Autoencoding Conditional GAN for Portfolio Allocation Diversification

Jun Lu[1], Shao Yi[2]

[1] Trexquant

[2] JJ Capital Fund

Correspondence: Jun Lu. E-mail: jun.lu.locky@gmail.com

Received: June 16, 2022 Accepted: August 1, 2022 Available online: August 5, 2022

doi:10.11114/aef.v9i3.5610 URL: https://doi.org/10.11114/aef.v9i3.5610

Abstract

Over the decades, the Markowitz framework has been used extensively in portfolio analysis though it puts too much emphasis on the analysis of the market uncertainty rather than on the trend prediction. While generative adversarial network (GAN) and conditional GAN (CGAN) have been explored to generate financial time series and extract features that can help portfolio analysis. The limitation of the CGAN framework stands in putting too much emphasis on generating series rather than keeping features that can help this generator. In this paper, we introduce an autoencoding CGAN (ACGAN) based on deep generative models that learns the internal trend of historical data while modeling market uncertainty and future trends. We evaluate the model on several real-world datasets from both the US and Europe markets, and show that the proposed ACGAN model leads to better portfolio allocation and generates series that are closer to true data compared to the existing Markowitz and CGAN approaches.

Keywords: Autoencoding conditional GAN (ACGAN), Conditional GAN, Time series, Portfolio analysis and allocation, Markowitz, Sharpe ratio, Financial markets, Synthetic series

1. Introduction

Financial portfolio management is largely based on linear models and the Markowitz framework (Markowitz, 1968, 1976) though the underlying data and information in today's market has increased countless times over that of many years ago. The fundamental idea behind the Markowitz framework is to create portfolio diversification while reducing specific risks and assessing the risk-return trade-offs for each asset. The Markowitz framework, on the other hand, has been criticized for making ideal assumptions about the financial system and data: the expected mean returns, and the covariance matrix of the return series are estimated from the past observations and assumed constant in the future. However, this is such a strong assumption that the market will find it impossible to meet this requirement in practice.

The classic portfolio assessment approach calculates portfolio risk indicators based on asset price series in the past period, such as variance, value at risk, and expected loss, as the evaluation results of cross-section risk. However, the traditional method has two obvious drawbacks. First, because the capital market is changing rapidly, historical data usually cannot be used to indicate the situation in the future; when a trusted long-term forecast is offered in a high efficient market, this prediction is absorbed by traders in the short term and has a direct impact on current price, while future price variations are unpredictable again (Timmermann & Granger, 2004). Secondly, the risk measurement indicators estimated by traditional methods usually only contain the linear components in the historical series, leaving out the nonlinear information contained therein, resulting in the deviation between the evaluation results and the real situation (Tsay, 2005).

On the other hand, the financial market is one of the most heavily impacted industries by AI advancements. Machine learning has been used in a variety of applications, including series generation, forecasting, customer service, risk management, and portfolio management (Huang et al., 2005; Kara et al., 2011; Takahashi et al., 2019). Specifically, generative adversarial networks (GANs) are a sort of neural network architectures that have shown promise in image generation and are now being used to produce time series and other financial data (Goodfellow et al., 2014; Esteban et al., 2017; Eckerli & Osterrieder, 2021). There are several techniques to model financial time series data, including models of the ARCH and GARCH family, which use classical statistics to model the change in variance over time in a time series by characterizing the variance of the current error component as a function of previous errors (Engle, 1982; Bollerslev, 1986; Lu & Yi, 2022). GANs are being used to address issue of the paucity of real data, as well as to optimize portfolios and trading methods which achieve better results (Takahashi et al., 2019; Mariani et al., 2019).

However, due to its highly noisy, stochastic, and chaotic nature, market price forecasting is still one of the key issues in

the time series literature. While previous work have tried to generate financial data based on historical trend, the internal features of the past series are not captured sufficiently so that the generated series is not close to the real market trend (Mariani et al., 2019).

In this light, we focus on GANs for better portfolio allocation that can both capture historical trends and generate series based on past data. We present a novel framework about portfolio analysis based on conditional GAN (CGAN) that incorporates autoencoder, hence the name *autoencoding CGAN (ACGAN)*, to overcome the issues and challenges encountered in portfolio management tasks. Similar to the CGAN model for portfolio analysis (Mariani et al., 2019), ACGAN can also directly model the market uncertainty via its complex multidimensional form, which is the primary driver of future price trends, such that the nonlinear interactions between various assets can be embedded effectively. We assess the proposed ACGAN method on two separate portfolios representing different markets (the US and the European markets) and industrial segments (e.g., Healthcare, Technology, Industrials, and Basic materials sectors). The empirical results show that the proposed approach is capable of realizing the risk-return trade-off and outperforms the classic Markowitz framework and CGAN-based methodology considerably.

2. Related Work

As aforementioned, there are several methods delving with portfolio allocation, including the Markowitz framework and the CGAN methodology (Markowitz, 1968; Mariani et al., 2019). The Markowitz framework relies on the assumption that the past trend can be applied in the future. While the CGAN methodology partly solves the drawback in the Markowitz framework by simulating future data based on historical trends, it still lacks full ability to capture the information and features behind the past data. The proposed ACGAN model introduces an extra *decoder* that can help the constructed networks to capture historical features and simulate data closer to real ones.

2.1 Markowitz Framework

Portfolio allocation is a kind of investment portfolio where the market portfolio has highest Sharpe ratio (SR) given the composition of assets (Markowitz, 1968). For simplicity, we here only consider long only portfolio. Denote $\boldsymbol{r}$ as the return on assets vector, $\boldsymbol{\Sigma}$ as the asset covariance matrix, $\boldsymbol{w}$ as the weight vector of each asset, and r_f as the risk-free interest rate. If we measure portfolio risk by variance (or standard deviation), then the overall return and risk of the portfolio are:

$$r_p = \boldsymbol{w}^\top \boldsymbol{r}, \qquad \sigma_p^2 = \boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}. \tag{1}$$

And the Sharpe ratio (Sharpe, 1966) can be obtained by

$$\mathrm{SR} = \frac{r_p - r_f}{\sigma_p} = \frac{\boldsymbol{w}^\top \boldsymbol{r} - r_f}{\sqrt{\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}}}. \tag{2}$$

According to the definition of portfolio allocation, the weight of each asset in the market portfolio is the solution to the following optimization problem:

$$\begin{aligned} &\arg\max_{\boldsymbol{w}} \frac{\boldsymbol{w}^\top \boldsymbol{r} - r_f}{\sqrt{\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}}}; \\ &\text{s.t.} \sum_{i=1}^{N} w_i = 1; \qquad 0 \le w_i \le 1, \forall\, i \in \{1, 2, \ldots, N\}, \end{aligned} \tag{3}$$

where N is the number of assets, and w_i is the i-th element of the weight vector $\boldsymbol{w}$.

2.2 Portfolio Analysis with GAN

We consider the matrix $\boldsymbol{A}$ to span the whole analysis length: $w = h + f$. The matrix $\boldsymbol{A}$ contains two components, the known historical series $\boldsymbol{A}_h$ of length h, and the unknown future $\boldsymbol{A}_f$ of length f. Given the number of assets N, the matrix $\boldsymbol{A}$ is of size $N \times w$; $\boldsymbol{A}_h$ has shape $N \times h$; and $\boldsymbol{A}_f$ is of shape $N \times f$.

Given the known historical series $\boldsymbol{A}_h \in \mathbb{R}^{N \times h}$ and a prior distribution of a random latent vector $\boldsymbol{z}$ (or size m: $\boldsymbol{z} \in \mathbb{R}^m$), we use a generative deep-neural network G to learn the probability distribution of future price trends $\boldsymbol{A}_f$ within the target future horizon f. Figure 1 depicts a graphical representation of the matrix $\boldsymbol{A}$, as well as the generator G's inputs and outputs. Formally the generative model generates a fake future matrix $\widetilde{\boldsymbol{A}}_f$ by

$$\widetilde{\boldsymbol{A}}_f = G(\boldsymbol{z}, \boldsymbol{A}_h), \tag{4}$$

where $\boldsymbol{z} \in \mathbb{R}^m$ is the latent vector sampled from a prior distribution (e.g., from a normal distribution). In practice, the latent vector $\boldsymbol{z}$ represents the unforeseeable future occurrences and phenomena that will have an impact on the marketplace.

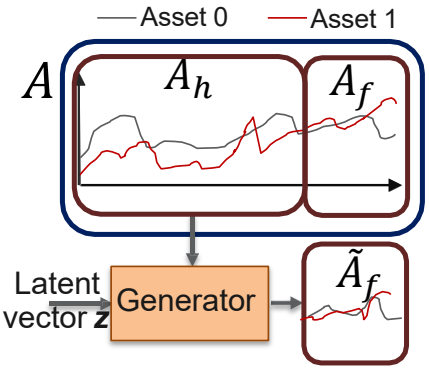


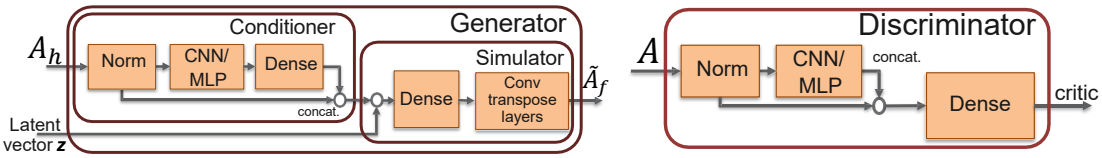
Figure 1. A conceptual overview of the CGAN and the proposed ACGAN generators' inputs and outputs

Based on the most recent market conditions, the known historical series $\boldsymbol{A}_h$ is used to extract features and condition the probability distribution of the future $\boldsymbol{A}_f$. Given the historical observation $\boldsymbol{A}_h$ and following the Wasserstein GAN-GP (WGAN-GP) by Gulrajani et al. (2017), the generative G is trained in adversarial mode against a discriminator network D with the goal of minimizing the Wasserstein distance between the real future series $\boldsymbol{A}_f$ and the fake series $\widetilde{\boldsymbol{A}}_f$. Formally, the process is described by the following optimization problem:

$$\begin{aligned} \max_{D} \quad & \mathbb{E}_{\boldsymbol{x}\sim p(\text{data})}\Big\{D(\boldsymbol{x}) - \mathbb{E}_{z\sim p(z)}[D(G(z,\boldsymbol{x}_h))]\Big\} - \lambda_1 \cdot \mathbb{E}_{\bar{\boldsymbol{x}}\sim p(\epsilon\,\text{data}+(1-\epsilon)G(z))}[\|\nabla_{\bar{\boldsymbol{x}}} D(\bar{\boldsymbol{x}})\|_2 - 1]^2; \\ \max_{G} \quad & \mathbb{E}_{\boldsymbol{x}\sim p(\text{data})}\Big\{\mathbb{E}_{z\sim p(z)}[D(G(z,\boldsymbol{x}_h))]\Big\}, \end{aligned} \tag{5}$$

where $\boldsymbol{x}_h$ contains the historical parts of the data $\boldsymbol{x}$ ($\boldsymbol{x}_h \in \boldsymbol{x}$), $G(\boldsymbol{z}, \boldsymbol{x}_h)$ indicates that the generator depends on the (historical) data $\boldsymbol{x}_h$, and λ_1 controls the gradient penalty. Theoretically, the optimization process finds the surrogate posterior probability distribution $p(\widetilde{\boldsymbol{A}}_f|\boldsymbol{A}_h)$ that approximates the real posterior probability distribution $p(\boldsymbol{A}_f|\boldsymbol{A}_h)$.

The main drawback of the CGAN methodology is in that it puts too much emphasis on the *conditioner* to extract features that can "deceive" the discriminator (Figure 2(a)). When the discriminator is perfectly trained, this issue is not a big problem. However, in most cases, especially due to the scarcity of financial data, the discriminator works imperfectly such that the conditioner may lose important information for the historical data. Whereas in the proposed ACGAN model, we find a balance between the information extraction and generation for cheating the discriminator via an embedded autoencoder providing the capability of keeping the intrinsic information of historical data.



(a) Generator (b) Discriminator

Figure 2. Architectures of the CGAN generative and discriminative models for portfolio analysis

3. Autoencoding Conditional GAN (ACGAN) for Portfolio Analysis

3.1 Proposed Methodology

The proposed ACGAN has the same discriminator structure as the CGAN. However, it contains an extra *decoder* in the generator as shown in Figure 3. And therefore we call the conditioner an *encoder* in the ACGAN context.

We use an *encoding* deep-neural network E to learn the features that can help the generator trick the discriminator and can find the internal information itself; and a *decoding* deep-neural network F to reconstruct the historical series so as to force the encoder to do so. Formally the encoding and decoding models reconstruct the historical matrix by

$$\boldsymbol{y} = E(\boldsymbol{A}_h), \qquad \widetilde{\boldsymbol{A}}_h = F(\boldsymbol{y}).$$

This process is known as the *autoencoding*, hence the name autoencoding conditional GAN (ACGAN). In a non-GAN context, the autoencoder is typically done by matrix decomposition or nonnegative matrix factorization via alternative

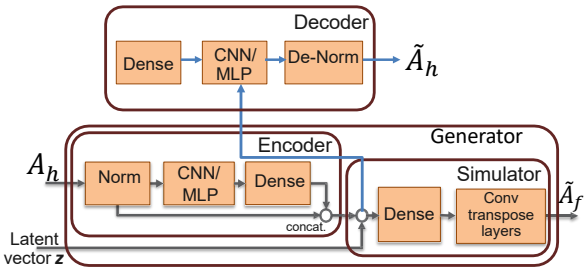


Figure 3. Architecture of the ACGAN generative model for portfolio analysis

least squares or Bayesian inference (Lee & Seung, 1999; Lu, 2021, 2022; Lu & Ye, 2022). Since we need to use the encoding part of the autoencoder to help "cheat" the discriminator as well, we here use deep-neural network instead. Formally the process is described by the following optimization:

$$\begin{aligned} &\max_{D} \quad \mathbb{E}_{\boldsymbol{x}\sim p(\text{data})}\Big\{D(\boldsymbol{x}) - \mathbb{E}_{z\sim p(z)}[D(G(\boldsymbol{z},\boldsymbol{x}_h))]\Big\} - \lambda_1 \mathbb{E}_{\overline{\boldsymbol{x}}\sim p(\epsilon\,\text{data}+(1-\epsilon)G(z))}\big[\|\nabla_{\overline{\boldsymbol{x}}} D(\overline{\boldsymbol{x}})\|_2 - 1\big]^2; \\ &\max_{G,E,F} \quad \mathbb{E}_{\boldsymbol{x}\sim p(\text{data})}\Big\{\mathbb{E}_{z\sim p(z)}[D(G(\boldsymbol{z},\boldsymbol{x}_h))] - \lambda_2\, f(\underbrace{F(E(\boldsymbol{x}_h))}_{\widetilde{\boldsymbol{x}}_h},\ \boldsymbol{x}_h)\Big\}, \end{aligned} \tag{6}$$

where $f(\cdot)$ denotes the loss function. In our work, we apply the mean squared error (MSE) as the loss function. The parameter λ_2 controls how large the penalization by the autoencoder, and we call the term *autoencoding penalty (AP)*. In the original CGAN scenario, the conditioner is used to extract features that can help the generator to cheat the discriminator; however, it may lose some important information that captures the features of the market trend. The ACGAN then finds a balance between cheating the discriminator and retaining its market information.

Other Extension In our work, we apply to find small discrepancy between $F(E(\boldsymbol{x}_h))$ and $\boldsymbol{x}_h$ in Eq. (6) so that the encoder can keep the original information of $\boldsymbol{x}_h$ as much as possible. If one believes the encoder should keep information to some other features, say $\boldsymbol{y}_h$ (e.g., the fat-tail, mean, skewness of the data), the ACGAN can be extended to the general *encoding CGAN (ECGAN)* that penalizes the following loss:

$$f(F(E(\boldsymbol{x}_h)),\ \boldsymbol{y}_h). \tag{7}$$

Discriminator The discriminator shown in Figure 2(b) (for both CGAN and ACGAN) takes as input either the real data matrix $\boldsymbol{A} = [\boldsymbol{A}_h, \boldsymbol{A}_f] \in \mathbb{R}^{N\times w}$ or the synthetic data matrix $\widetilde{\boldsymbol{A}} = [\boldsymbol{A}_h, \widetilde{\boldsymbol{A}}_f] \in \mathbb{R}^{N\times w}$.

3.2 Data Normalization

Following Mariani et al. (2019), we consider the *adjusted closing price* $\boldsymbol{p} \in \mathbb{R}^w$ series for each asset. Given the frame window of $w = h + f$ days (h for the historical length, f for the future length. The historical series is denoted by $\boldsymbol{p}_{1:h} \in \mathbb{R}^h$, and the real future series can be obtained by $\boldsymbol{p}_{h+1:w} \in \mathbb{R}^f$), we unit-normalize the price series $\boldsymbol{p}$ for each asset to fill in the range $[-1, 1]$ for the initial h days. In practice, the unit-normalization can be done by 3-sigma normalization: given the mean μ and standard deviation σ of $\boldsymbol{p}_{1:h} \in \mathbb{R}^h$, the normalization is done by

$$\widetilde{\boldsymbol{p}} = \frac{\boldsymbol{p} - \mu}{3\sigma}. \tag{8}$$

This normalization procedure can help us to expose the neural networks values limited within a suitable range that removes price-variability over multiple assets within the specified window.

After generating the surrogate future series $\widetilde{\boldsymbol{p}}_{h+1:w}$, we apply again a de-normalization procedure:

$$\widehat{\boldsymbol{p}}_{h+1:w} = \widetilde{\boldsymbol{p}}_{h+1:w} \times 3\sigma + \mu. \tag{9}$$

3.3 Statistical Properties of Financial Time Series

We have several statistical properties (stylized facts) of financial time series (Müller et al., 1997; Cont, 2001; Chakraborti et al., 2011; Takahashi et al., 2019). Given the price series p_t of an asset at time t, the log return of price can be obtained by

$$r_t = \log p_{t+1} - \log p_t.$$

	Ticker	Type	Sector	Company	Curr.
US Region	GOOG	Share	IT	Alphabet	USD
	MSFT	Share	IT	Microsoft	USD
	PFE	Share	Healthcare	Pfizer	USD
	HES	Share	Energy	Hess	USD
	XOM	Share	Energy	Exxon Mobil	USD
	KR	Share	Consumer staples	The Kroger	USD
	WBA	Share	Consumer staples	Walgreens Alliance	USD
	IYY	ETF	Dow Jones	iShares Dow Jones	USD
	IYR	ETF	Real estate	iShares US Real Estate	USD
	SHY	ETF	US treasury bond	iShares Treasury Bond	USD
EU Region	^FCHI	Index	French market	CAC 40	EUR
	^GDAXI	Index	German market	DAX	EUR
	BMW.DE	Share	Automotive	BMW	EUR
	VOW3.DE	Share	Automotive	Volkswagen	EUR
	SOI.PA	Share	Industrials	Soitec S.A.	EUR
	VK.PA	Share	Industrials	Vallourec S.A.	EUR
	BAS.DE	Share	Basic materials	BASF SE	EUR
	SAP.DE	Share	Technology	SAP SE	EUR
	DTE.DE	Share	Technology	Deutsche Telekom AG	EUR
	BAYN.DE	Share	Healthcare	Bayer AG	EUR

Table 1. Summary of the underlying portfolios in the US and EU markets, 10 assets for each market respectively. In each region, we include assets from various sectors (e.g., IT, Healthcare, Industrials, and Technology sectors) to favor a somehow sector-neutral strategy

Asset	CGAN	ACGAN	Asset	CGAN	ACGAN
GOOG	9.968	**9.972**	^FCHI	**9.875**	9.838
MSFT	**9.967**	9.961	^GDAXI	9.849	**9.870**
PFE	9.836	**9.851**	BMW.DE	9.824	**9.867**
HES	9.831	**9.871**	VOW3.DE	9.732	**9.745**
XOM	**9.888**	9.877	SOI.PA	9.873	**9.895**
KR	9.902	**9.948**	VK.PA	9.965	**9.969**
WBA	9.690	**9.750**	BAS.DE	**9.859**	9.854
IYY	**9.948**	9.945	SAP.DE	9.063	**9.138**
IYR	9.833	**9.839**	DTE.DE	9.879	**9.901**
SHY	**9.934**	9.933	BAYN.DE	9.502	**9.513**
GOOG	**9.958**	9.954	^FCHI	**9.865**	9.864
MSFT	9.940	**9.948**	^GDAXI	**9.866**	9.864
PFE	9.847	**9.860**	BMW.DE	9.837	**9.873**
HES	**9.848**	9.840	VOW3.DE	9.721	**9.765**
XOM	**9.873**	9.862	SOI.PA	9.914	**9.927**
KR	9.921	**9.928**	VK.PA	9.976	**9.977**
WBA	9.495	**9.659**	BAS.DE	9.824	**9.825**
IYY	9.930	**9.939**	SAP.DE	9.364	**9.466**
IYR	9.810	**9.833**	DTE.DE	9.889	**9.890**
SHY	9.904	**9.916**	BAYN.DE	9.615	**9.616**

Table 2. Mean Pearson correlation between the true series and the generated series. The fake series are generated by the generator at the 100-th epoch (upper table) and the 1,000-th epoch (lower table). The values are multiplied by 10 for clarity

We then review a few facts on the return series.

Linear Unpredictability The most important fundamental property of the return series is its linear unpredictability (a.k.a., absence of autocorrelations) that is quantified by its diminishing autocorrelation of the return series:

$$\text{Corr}(r_t, r_{t+k}) = \frac{\mathbb{E}[(r_t - \mu)(r_{t+k} - \mu)]}{\sigma^2} \approx 0, \forall\ k \geq 1,$$

where μ, σ are the mean and standard deviation of the return series respectively.

Fat-Tailed Distribution The probability distribution function of the return series $p(r)$ is empirically known to have a power-law decay [1] in the tails:

$$p(r) \propto r^{-\alpha}.$$

Empirically, the exponent α ranges $3 \leq \alpha \leq 5$.

Leverage Effect The leverage effect means that there is a negative correlation between past price return and future volatility (Bouchaud et al., 2001). In other words, if the market declines significantly in the past price, the future volatility will increase; while if the market increases significantly in the past price, the future volatility will decrease. The leverage effect is quantified by the following lead-lag correlation function

$$L(k) = \frac{\mathbb{E}[r_t |r_{t+k}|^2] - \mathbb{E}[r_t] \cdot \mathbb{E}[|r_t|^2]}{\mathbb{E}[|r_t|^2]^2}.$$

Bouchaud et al. (2001) show that $L(k)$ has a negative value for $1 \leq k \leq 10$ and the distribution follows the exponential decay.

Coarse-Fine Volatility Correlation The coarse-fine volatility correlation is a multi-time-scale analysis of volatility (Müller et al., 1997; Rydberg, 2000). Define the coarse volatility v_c^τ and the fine volatility v_f^τ as

$$v_c^\tau(t) = \left| \sum_{i=1}^{\tau} r_{t-i} \right|, \qquad v_f^\tau = \sum_{i=1}^{\tau} |r_{t-i}|,$$

[1] The distribution is known as the Zipf distribution or Zeta distribution.

where the coarse volatility is the absolute value of the movement in τ days, and the fine volatility is the sum of absolute return in τ days. Then we calculate the correlation between the current fine volatility and k-th lagged coarse volatility:

$$\rho^{\tau}_{cf}(k) = \mathrm{Corr}(v^{\tau}_{c}(t+k), v^{\tau}_{f}(t)).$$

Then there exists the negative asymmetry (especially when k is small) of the lead-lag correlation quantified by the following difference

$$\Delta\rho^{\tau}_{cf}(k) = \rho^{\tau}_{cf}(k) - \rho^{\tau}_{cf}(-k) < 0.$$

The fact that $\Delta\rho^{\tau}_{cf}(k)$ is negative means that fine volatility can predict coarse volatility.

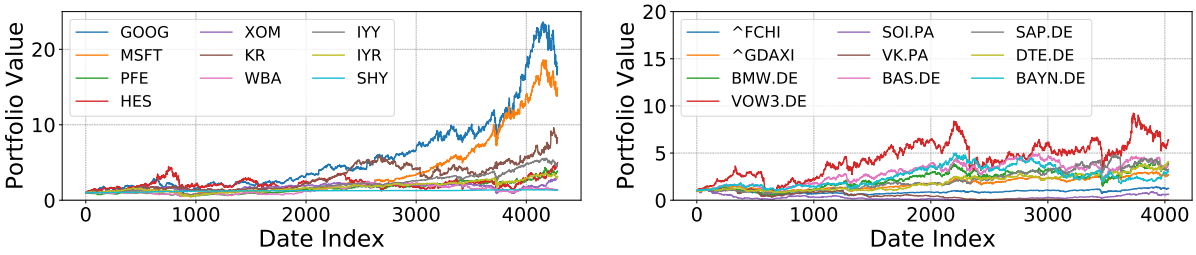


Figure 4. Different portfolios for the US (left) and EU (right) markets with a unit initial value

4. Experiments

Algorithm 1 Training and testing process for the ACGAN and CGAN models.

General Input: Choose parameters $w = h + f$; number of assets N; number of epoches T; latent dimension m;
Training Input: Training data matrix $\boldsymbol{M} \in \mathbb{R}^{N\times D}$;
Decide index set $\mathcal{S}_1 = \{1, 2, \ldots, D - w + 1\}$ and **draw without replacement**;
for $t = 1$ to T **do**
  for $i \in$ random$(\mathcal{S}_1)$ **do**
    $\boldsymbol{A} = [\boldsymbol{A}_h, \boldsymbol{A}_f] = \boldsymbol{M}[:, i : i + w - 1] \in \mathbb{R}^{N\times w}$;
    Randomly sample latent vector $\boldsymbol{z} \in \mathbb{R}^m$;
    Backpropatation for generator in Eq. (6) or (5);
    Generate surrogate $\widetilde{\boldsymbol{A}}_f = G(\boldsymbol{z}, \boldsymbol{A}_h) \in \mathbb{R}^{N\times f}$;
    Backpropatation for discriminator in Eq. (6) or (5);
  end for
end for
Inference Input: Testing data matrix $\boldsymbol{X} \in \mathbb{R}^{N\times K}$;
Inference Output: Testing data matrix $\boldsymbol{Y} \in \mathbb{R}^{N\times K}$;
Decide index set $\mathcal{S}_2 = \{h + 1, h + f + 1, \ldots\}$;
Copy the first h days data $\boldsymbol{Y}[:, 1 : h] = \boldsymbol{X}[:, 1 : h]$;
for $i \in$ ordered$(\mathcal{S}_2)$ **do**
  $\boldsymbol{A} = [\boldsymbol{A}_h, \boldsymbol{A}_f] = \boldsymbol{X}[:, i : i + w - 1] \in \mathbb{R}^{N\times w}$;
  Randomly sample latent vector $\boldsymbol{z} \in \mathbb{R}^m$;
  Generate $\boldsymbol{Y}[:, i : i + f - 1] = G(\boldsymbol{z}, \boldsymbol{A}_h) \in \mathbb{R}^{N\times f}$ with de-normalization in Eq. (9);
end for
Output the synthetic series $\boldsymbol{Y}$;

To evaluate the strategy and demonstrate the main advantages of the proposed ACGAN method, we conduct experiments with different analysis tasks; datasets from different geopolitical markets including the US and the European (EU) markets, and various industrial segments including Healthcare, Automotive, Energy and so on. We obtain publicly available data from Yahoo Finance [2].

For the US market, we obtain data for a 17-year period, i.e., 2005-05-24 to 2022-05-27, where the data between 2005-05-24 and 2019-03-28 is considered training data; while data between 2019-03-28 and 2022-05-27 is taken as the test data (800 trading days). For the EU market, we obtain data for a 16-year period, i.e., 2006-07-18 to 2022-06-07, where the

[2] https://finance.yahoo.com/.

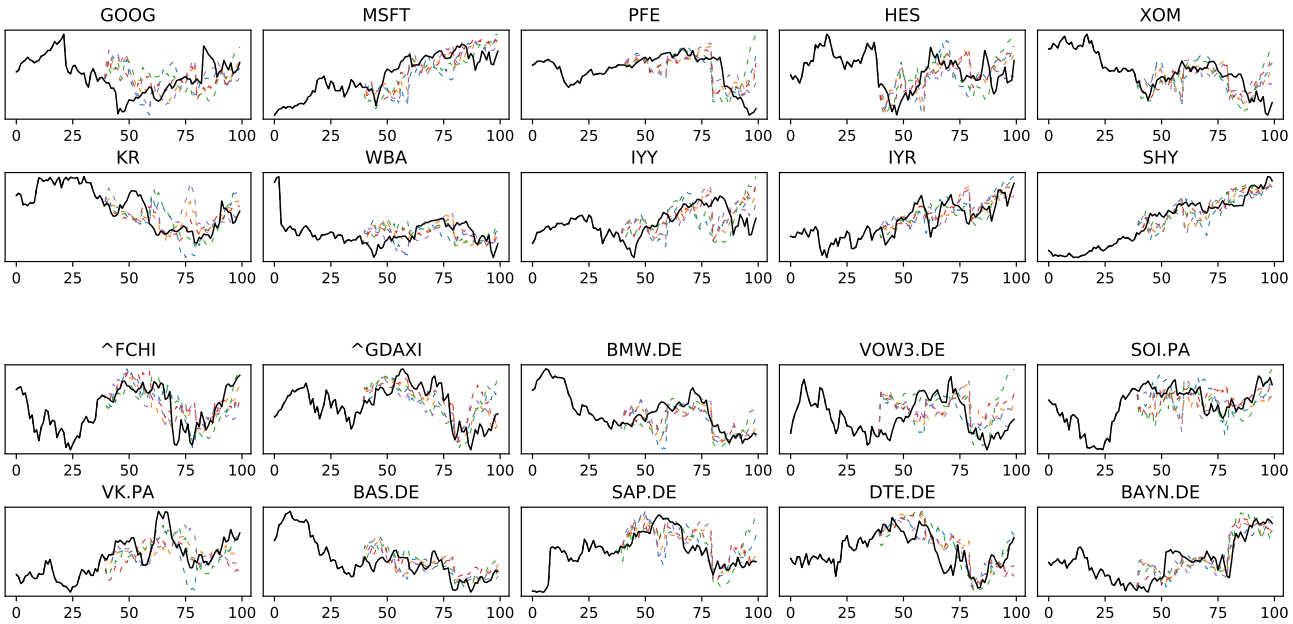


Figure 5. Actual price trend (black solid line) of the US assets (upper two rows) and the EU assets (lower two rows) for the first 100 trading days in the test set, and five representative simulations generated by ACGAN (colored dashed lines). The simulated price series of the whole period (800 trading days) and simulated price series of the CGAN model (100 and all trading days) can be found in Figure 9, 10, and 11 respectively

data between 2006-07-18 and 2019-04-09 is considered training data; while data between 2019-04-10 and 2022-06-07 is taken as the test data (800 trading days). The underlying portfolios are summarized in Table 1:

- *US market*: 10 assets of US companies from different industrial segments, i.e., Alphabet and Microsoft (from IT sector), Pfizer (from Healthcare sector), Hess and Exxon Mobil (from Energy sector), Kroger and Walgreens Alliance (from Consumer staples sector), and three ETFs (IYY, IYR, SHY).
- *EU market*: 10 portfolios of EU companies from different industrial segments, i.e., BMW and Volkswagen (from Automotive sector), Soitec and Vallourec (from Industrials sector), BASF (from Basic materials sector), SAP and Deutsche Telekom (from Technology sector), Bayer (from Healthcare sector), and two indices, ^FCHI and ^GDAXI, that track the German and French stock markets respectively.

The specific time periods and assets are chosen by following the four criteria. *1). Data availability*: we want to cover as longer period as possible; the periods are selected to make all the assets have same frame length; *2). Data diversity*: in each market, we include firms from different segments so that the end strategies are somewhat sector-neutral; *3). Currency homogeneity*: in each marketplace, the currencies are the same; *4). Data correctness*: given the Yahoo Finance data source, we only include the data that do not have NaN values. Figure 4 shows the series of different assets where we initialize each portfolio with a unitary value for clarity.

In all scenarios, same parameter initialization is adopted when conducting different tasks. We compare the results in terms of series correlation and performance of portfolio allocation. In a wide range of scenarios across various tasks, ACGAN improves portfolio evaluation, and leads to return-risks performances that are as good or better than the existing Markowitz framework and CGAN methodology.

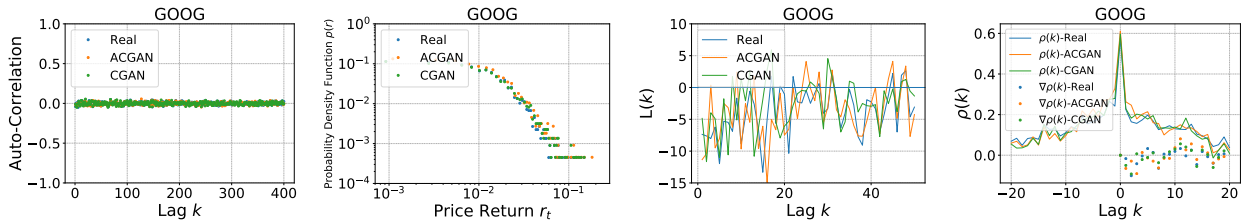
Hyperparameters Network structures for the conditioner (in CGAN), encoder, decoder (in ACGAN), generator, and discriminator (in both CGAN and ACGAN) are provided in Appendix A. In all experiments, we train the network with 1,000 epochs. For simplicity, we set the risk-free interest $r_f = 0$ to assess the Sharpe ratio evaluations.

4.1 Generating Analysis

We follow the training and testing procedures in Algorithm 1. Given the training matrix $\boldsymbol{M}$ of size $N \times D$ (where N is the number of assets and D is the number of days in the daily analysis context) and the window size w ($w = h+f$ where h is the

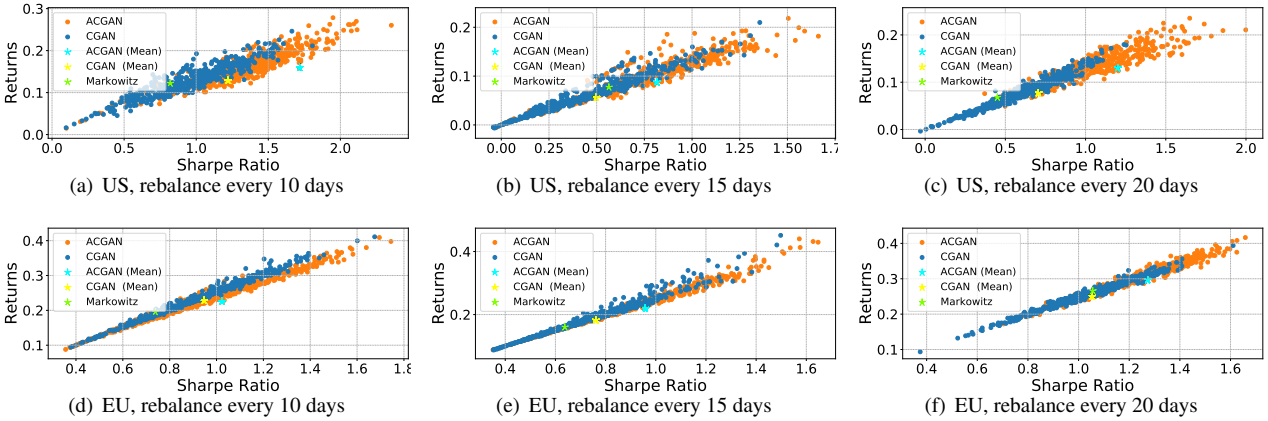
Statistics	Real (US)	ACGAN (US)	CGAN (US)	Real (EU)	ACGAN (EU)	CGAN (EU)
Autocorrelation	0.027	**0.035**	**0.036**	0.018	**0.032**	**0.034**
Fat-tail	3.819	**3.477**	3.423	4.130	**3.611**	3.552
Leverage effect	-9.328	**-7.254**	-6.384	-10.228	-7.660	**-8.815**
Coarse-fine	-0.068	**-0.040**	-0.011	-0.074	**-0.033**	**-0.038**
Kurtosis	11.305	**22.366**	30.019	34.002	24.144	**40.297**
Skewness	-0.324	**-0.403**	0.179	-1.119	**-0.311**	-0.213

Table 3. Statistical properties of the real and generated series. Autocorrelation value is the mean of the first 10-th lags of autocorrelation coefficients; Fat-tail value is the fitted powerlaw coefficient α, leverage effect value is the mean of the first 10-th lags of leverage effect coefficients; and coarse-fine value is the difference between ± 1-th order of lead-lag coefficients. In most cases, the coefficients of the ACGAN are closer to the true ones. The Kurtosis of a normal distribution is 3 and for a fat-tail distribution, the value is larger than 3



(a) Linear unpredictability (b) Fat-tail distribution (c) Leverage effect (d) Coarse-fine volatility correlation

Figure 6. Example of autocorrelation, fat-tail distribution, leverage effect, and coarse-fine volatility correlation for one asset



(a) US, rebalance every 10 days (b) US, rebalance every 15 days (c) US, rebalance every 20 days

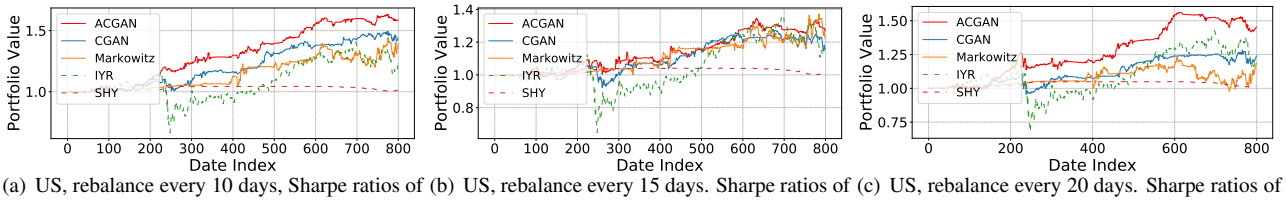
(d) EU, rebalance every 10 days (e) EU, rebalance every 15 days (f) EU, rebalance every 20 days

Figure 7. (Annual) return-SR measured on the test period by randomly sampling 1000 series

length of the historical window and f is the length of the future window), we define the index set $\mathcal{S}_1 = \{1, 2, \ldots, D-w+1\}$ so that $D - w$ samples can be extracted for each training epoch. While at the testing stage, given the testing matrix $\boldsymbol{X} \in \mathbb{R}^{N\times K}$, the index set is obtained by $\mathcal{S}_2 = \{h+1, h+f+1, \ldots\}$ so that $(K-h)/f$ samples can be obtained (supposed here $(K-h)$ can be divided by f). The output $\boldsymbol{Y} \in \mathbb{R}^{N\times K}$ of Algorithm 1 is the financial market simulation of the N assets in K days (here $N = 10$ and $K = 800$ in our datasets from US and EU regions). To be more concrete, the first h days of $\boldsymbol{Y}$ are just copies of $\boldsymbol{X}$, while the next f days are the synthetic series based on the data of the first h days; the next f days are the synthetic series based on the data between the f-th and $(f+h)$-th days; and so on.

We set window size $h = 40, f = 20$ and $w = 60$ in all experiments. Figure 5 shows the actual price trend (black solid line) of the US assets and the EU assets for the first 100 trading days in the test set, and five representative simulations generated by ACGAN (colored dashed lines). The simulated price series of the whole period (800 trading days) and simulated price series of the CGAN model (100 and all trading days) can be found in Figure 9, 10, and 11 respectively.

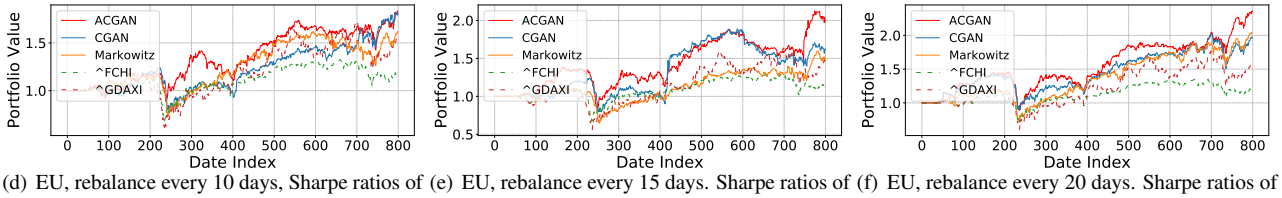
We then calculate the Pearson correlation between the real series and the synthetic series by ACGAN and CGAN models. The Pearson correlation between two series $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined by $\mathrm{Corr}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\mathbb{E}[(\boldsymbol{x}-\mu_x)(\boldsymbol{x}-\mu_y)]}{\sigma_x\sigma_y}$, where μ_x, σ_x are the mean



(a) US, rebalance every 10 days, Sharpe ratios of ACGAN, CGAN, and Markowitz are **1.72**, 1.22, and, 0.82 respectively

(b) US, rebalance every 15 days. Sharpe ratios of ACGAN, CGAN, and Markowitz are **0.82**, 0.49, and 0.56 respectively.

(c) US, rebalance every 20 days. Sharpe ratios of ACGAN, CGAN, and Markowitz are **1.17**, 0.71, and 0.45 respectively



(d) EU, rebalance every 10 days, Sharpe ratios of ACGAN, CGAN, and Markowitz are **1.02**, 0.95, and, 0.74 respectively

(e) EU, rebalance every 15 days. Sharpe ratios of ACGAN, CGAN, and Markowitz are **0.96**, 0.76, and 0.64 respectively

(f) EU, rebalance every 20 days. Sharpe ratios of ACGAN, CGAN, and Markowitz are **1.27**, 1.05, and 1.06 respectively

Figure 8. Portfolio values for different diversification risk settings. Reference benchmarks are shown with dashed lines (Index or ETF assets). CGAN, ACGAN, and Markowitz with solid lines

and standard deviation of $\boldsymbol{x}$. We generate 1,000 synthetic series. Table 2 presents the mean correlation for different assets. In most cases, the correlation between the real series and synthetic series by ACGAN achieves larger values making the ACGAN a better model to generate financial series closer to the real ones. While there exist other correlation measurements, e.g., the Cosine correlation, and time-varying correlation (Tulchinsky, 2019), the results do not alter significantly.

Moreover, we analyze different statistical properties of the generated series as discussed in Section 3.3 [3]. Figure 6(a) shows the decay of the autocorrelation function of the price return on a daily timeframe (for GOOG asset in the US market). The absence of linear correlation in the price return on a daily timeframe suggests that the generated financial series are efficient to a certain extent. Similarly, the fat-tail, leverage effect, and coarse-fine properties of the generated data by CGAN and ACGAN all satisfy the statistical properties we have mentioned previously. Table 3 summarizes the coefficients of the real market data and data generated by CGAN and ACGAN, where we also include the Kurtosis and skewness of these assets. In practice, the Kurtosis of a normal distribution is 3 and the value is larger than 3 for a fat-tailed distribution. In most cases, the coefficients of the ACGAN are closer to the true ones.

4.2 Portfolio Analysis

After generating the synthetic series for each asset, we optimize over the fake series to generate minimal Sharpe ratio weight allocations (for ACGAN and CGAN). For Markowitz framework, the optimization is done over the past data. We consider three rebalance settings: a *defensive setting* with rebalancing every $\eta = 10$ days; a *balanced setting* with $\eta = 15$; and an *aggressive setting* with $\eta = 20$. Figure 7 presents the distribution of return-SR (Sharpe ratio) scatters with 1,000 draws from ACGAN and CGAN models, and the one from Markowitz framework. The *ACGAN (Mean)* and *CGAN (Mean)* are strategies by taking the average weight from these 1,000 draws on each rebalancing date. The points in the upper-right corner are the better ones. In all scenarios, the mean strategies of the proposed ACGAN model have better returns and Sharpe ratios than those of the Markowitz model and the CGAN model. When we apply the mean strategy in the US region, the ACGAN achieves both better return and Sharpe ratio evaluations compared to the mean strategy of CGAN. Similar results are observed in the EU region with balanced and aggressive settings with $\eta = 15, 20$. The return results of the ACGAN and CGAN models are close in the defensive setting for the EU region; while ACGAN has better Sharpe ratio such that it takes smaller risk.

Figure 8 shows the portfolio value series of mean strategies for ACGAN and CGAN, and the one from Markowitz framework along the test period where we initialize each portfolio with a unitary value. ACGAN dominates the other approaches in terms of the final portfolio value.

5. Conclusion

The aim of this paper is to solve the issue of poor prediction ability in the CGAN methodology for portfolio analysis. We propose a simple and computationally efficient algorithm that requires little extra computation and is easy to implement

[3] Other than these single-asset properties, we also report some cross-asset statistical properties in Appendix C.

for conditional time series generation. Overall, we show that the proposed ACGAN model is a versatile framework that synthesizes series with larger correlation to the true time series and thus yields better portfolio allocation. ACGAN is able to keep as much original information as possible while still enables the generator to generate data closer to the ones receiving high scores from discriminator. While it still remains interesting if the ACGAN model can be applied in computer vision context to generate images that are closer to the real ones.

Acknowledgments We greatly appreciate insightful discussions with Giovanni Mariani on the data normalization and the framework of the PAGAN (CGAN) methodology.

References

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, *31(*3), 307–327. https://doi.org/10.1016/0304-4076(86)90063-1.

Bouchaud, J.-P., Matacz, A., & Potters, M. (2001). Leverage effect in financial markets: The retarded volatility model. *Physical review letters*, *87(*22), 228701. https://doi.org/10.2139/ssrn.255868.

Chakraborti, A., Toke, I. M., Patriarca, M., & Abergel, F. (2011). Econophysics review: I. empirical facts. *Quantitative Finance*, *11(*7), 991–1012. https://doi.org/10.1080/14697688.2010.539248.

Cont, R. (2001). Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative finance*, *1(*2), 223. https://doi.org/10.1080/713665670.

Eckerli, F. & Osterrieder, J. (2021). Generative adversarial networks in finance: an overview. *arXiv preprint arXiv:2106.06364*. https://doi.org/10.2139/ssrn.3864965.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pages 987–1007. https://doi.org/10.2307/1912773.

Esteban, C., Hyland, S. L., & Rätsch, G. (2017). Real-valued (medical) time series generation with recurrent conditional GANs. *arXiv preprint arXiv:1706.02633*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, *27*.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of Wasserstein GANs. *Advances in neural information processing systems*, *30*.

Huang, W., Nakamori, Y., & Wang, S.-Y. (2005). Forecasting stock market movement direction with support vector machine. *Computers & operations research*, *32(*10), 2513–2522. https://doi.org/10.1016/j.cor.2004.03.016.

Kara, Y., Boyacioglu, M. A., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the istanbul stock exchange. *Expert systems with Applications*, *38(*5), 5311–5319.

Lee, D. D. & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401(*6755), 788–791. https://doi.org/10.1038/44565.

Lu, J. (2021). Numerical matrix decomposition and its modern applications: A rigorous first course. *arXiv preprint arXiv:2107.02579*.

Lu, J. (2022). Matrix decomposition and applications. *arXiv preprint arXiv:2201.00145*.

Lu, J. & Ye, X. (2022). Flexible and hierarchical prior for Bayesian nonnegative matrix factorization. *arXiv preprint arXiv:2205.11025*.

Lu, J. & Yi, S. (2022). Reducing overestimating and underestimating volatility via the augmented blending-ARCH model. *Applied Economics and Finance*, *9(*2), 48–59. https://doi.org/10.11114/aef.v9i2.5507.

Mariani, G., Zhu, Y., Li, J., Scheidegger, F., Istrate, R., Bekas, C., & Malossi, A. C. I. (2019). PAGAN: Portfolio analysis with generative adversarial networks. *arXiv preprint arXiv:1909.10578*.

Markowitz, H. M. (1968). Portfolio selection. In *Portfolio selection*. Yale university press.

Markowitz, H. M. (1976). Markowitz revisited. *Financial Analysts Journal*, *32(*5), 47–52.

Müller, U. A., Dacorogna, M. M., Davé, R. D., Olsen, R. B., Pictet, O. V., & Von Weizsäcker, J. E. (1997). Volatilities of different time resolutions—analyzing the dynamics of market components. *Journal of Empirical Finance*, *4(*2-3), 213–239.

Rydberg, T. H. (2000). Realistic statistical modelling of financial data. *International Statistical Review*, *68(*3), 233–258. https://doi.org/10.1111/j.1751-5823.2000.tb00329.x.

Sharpe, W. F. (1966). Mutual fund performance. *The Journal of business*, *39(*1), 119–138. https://doi.org/10.1086/294846.

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

Timmermann, A. & Granger, C. W. (2004). Efficient market hypothesis and forecasting. *International Journal of forecasting*, *20(*1), 15–27. https://doi.org/10.1016/S0169-2070(03)00012-8.

Tsay, R. S. (2005), *Analysis of financial time series*. John wiley & sons. https://doi.org/10.1002/0471746193.

Tulchinsky, I. (2019), *Finding Alphas: A quantitative approach to building trading strategies*. John Wiley & Sons. https://doi.org/10.1002/9781119571278.

A. Network Structures

We provide detailed structure for the neural network architectures we used in our experiments in this section. Given the number of assets N, historical length h, future length f ($w = h + f$), and latent dimension m for the prior distribution vector z, we consider multi-layer perceptron (MLP) structures, the detailed architecture for each fully connected layer is described by F($\langle$*num inputs*$\rangle$: $\langle$*num outputs*$\rangle$: $\langle$*activation function*$\rangle$); for an activation function of LeakyRelu with parameter p is described by LR($\langle p \rangle$); and for a dropout layer is described by DP($\langle$*rate*$\rangle$). The *conditioner* in CGAN shares the same structure as the *encoder* in the ACGAN model (see Figure 2 and Figure 3). Then the network structures we use can be described as follows:

$$\textbf{Conditioner} = \textbf{Encoder} = \text{F}(N \cdot h\text{: } 512\text{: LR}(0.2)) \cdot \text{F}(512\text{:}512\text{:LR}(0.2)) \cdot \text{DP}(0.4) \cdot \text{F}(512\text{:}16) \tag{10}$$

$$\textbf{Decoder} = \text{F}(16\text{:}512\text{:LR}(0.2)) \cdot \text{F}(512\text{:}512\text{:LR}(0.2)) \cdot \text{DP}(0.4) \cdot \text{F}(512\text{:}N \cdot h) \tag{11}$$

$$\textbf{Simulator} = \text{F}(m\text{+}16\text{:}128\text{:LR}(0.2)) \cdot \text{F}(128\text{:}256\text{:LR}(0.2)) \cdot \text{F}(256\text{:}512\text{:LR}(0.2)) \cdot \text{F}(512\text{:}1024\text{:LR}(0.2)) \cdot \text{F}(1024\text{:}N \cdot f\text{:TanH}) \tag{12}$$

$$\textbf{Discriminator} = \text{F}(N \cdot (h\text{+}f)\text{:}512\text{:LR}(0.2)) \cdot \text{F}(512\text{:}512\text{:LR}(0.2)) \cdot \text{DP}(0.4) \cdot \text{F}(256\text{:}512\text{:LR}(0.2)) \cdot \text{F}(512\text{:}1) \tag{13}$$

We trained networks using Adam's optimizer with learning rate 2×10^{-5}, $\beta_1 = 0.5$, and $\beta_2 = 0.999$. We set the penalization parameters $\lambda_1 = 10$ and $\lambda_2 = 3$. The latent dimension is $m = 100$. And we trained models for 1,000 epochs.

B. More Samples for the Generative Models

In this section, we provide more synthetic results for both ACGAN and CGAN models. We set window size $h = 40$, $f = 20$ and $w = 60$ in all experiments. As shown in the main paper, Figure 5 presents the actual price trend (black solid line) of the US assets and the EU assets for the first 100 trading days in the test set, and five representative simulations generated by ACGAN (colored dashed lines). Figure 9 reports the simulated price series of the whole period (800 trading days). And simulated price series of the CGAN model (100 and all trading days) can be found in Figure 10, and 11 respectively.

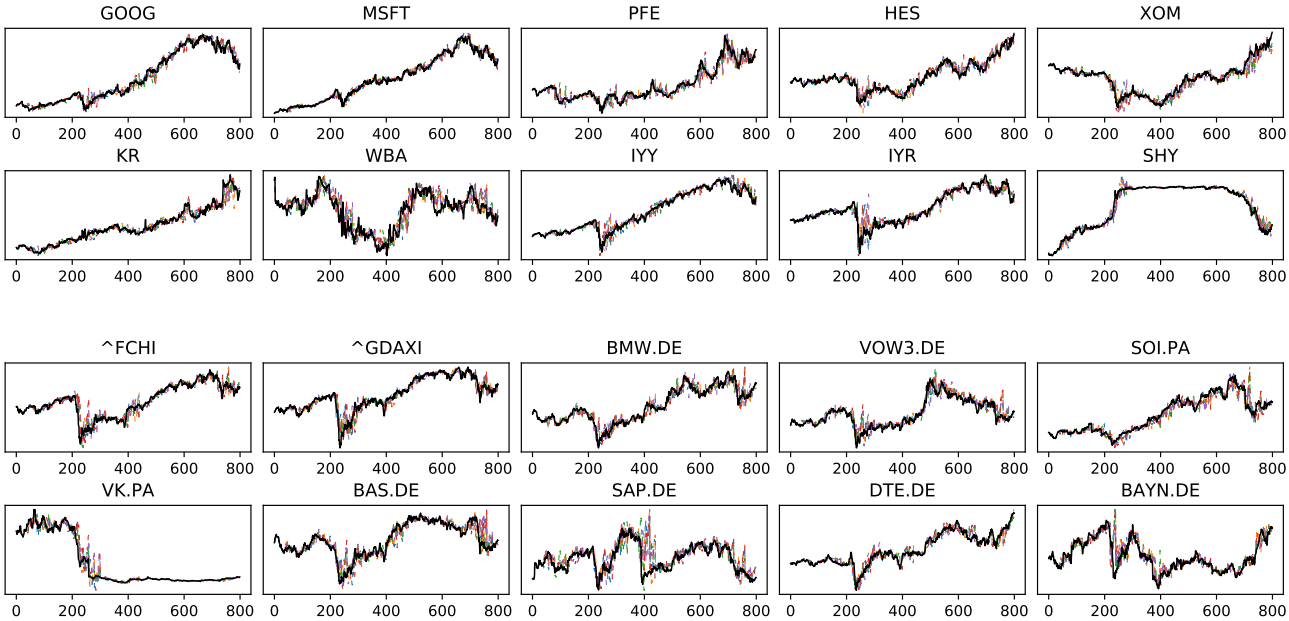


Figure 9. **ACGAN:** Actual price trend (black solid line) of the US assets (upper) and the EU assets (lower) for 800 trading days in the test set (the whole period of the test set), and five representative simulations generated by ACGAN (colored dashed lines)

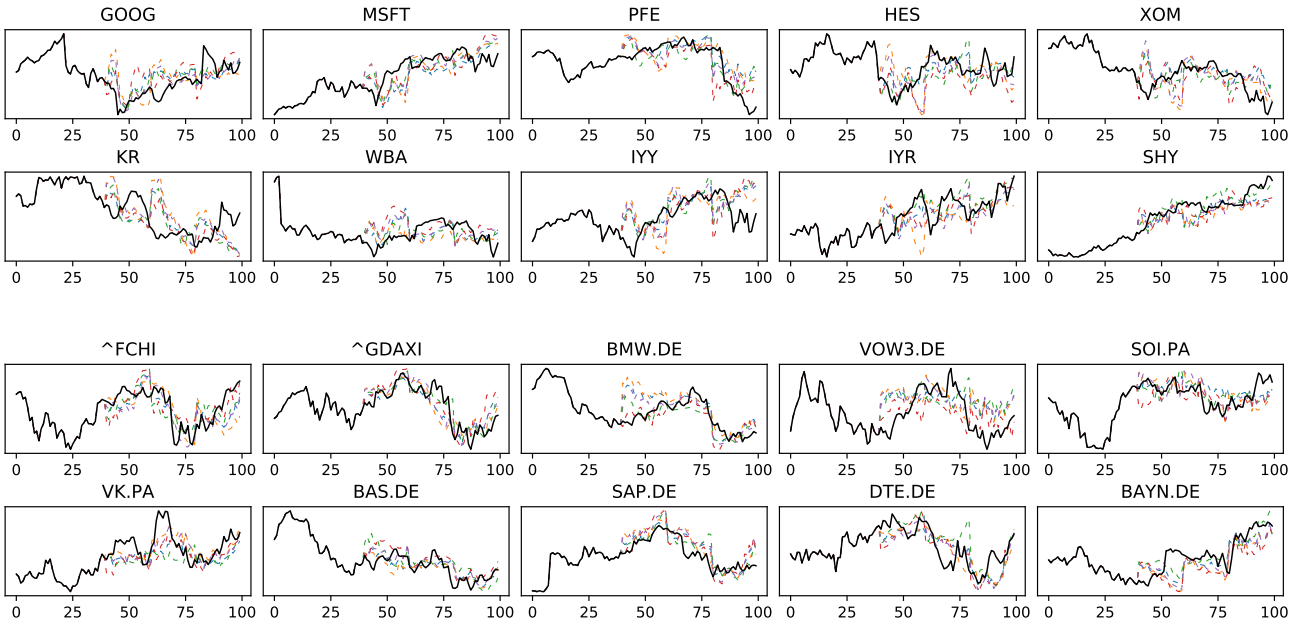


Figure 10. **CGAN:** Actual price trend (black solid line) of the US assets (upper) and the EU assets (lower) for the first 100 trading days in the test set, and five representative simulations generated by CGAN (colored dashed lines)

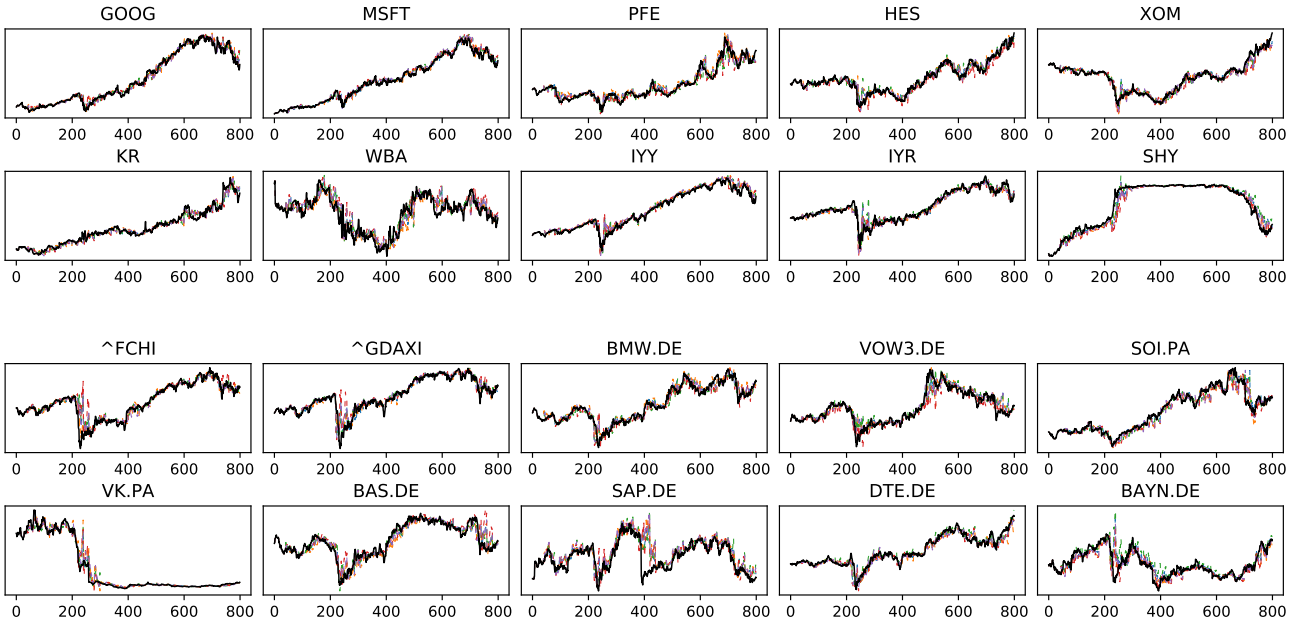


Figure 11. **CGAN:** Actual price trend (black solid line) of the US assets (upper) and the EU assets (lower) for 800 trading days in the test set (the whole period of the test set), and five representative simulations generated by CGAN (colored dashed lines)

C. Statistical Properties for Multiple Assets

Following Cont (2001), we also consider the stylized facts across different assets, namely the cross-asset correlation, volatility correlation, and cross-asset leverage effect.

Cross-Asset Correlation Given two assets i, j whose return values at time t are denoted by $r_{i,t}, r_{j,t}$ respectively. *Cross-asset correlation* considers the correlation of two assets at different time frames. Under weak market efficiency, there

Statistics	Real (US)	ACGAN (US)	CGAN (US)	Real (EU)	ACGAN (EU)	CGAN (EU)
Cross correlation	0.0204	**0.0223**	**0.0219**	0.0190	**0.0205**	**0.0191**
Volatility correlation	0.1855	**0.1872**	0.1748	0.1279	**0.1410**	**0.1405**
Cross leverage effect	-10.2414	**-6.5860**	0.9129	-6.4018	**-3.6489**	-2.9875

Table 4. Cross statistical properties of the real and generated series. Cross correlation value is the mean of the first 10-th lags of cross correlation coefficients (across different combinations of assets); Similarly, the volatility correlation and cross leverage effect shown in the table consider the mean of first 10-th lags of leverage effect coefficients (across different combinations of assets). ACGAN shows promise in terms of the cross leverage effect

should be no obvious lead-lag cross correlation between different assets; otherwise, it will provide arbitrage space:

$$\text{Corr}(r_{i,t}, r_{j,t+k}) = \frac{\mathbb{E}[(r_{i,t} - \mu_i)(r_{j,t+k} - \mu)]}{\sigma_i \sigma_j} \approx 0, \forall\ k \geq 2,$$

where μ_i, σ_i are the mean and standard deviation of return series $\boldsymbol{r}_i$ of asset i. In practice, the cross correlation presents positive correlation for the 0-th and 1-th lag values; and no significant correlation for higher order values. Table 4 considers the mean coefficients of the first 10 orders across different combinations of assets. For 10 assets, there are 45 such combinations.

Volatility Correlation The *volatility correlation* consider the lead-lag cross correlation between the absolute values of different asset returns:

$$\text{Corr}(|r_{i,t}|, |r_{j,t}|).$$

Although there is no significant cross correlation between different asset return series under the efficient markets hypothesis, its volatility may show a significant cross correlation. We may observe that the increasing in the volatility of one asset can lead to the rise of the volatility of another asset. In the short term, there is a large positive correlation between volatilities of different assets; and the cross correlation will gradually decline over time. Table 4 considers the mean coefficients of the first 10 orders across different combinations of assets.

Cross-Asset Leverage Effect Similar to the single-asset leverage effect (Section 3.3), the *cross-asset leverage effect* means there is a negative correlation between the past price returns of one asset and the future volatility of another asset:

$$L_{ij}(k) = \frac{\mathbb{E}[r_{i,t}|r_{j,t+k}|^2] - \mathbb{E}[r_{i,t}] \cdot \mathbb{E}[|r_{j,t}|^2]}{\mathbb{E}[|r_{j,t}|^2]^2}.$$

Again, Table 4 considers the mean coefficients of the first 10 orders across different combinations of assets.

We can observe from Table 4 that the cross-asset correlation and volatility correlation of ACGAN and CGAN are close. However, ACGAN shows promise in terms of the cross-asset leverage effect.