# APPLICATION OF A MULTICRITERIA REWARD FUNCTION TO DYNAMIC RISK MANAGEMENT IN MARKET-NEUTRAL PORTFOLIOS

B. E. Belyakov

HSE Tikhonov Moscow Institute of Electronics and Mathematics (HSE MIEM), Moscow, Russia
National Research University Higher School of Economics (HSE), Moscow, Russia

✉ work.belyakov@gmail.com

**Abstract.** In this paper, a multicriteria reward function is developed and empirically validated within a market-neutral portfolio management framework based on deep reinforcement learning. This function formalizes the optimal capital allocation problem as a Markov decision process in a nonstationary environment. The reward function simultaneously optimizes excess return per unit of risk, market neutrality via an explicit penalty on the correlation with a target market index, and transaction costs. As a result, systemic risk control and the cost of portfolio rebalancing are embedded into the modeling and training procedure. The model is trained and analyzed using cross-validation to mitigate the bias and adapt to regime shifts. According to the experimental validation results on leading equity indices of the USA, Europe, and Asia over a ten-year horizon, the above framework is effective, particularly showing consistent improvements in key metrics: higher Sharpe ratios, lower maximum drawdowns, and a substantially reduced correlation with the market index relative to classical optimization algorithms and reinforcement learning approaches with a single criterion.

**Keywords**: market-neutral strategy, portfolio optimization, reinforcement learning, market-neutral portfolio, deep learning, portfolio management.

## INTRODUCTION

Financial markets are characterized by stochastic returns, alongside persistent behavioral and structural anomalies such as momentum, excess volatility, and multifactor premiums [1–4]. Given these characteristics, market-neutral strategies have formed the core of modern quantitative trading, where the agent's objective is to extract excess returns with minimal exposure to systemic risk, i.e., to gain an investment advantage that cannot be explained by passive market factors [2, 5, 6]. Historically, this approach stems from the early hedge funds of Alfred Jones (the 1940s), which used combinations of long and short positions to hedge systemic risks [7]. The theoretical foundation was laid by factor models (Fama–French, Carhart) and classical relative-value strategies, including pair trading or synthetic contracts [1, 2, 8].

Despite their practical relevance, conventional implementations of such models rest on static factor structures with fixed exposures and are subject to several systemic vulnerabilities: factor degradation during shifts in market regimes [4, 9, 10], overfitting in high-dimensional feature spaces [9, 11, 12], and the absence of dynamic adaptation mechanisms for volatility clusters and macroeconomic shocks [4, 10, 13]. Recent developments in the field of factor premium timing and adaptive allocation demonstrate the benefits of dynamic predictors over static arbitrage strategies [11, 14]; however, the practical utility of such models diminishes when the problem formulation requires zero exposure to the benchmark under realistic market conditions and transaction costs [3, 5, 14].

Reinforcement learning naturally formulates portfolio management as a Markov decision process in which an agent learns a rebalancing policy optimizing the long-term reward function under transaction costs, constraints on working or borrowed capital, and the risk of drawdowns (decreases in portfolio value) [14–18]. The proposed problem statement allows updating representations when receiving new data [15, 17], exploiting the dynamic structure of returns and covariances [9, 11, 17], and implementing the closed "observation–action–feedback" loop, while eliminating the need for rigidly defined sets of factors [14, 15, 17].

For the practical realization and stable training of policies, it is useful to involve modern optimization algorithms and techniques for regularization and prevention of overfitting in high-dimensional feature spaces [12, 16, 19].

The goal of this work is to develop and empirically validate a methodology for dynamic market-neutral capital allocation based on deep reinforcement learning (Fig. 1). Portfolio management is formalized as a Markov process of sequential decisions in a non-stationary environment: an agent selects a portfolio weight vector under constraints on exposure (the allocated share of capital in a single asset and in the entire strategy) and turnover (changes in weights).

A key element of this methodology is a multicriteria reward function, specifically designed to consider the three interrelated criteria simultaneously:

1) maximizing excess return per unit of portfolio volatility;

2) minimizing the coupling with a target market index through a correlation penalty;

3) limiting trading volume through a penalty for transaction costs.

Thus, control of risk, market neutrality, and rebalancing costs is embedded directly into the agent's objective function, instead of being implemented as external post-processing or hard constraints. This has not previously been considered in similar problems.

The methodology's architecture includes a conveyor of alpha signals (predictors of excess return; indicators related to future asset price dynamics), a hybrid feature extraction module based on deep neural networks, and a recurrent actor-critic algorithm that maps the hidden state to a vector of target weights [16, 17]. Evaluation is performed using cross-validation with daily portfolio rebalancing, with consideration of slippage and commissions, including the cost of short positions [5, 10, 13].

Taken together, the above peculiarities distinguish this work from currently known approaches, which are primarily focused on maximizing the agent's return without explicit control over neutrality and trading volume [10, 13]. The approach proposed below is empirically validated on data of key regional indices (the USA, Europe, and Asia), consistently demonstrating an increase in the Sharpe ratio, a reduction in the portfolio's maximum drawdown (the decline in portfolio value relative to the last peak), and the maintenance of low or zero correlation with benchmarks.

The scientific novelty of this work lies in the integration of a multicriteria reward function, simultaneously incorporating risk-adjusted returns and penalties for the market correlation and transaction costs, into the agent's recurrent neural network for constructing market-neutral portfolios. The problem statement and reward function proposed below have not previously been systematically considered in the context of agent-based modeling and reinforcement learning for market-neutral strategies.

The following sections of the paper provide a literature review, a formalization of the decision process, a detailed description of the architecture and objective function, the experimental setup and metrics, the results of numerical experiments, and directions for further research.

## 1. LITERATURE REVIEW

### 1.1. Market-Neutral Portfolios

The origins of market-neutral strategies are traced back to A. Jones's early hedge funds, where returns were generated through stock selection while simultaneously hedging market risk via long and short positions [5, 7]. The theoretical framework is based on multifactor asset pricing models: the Fama–French three-factor model and its extension by Carhart to include the momentum factor [1, 2, 6]. Practical methods include regression factor neutralization and optimization formulations with zero exposure to systemic factors [9, 13]. Classical relative value strategies—pair trading, convertible bond arbitrage, and merger arbitrage—have demonstrated economically significant premiums; for pair trading in US stocks over 1962–2002, statistically robust excess returns of approximately 11% per annum were shown in [8, 20].

Advances in machine learning have expanded the toolkit for statistical arbitrage: ensemble models and deep neural networks are used to construct long-short portfolios and have demonstrated positive results in empirical pricing and return forecasting [10, 11, 21]. Deep learning and reinforcement learning methods are used for joint portfolio construction as well as for modeling and optimization of trading rules [10, 14, 15]. In regression models, returns are projected onto sectoral and market factors, followed by the neutralization of factor loadings (e.g., via the Fama–French factor regression) [1, 2, 9]. In optimization settings, the Markowitz criterion serves as the basis for formulations with strict constraints on exposures and risks, which is also applicable to small-cap portfolios when covariances are correctly estimated in high dimensions [6, 9]. The concept of neutrality is heterogeneous: different operational criteria (regression neutralization, direct constraints on exposures, or target optimization functions) lead to different practical outcomes of strategy implementation [2, 9, 13]. Alternative optimization formulations aimed at reducing the correlation

with a benchmark or controlling trading volume potentially provide stricter control of neutrality compared to regressions; however, a comparison of their effects depends on a particular problem statement and data and requires thorough empirical validation [13, 22]. Finally, despite significant progress in reinforcement learning methods for portfolio management and accounting for market constraints [10, 23–25], their application to market-neutral portfolios and integration into the reward function remain an underdeveloped topic in the literature.

### 1.2. Reinforcement Learning in Trading

Reinforcement learning contrasts with conventional econometric approaches due to its ability to learn online and utilize deep neural network architectures; therefore, it is particularly well-suited for the conditions of noise, nonstationarity, and heavy tails in financial time series distributions [10, 11, 14, 15, 26].

Comparative reviews indicate that key families of algorithms (policy gradient methods, the actor-critic algorithm) are commensurably effective in portfolio management problems, with the choice of a specific approach determined by the problem statement and structural constraints [15, 16, 26]. In practice, several studies have demonstrated the applicability of deep neural network algorithms and their recurrent modifications to partially observable financial problems, also highlighting the advantages of policy gradient methods and the actor-critic algorithm (including their deterministic and stochastic variants, such as proximal policy optimization), for optimizing portfolio objective functions [10, 16, 18, 19]. Note that the specification of a Markov decision process and the choice of a feature space remain critical factors. According to a number of research results, the inclusion of nonfinancial indicators (such as market sentiment indicators), alongside price characteristics, contributes to improving the predictive properties of models and leads to higher returns and lower risk characteristics in empirical experiments. At the same time, the degree of this effect depends significantly on the composition of features, the chosen regularization scheme, and the methods for estimating covariance structures in high-dimensional spaces [10, 12, 21, 26]. In market-making (maintaining the liquidity of a trading instrument by regularly placing buy and sell bids) and multiasset capital allocation, reinforcement learning scales effectively to large, including continuous, action spaces and reduces the dependence on manual parameterization through deep policies and deterministic training schemes [16–18, 19, 26]. Risk-adjusted reward functions allow formalizing the balance between return and risk [22, 27], and the consideration of transaction costs and slippage is a key condition for correct evaluation [5, 13, 20]. These arguments underlie the methodological framework developed below, which establishes a new approach to training agents in market-neutral strategies.

## 2. METHODOLOGY

### 2.1. Notation

Consider discrete time $t = 0, 1, ..., T$ and a set of assets consisting of $N$ financial instruments. Let

$$r_{t+1} = \left(r_{t+1}^1, ..., r_{t+1}^N\right)^{\mathrm{T}}$$

denote the vector of asset returns over the period $[t, t+1]$. Let $m_{t+1}$ denote the return of a selected market index (benchmark) over the same period.

At a time instant $t$, the portfolio is described by the vector of weights

$$w_t = \left(w_t^1, ..., w_t^N\right)^{\mathrm{T}},$$

where $w_t^i$ is interpreted as the position weight (the exposure of the $i$th instrument, expressed as a share of the capital allocated to the asset). For instance, $w_t^i > 0$ corresponds to a long position, $w_t^i < 0$ to a short position, and $w_t^i = 0$ to no position. Unlike the classical fully invested portfolio with the budget constraint $\sum_{i=1}^{N} w_t^i = 1$, in the market-neutral problem statement, the portfolio size is fixed by a constraint on gross exposure; see formula (2).

The requirement for the portfolio's market-neutral net exposure is formalized by the condition

$$\sum_{i=1}^{N} w_t^i = 0, \tag{1}$$

and the constraint on the portfolio's gross exposure is described by the inequality

$$\sum_{i=1}^{N} \left|w_t^i\right| \le \Gamma, \tag{2}$$

where $\Gamma > 0$ is a given threshold for the maximum aggregate position.

The constraint (2) sets an upper bound on the portfolio's gross exposure, i.e., limits borrowed capital. In strategies, the threshold $\Gamma$ reflects the maximum allowable gross exposure (the sum of the absolute values of long and short positions) and limits borrowed

capital. In this paper, the threshold $\Gamma$ is fixed at 100% (a conservative limit on the aggregate position that ensures zero net exposure).

Together with condition (1), this means that the total long exposure equals the total short exposure (in absolute terms), and each does not exceed $\Gamma/2$.

The change in weights during portfolio rebalancing at a time instant $t$ is

$$\Delta w_t = w_t - w_{t-1}.$$

The transaction costs $TC_t$ are defined as a linear-quadratic function of the change in the portfolio:

$$TC_t = c_{\text{lin}} \sum_{i=1}^{N} \left| \Delta w_t^i \right| + c_{\text{quad}} \sum_{i=1}^{N} \left( \Delta w_t^i \right)^2, \qquad (3)$$

where $c_{\text{lin}}, c_{\text{quad}} > 0$ are the commission and slippage parameters, respectively.

In view of the costs over the period $[t, t+1]$, the return on the portfolio $p$ (the capital return per unit of allocated capital) is defined as

$$r_{p,t+1} = \sum_{i=1}^{N} w_t^i r_{t+1}^i - TC_t. \qquad (4)$$

Formula (4) is written for the specified position weights.

With $q_t^i$ and $C_t$ designating the monetary size of the position and the strategy's capital, we have $w_t^i = \dfrac{q_t^i}{C_t}$ and $\dfrac{\sum_{i=1}^{N} w_t^i r_{t+1}^i}{C_t} = \sum_{i=1}^{N} w_t^i r_{t+1}^i$.

Thus, $r_{p,t+1}$ is the return on the strategy's capital, whereas the net nominal value $\sum_{i=1}^{N} q_t^i$ in a market-neutral portfolio is zero.

The transaction costs $TC_t$ in (3) are also specified as a share of capital (via $\Delta w_t$) and are therefore subtracted directly in formula (4).

The current market coupling of the portfolio is estimated via the correlation coefficient between the portfolio's return and the benchmark's return over a sliding window of length $L_{\text{corr}}$:

$$\rho_t = \text{Corr}\left( r_{p,t+1}, m_{t+1} \right), \qquad (5)$$

based on data over the interval $[t - L_{\text{corr}} + 1, t+1]$.

At a time instant $t$, the vector of observed environmental features is denoted by

$$x_t \in R^d,$$

where $d$ is the number of indicators; this vector includes price, macroeconomic, and alternative variables.

### 2.2. Problem Statement in Terms of Reinforcement Learning

Reinforcement learning is an optimal control paradigm in which an agent interacts with an environment over time, selecting actions and receiving quality estimates (rewards) [15]. The agent's objective is to maximize the expected value of the total discounted reward.

The environment is described by a Markov decision process defined as a five-tuple

$$M = (S, A, P, R, \gamma),$$

where $S$ denotes the set of states; $A$ is the set of actions; $P(s \mid s, a)$ is a transition probability function; $R(s, a)$ is the reward; finally, $\gamma \in (0, 1)$ is the discount factor.

The agent's policy $\pi(a \mid s)$ specifies a conditional distribution of actions $a$ given a state $s$. In a financial market setting, the agent observes not the full state but only a finite set of features $x_t$. To take partial observability into account, a recurrent hidden state $h_t$ is introduced, which accumulates information about past observations.

This paper considers an approximate formulation of the problem where the state is defined by the vector

$$s_t = (x_t, w_{t-1}, h_{t-1}) \in S.$$

The action is the vector of target weights

$$\hat{w}_t \in R^N,$$

generated by the agent's policy [10, 17, 18, 26].

The actual portfolio weights are obtained by projection onto the set of feasible portfolios:

$$w_t = \Pi(\hat{w}_t),$$

where the operator $\Pi(\cdot)$ realizes the constraints (1) and (2); the environment's transition $s_t \to s_{t+1}$ is determined by realizing the market returns $r_{t+1}$, updating the weights $w_t$, recalculating the features $x_{t+1}$, and updating the hidden state $h_t$; at a time instant $t$, the reward $R_t$ is given by the author's multicriteria reward function

$$R_t = r_{p,t+1} - \lambda_1 \rho_t^2 - \lambda_2 TC_t,$$

where the first term corresponds to the portfolio return after deducting the costs (4), the second to the penalty for the correlation with the benchmark (5), and the third to the penalty for the transaction costs, which depend on the trading volume (3); finally, the coefficients $\lambda_1 > 0$ and $\lambda_2 > 0$ specify the relative weights of market neutrality requirements and cost level requirements, respectively.

In the experimental setting, the penalty coefficients were fixed at $\lambda_1 = 10$ (the penalty for the correlation with the benchmark) and $\lambda_2 = 5$ (the penalty for the transaction costs).

The above values were used in all main experiments and were chosen based on cross-validation results. This choice of the coefficients ensures a balance between reducing the correlation with the benchmark and controlling the transaction costs during portfolio rebalancing.

The agent's objective is to find a policy $\pi$ that maximizes the discounted sum of rewards:

$$J(\pi) = E_\pi\left[\sum_{t=0}^{T-1}\gamma^t R_t\right].$$

In the author's methodology, the policy $\pi_\theta(a|s)$ is parameterized by a deep neural network. The parameter $\theta$ is updated using the Proximal Policy Optimization (PPO) algorithm; the recurrent hidden state $h_t$ is implemented based on the Gated Recurrent Unit (GRU) to approximate the hidden state of a partially observable environment.

### 2.3. Solution Architecture

The architecture proposed in this paper uses a hierarchical feature extractor based on *convolutional neural networks* (CNNs) and GRUs, with a recurrent actor-critic algorithm implemented above this extractor (see Fig. 1). This architecture is intended to construct a spatiotemporal representation of the agent's environment state for portfolio optimization, with the simultaneous consideration of the differences between assets and dynamics over time.

Let a feature tensor of the form

$$X_t \in R^{N\times L\times F}$$

be supplied to the input at a time instant $t$; here, $N$, $L$, and $F$ are the numbers of assets, time lags, and features characterizing the asset, respectively. This tensor is interpreted as a cross-sectional feature map: assets are listed along one axis, lags along the second, and feature descriptions along the third.

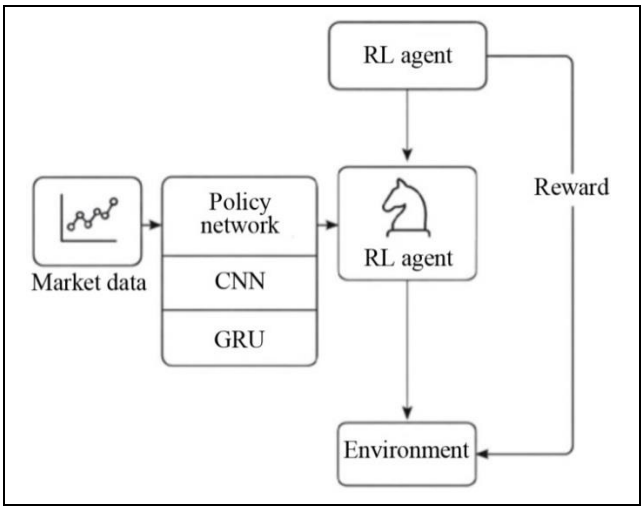


**Fig. 1. The high-level diagram of the system with the integrated reward function for agent control.**

The convolutional layers applied to the tensor $X_t$ perform local filtering along the asset–lag axis to identify spatial and short-term temporal dependencies [10, 12, 26].

This block outputs the feature tensor

$$Z_t = CNN(X_t) \in R^{N\times F},$$

which is further aggregated into the vector $z_t \in R^{d_z}$ of fixed dimension $d_z$ (the dimension of the aggregated feature vector after the convolutional block).

The vector sequence $\{z_t\}$ is supplied to the GRU to implement the evolution of the hidden state:

$$h_t = GRU(z_t, h_{t-1}),\ h_t \in R^{d_h},$$

where $d_h$ is the dimension of the hidden state of the GRU.

The hidden state $h_t$ accumulates information about past feature dynamics to approximate the state in a partially observable environment [17, 18, 26]. Two outputs are constructed above the hidden state:

- the actor $\pi_\theta$, which maps the state $h_t$ (together with positional information, including $w_{t-1}$) to the vector of action parameters $\hat{w}_t$, being then projected onto the feasible set of portfolio weights;
- the critic $V_\phi$, which maps the state $h_t$ to a scalar value estimate $V_\phi(h_t)$, being used to calculate the rewards and stabilize the training process [12, 13, 16, 19, 22].

Thus, in the resulting actor-critic architecture, the hidden state $h_t$ serves as a common representation for the policy and the value function.

Therefore, the combination of neural networks allows simultaneously capturing local spatial patterns and long-term temporal dynamics and forming an extended state representation; based on the latter, the recurrent actor-critic algorithm optimizes the investment decision strategy [10, 17, 18].

Figure 2 shows the diagram of the solution architecture, including the main blocks of neural networks and their integration into the algorithm.

## 3. EXPERIMENTAL VALIDATION OF THE METHODOLOGY

### 3.1. Data and Methods

A panel market dataset with the components of stock indices of the USA, Europe, and Asia was used to validate the methodology. The sample included the following indices: S&P 500 (GSPC), NASDAQ-100 (NDX), DJIA (DJI), FTSE 100 (FTSE), DAX (GDAXI), Hang Seng (HSI), and SSE Composite (SSEC). The data were downloaded from the Bloomberg terminal and cover the period from January 2010 to December 2020, with an observation frequency of 1 day. For each index, the full collection of components in effect at the start of the corresponding training window was used; changes in the collection were automatically accounted for when forming the panel dataset. For each asset, a feature vector was constructed from the following groups of variables: price characteristics (logarithmic returns over 1-, 5-, 10-, and 20-day horizons); moving averages of the prices; realized volatility; liquidity and volume indicators: normalized trading volume, spread, and turnover; risk measures (the coefficients of variation of the returns and market volatility indicators); macroeconomic and alternative features (interest rates, inflation indicators, business activity indices, and sentiment indicators), i.e., standard indicators from the Bloomberg terminal.

For each time instant $t$, the feature tensor

$$X_t \in R^{N_t \times L \times F}$$

was formed, where $N_t$ is the number of assets, and $L$ is the number of time lags.

Preprocessing was performed separately on each training window using the following procedures: calendar synchronization and shift of the last observation in case of missing data; winsorization by the 1st and 99th percentiles; standardization of all features across the training segment; creation of time lags for the indicators; formation of positional features (portfolio weights at the previous step, as well as aggregated risk and liquidity indicators). This procedure prevents information leakage and ensures the correct data division into time segments.

The model's generalization ability was assessed using a stepwise optimization approach. Each cycle consisted of a 750-day training window, followed by a 250-day validation window, and then a 250-day test window. Upon cycle completion, the window was shifted by 250 days, and the procedure was repeated. The model was retrained in each cycle. During the test period, the portfolio was rebalanced daily, considering the transaction costs and the cost of holding short positions. The performance metrics were calculated strictly on the test data and averaged across all test windows. The state of the environment included the observation tensor, the portfolio weight vector, and additional aggregated parameters (liquidity, volatility, and the hidden state of the GRU).

The pseudocode of the algorithm is provided in the Appendix.

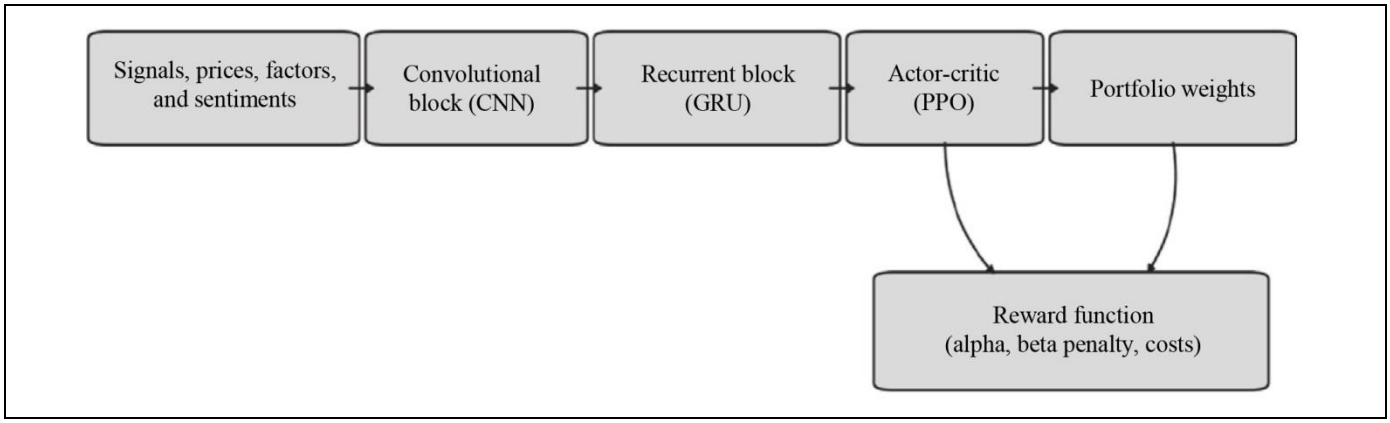


**Fig. 2. The closed loop of agent's training and decision-making in the proposed methodology.**

### 3.2. Comparative Models and Metrics

The model was compared with several reference approaches for an in-depth analysis of the methodology under consideration:

- Mean–Variance, Markowitz's classical portfolio optimization, where expected returns and the covariance matrix are estimated on each training window, and then a portfolio is constructed by minimizing the variance under a given target return; for validation, this approach is applied with realistic transaction costs and daily rebalancing [5, 6, 9, 20].
- FF3, the factor strategy based on the Fama–French model, implemented through the replication of factor portfolios; it serves as a benchmark for measuring the portfolio's systematic factor exposure [1, 2].
- Pairs Trading, the pair strategy based on selecting cointegrated liquid pairs from the index; it is triggered and utilized for the spread's deviations from an equilibrium value. Such a strategy provides a local market-neutral position but remains sensitive to liquidity and the parameters for selecting trading pairs [5, 8, 20].
- RL-Baseline, a primitive reinforcement learning agent that uses the same input features and validation scheme as the advanced agent; it serves as a baseline for assessing the advantages of the advanced agent's deep, complex planning methods [10, 17, 21, 26].
- AlphaZeroBeta, the proposed model combining the recurrent neural network architecture with the multi-criteria reward function designed to simultaneously increase risk-adjusted returns, reduce correlation with the market index, and limit turnover.

Three metrics were used:

- *the Sharpe ratio*, which characterizes risk-adjusted return, i.e., the ratio of the portfolio's average excess return to the volatility of that return; higher values indicate a more efficient combination of return and risk [28]:

$$\text{Sharpe} = \frac{E\left[r_{p,t+1} - r_f\right]}{\sqrt{\text{Var}\left(r_{p,t+1}\right)}},$$

where $r_f$ is the risk-free rate (the rate converted to a daily basis from an annual rate);

- *Max Drawdown*, which reflects the worst relative decline in the portfolio's accumulated value from the previous peak to the local minimum over the period under consideration and indicates the strategy's vulnerability to large drawdowns [13, 22]:

$$MDD = \max_{0<t<s<T} \frac{W_t - W_s}{W_t},$$

where $W_t$ is the cumulative value of the portfolio at a time instant *t*, i.e., its value before the decline began; $W_s$ is the cumulative value of the portfolio at a later time instant *s* (after the decline).

- Correlation with the market index, characterizes the linear relationship between the returns of the portfolio and the benchmark, serving as a proximal estimate for the exposure to market beta (a positive correlation means that the portfolio value moves in tandem with the market) [2, 6]:

$$\text{Corr} = \frac{\text{Cov}\left(r_{p,t+1},\, m_{t+1}\right)}{\sqrt{\text{Var}\left(r_{p,t+1}\right)\text{Var}\left(m_{t+1}\right)}}.$$

### 3.3. Model, Architecture, and Data Parameters

The following fixed parameters were adopted in the experiments:

- For step-by-step optimization, windows of $T_{\text{train}} = 750$ days (nearly 3 years) were used for training, $T_{\text{train}} = 250$ days (nearly 1 year) for hyperparameter tuning and early stopping, and $T_{\text{test}} = 250$ days for metric evaluation, followed by a shift of $T_{\text{test}}$.
- The feature time lag was set to $L = 20$ days.
- The CNN architecture included two convolutional layers with 32 and 64 filters and a $3\times3$ kernel, followed by aggregation and a fully connected layer of dimension 128. The GRU had a hidden state of dimension $d_h = 128$.
- The actor was represented by a two-layer fully connected network $128 \to 64 \to N$, and the critic by a network $128 \to 64 \to 1$.
- Training was performed using the PPO algorithm with the following parameters: the discount factor $\gamma = 0.99$, the generalized advantage estimation coefficient $\lambda_{\text{GAE}} = 0.95$, and the clipping coefficient $\varepsilon_{\text{PPO}} = 0.20$.
- The critic regularization coefficient was set to $c_{\text{v}} = 1.0$.
- The entropy regularization coefficient was set to $c_{\text{ent}} = 0.01$.
- The window length was 64 steps, the mini-batch size was 256, with 10 PPO optimization epochs per training step.
- The penalty coefficients of the reward function were set to $\lambda_1 = 10$ (the correlation with the benchmark) and $\lambda_2 = 5$ (the transaction costs).

• The values of the linear and quadratic cost components were set to $c_{\text{lin}} = 1 \cdot 10^{-4}$ and $c_{\text{quad}} = 1 \cdot 10^{-6}$, respectively.

• The portfolio limit parameters were $\Gamma = 1.0$ (the maximum gross exposure) and a weight change limit of 0.2 per unit of invested capital.

The Bloomberg terminal was used as the data source; for the components of the GSPC, NDX, DJI, FTSE, GDAXI, HSI, and SHCOMP indices, the data series PX_LAST, PX_OPEN, PX_HIGH, and PX_LOW (prices) and VOLUME (volume) were applied, in combination with the liquidity indicators (BID, ASK, and BID_ASK_SPREAD, where available). The market and macroeconomic factors were constructed using the following data: the volatility indicators VIX, V2X, and HSIV; the interest rates USGG2YR, USGG5YR, USGG10YR, and US0003M; the inflation and macroeconomic series CPI YOY, PPI YOY, GDP CQOQ, UNRATE, and IP CHNG; the business activity indicators NAPMPMI, NAPMNMI, ECOPMIM, and CHPMMAN; the sentiment and risk appetite proximal estimates NHBUS, CPSENT, BAMLELL, and NEWS_SENTIMENT; the option metrics SDEX, SPX SKEW, GVZ, and OVX; finally, the global currency and commodity indicators DXY, EURUSD, JPYUSD, CL1, and GC1. All daily data for the period 2010–2020 were downloaded, calendar-synchronized, and normalized based on training window statistics.

### 3.4. Results and Training Diagnosis

The results of the performance analysis (Tables 1–3) demonstrate that the AlphaZeroBeta approach under consideration leads confidently in terms of the Sharpe ratio, maintains a nearly zero correlation with the market index, and significantly reduces the portfolio's maximum drawdown during market shock periods. This combination of high risk-adjusted returns and low systematic dependence indicates the investment advantage of the approach, as well as effective risk management mechanisms [5, 10, 13, 20, 22, 28].

*Table 1*

**Sharpe ratio values for different strategies**

| Strategy | USA | Europe | Asia |
|---|---|---|---|
| Mean–Variance | 0.90 | 0.80 | 0.70 |
| FFE (Fama–French) | 0.95 | 0.85 | 0.75 |
| Pairs Trading | 0.80 | 0.70 | 0.65 |
| RL-Baseline | 1.10 | 0.90 | 0.80 |
| AlphaZeroBeta (the proposed approach) | 1.35 | 1.20 | 1.05 |

*Table 2*

**The correlation of strategies with market indices**

| Strategy | USA | Europe | Asia |
|---|---|---|---|
| Mean–Variance | 0.55 | 0.60 | 0.50 |
| FFE (Fama–French) | 0.45 | 0.50 | 0.40 |
| Pairs Trading | 0.35 | 0.30 | 0.25 |
| RL-Baseline | 0.20 | 0.15 | 0.10 |
| AlphaZeroBeta (the proposed approach) | 0.02 | 0.01 | 0.00 |

*Table 3*

**The maximum drawdown of strategies, %**

| Strategy | US | Europe | Asia |
|---|---|---|---|
| Mean–Variance | –35 | –32 | –30 |
| FFE (Fama–French) | –28 | –26 | –24 |
| Pairs Trading | –25 | –22 | –20 |
| RL-Baseline | –18 | –17 | –15 |
| AlphaZeroBeta (the proposed approach) | –12 | –11 | –10 |

According to the experimental results, the agent adaptively optimizes capital allocation while maintaining efficiency during crises and volatility periods. Incorporating a penalty for the correlation with the benchmark in the reward function maintains near-zero market exposure without strict constraints, which reduces sensitivity to macroeconomic shocks [1, 2, 13]. Accounting for the turnover and transaction costs directly during the training stage enhances practical feasibility and reduces the risk of systematic overfitting in testing [5, 13, 14].

Overall, the algorithm demonstrates improved risk-adjusted metric values and lower maximum portfolio drawdowns compared to the alternative approaches. Key limitations are associated with the calibration of objective function weights, data quality, and the cost model. The method stands out for its targeted optimization of neutrality through the reward component (transforming "zero beta" from a side effect into a target characteristic of the strategy); the hybrid neural network extractor combined with a recurrent policy enhances stability to regime shifts and partial observability, while incorporating the turnover brings the model's performance closer to real trading conditions [5, 16–18, 20, 22, 28].

All computations are performed on V100/A100-class GPU infrastructure; a single run takes several hours to train, and a full series of 20–30 sliding windows per index needs 5–10 days on a single GPU (or 1–3 days with parallelized runs on 4–8 GPUs).

The Sharpe ratios of the five strategies (Mean–Variance, FF3, Pairs Trading, RL-Baseline, and Al-

phaZeroBeta) for the three regions (the USA, Europe, and Asia) are compared in Fig. 3; an *out-of-sample* dataset was used to calculate the Sharpe ratio (standard practice for evaluating algorithm performance). Clearly, the proposed approach demonstrates the highest Sharpe ratios for all regions, excelling both the classical portfolio models and the portfolios of single-factor RL agents [5, 10, 20, 26, 28]. Figure 4 shows the dynamics of the correlation (market beta) of the strategies with the corresponding benchmarks: the approach maintains a near-zero or negative correlation due to the target penalty for the market correlation in the reward function, whereas the reference approaches (including factor and pair strategies) retain a noticeable positive correlation with the index [1, 2, 6, 8, 13].

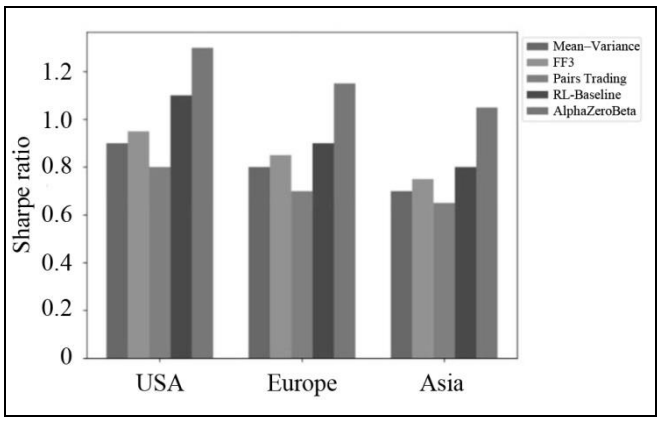


**Fig. 3. The average values of the Sharpe ratio on the out-of-sample dataset for different strategies by region.**

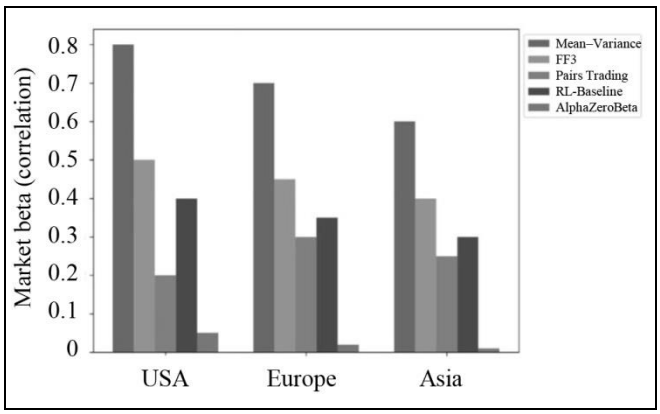


**Fig. 4. The average Pearson correlation coefficients on the OOS dataset with benchmarks.**

Feature normalization and metric calculation were performed strictly on the test segments, excluding any information from the training windows to avoid data leakage [9, 11]. The transaction costs were modeled and subtracted when forming the portfolio returns, which brings the results closer to realistic execution [5, 13, 14].

## 4. AREAS FOR FURTHER RESEARCH

The following priority areas of further research can be outlined:

- Enhancing the interpretability of the model—applying machine learning model analysis methods to compare the contribution of strategy signals with classical factors (FF3, Carhart) directly and demonstrate the economic interpretability and attribution of the identified investment advantage; such tools help distinguish the "black box" from a reproducible economic effect and link the investment advantage to known factors from the fundamental economic literature.
- Adapting the algorithm to operate across multiple trading universes, considering margin requirements and clearing constraints; it will require integrating liquidity models into the agent's trading environment and conducting stress tests under various margin shocks.
- Developing and formalizing online learning and stabilization mechanisms, such as limiting divergence when updating the agent's policy.
- Enhancing the model's stability through simulation of liquidity shocks, feature distortion, and the application of risk-sensitive metrics (more advanced risk metrics) during optimization and reward evaluation to control portfolio exposure in stress scenarios.
- Validating systematically the architectural components, analyzing the model's sensitivity with respect to the parameters of the reward function and slippage; documenting and taking snapshots of the data and metrics for reproducibility.
- Combining the listed techniques with infrastructural requirements, such as execution delays and realistic asset pricing models, to ensure the reproducibility of backtesting results on historical data under real conditions.

## CONCLUSIONS

This paper has presented a methodology for constructing market-neutral strategies that formalizes capital allocation as a sequential decision process, taking into account transaction costs and direct control of correlation with a market index. The inclusion of a realistic model of costs and operational constraints reduces the risk of systematic overestimation of results and enhances the practical utility of the approach by considering costs during the training stage. A multicriteria reward function balancing returns, penalties for market exposure, and trading volume has been combined with a recurrent actor-critic architecture. This combination ensures adaptability to non-stationarity and stability to market regime shifts, as

well as reduces the variance of gradient estimates. Static factor methods are simpler from the standpoint of implementation and interpretability, but they are inferior to the proposed approach in terms of adaptability and consideration of operational constraints. For further risk control, it is important to account for the share of risk in the objective function, which will enhance the economic interpretability and stability of results in stress scenarios.

**Acknowledgments.** *The author is grateful to HSE colleagues for their valuable discussions, constructive feedback, and continued support of this work.*

## APPENDIX

### Pseudocode of the algorithm

Algorithm 1. Training and testing of a market-neutral strategy
Input data: *D*—market data panel, *M*—benchmark returns, *U*—the set of assets, and *H*—the hyperparameters of the model, training, and portfolio constraints.
Output: the best trained policy for each window and out-of-sample metrics across all windows

```
S ← Construct a sequence of sliding windows (training, validation, and test)
For each window s ∈ S:
    (X_train, R_train, M_train, stats) ← Preprocessing of (D_train, M_train, only on the training window)
    (X_val,   R_val,   M_val)          ← Preprocessing of (D_val, M_val, stats)
    (X_test,  R_test,  M_test)         ← Preprocessing of (D_test, M_test, stats)
Initialize the encoder, actor, and critic parameters
best_score ← −∞
Until the early stopping condition is satisfied:
    Clear the path buffer
    h ← 0; w_prev ← 0
    Initialize the state of the online correlation estimator
    For each time instant t in the training window:
        h_t ← Encoder(X_train[t], h_{t−1})
        a_t ← Actor(h_t)
        w*_t ← Transform action into target weights
        w_t  ← Project onto the feasible set
        w_pre ← Update the weights after market movement
        Δw_t  ← Limit the change in weights by turnover
        TC_t  ← Calculate the transaction costs (Δw_t)
        w_t   ← Execute the transaction
        p_t^gross ← Portfolio return for the next step before deducting costs
        ρ_t ← Update the online correlation estimate (p_t^gross, M_train[t+1])
        r_t ← p_t^gross − λ_corr · ρ_t^2 − λ_tc · TC_t
        V_t ← Critic(h_t)
        Save (h_t, a_t, w_t, p_t^gross, ρ_t, TC_t, r_t, V_t) to the buffer
        w_prev ← w_t
    Estimate the benefits and target values of the critic based on the buffer
    Update the model parameters using PPO on truncated sequences
    score_val ← Test the policy on the validation window using the selected criterion
    If score_val is better than best_score:
        Save the current parameters as the best
        best_score ← score_val
    Load the best model for this window
    test_metrics ← Test the policy on the test window
    Save test_metrics
Aggregate metrics across all windows
Return the aggregated results and the results by windows
```

## REFERENCES


1. Carhart, M.M., On Persistence in Mutual Fund Performance, *The Journal of Finance*, 1997, vol. 52, no. 1, pp. 57–82.
2. Fama, E.F. and French, K.R., Common Risk Factors in the Returns on Stocks and Bonds, *Journal of Financial Economics*, 1993, vol. 33, no. 1, pp. 3–56.
3. Goyal, A. and Welch, I., A Comprehensive Look at the Empirical Performance of Equity Premium Prediction, *The Review of Financial Studies*, 2008, vol. 21, no. 4, pp. 1455–1508.
4. Lettau, M. and Ludvigson, S., Consumption, Aggregate Wealth, and Expected Stock Returns, *The Journal of Finance*, 2001, vol. 56, no. 3, pp. 815–849.
5. Pedersen, L.H., *Efficiently Inefficient: How Smart Money Invests and Market Prices Are Determined*, Princeton: Princeton University Press, 2015.
6. Markowitz, H., Portfolio Selection, *The Journal of Finance*, 1952, vol. 7, no. 1, pp. 77–91.
7. Loomis, C.J., The Jones Nobody Keeps up with, *Fortune Magazine*, April 1, 1966, pp. 237–247.
8. Gatev, E., Goetzmann, W.N., and Rouwenhorst, K.G., Pairs Trading: Performance of a Relative-Value Arbitrage Rule, *The Review of Financial Studies*, 2006, vol. 19, no. 3, pp. 797–827.
9. Choi, I., Efficient Estimation of Factor Models, *Econometric Theory*, 2012, vol. 28, no. 2, pp. 274–308.
10. Jiang, R., Xu, D., and Liang, Z., A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem, *arXiv:1706.10059*, 2017. DOI: https://doi.org/10.48550/arXiv.1706.10059
11. Gu, S., Kelly, B., and Xiu, D., Empirical Asset Pricing via Machine Learning, *The Review of Financial Studies*, 2020, vol. 33, no. 5, pp. 2223–2273.
12. Srivastava, N., Hinton, G., Krizhevsky, A., et al., Dropout: A Simple Way to Prevent Neural Networks from Overfitting, *Journal of Machine Learning Research*, 2014, vol. 15, no. 56, pp. 1929–1958.
13. Buehler, H., Gonon, L., Teichmann, J., and Wood, B., Deep Hedging, *Quantitative Finance*, 2019, vol. 19, no. 8, pp. 1271–1291.
14. Moody, J. and Saffell, M., Learning to Trade via Direct Reinforcement, *IEEE Transactions on Neural Networks*, 2001, vol. 12, no. 4, pp. 875–889.
15. Sutton, R.S. and Barto, A.G., *Reinforcement Learning: An Introduction*, 2nd ed., Cambridge, MA: MIT Press, 2018.
16. Schulman, J., Wolski, F., Dhariwal, P., et al., Proximal Policy Optimization Algorithms, *arXiv:1707.06347*, 2017. DOI: https://doi.org/10.48550/arXiv.1707.06347
17. Heess, N., Wayne, G., Silver, D., et al., Memory-Based Control with Recurrent Neural Networks, *arXiv:1512.04455*, 2015. DOI: https://doi.org/10.48550/arXiv.1512.04455
18. Hausknecht, M. and Stone, P., Deep Recurrent Q-Learning for Partially Observable MDPs, *arXiv:1507.06527*, 2015. DOI: https://doi.org/10.48550/arXiv.1507.06527
19. Silver, D., Lever, G., Heess, N., et al., Deterministic Policy Gradient Algorithms, *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, Beijing, 2014, vol. 32, no. 1, pp. 387–395.
20. Bali, T.G., Brown, S.J., and Caglayan, M.O., Macroeconomic Risk and Hedge Fund Returns, *Journal of Financial Economics*, 2014, vol. 114, no. 1, pp. 1–19.
21. Kogan, L. and Papanikolaou, D., Firm Characteristics and Stock Returns: The Role of Investment-Specific Shocks, *The Review of Financial Studies*, 2013, vol. 26, no. 11, pp. 2718–2759.
22. Rockafellar, R.T. and Uryasev, S., Optimization of Conditional Value-at-Risk, *Journal of Risk*, 2000, vol. 2, no. 3, pp. 21–41.
23. Liu, X., Xiong, Z., Zhong, S., et al., Practical Deep Reinforcement Learning Approach for Stock Trading, *arXiv:1811.07522*, 2018. DOI: https://doi.org/10.48550/arXiv.1811.07522
24. Liu, X., Yang, H., Gao, J., and Wang, C.D., FinRL: Deep Reinforcement Learning Framework to Automate Trading in Quantitative Finance, *arXiv:2111.09395*, 2021. DOI: https://doi.org/10.48550/arXiv.2111.09395
25. Gu, F., Jiang, Z., García-Fernández, Á.F., et al., MTS: A Deep Reinforcement Learning Portfolio Management Framework with Time-Awareness and Short-Selling, *arXiv:2503.04143*, 2025. DOI: https://doi.org/10.48550/arXiv.2503.04143
26. Arulkumaran, K., Deisenroth, M.P., Brundage, M., and Bharath, A.A., Deep Reinforcement Learning: A Brief Survey, *IEEE Signal Processing Magazine*, 2017, vol. 34, no. 6, pp. 26–38.
27. Choudhary, H., Orra, A., Sahoo, K., and Thakur, M., Risk-Adjusted Deep Reinforcement Learning for Portfolio Optimization: A Multi-reward Approach, *International Journal of Computational Intelligence Systems*, 2025, vol. 18, no. 1, pp. 1–19.
28. Sharpe, W.F., Mutual Fund Performance, *The Journal of Business*, 1966, vol. 39, no. 1, pp. 119–138.


*This paper was recommended for publication by V.V. Klochkov, a member of the Editorial Board.*

*Received October 3, 2025, and revised March 11, 2026. Accepted April 8, 2026.*

**Author information**

**Belyakov, Boris Evgen'evich.** Postgraduate, HSE Tikhonov Moscow Institute of Electronics and Mathematics (HSE MIEM), Moscow, Russia; visiting lecturer, National Research University Higher School of Economics (HSE), Moscow, Russia
✉ work.belyakov@gmail.com
ORCID iD: https://orcid.org/0009-0007-8237-9833

**Cite this paper**

Belyakov, B.E., Application of a Multi-Objective Reward Function to Dynamic Risk Management in Market-Neutral Portfolios. *Control Sciences* **3**, 42–52 (2026).

Original Russian Text © Belyakov, B.E., 2026, published in *Problemy Upravleniya*, 2026, no. 3, pp. 51–63.



Translated into English by *Alexander Yu. Mazurov*,
Cand. Sci. (Phys.–Math.),
Trapeznikov Institute of Control Sciences,
Russian Academy of Sciences, Moscow, Russia
✉ alexander.mazurov08@gmail.com