the date of this Material, are for informational purposes only and are subject to change without notice. It is not a research report and is not intended as such. Past performance is not indicative of future results. This Material is not the product of J.P. Morgan's Research Department and therefore, has not been prepared in accordance with legal requirements to promote the independence of research, including but not limited to, the prohibition on the dealing ahead of the dissemination of investment research. This Material is not intended as research, a recommendation, advice, offer or solicitation for the purchase or sale of any financial product or service, or to be used in any way for evaluating the merits of participating in any transaction. Please consult your own advisors regarding legal, tax, accounting or any other aspects including suitability implications for your particular circumstances. J.P. Morgan disclaims any responsibility or liability whatsoever for the quality, accuracy or completeness of the information herein, and for any reliance on, or use of this material in any way. Important disclosures at: `www.jpmorgan.com/disclosures`.