NEURAL FUNCTIONALLY GENERATED PORTFOLIOS

MICHAEL MONOYIOS AND OLIVIA PRICILIA

Abstract. We introduce a novel neural-network-based approach to learning the generating function $G(\cdot)$ of a functionally generated portfolio (FGP) from synthetic or real market data. In the neural network setting, the generating function is represented as $G_\theta(\cdot)$, where θ is an iterable neural network parameter vector, and $G_\theta(\cdot)$ is trained to maximise investment return relative to the market portfolio. We compare the performance of the Neural FGP approach against classical FGP benchmarks. FGPs provide a robust alternative to classical portfolio optimisation by bypassing the need to estimate drifts or covariances. The neural FGP framework extends this by introducing flexibility in the design of the generating function, enabling it to learn from market dynamics while preserving self-financing and pathwise decomposition properties.

1. Introduction

Stochastic Portfolio Theory (SPT), introduced by Fernholz in [6] and developed further in the monograph [5], provides a descriptive framework for modelling equity markets that aligns closely with observed empirical behaviour. Unlike classical models based on strong no-arbitrage assumptions, SPT operates under a weaker assumption: no unbounded profit with bounded risk (NUPBR), which is equivalent to the existence of a non-empty set of local martingale deflators (LMDs), as established by Karatzas and Kardaras in [12]. It does not require the existence of an equivalent local martingale measure (ELMM), which (as established by [2]) is equivalent to the stronger no-arbitrage condition of No Free Lunch with Vanishing Risk (NFLVR) and corresponds to the scenarios when the LMDs are in fact true martingales.

Despite relying on weaker assumptions, SPT offers several key advantages. In particular, it can often develop investment strategies which avoid the need to estimate the drift or volatility of individual assets, making it more robust to model mis-specification. A central objective in this framework is the construction of portfolios that almost surely outperform the market over finite horizons, a concept known as *relative arbitrage*.

This idea is made tractable through the concept of **Functionally Generated Portfolios** (FGPs), introduced by Fernholz [7]. FGPs are constructed from deterministic generating functions of observable market weights, allowing one to bypass latent quantities such as drift and covariance. The pathwise nature of their construction makes them both theoretically elegant and practically appealing. We review the theory and properties of FGPs in Section 2.

Functionally Generated Portfolios (FGPs) can thus offer a robust alternative to classical portfolio optimisation by avoiding the need to estimate drifts or covariances. The Neural FGP framework extends this approach by introducing flexibility in the design of the generating function, allowing it to learn directly from market dynamics while preserving self-financing properties and pathwise wealth decomposition. The full methodology of the Neural FGPs is presented in Section 3.

The remainder of the paper is organised as follows: Section 2 introduces the fundamentals of functionally generated portfolios (FGPs) and discusses briefly their potential robustness to

Date: June 24, 2025.

Key words and phrases. Stochastic Portfolio Theory, Functionally Generated Portfolios, Neural Networks, Robust Portfolio Optimisation, Relative Arbitrage, Financial Machine Learning.

Olivia Pricilia acknowledges funding support from an EPSRC CDT Studentship.

model uncertainty. It also provides a brief overview of relative arbitrage, optimal relative arbitrage, and the motivation for studying Neural FGPs. Section 3, the core of the paper, presents the Neural FGP methodology along with performance evaluation results. Finally, Section 4 highlights the main contributions, concludes the paper, and outlines potential directions for future research.

2. Functionally generated portfolios and model uncertainty

Functionally Generated Portfolios (FGPs), introduced by Fernholz [7], are central to Stochastic Portfolio Theory (SPT), enabling the construction of self-financing stock-only portfolios based solely on observable market weights. Their defining feature is a pathwise decomposition of relative wealth (to be defined precisely later), which does not require drift or volatility estimation.

In a market with n stocks, let $\mu(t) \in \Delta^n_+$ denote the vector of strictly positive market weights at time $t \geq 0$, where each component $\mu_i(t)$, $i = 1, \dots, n$ represents the capitalisation $X_i(t)$ of the i^{th} stock divided by the total market capitalisation $\sum_{j=1}^n X_j(t)$.[1] Here, Δ^n_+ denotes the interior of the positive unit simplex:

$$\Delta^n_+ := \{a \in \mathbb{R}^n : \textstyle\sum_{i=1}^n, \, a_i = 1 \text{ and } a_i > 0 \text{ for all } i = 1, \dots, n\}.$$

Let $V^{w,\pi}(\cdot)$ be the wealth process starting from initial capital $w > 0$ when using a portfolio $\pi(t)$, $t \geq 0$, where $\pi(t)$ is a vector of proportions of wealth allocated to each stock at time $t \geq 0$. For notational simplicity, we write $V^\pi(\cdot) := V^{1,\pi}(\cdot)$, for initial capital $w = 1$. We assume the ***admissibility*** condition

$$\mathbb{P}\left(V^{w,\pi}(t) \geq 0, \ \forall t \in [0,T]\right) = 1.$$

With this notation, the *market portfolio* corresponds to the choice $\pi \equiv \mu$.

A portfolio π is said to be *functionally generated* if there exists a C^2, strictly positive function $G : U \mapsto (0,\infty)$, defined on an open neighbourhood $U \supseteq \Delta^n_+$, such that the portfolio weights take the form $\pi_i(\cdot) \equiv \pi_i(\mu(\cdot))$, given by

$$\pi_i(t) = \left(D_i \log(G(\mu(t))) + 1 - \sum_{j=1}^n \mu_j(t) D_j \log G(\mu(t)) \right) \mu_i(t), \quad i = 1, \dots, n, \tag{2.1}$$

where $D_i \equiv \partial/\partial x_i$. This prescription ensures that the strategy is self-financing and remains within the simplex. Some examples of FGPs are shown in Table 1.

Table 1. Examples of Functionally Generated Portfolios

Portfolio	Generating Function $G(\mu)$	Weights $\pi_i(\mu)$
Equally Weighted Portfolio (EWP)	$\prod_{i=1}^n \mu_i^{1/n}$	$\pi_i = 1/n$
Diversity Weighted Portfolio (DWP)	$\sum_{i=1}^n \mu_i^p$, $0 < p < 1$	$\pi_i = \mu_i^p / \sum_j \mu_j^p$
Entropy Weighted Portfolio	$-\sum_{i=1}^n \mu_i \log \mu_i$	$\propto -\log \mu_i$
Market Portfolio	Constant: $G(\mu) = c > 0$	$\pi_i = \mu_i$

In a continuous time setting, with the market capitalisations following Itô processes, it turns out that the *relative wealth* $V(\cdot)$ of an FGP π relative to the market portfolio μ, defined (for the case of unit initial capital) by

$$V(t) := V^\pi(t)/V^\mu(t), \quad t \geq 0, \tag{2.2}$$

[1] Without loss of generality, we follow the usual convention in SPT that there is one share of each company in the market with all stock positions infinitely divisible, so $X_i(\cdot)$ also represents the i^{th} stock price process.

satisfies, over a time horizon $[0, T]$, the so-called *master equation*

$$\log\left(\frac{V^\pi(T)}{V^\mu(T)}\right) = \log\left(\frac{G(\mu(T))}{G(\mu(0))}\right) + \int_0^T q(t)\,\mathrm{d}t, \tag{2.3}$$

where the drift term is given by

$$q(t) := \frac{-1}{2G(\mu(t))}\sum_{i,j=1}^{n} D^2_{ij}G(\mu(t))\mu_i(t)\mu_j(t)\tau^\mu_{ij}(t), \quad t \geq 0, \tag{2.4}$$

and with $\tau^\mu_{ij}(\cdot)$ denoting the covariance structure of the market weights (the covariances of individual stocks relative to the entire market), that is:

$$\tau^\mu_{ij}(t) := \langle \log\mu_i, \log\mu_j\rangle(t), \quad t \geq 0. \tag{2.5}$$

Fernholz's Master Formula (2.3) (see Fernholz [5, Chapter 3] for a derivation), allows for the explicit construction of portfolios with relative performance guarantees.

The FGP weights (2.1) (or, indeed, the master formula, given that (2.5) holds) shows that the wealth process V^π of an FGP relative to the market depends only on observable quantities. This robustness makes FGPs potentially attractive under high model uncertainty, where latent parameters such as expected returns are difficult to estimate reliably.

2.1. **On Relative Arbitrage and Optimal Relative Arbitrage.** As alluded to in the Introduction, a central objective of Stochastic Portfolio Theory (SPT) is to construct portfolios that almost surely outperform the market over finite horizons. This concept, known as *Relative Arbitrage (RA)*, is fundamental to the SPT framework.

Definition 2.1 (Relative Arbitrage)**.** [8, Definition 6.1] [9, (2.10) and (2.11)]

A portfolio $\pi_1(.)$ is a ***relative arbitrage (RA)*** with respect to portfolio π_2 over the time interval $[0, T]$ if the associated wealth process (starting from the same initial wealth w) satisfies:

$$\mathbb{P}(V^{w,\pi_1}(t) \geq V^{w,\pi_2}(t)) = 1, \text{ and } \mathbb{P}(V^{w,\pi_1}(t) > V^{w,\pi_2}(t)) > 0. \tag{2.6}$$

We call this a ***strong relative arbitrage*** if it satisfies:

$$\mathbb{P}(V^{w,\pi_1}(t) > V^{w,\pi_2}(t)) = 1. \tag{2.7}$$

Beyond constructing arbitrage, one can ask: what is the *best* way to beat the market? This leads to the notion of *optimal relative arbitrage*. As formulated by D. Fernholz and I. Karatzas in [3], this refers to the minimum initial capital required to outperform the market almost surely over a finite time horizon:

$$u(T) := \inf\left\{w > 0 : \exists\,\pi \text{ such that } \mathbb{P}\left(V^{w,\pi}(T) \geq V^{w,\mu}(T)\right) = 1\right\}. \tag{2.8}$$

This can be reformulated as the problem of maximising, over the portfolio choice π, the return on investment, relative to the market, over the time horizon $[0, T]$:

$$\log\left(\frac{V^\pi(T)}{V^\mu(T)}\right) = \log\left(V(T)\right) \to \max! \tag{2.9}$$

where, in the case that π is an FGP, and in a continuous-time Itô process setting for the market capitalisations, the relative wealth process $V(\cdot)$ of (2.2) satisfies the master formula (2.3).

In a complete Markovian diffusion setting, the problem of maximising return relative to the market was studied by D. Fernholz and I. Karatzas [3] (see also their associated book chapter [4]). The best possible relative arbitrage in this setting is characterised via the smallest solution of a linear parabolic Cauchy problem. In particular, the highest relative return that can be achieved over $[0, T]$ is given by $1/U(T, \mathbf{X}(0))$, where $\mathbf{X}(0)$ is the initial configuration of the market weight vector, and $U(T - t, \mathbf{x})$, $t \in [0, T]$ (sometimes called the "arbitrage function" by Fernholz and Karatzas) is the solution to the associated Cauchy problem. There are also probabilistic characterisations involving the probability, under the so-called exit measure of Föllmer [10, 11] that the market weights remain in the unit simplex.

The salient point from our perspective is that the portfolio which realises the best return *turns out to be functionally generated* (see, for example, [3, Remark 6]) with generating function directly related to the "arbitrage function" that arises as the solution to a Cauchy problem.

The notion that FGPs emerge naturally as portfolios that can best beat the market, and in particular when there is some form of model uncertainty, has also emerged in other contexts, such as problems of robust long-term growth, when the expected returns of assets are unknown, as shown by Kardaras and Robertson [13, 14].

These properties of FGPs, that they often arise as optimal solutions to relative outperformance problems in the face of model uncertainty, serve as motivation for the approach we adopt in this paper, namely a neural network-based methodology to seek optimal relative return, when the portfolio is constrained to be functionally generated. Our approach is flexible and readily scalable to high-dimensional settings, avoiding some of the implementation limitations of the classical approach in [3], which would be computationally challenging in high dimensions, as it involves solving an associated parabolic PDE system.

2.2. **From Classical to Neural FGPs.** Building on the theoretical foundation of classical FGPs, we propose a **Neural FGP** framework that preserves the key structural properties: self-financing evolution, pathwise decomposition (as in (2.3), with portfolio weights given by (2.1)), and direct dependence on observable quantities, while allowing for greater flexibility in the design of the generating function. As mentioned above, our methodology avoids the need to solve a partial differential inequality, as in [3].

In our framework, the generating function $G(\cdot)$ is parameterised by a neural network trained on historical market data. This allows the portfolio to adapt to complex, possibly nonlinear features in the market weight dynamics, without sacrificing the interpretability and robustness of the original FGP formulation. The Neural FGP framework thus offers a hybrid approach that combines the theoretical rigour of SPT with the data-adaptivity of modern machine learning.

3. Neural Functionally Generated Portfolios (Neural FGPs)

We introduce a neural-network-based extension of Functionally Generated Portfolios (FGPs), in which the generating function is learned directly from data. This approach preserves the theoretical structure of classical FGPs while enhancing flexibility and adaptability to market dynamics. To the best of our knowledge, this extension has not appeared in the existing SPT or portfolio optimisation literature. The following sections detail the methodology and evaluate its performance against classical FGP benchmarks.

3.1. Neural FGP Architecture and Training Framework.

3.1.1. *Neural Network Approximation of the Generating Function.* To construct a neural FGP, we approximate the generating function $G(\cdot)$ using a parametrised version $G_\theta(\cdot)$, where θ is the learning vector in a neural network. From this point onwards, we use x in place of μ to preserve notational consistency with standard neural network frameworks. We compute the portfolio weights according to the classical prescription (2.1). In the current notation, we have a portfolio weight for the i^{th} stock similarly parametrised by the learning vector, and given by

$$\pi_{\theta,i}(x) = \left(\frac{\partial \log G_\theta(x)}{\partial x_i} + 1 - \sum_j x_j \frac{\partial \log G_\theta(x)}{\partial x_j} \right) x_i, \quad i = 1, \dots, n. \tag{3.1}$$

This construction ensures that the resulting portfolio remains long-only and self-financing. We employ automatic differentiation on $\log G_\theta(\cdot)$ to compute the derivatives in (3.1), in a manner that thus mirrors the classical FGP structure.

3.1.2. *Enforcing Concavity via Input Convex Neural Networks (ICNN).* Traditional neural networks may not enforce the concavity required for generating functions in FGPs. To resolve this, we adopt ***Input Convex Neural Networks (ICNN)***, first introduced in [1], and specifically designed to generate convex outputs based on their inputs.

We ensure the concavity of the learned generating function $G_\theta(\cdot)$ by defining it as the negative of an Input Convex Neural Network (ICNN) output:

$$G_\theta(x) := -f(x),$$

where $f(\cdot)$ is a convex function constructed using an ICNN architecture, defined below.

Let the network input be $x \in \mathbb{R}^n$, where n is the number of stocks. The ICNN defines a sequence of intermediate variables $z_k \in \mathbb{R}^{m_k}$ recursively, as follows:

$$\begin{aligned} z_0 &= 0, \\ z_1 &= \phi(W_0 x + b_0), \\ z_2 &= \phi(W_1 z_1 + U_1 x + b_1), \\ &\vdots \\ z_K &= \phi(W_{K-1} z_{K-1} + U_{K-1} x + b_{K-1}), \end{aligned}$$

with final output

$$f(x) = w^\top z_K + u^\top x + c.$$

The full set of learnable parameters of the ICNN is denoted by θ, defined by:

$$\theta := \left\{ \{W_k, U_k, b_k\}_{k=0}^{K-1}, \, w, u, c \right\}.$$

These parameters are subject to the following architectural constraints:

- $W_k \in \mathbb{R}^{m_{k+1} \times m_k}$: unconstrained weight matrices,
- $U_k \in \mathbb{R}^{m_{k+1} \times n}$: weight matrices constrained to have non-negative entries (that is, $U_k \geq 0$, elementwise),
- $b_k \in \mathbb{R}^{m_{k+1}}$: bias vectors for $k = 0, \ldots, K-1$,
- $w \in \mathbb{R}^{m_K}$, $u \in \mathbb{R}^n$, $c \in \mathbb{R}$: output layer parameters.

Here, $m_k \in \mathbb{N}$ denotes the width of the k^{th} hidden layer, and $\phi(\cdot)$ is a convex and non-decreasing activation function (for example, `Softplus` or ReLU). These structural constraints ensure that $f(x)$ is convex in x, and thus that $G_\theta(x)$ is concave.

3.1.3. *Training Objective.* We train the model to maximise the logarithmic return on investment relative to the market, that is, the objective in (2.9). Let $x_i(s)$ denote the *relative* capitalisation of the i^{th} stock at time step s, that is, the capitalisation of stock i stock divided by the total market capitalisation. For notational convenience, we denote this as $x_{s,i}$, and similarly denote the relative wealth process by $V_t \equiv V(t)$. The discrete-time evolution of the relative wealth process V is given by

$$V_t = \prod_{s=1}^{t} \sum_{i=1}^{n} \pi_i(x_{s-1}) \frac{x_{s,i}}{x_{s-1,i}}, \tag{3.2}$$

with $\pi_i(\cdot)$ defined in (2.1) (with its corresponding neural-vector parametrised form as in (3.1)).

We optimise the *logarithmic return relative to the market*, given by $\log V_T$, the logarithm of relative wealth at the terminal time T. This objective is consistent with the continuous-time formulation (2.9).

The objective is recast as a loss function, incorporating an ℓ_2 regularisation term to discourage overly concentrated portfolios and promote numerical stability during training:

$$\mathcal{L}(\theta) = -\frac{1}{T}\log(V_T) + \lambda\|\pi(x)\|_2. \tag{3.3}$$

This discrete-time objective mirrors the continuous-time formulation associated with optimal relative arbitrage discussed in Section 2, ensuring theoretical consistency in our learning framework.

Remark 3.1 (ICNN versus a Hessian Penalty)*.* To encourage concavity in the function $G(\cdot) = -f(\cdot)$, another possible approach would involve adding a **Hessian penalty** to the loss function, which discourages regions of positive curvature by penalising the positive eigenvalues of the Hessian of $G_\theta(\cdot)$. This would yield a total loss of the form

$$\mathcal{L}_{\text{total}}(\theta) = \mathcal{L}(\theta) + \lambda \cdot \sum_{i=1}^{n} \max\left(\lambda_i^{(x)}, 0\right)^2, \tag{3.4}$$

where $\lambda_i^{(x)}$ are the eigenvalues of the Hessian at input x.[2]

While this method has a sound theoretical basis, it is numerically unstable and computationally inefficient. Moreover, it only encourages concavity rather than enforcing it and requires careful tuning of the regularisation parameter λ. In contrast, Input Convex Neural Networks (ICNNs) impose convexity by architectural design, thereby ensuring that $G(\cdot)$ is concave by construction.

3.2. Implementation and Experiments.

3.2.1. *Simulation Workflow and Implementation Details.* To evaluate the performance of the Neural FGP with ICNN, we conduct simulations on both synthetic and real-world datasets. For synthetic data, we generate geometric Brownian motion (GBM) paths for asset prices and normalise them to compute market weights. For real data, we fetch historical prices from Yahoo Finance. The neural network is trained via back-propagation using these inputs, and we derive the resulting portfolio weights through the gradient of the learned generating function. The procedure involves the following steps:

(1) **Data type and input configuration.**
The user specifies whether to use real equity market data or synthetic geometric Brownian motion (GBM) paths. The following input parameters are defined:
- `use_real`: Boolean flag indicating real vs synthetic data
- `n`: Number of assets in the portfolio
- `y`: Time window (in years) of historical data (if using real data)
- `p_vals`: Exponents used for DWP benchmark comparisons

(2) **Data loading and normalisation.**
For synthetic data, we simulate GBM price paths with randomly sampled drifts and volatilities over 1000 days. For real data, we fetch daily adjusted closing prices for the selected tickers (e.g., AAPL, MSFT, GOOG, AMZN, META) over the past $y = 5$ years via Yahoo Finance, ending on Friday, 11 April 2025. In both cases, the price series is normalised at each time step to compute the market weight vector $x_t \in \Delta_+^n$. This normalisation implicitly sets the market portfolio as the numéraire; that is, each asset's capitalisation is divided by total market capitalisation at each time.

(3) **Model architecture and portfolio rule.**
An Input Convex Neural Network (ICNN) is initialised with input dimension equal to the number of assets. The model outputs a convex function $G(x)$, and portfolio

[2]Recall that a function is concave if the Hessian is negative semidefinite, that is, all eigenvalues are nonpositive.

weights are computed via the gradient $\nabla \log G(x)$, following the structure of functionally generated portfolios (FGPs). The gradient is projected onto the simplex and scaled for numerical stability.

(4) **Training and optimisation.**
The ICNN is trained via back-propagation to maximise logarithmic portfolio returns relative to the market over a window of 200 trading days. A regularisation term penalising extreme weight concentration is included to encourage robustness. The training objective is given by the loss function in (3.3), and the optimisation is carried out using the Adam optimiser.

(5) **Walk-forward evaluation.**
A sliding window approach is used: for each 200-day training segment, we evaluate the trained model on the subsequent 20-day out-of-sample period. This walk-forward setup mimics realistic deployment by retraining on rolling windows and helps mitigate over-fitting.

3.2.2. *Benchmark Strategies and Evaluation.* To evaluate the performance of the neural FGP portfolio, we compare it against three classical FGP benchmark strategies:

- **Equally-Weighted Portfolio (EWP):** assigns equal weights to each asset, that is, $\pi_i^{\text{equal}} = 1/n$
- **Market Portfolio (MP):** invests proportionally to market weights, that is, $\pi^{\text{market}}(x) = x$
- **Diversity-Weighted Portfolio (DWP):** defined as $\pi_i^{\text{DWP}}(x) = x_i^p / \sum_j x_j^p$, with $p = 0.3, 0.5, 0.8$

We track the relative wealth evolution V as defined in (3.2) for each portfolio strategy using the walk-forward setup described earlier. Specifically, the model is retrained every 200 trading days and evaluated on the subsequent 20-day out-of-sample window. For each test window, V_T denotes the terminal relative wealth at the end of that 20-day period, which we denote by V_{T_k} for the k^{th} evaluation.

The resulting trajectories of V_{T_k} across all walk-forward windows are shown in Figure 1 (synthetic data) and Figure 2 (real data). Since we explicitly normalise by the market portfolio, $V_{T_k} = V^\pi(T_k)/V^\mu(T_k)$ captures portfolio performance *relative to the market* at time T_k.

In both settings, the Neural FGP consistently outperforms all benchmark strategies across nearly all evaluation periods. We report the average of the logarithmic relative return across all test windows in Table 2 (synthetic data) and Table 3 (real data). *Note that this is not a time-average over each* 20*-day testing window, but rather an average of terminal relative log-returns* $\log(V_{T_1}), \log(V_{T_2}), \ldots, \log(V_{T_K})$*, where each* T_k *denotes the end of the* k^{th} 20*-day test period. The quantity reported is the average terminal log-relative return* $(1/K)\sum_{k=1}^{K} \log(V_{T_k})$*. For the simulated dataset, we have* $K = (1000 - 220)/20 = 39$*, and for the real dataset,* $K = (252 \times 5 - 220)/20 = 52$.

Table 2. Average Logarithmic Relative Return $(1/K)\sum_{k=1}^{K} \log(V_{T_k})$ over Walk-Forward Test Windows (Synthetic Data)

Strategy	$(1/K)\sum_{k=1}^{K} \log(V_{T_k})$
FGP	0.078694
EWP	0.0092019
Market	-9.79007e-07
DWP $p = 0.3$	0.00660325
DWP $p = 0.5$	0.00478323
DWP $p = 0.8$	0.0019446

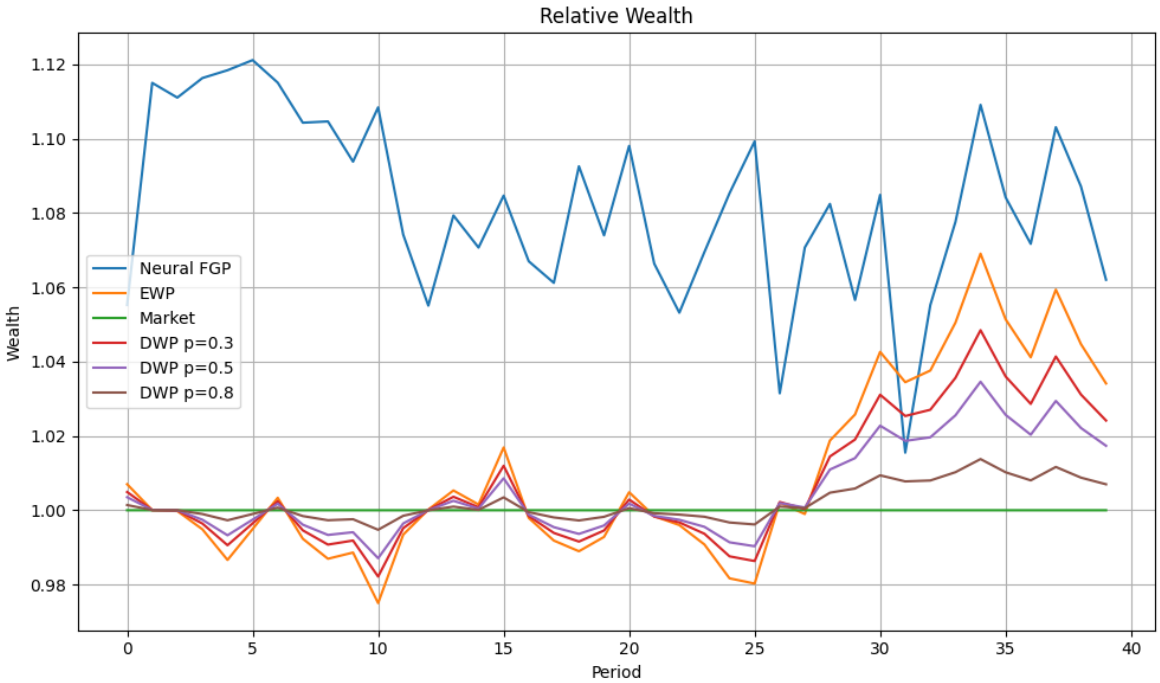

FIGURE 1. Relative terminal wealth V_{T_k} for $k = 1, \ldots, 39$, evaluated on synthetic data.

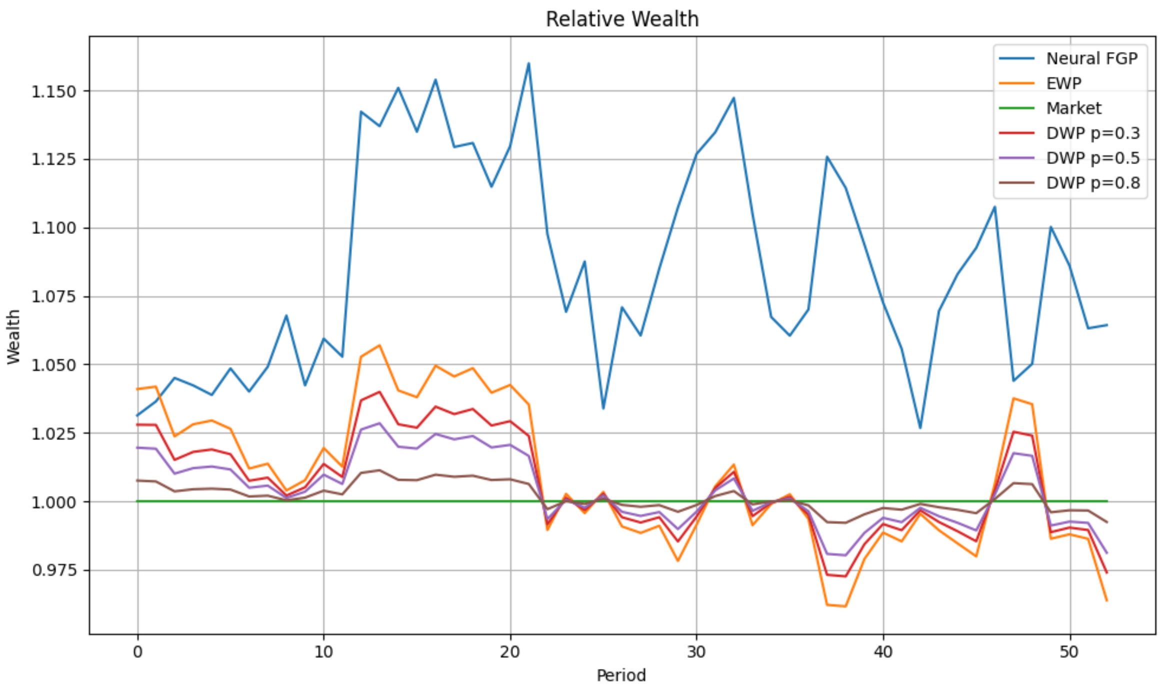

FIGURE 2. Relative terminal wealth V_{T_k} for $k = 1, \ldots, 52$, evaluated on real data.

4. CONCLUSION AND OUTLOOK

4.1. **Contribution and Novelty.** Our main contribution is the integration of neural networks with the FGP framework in a manner that preserves the FGP structure. In contrast to

TABLE 3. Average Logarithmic Relative Return $(1/K)\sum_{k=1}^{K}\log(V_{T_k})$ over Walk-Forward Test Windows (Real Data)

Strategy	$(1/K)\sum_{k=1}^{K}\log(V_{T_k})$
FGP	0.0785746
EWP	0.0121577
Market	-1.07738e-06
DWP $p = 0.3$	0.00825373
DWP $p = 0.5$	0.00576254
DWP $p = 0.8$	0.00221836

Vervuurt and Kom Samo [15], who addressed the inverse problem using Gaussian Processes and focused on learning portfolio weights directly, our method retains a differentiable generating function. This approach allows us to remain within the classical SPT framework while benefiting from data-driven adaptivity. Specifically, our approach:

- Trains $G_\theta(\cdot)$ directly, maintaining the structure of classical FGPs.
- Enforces concavity through ICNN.
- Evaluates performance under a natural logarithmic relative return criterion.

4.2. **Concluding remarks, future directions.** We have developed a hybrid model that integrates the theoretical structure of functionally generated portfolios (FGPs) with the flexibility and adaptivity of neural networks. This neural FGP framework maintains the self-financing, pathwise decomposition properties of classical FGPs while learning the generating function $G(\cdot)$ directly from synthetic or real market data. The architecture allows the portfolio to adapt dynamically to evolving data patterns without drift or volatility estimation. The performance of neural FGP is unquestionable, with evidence from our empirical evaluations showcasing its ability to outperform the classical benchmark, including market, equal-weighted, and diversity-weighted portfolios. Neural FGP is both interpretable and robust, offering a promising data-driven approach that remains grounded in the SPT framework.

As future research directions, we can make Neural FGP more realistic for practitioners. This includes exploring the robustness of Neural FGP performance across different market regimes and market characteristics, accounting for transaction costs and portfolio turnover, and addressing the scalability and computational efficiency issues.

References

[1] B. Amos, L. Xu, and J. Z. Kolter, *Input convex neural networks*, in International conference on machine learning, PMLR, 2017, pp. 146–155.
[2] F. Delbaen and W. Schachermayer, *A general version of the fundamental theorem of asset pricing*, Mathematische Annalen, 300 (1994), pp. 463–520.
[3] D. Fernholz and I. Karatzas, *On optimal arbitrage*, The Annals of Applied Probability, 20 (2010), pp. 1179 – 1204.
[4] D. Fernholz and I. Karatzas, *Probabilistic aspects of arbitrage*, in Contemporary Quantitative Finance: Essays in Honour of Eckhard Platen, Springer, 2010, pp. 1–17.
[5] E. R. Fernholz, *Stochastic portfolio theory*, Springer, 2002.
[6] R. Fernholz, *On the diversity of equity markets*, Journal of Mathematical Economics, 31 (1999), pp. 393–417.
[7] R. Fernholz, *Portfolio Generating Functions*, in Quantitative Analysis In Financial Markets Collected Papers of the New York University Mathematical Finance Seminar, M. Avellaneda, ed., World Scientific Book Chapters, World Scientific Publishing Co. Pte. Ltd., September 1999, ch. 15, pp. 344–367.
[8] R. Fernholz and I. Karatzas, *Stochastic portfolio theory: an overview*, in Special Volume: Mathematical Modeling and Numerical Methods in Finance, A. Bensoussan and Q. Zhang, eds., vol. 15 of Handbook of Numerical Analysis, Elsevier, 2009, pp. 89–167.
[9] R. Fernholz, I. Karatzas, and C. Kardaras, *Diversity and relative arbitrage in equity markets*, Finance and Stochastics, 9 (2005), pp. 1–27.

[10] H. Föllmer, *The exit measure of a supermartingale*, Z. Wahrscheinlichkeitstheorie und Verw. Gebiete, 21 (1972), pp. 154–166.
[11] ———, *On the representation of semimartingales*, Ann. Probability, 1 (1973), pp. 580–589.
[12] I. Karatzas and C. Kardaras, *The numéraire portfolio in semimartingale financial models*, Finance and Stochastics, 11 (2007), pp. 447–493.
[13] C. Kardaras and S. Robertson, *Robust maximization of asymptotic growth*, Ann. Appl. Probab., 22 (2012), pp. 1576–1610.
[14] ———, *Ergodic robust maximization of asymptotic growth*, Ann. Appl. Probab., 31 (2021), pp. 1787–1819.
[15] Y.-L. Kom Samo and A. Vervuurt, *Stochastic portfolio theory: A machine learning perspective*, in Proceedings of the 32nd Conference on Uncertainty in Artificial Intelligence (UAI), 2016. Available at SSRN: https://ssrn.com/abstract=2777631.

Email address: monoyios@maths.ox.ac.uk

Email address: pricilia@maths.ox.ac.uk

Mathematical Institute, University of Oxford, Radcliffe Observatory Quarter, Woodstock Road, Oxford OX2 6GG, UK