Deep Stochastic Control for Optimal Trade Execution under Renewable-Induced Uncertainty in Intraday Electricity Markets

Alexander Laloi Dybdahl
NTNU
alexanld@ntnu.no

Christopher Laloi Dybdahl
NTNU
christdy@alumni.ntnu.no

Jacob Kooter Laading
NTNU
jacob.laading@ntnu.no

Syver Verlo Nes
NTNU
syver_99@icloud.com

Ståle Størdal
University of Inland Norway
stale.stordal@inn.no

Sjur Westgaard
NTNU
sjur.westgaard@ntnu.no

Abstract

We consider an Almgren–Chriss execution problem in continuous intraday electricity markets with renewable forecast uncertainty, price impact, and bid–ask spreads. Including spreads yields a Hamilton–Jacobi–Bellman equation that does not admit a closed-form solution except under restrictive assumptions. We address this problem using a deep backward stochastic differential equation (BSDE) method that approximates the value function through a forward–backward stochastic differential equation and derives a deterministic feedback policy via Hamiltonian minimization. The model is trained on simulated trajectories using losses that enforce BSDE consistency, the control objective, and physics-informed residuals. In numerical experiments, the learned policy matches the analytical solution in the zero-spread case and exhibits threshold-based trading behavior when spreads are present. When calibrated to the German continuous intraday market, the deep BSDE policy outperforms a time-weighted average price strategy and the zero-spread analytical benchmark across spread levels, although under high imbalance penalties the analytical benchmark achieves a lower mean cost.

Stochastic control, Deep learning, Optimal execution, Electricity market

1 Introduction

The accelerating transition toward renewable energy has introduced new complexities into electricity markets. As intermittent wind and solar generation become increasingly important sources of supply, market participants must manage the inherent uncertainty associated with weather dependent production. This unpredictability has made forecast errors a central operational challenge and has highlighted the need for trading mechanisms that complement the traditional day ahead market.

Electricity differs from other commodities because it cannot be stored economically, requiring supply and demand to be balanced continuously. Liberalized electricity systems therefore organize trading through a sequence of markets operating over decreasing time horizons, including the day ahead auction, intraday auctions, continuous intraday trading, and final balancing operated by transmission system operators. Within this structure, the continuous intraday market has emerged as a critical platform for participants to adjust their positions as forecasts are refined closer to delivery. As the share of renewable generation increases, forecast uncertainty is amplified, raising the risk of costly imbalances and increasing reliance on intraday trading to maintain system reliability (Katarzyna et al., 2019). Trading volumes on exchanges such as EPEX SPOT have increased by more than 200 percent since 2019, underscoring the growing operational and economic importance of intraday markets (EPEX SPOT, 2021, 2025; Kath Ziel, 2020).

Unlike auction based markets with uniform clearing prices, the continuous intraday market operates through a pay as bid limit order book. Continuous limit order book systems now dominate most financial and commodity markets (Angel et al., 2011; Harris, 2003; Nord Pool, 2025; Intercontinental Exchange, 2025), and similar mechanisms underpin intraday electricity trading. The bid-ask spread reflects prevailing liquidity conditions and constitutes a key transaction cost (Fleming, 2003; Harris, 2003). Large trades may consume liquidity across successive price levels, inducing adverse price movements commonly referred to as market impact (Bouchaud, 2010; Ozenbas et al., 2022; Frey Sandås, 2017; Kwok Lau, 2005). As a result, traders in the intraday market must manage both price risk and execution costs. In addition, regardless of whether a position is carried over from the day ahead market or initiated intraday, traders must continually respond to updated forecasts of demand and generation, which introduce significant uncertainty regarding the final net volume to be traded.

Despite extensive work on optimal execution following the framework of Almgren Chriss (2001), existing solution approaches remain limited in their ability to handle realistic intraday market features. Analytical models typically rely on strong assumptions on state dynamics in order to obtain explicit solutions, as in Aïd et al. (2016), which in turn prevents the inclusion of nonsmooth execution costs such as bid-ask spreads. Numerical approaches such as Glas et al. (2020) relax some of these assumptions and allow for richer dynamics, including the joint optimization of trading and generation decisions, but they rely on discretization based methods whose complexity grows rapidly with the dimension of the state space, effectively constraining practical implementations to a limited number of state variables.

The objective of this study[1] is to develop a solution method that can accommodate bid-ask spreads, allow for less restrictive assumptions on the state dynamics, and enable extensions to higher dimensional settings through the inclusion of additional state variables. We propose a computational approach that builds on recent work in solving nonlinear partial differential equations based on backward stochastic differential equations (BSDE) by deep learning (Han et al., 2018), as well as on recent work uncovering efficient neural architectures (Pereira et al., 2019), and adapt it to allow for noncontinuous cost structures. To our knowledge, this is the first work to apply a BSDE-based neural solver to the Almgren–Chriss optimal execution framework for learning an optimal closed-loop control policy.

The performance of the learned execution policy is evaluated against two benchmark strategies using expected execution cost as the primary metric, including an analytical zero-spread benchmark

[1] An earlier version of this work appeared as a master's thesis. In accordance with the journal's double-blind review policy, details of this prior work are omitted during review and will be fully disclosed upon acceptance.

and a time-weighted average price strategy. Across calibrated intraday market scenarios, the proposed deep stochastic optimal control approach delivers systematically lower execution costs, demonstrating its ability to extend classical Almgren–Chriss models to more realistic market settings.

The remainder of this article is structured as follows. Section 2 reviews the relevant literature, covering diffusion-based execution models, stochastic optimal control and reinforcement learning formulations, and deep learning methods for high-dimensional control problems. Section 3 introduces the market setting and the core components of the execution model, including price dynamics, trading costs, and generation forecast uncertainty. Section 4 formulates the optimal execution problem as a stochastic control problem, derives the associated Hamilton–Jacobi–Bellman equation, and presents the Deep Stochastic Optimal Control solution based on a forward–backward stochastic differential equation representation. Section 5 describes the data and calibration procedure, derives analytical benchmarks under simplified assumptions, and evaluates the proposed method through numerical experiments. Finally, Section 6 summarizes the main findings and discusses directions for future research.

The accelerating transition toward renewable energy has introduced new complexities into the electricity markets. As intermittent wind and solar generation becomes increasingly important sources of supply, market participants must manage the inherent uncertainty associated with weather-dependent production. This unpredictability has made forecast errors a central operational challenge and highlighted the need for mechanisms that complement the traditional day-ahead market (DAM).

The continuous intraday market (IDM) has emerged as a critical platform for participants to adjust their positions as forecasts are refined closer to real time. During the past five years, the trading volumes in the IDM have increased by more than 200%, underscoring its growing importance to maintain the balance of the system in an increasingly volatile and renewable-driven landscape (EPEX SPOT, 2021, 2025). However, unlike auction-based markets with uniform clearing prices, the IDM operates through a pay-as-bid limit order book (LOB), exposing participants to price volatility, bid–ask spreads, and market impact. As a result, traders must navigate both price variation and execution costs. In addition, regardless of whether a trader carries a position from DAM to IDM, they must continually respond to updated forecasts of demand and generation. These revisions introduce significant uncertainty around the target net volume to be traded.

Electricity differs from other commodities because it cannot be stored economically, requiring real-time balancing of supply and demand. Liberalized electricity systems organize trading through sequential markets: the day-ahead auction, intraday auctions, continuous intraday trading, and final balancing operated by transmission system operators (TSOs). The continuous intraday market (IDM) allows participants to adjust their positions as forecasts and prices evolve (Bush, 2023).

The increasing share of renewable generation amplifies forecast uncertainty, raising the risk of costly imbalances (Katarzyna et al., 2019). The IDM thus serves as the final opportunity for market participants to offset forecast errors before gate closure, playing a key role in maintaining system reliability. Trading volumes on exchanges such as EPEX SPOT have more than doubled since 2019, underscoring the growing operational and economic importance of intraday trading (EPEX SPOT, 2021, 2025; Kath Ziel, 2020).

Continuous limit order book (LOB) systems are now dominating most financial and commodity markets (Angel et al., 2011; Harris, 2003; Nord Pool, 2025; Intercontinental Exchange, 2025). In a LOB, traders post limit orders specifying price and quantity, while market orders execute immediately against the best available prices (Cartea et al., 2015, p. 18). The bid–ask spread reflects market liquidity and constitutes a key transaction cost (Fleming, 2003; Harris, 2003). Large trades may "walk the book," consuming liquidity at successive price levels and inducing temporary or permanent price impacts (Ozenbas et al., 2022; Frey Sandås, 2017; Kwok Lau, 2005). The rise of electronic LOBs has transformed trading into a computational activity dominated by algorithmic execution strategies (Breuer et al., 2011). While originally developed for equities, similar mechanisms now underpin the *continuous intraday electricity market*, where trading ensures real-time balance between production and consumption.

This optimization problem resulting from the minimization of costs is commonly referred to in the literature as the *optimal order execution problem.* Although it has been extensively studied in equity markets and has been increasingly applied to electricity trading, existing approaches often simplify the problem to obtain analytical solutions or rely on classical numerical methods that do not scale well in high-dimensional settings.

The objective of this study is to develop a scalable solution method for the optimal trade execution problem in continuous intraday electricity market trading, capable of handling non-smooth cost dynamics that preclude closed-form solutions. To achieve this, we adopt the stochastic control framework introduced by Almgren Chriss (2001) and solve the resulting Hamilton–Jacobi–Bellman (HJB) equation using a modified variant of the deep neural backwards stochastic differential equation solver (Deep BSDE) proposed by Han et al. (2018), adapted to incorporate closed-loop controls[2]. This method approximates the value function of the HJB equation and computes the optimal controls from its gradients.

The performance of the learned policy is evaluated against two benchmark strategies, using expected execution cost as the primary metric: (i) an analytical solution assuming zero bid–ask spread, (ii) a time-weighted average price (TWAP) strategy. Our findings are twofold. First, the Deep Stochastic Optimal Control (SOC) proves to consistently outperform all benchmarks across the evaluated runs, except under heightened imbalance-penalty conditions, where the analytical policy achieves lower total cost. Second, the training process underscores the usual but critical importance of neural network tuning, as performance is strongly affected by the selection of hyperparameters, network architecture, and training pipeline.

To our knowledge, this is the first study to apply a neural stochastic control solver to the Almgren-Chriss framework, enabling it to scale beyond low-dimensional settings and incorporate realistic market dynamics. It also presents the first application of such a method in the context of the continuous intraday electricity market, where nonlinear execution costs, bid-ask spreads, and volume uncertainty pose structural challenges not captured by classical models. Lastly, by leveraging the Deep BSDE architecture and Physics-Informed Neural Networks, we propose a general framework for solving nonsmooth stochastic control problems through viscosity solution approximations of

[2] A closed-loop (or feedback) control is a policy where decisions at each time depend on the current state of the system

HJB equations.

It is shown that the Deep SOC solver can approximate closed-loop control policies in stochastic execution problems with non-smooth costs and offers a scalable alternative to classical approximation methods, which are known to have dimensional limitations. Considering that this is a model-based approach, the outperforming results suggest it can be worthwhile to approximate control policies with neural networks rather than rely on biased analytical models.

ET FORSØK PÅ Å OMSKRIVE LITT: The objective of this study is to develop a scalable and theoretically grounded solution method for the optimal trade execution problem in continuous intraday electricity market trading, capable of handling non-smooth and nonlinear cost dynamics that preclude closed-form solutions. Building on the stochastic control framework of Almgren (2012), we formulate the trader's optimization problem as a Hamilton–Jacobi–Bellman (HJB) equation and solve it using a modified version of the Deep Backward Stochastic Differential Equation (Deep BSDE) solver of
citetHanJentzenE2018, adapted to incorporate closed-loop (feedback) controls.

The proposed Deep Stochastic Optimal Control (Deep SOC) method approximates the value function of the HJB equation and derives optimal control policies directly from its gradients. Performance is evaluated against two benchmarks using expected execution cost as the primary metric: (i) the analytical solution under zero bid–ask spread, and (ii) a time-weighted average price (TWAP) strategy. The Deep SOC policy consistently outperforms both benchmarks across most market conditions, except under extreme imbalance-penalty regimes, where the analytical strategy remains marginally superior. The training results also highlight the sensitivity of performance to neural network hyperparameters, emphasizing the importance of architecture design and training stability.

To our knowledge, this is the first study to apply a neural stochastic control solver to the Almgren–Chriss execution framework, extending it beyond low-dimensional or smooth-cost assumptions and introducing realistic market features such as bid–ask spreads, nonlinear impact, and volume uncertainty. Furthermore, by embedding the Deep BSDE solver within a viscosity solution framework, we provide a generalizable approach for solving non-smooth stochastic control problems. The results demonstrate that neural approximations can effectively recover closed-loop optimal controls in continuous-time execution settings, offering a scalable and model-consistent alternative to traditional discretization-based methods.

2 Literature review

The literature relevant to this study spans three closely related areas: (i) diffusion-based models of trade execution, (ii) stochastic optimal control, and (iii) deep learning methods for solving PDEs.

2.1 Diffusion-based models of optimal execution

Early work on optimal execution models trading dynamics using continuous-time diffusion approximations. Bertsimas Lo (1998) introduced a dynamic programming formulation for minimizing execution costs over a fixed horizon. Almgren Chriss (2001) proposed a continuous-time model in which prices follow an arithmetic Brownian motion and trades generate linear temporary and permanent price impact, yielding closed-form optimal execution schedules under a mean-variance objective.

Subsequent extensions introduced nonlinear impact functions (Almgren, 2003), stochastic liquidity and volatility (Almgren, 2012), and alternative risk and utility formulations (Marzo et al., 2011; Gatheral Schied, 2012; Schied, 2013; Forsyth et al., 2012; Cheng et al., 2017, 2019; Bulthuis et al., 2017). These diffusion-based models provide clear economic structure but typically rely on low-dimensional state spaces and smooth cost functions.

In electricity markets, execution models have been adapted to account for stochastic volume targets driven by forecast uncertainty. Aïd et al. (2016) introduced uncertain generation into an execution framework and derived analytical solutions under simplifying assumptions. Later work incorporated multiple markets and stochastic generation dynamics (Tan Tankov, 2018; Glas et al., 2020), while empirical studies calibrated Almgren-Chriss-type models to intraday electricity data and limit order book dynamics (Kath Ziel, 2020; Vaes Hauser, 2022).

2.2 Stochastic optimal control

Diffusion-based execution models can be viewed as stochastic optimal control problems in which trading decisions affect the drift of a controlled diffusion. In this setting, the value function solves a nonlinear PDE, namely the Hamilton-Jacobi-Bellman (HJB) equation, and the optimal policy is obtained via pointwise minimization of the Hamiltonian (Yong Zhou, 1999). When the HJB equation admits no closed form solution, it must be approximated using numerical PDE methods.

Reinforcement learning provides a model-free alternative by learning execution policies directly from data. Nevmyvaka et al. (2006) showed that agents could learn execution strategies through market interaction without specifying price dynamics. Hybrid approaches combining reinforcement learning with Almgren-Chriss-type structure were proposed by Hendricks Wilcox (2014), while recent deep reinforcement learning methods further extend this model (Ning et al., 2021; Lin Beling, 2020).

2.3 Deep learning methods for solving PDEs

Following E et al. (2017), there has been substantial progress in using deep learning to solve partial differential equations, including problems in high dimensions where classical grid-based numerical methods become impractical.

A central line of work reformulates HJB equations as forward–backward stochastic differential equations. The deep BSDE method of Han et al. (2018) parametrizes the initial value of the control problem and the associated gradient processes using neural networks, and trains these networks by enforcing consistency with the BSDE dynamics along simulated sample paths. Training proceeds via Monte Carlo simulation of the underlying diffusion, and deviations from the BSDE representation are penalized through a loss function defined over the forward trajectories.

Crucially, this class of methods relies on a sampling policy to generate state trajectories during training. Similar to reinforcement learning, the training policy may be chosen on-policy or off-policy, but must adequately explore regions of the state space that are relevant under the optimal control.

For example, Han et al. (2018) employ an off-policy exploration scheme based on white-noise, while Ata et al. (2025) use a constant reference policy to sample trajectories of the reflected Brownian motion, whose ergodic dynamics ensure visitation of the relevant state space in the long run. In contrast, Pereira et al. (2019) adopt an on-policy exploration strategy, where trajectories are generated under the current control implied by the learned value gradients. To mitigate poor exploration and variance, they exploit Girsanov's theorem to perform importance sampling by modifying the drift during simulation.

Beyond BSDE based approaches, a broad class of alternative deep learning methods has been developed for approximating solutions of PDEs. These include primal dual and variational formulations such as the Deep Ritz method (Henry-Labordere, 2017), deep backward and splitting schemes for semilinear equations (Huré et al., 2020), and methods targeting fully nonlinear PDEs (Beck et al., 2019). Related mesh free approaches include the Deep Galerkin Method (Sirignano Spiliopoulos, 2018) and Physics-Informed Neural Networks (Raissi et al., 2019), which enforce the PDE structure through residual based penalties rather than stochastic representations. Comprehensive overviews of these methodologies can be found in the survey articles of Beck et al. (2019) and E et al. (2022).

Ser ut som Baris Ata liker å gi en oversikt på hvilken litteratur som blir gjennomgått helt i starten av lit. rew. for eksempel slik: 'Two of the most relevant streams of literature are (i) drift rate control problems and (ii) solving PDEs using deep learning.'

Videre, i mange av sine papers om control problems, er han ganske spesifikk i kategoriseringen av control problem (e.g. drift, diffusion, eller optimal stopping control problems), horisont (e.g. infinite/finite), og prosess (e.g. Brownian-/reflected-Brownian motion).

På grunn av: - Ovenfornevnte punkt (uten at det skal bli for 'tilpasset' han) - At innholdet ikke kan være for 'niche' - Vi ikke er de første som formulerer IDM trading som et optimal execution problem - For å sette i kontekst av bredere litteratur

ser jeg for meg at det kan være gunstig å nevne at vi har:

- Drift control problem (linear) - Brownian motion - Finite horizon - Non-smooth cost (edge)

med matematisk modell:

- closed form q = argmin Hamiltonian

løst med:

- deep learning for PDE med FBSDE system (i kontrast med andre metoder, e.g. PINN / deep Galerkin)

innenfor denne løsningsmetoden:

- finner vi andre som har implementert dette på "teoretisk" data: Ata et. al (2024) for infinite horizon reflected BM, og

vi kombinerer elementer fra forskjellige eksisterende arktitekturer:

- to nevrale nettverk (inspirert av Pereira et al. 2019, de bruker også importance sampling) 1) én som lærer den initielle verdi funksjonen 2) én som direkte lærer gradientene til verdifunksjonen per tidssteg

- nettverk 1) benytter en NAISNet-arkitektur for input-output stabilitet (inspirert av Güler et al. 2019). - nettverk 2) følger en LSTM-arkitektur for å modellere temporale relasjoner, med NAIS-lignende stabilitetsbegrensninger i den rekurrente dynamikken. (tillater andre tidssteg enn det den ble trent på) - reinforcement style + PINN + terminal loss

anvendt på elektrisitet IDM trading:

- Glas et al. har sammeverk som vi har. - med potensial å utvide states (likviditet, spreads, imbalance cost)

3 Model Overview

In this section, we present the key components of our model and outline how they interact. The model keeps the structure of the Almgren Chriss (2001) model, while incorporating uncertain volume target from Aïd et al. (2016).

3.1 Market and Agent Dynamics

We consider a continuous-time control problem on the interval $[0, T]$, where T denotes the terminal delivery time. The trading rate $q(t)$ is modeled as a progressively measurable control process belonging to the admissible set $\mathcal{A} \subset L^2_{\mathcal{F}}([0, T]; \mathbb{R})$, defined on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{P}, (\mathcal{F}_t)_{t\in[0,T]})$. At each time $t \in [0, T]$, the control process satisfies the pointwise constraint $q(t, \omega) \in \mathcal{Q}_{\text{ad}} \subseteq \mathbb{R}$ almost surely.

Trading Position

The cumulative traded position, denoted by $X^q(t)$, evolves according to:

$$dX^q(t) = q(t)\, dt, \quad X(0) = X_0, \tag{1}$$

where X_0 represents the initial position determined by the agent's day-ahead market commitments. In other words, $X^q(t)$ reflects the total volume of electricity bought or sold up to time t. The superscript q indicates that the position $X^q(t)$ evolves as a result of applying the control strategy $q(t)$ over time. The cumulative position is thus obtained by integrating the trading activity over the trading horizon:

$$X^q(t) = X_0 + \int_0^t q(s)\, ds. \tag{2}$$

Price Dynamics

The mid-price $P^q(t)$ follows the dynamics:

$$\begin{aligned} dP^q(t) =& [\mu_P + g(t, q(t))]\, dt + \sigma^q_P(t)\, dW^P(t), \\ P(0) =& P_0, \end{aligned} \tag{3}$$

where P_0 is the initial price, μ_P is the drift term, $g(t, q(t))$ represents the permanent price impact as a function of the trading rate, and $\sigma_P(t)$ is a time-dependent volatility coefficient. Here, $W^P(t)$ is a standard Brownian motion. The superscript q reflects that the mid-price $P^q(t)$ is affected by the agent's trading activity through the permanent price impact function $g(t, q(t))$, and thus by $q(t)$.

We define the permanent price impact function $g(t, q(t))$ used in our model:

$$g(t, q(t)) = \nu(t)q(t) \tag{4}$$

This function takes various forms in the literature. The majority of papers assume a linear relationship, as in Almgren Chriss (2001), often with a constant impact coefficient ν (Aïd et al., 2016; Glas et al., 2020).

Execution Price

The execution price $\tilde{P}(t, P^q, q)$ reflects the total cost of executing a trade at time t, given by:

$$\tilde{P}(t, P^q(t), q(t)) = \underbrace{P^q(t)}_{\text{mid price}} + \underbrace{\text{sign}(q(t))\psi(t)}_{\text{half-spread}} + \underbrace{\varphi(t, q(t))}_{\text{execution cost}}, \tag{5}$$

where $P^q(t)$ denotes the mid price, $\psi(t)$ is the half-spread, and $\varphi(t, q(t))$ captures temporary execution cost due to liquidity impact. The sign term accounts for directional spread costs, increasing the price for buys and reducing it for sells.

In our model, temporary impact is modeled linearly as

$$\varphi(t, q(t)) = \gamma(t)q(t), \tag{6}$$

following the standard assumption in optimal execution (Almgren Chriss, 2001; Glas et al., 2020). The coefficient $\gamma(t)$ captures market liquidity and may vary over time to reflect changing conditions.

Generation Forecasting

The residual generation forecast $G(t)$ evolves according to:

$$\begin{aligned} dG(t) =& \mu_G \, dt + \sigma_G(t) \left(\rho \, dW^P(t) + \sqrt{1-\rho^2} \, dW^G(t) \right), \\ G(0) =& G_0, \end{aligned} \tag{7}$$

where G_0 is the initial generation forecast, $\mu_G(t)$ is the drift of the generation forecast, $\sigma_G(t)$ is the time-dependent volatility, and $W^P(t), W^G(t)$ are independent standard Brownian motions. The correlation parameter $\rho \in [-1, 1]$ models dependence between price and generation. We assume $\mu_G = 0$, reflecting that the forecast is unbiased and incorporates all available information.

Running Cost

The instantaneous trading cost is given by:

$$f(t, P^q(t), q(t)) = q(t)\tilde{P}(t, P^q(t), q(t)), \tag{8}$$

where $q(t)$ denotes the rate at which the agent is trading (positive for buying, negative for selling), and $\tilde{P}(t, P^q(t), q(t))$ is the execution price defined in (5).

Terminal Payoff

At the terminal time T, the agent is penalized for any mismatch between cumulative trading $X(T)$ and realized residual generation $G(T)$. The terminal payoff is modeled as a quadratic imbalance cost:

$$h(X^q(T), G(T)) = \frac{\eta}{2}\left(X^q(T) - G(T)\right)^2, \tag{9}$$

where η reflects the agent's aversion to delivery imbalance and encodes both market penalties and internal cost from deviations.

3.2 Control Problem and Objective

The agent aims to minimize the expected total cost incurred from intraday trading and terminal imbalance. Using the dynamics (1), (3), and (7), the state vector $\boldsymbol{y}(t) = (X(t),\ P(t),\ G(t))^\top$ evolves according to

$$d\boldsymbol{y}^q(t) = \boldsymbol{b}(t, q(t))\,dt + \Sigma(t)\,dW(t),$$

where the drift vector $\boldsymbol{b}(t, q(t)) \in \mathbb{R}^3$ is given by:

$$\boldsymbol{b}(t, q(t)) = \begin{pmatrix} q(t) \\ \mu_P + g(t, q(t)) \\ 0 \end{pmatrix},$$

with covariance structure of the state diffusion is given by:

$$\Sigma\Sigma^\top(t) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \sigma_P^2(t) & \rho\,\sigma_P(t)\sigma_G(t) \\ 0 & \rho\,\sigma_P(t)\sigma_G(t) & \sigma_G^2(t) \end{bmatrix}.$$

The associated cost functional is defined as:

$$\begin{aligned} J(t, \boldsymbol{y}; q) = \mathbb{E}^{t,\boldsymbol{y}}\Bigg[& \underbrace{\int_t^T f(s, \boldsymbol{y}^q(s), q(s))\,ds}_{\text{Running cost from intraday trading}} \\ & + \underbrace{h(\boldsymbol{y}^q(T))}_{\text{Terminal imbalance penalty}} \Bigg] \end{aligned} \tag{10}$$

where f denotes the instantaneous trading cost and h the terminal imbalance penalty.

For brevity, we write $h(X, G) = h(\boldsymbol{y})$ and $f(t, P, q) = f(t, \boldsymbol{y}, q)$ when their dependence on specific state components is clear. Throughout, we omit the superscript q on state variables, with the understanding that their evolution is induced by the candidate control strategy.

The goal of the agent is to find a control strategy $q \in \mathcal{A}$ that minimizes the total expected cost defined in (10). To formalize this, we define the *value function* $V(t, \boldsymbol{y})$ as the minimal cost achievable when starting from time t and state $\boldsymbol{y}$:

$$V(t, \boldsymbol{y}) := \inf_{q \in \mathcal{A}} J(t, \boldsymbol{y}; q), \tag{11}$$

where $\mathcal{A}$ denotes the set of admissible control strategies.

The goal of the agent is to find a control strategy $q \in \mathcal{A}$ that minimizes the expected total cost (10). An optimal control $\bar{q} \in \mathcal{A}$ is one that attains the infimum in (11).

Under standard assumptions, optimal controls can be expressed in feedback (closed-loop) form, meaning that the trading decision at time t depends only on the current state $\boldsymbol{y}$.

4 Stochastic Control Formulation and Solution Method

4.1 Dynamic Programming and HJB Formulation

The value function $V(t, \boldsymbol{y})$ denotes the minimal expected cost-to-go from state $\boldsymbol{y}$ at time t. By the stochastic Bellman principle (Yong Zhou, 1999), it satisfies a recursive optimality condition that, under standard smoothness assumptions, yields the HJB equation

$$\begin{aligned} 0 = \partial_t V(t, \boldsymbol{y}) + \inf_{q \in \mathcal{Q}_{\text{ad}}} \Big\{ & f(t, \boldsymbol{y}, q) + \langle \nabla_{\boldsymbol{y}} V, \, \boldsymbol{b}(t, q) \rangle \\ & + \tfrac{1}{2} \operatorname{Tr} \left(\Sigma \Sigma^\top (t) \nabla^2_{\boldsymbol{y}\boldsymbol{y}} V \right) \Big\}, \end{aligned} \tag{12}$$

subject to terminal condition $V(T, \boldsymbol{y}) = h(\boldsymbol{y})$. The optimal control follows as the minimizer of the Hamiltonian,

$$\bar{q}(t, \boldsymbol{y}) = \arg \min_{q \in \mathcal{Q}_{\text{ad}}} \left\{ f(t, \boldsymbol{y}, q) + \langle \nabla_{\boldsymbol{y}} V, \, \boldsymbol{b}(t, q) \rangle \right\}. \tag{13}$$

When the Hamiltonian minimization does not admit a closed-form solution, the optimal control can be obtained by jointly parameterizing the value function and the control policy and enforcing Hamiltonian optimality conditions during training. In this setting, the resulting policy remains a deterministic state-feedback control derived from the specified stochastic model.

In implementation, controls are evaluated on a discrete grid and held constant over intervals $[t_i, t_{i+1})$, converging to the continuous formulation as $\Delta t \to 0$.

Non-smooth execution costs may prevent differentiability of V. In this case, the appropriate solution concept is a viscosity solution, which ensures existence and uniqueness under standard Lipschitz conditions (Yong Zhou, 1999) while preserving the interpretation of V as the value function.

Forward-Backward SDE Representation

Following Han et al. (2018), application of Itô's lemma to $V(t, \boldsymbol{y}(t))$ yields the equivalent forward–backward stochastic differential system

$$\begin{aligned} d\boldsymbol{y}(t) &= \boldsymbol{b}(t, q(t))\, dt + \Sigma(t)\, dW(t), \\ dY(t) &= -f(t, \boldsymbol{y}(t), q(t))\, dt + Z(t)^\top dW(t), \\ Y(T) &= h(\boldsymbol{y}(T)), \end{aligned} \tag{14}$$

where $Y(t) = V(t, \boldsymbol{y}(t))$ and $Z(t) = \Sigma^\top(t)\nabla_{\boldsymbol{y}} V(t, \boldsymbol{y}(t))$. This representation provides a basis for numerical approximation through simulated trajectories and is used in the Deep Stochastic Optimal Control approach introduced in the next section.

4.2 Computational Model

Having reformulated the HJB equation as a forward-backward stochastic differential equation, we now introduce a neural network-based approach for approximating its solution. This method, enables learning both the value function $V(t, \boldsymbol{y})$ and the corresponding optimal control $q(t, \boldsymbol{y})$ in high-dimensional stochastic control problems where classical discretization schemes become infeasible.

Neural Network Parameterization

To represent the value process over time, we adopt a hybrid parameterization that focuses on local structure: instead of directly predicting the value $Y(t_i)$ at each time step, we model its temporal and spatial derivatives $(\partial_t Y, \nabla_{\boldsymbol{y}} Y)$, which appear explicitly in the BSDE dynamics and the Hamiltonian. The value function is then incrementally reconstructed using a first-order Taylor expansion.

We define a parameterized function $Y^\theta(t, \boldsymbol{y})$ to approximate the value function, where θ denotes the neural network weights. The associated backward and control quantities are:

$$\begin{aligned} Y^\theta(t, \boldsymbol{y}) &\approx Y(t), \\ Z^\theta(t, \boldsymbol{y}) &:= \Sigma(t)^\top \nabla_{\boldsymbol{y}} Y^\theta(t, \boldsymbol{y}), \\ q^\theta(t, \boldsymbol{y}) &:= \arg\min_{q \in \mathcal{Q}_{\mathrm{ad}}} H(t, \boldsymbol{y}, q; \theta), \end{aligned}$$

where Z^θ represents the backward stochastic term in the BSDE, and q^θ is the feedback control obtained by minimizing the Hamiltonian using the learned gradient of the value function.

In problems where this minimization does not admit a closed-form solution, it can be approximated by a separately parameterized control network, while remaining within a model-based framework. Related approaches based on the stochastic maximum principle directly parameterize the control and enforce first-order optimality conditions during training; see, for example, Ji et al. (2022).

From a reinforcement learning perspective, the proposed approach bears similarities to actor–critic methods. The neural approximation of the value function plays the role of a critic, while the control policy is obtained analytically as the minimizer of the Hamiltonian using the learned value gradients. Unlike model-free actor–critic algorithms, both components are tied through the HJB structure and trained jointly to satisfy the underlying stochastic control equations, rather than through temporal-difference updates.

Discretization Scheme

We discretize the time interval $[0,T]$ into N steps with grid points $t_i = i\Delta t$. At each step, we simulate the forward state process and draw independent Brownian increments $\Delta W_i \sim \mathcal{N}(0, \Delta t\, I)$. The state at time t_i is written as $\boldsymbol{y}_i = (X(t_i), P(t_i), G(t_i))^\top$. For the neural approximation, we denote $Y_i^\theta = Y^\theta(t_i, \boldsymbol{y}_i)$, $Z_i^\theta = Z^\theta(t_i, \boldsymbol{y}_i)$, and $q_i^\theta = q^\theta(t_i, \boldsymbol{y}_i)$.

The forward and backward components are then updated recursively as

$$\boldsymbol{y}_{i+1} \approx \boldsymbol{y}_i + \boldsymbol{b}(t_i, q_i^\theta)\Delta t + \Sigma(t_i)\Delta W_i, \tag{15}$$

$$Y_{i+1}^\theta \approx Y_i^\theta + \partial_t Y_i^\theta\, \Delta t + \nabla_{\boldsymbol{y}} Y_i^{\theta\top}(\boldsymbol{y}_{i+1} - \boldsymbol{y}_i), \tag{16}$$

which links the simulated state dynamics with the backward evolution of the value function.

Training Loss

The total training loss consists of several terms that enforce consistency with the BSDE formulation, satisfaction of the HJB equation, and alignment with the terminal and cost objectives. These are combined using an adaptive multi-task weighting scheme inspired by Kendall et al. (2018), where each loss term is weighted by a learnable uncertainty parameter:

$$\mathcal{L}_{\text{total}}(\theta) := \sum_i \lambda_i^{\text{task}} \cdot \left(\frac{1}{\sigma_i^2} \mathcal{L}_i(\theta) + \log \sigma_i \right), \tag{17}$$

where λ_i^{task} is a manually specified importance weight, and σ_i^2 is a trainable parameter reflecting the model's confidence in task i. This encourages automatic trade-offs between competing objectives.

BSDE Consistency Loss: This term ensures consistency with the discretized BSDE dynamics (14):

$$\begin{aligned}\mathcal{L}_{\text{BSDE}} := \mathbb{E}\Big[\sum_{i=0}^{N-1} \Big\| Y_{i+1}^\theta - \Big(Y_i^\theta - f(t_i, \boldsymbol{y}_i, q_i^\theta)\Delta t \\ + Z_i^{\theta\top} \Delta W_i \Big) \Big\|^2 \Big].\end{aligned} \tag{18}$$

Terminal Condition Loss: This loss ensures the terminal value of the BSDE matches the prescribed final payoff:

$$\mathcal{L}_{\text{T}} := \mathbb{E}\left[\left\| Y_N^\theta - h(\boldsymbol{y}_N) \right\|^2 \right]. \tag{19}$$

HJB Residual Loss: Inspired by the Physics-Informed Neural Network (PINN) framework Raissi et al. (2019), this loss penalizes violations of the HJB equation at collocation points across the domain:

$$\begin{aligned}\mathcal{L}_{\mathrm{PINN}} := \mathbb{E}\Bigg[\Bigg\| &\partial_t Y^\theta + \nabla_{\boldsymbol{y}} Y^{\theta\top} \boldsymbol{b}(t, q^\theta) \\ &+ \frac{1}{2}\operatorname{Tr}[\Sigma\Sigma^\top(t)\nabla^2_{\boldsymbol{yy}} Y^\theta] + f(t, \boldsymbol{y}, q^\theta)\Bigg\|^2\Bigg].\end{aligned} \tag{20}$$

This term directly enforces the residual of the HJB equation.

Cost Objective Loss: This final term encodes the stochastic control objective directly:

$$\mathcal{L}_{\mathrm{cost}} := \mathbb{E}\left[\int_0^T f(t, \boldsymbol{y}_t, q_t^\theta)\, dt + h(\boldsymbol{y}_T)\right]. \tag{21}$$

It represents the expected total cost-to-go under the learned control policy q^θ, analogous to a reinforcement learning objective. Unlike the other terms, it is excluded from the uncertainty-weighted loss (17), since it does not arise from a homoscedastic Gaussian likelihood and lacks a natural variance interpretation.

Neural Network Architecture

The architecture consists of two components. One component predicts the initial value function Y_0^θ. The other provides the gradient vector $(\partial_t Y_i^\theta, \nabla_{\boldsymbol{y}} Y_i^\theta)$ that is used at each time step for the Taylor reconstruction of the value process along a simulated rollout driven by sampled diffusion increments ΔW_i which act as training samples.

The first component is a feedforward NAISNet. Inputs are normalized and outputs rescaled to improve robustness to the scale of the state. Its design encourages stable feature propagation through structured weight constraints (Ciccone et al., 2018), which supports reliable initialization of the value process.

The second component is an LSTM that models temporal evolution along the simulated trajectories. At each step, the LSTM processes the current time and state $(t_i, \boldsymbol{y}_i)$ to produce a latent representation that is mapped to the local derivatives. If required, second order and cross derivative terms ($\partial_{\boldsymbol{yy}} Y^\theta$, $\partial_{tt} Y^\theta$, and $\partial_{t\boldsymbol{y}} Y^\theta$), obtained by automatic differentiation, are included to improve approximation accuracy. At inference the control, which is the quantity of practical interest, is obtained directly through a single forward evaluation of the LSTM, whereas the value function is used primarily to enforce BSDE and terminal cost consistency within the training loss and is not recovered explicitly at prediction time.

The design reflects ideas from Güler et al. (2019), who demonstrate the role of stability in neural BSDE solvers and motivate NAISNet as a stable architecture, and from Pereira et al. (2019), who show that recurrent networks such as LSTMs effectively capture temporal dependency in deep BSDE formulations. Figure 1 illustrates how the architecture interacts with the forward state in closed loop, using the predicted gradients to generate controls and reconstruct the value function over time along the forward rollout.

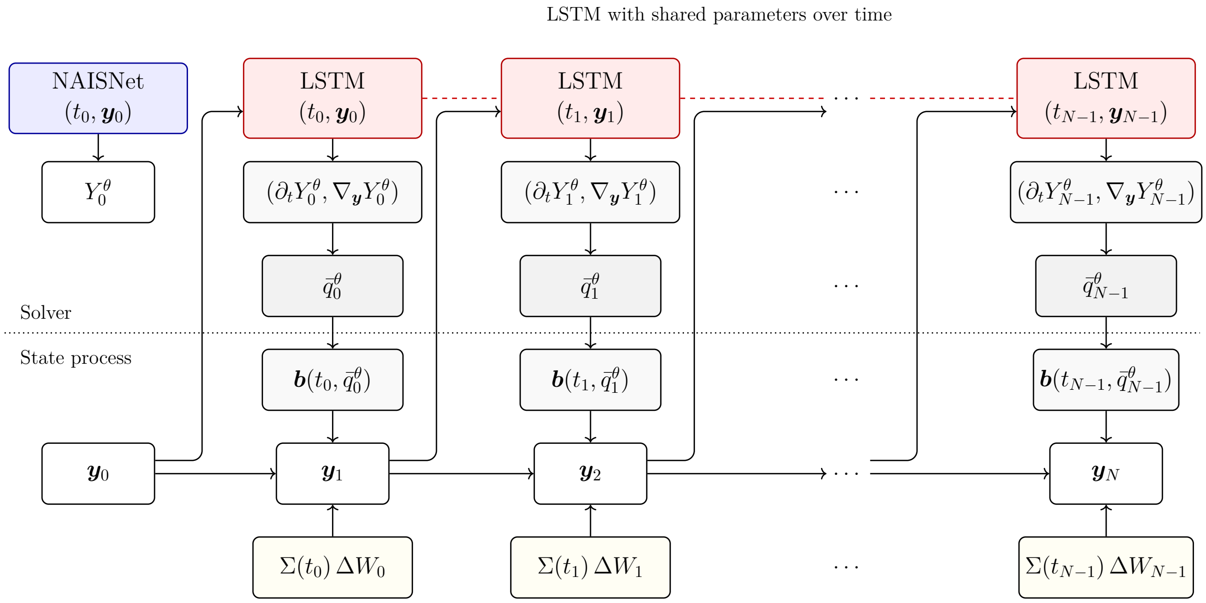

LSTM with shared parameters over time
NAISNet
$(t_0, \boldsymbol{y}_0)$
LSTM
$(t_0, \boldsymbol{y}_0)$
LSTM
$(t_1, \boldsymbol{y}_1)$
LSTM
$(t_{N-1}, \boldsymbol{y}_{N-1})$
Y_0^θ
$(\partial_t Y_0^\theta, \nabla_{\boldsymbol{y}} Y_0^\theta)$
$(\partial_t Y_1^\theta, \nabla_{\boldsymbol{y}} Y_1^\theta)$
$(\partial_t Y_{N-1}^\theta, \nabla_{\boldsymbol{y}} Y_{N-1}^\theta)$
$\bar{q}_0^\theta$
$\bar{q}_1^\theta$
$\bar{q}_{N-1}^\theta$
Solver
State process
$\boldsymbol{b}(t_0, \bar{q}_0^\theta)$
$\boldsymbol{b}(t_1, \bar{q}_1^\theta)$
$\boldsymbol{b}(t_{N-1}, \bar{q}_{N-1}^\theta)$
$\boldsymbol{y}_0$
$\boldsymbol{y}_1$
$\boldsymbol{y}_2$
$\boldsymbol{y}_N$
$\Sigma(t_0)\,\Delta W_0$
$\Sigma(t_1)\,\Delta W_1$
$\Sigma(t_{N-1})\,\Delta W_{N-1}$
. . .

Figure 1: The figure shows the architecture of the solver and its interaction with the forward state evolution. The initial value Y_0^θ is computed by a feedforward NAISNet using the initial time and state $(t_0, \boldsymbol{y}_0)$. For each time t_i, an LSTM with shared parameters takes the current time and state $(t_i, \boldsymbol{y}_i)$ as input and outputs the local derivatives $(\partial_t Y_i^\theta, \nabla_{\boldsymbol{y}} Y_i^\theta)$. These derivatives are used to compute the control q_i^θ, which determines the drift term $\boldsymbol{b}(t_i, q_i^\theta)$ and the state increments. In parallel, the diffusion term $\Sigma(t_i)\Delta W_i$ contributes to the state update. Together these components drive the forward state process in closed loop, while the value function along the trajectory is reconstructed from Y_0^θ and the sequence of derivatives using a Taylor update scheme.

4.3 Model Specification and Optimal Control

We derive a closed-form benchmark under zero drift, constant volatilities, no bid-ask spread, and linear temporary and permanent price impact, i.e., $\sigma_P(t) = \sigma_P$, $\sigma_G(t) = \sigma_G$, $\psi(t) = 0$, $\varphi(t, q) = \gamma q$, and $g(t, q) = \nu q$. Under these assumptions, the running cost and the drift term becomes:

$$f(t, \boldsymbol{y}, q) = q\tilde{P}(t, \boldsymbol{y}, q) = q(P + \gamma q), \quad \boldsymbol{b}(t, q) = \begin{pmatrix} q \\ \nu q \\ 0 \end{pmatrix},$$

with corresponding Hamiltonian:

$$H(t, \boldsymbol{y}, q, \nabla v) = \gamma q^2 + (P + \partial_X v + \nu \partial_P v)\, q.$$

Minimizing this quadratic function over q yields the optimal trading rate:

$$\bar{q}(t, \boldsymbol{y}) = \arg\min_{q \in \mathbb{R}} H(t, \boldsymbol{y}, q, \nabla v) = -\frac{1}{2\gamma} (P + \partial_X v + \nu \partial_P v)\,. \tag{22}$$

The optimal control (22) corresponds directly to the implementation in the neural network-based solver, where the control policy is computed from the gradient of the learned value function:

$$q^\theta(t, \boldsymbol{y}) = -\frac{1}{2\gamma} \Big(P + \partial_X Y^\theta(t, \boldsymbol{y}) + \nu \partial_P Y^\theta(t, \boldsymbol{y})\Big)\,. \tag{23}$$

Here, the partial derivatives are obtained via automatic differentiation. This closed-form policy provides a baseline for evaluating the learned control.

The same functional form remains optimal when volatilities vary deterministically in time, with only the value function coefficient adapting through an additional integral term. When a bid–ask spread is introduced, execution becomes regime-dependent: letting $\Lambda(t, \boldsymbol{y}) = P + \partial_X v + \nu\, \partial_P v$ denote the marginal execution value, the Hamiltonian minimizer takes the piecewise form

$$\bar{q}(t, \boldsymbol{y}) = \begin{cases} -\dfrac{\Lambda(t, \boldsymbol{y}) + \psi(t)}{2\gamma}, & \Lambda(t, \boldsymbol{y}) > \psi(t), \\ -\dfrac{\Lambda(t, \boldsymbol{y}) - \psi(t)}{2\gamma}, & \Lambda(t, \boldsymbol{y}) < -\psi(t), \\ 0, & |\Lambda(t, \boldsymbol{y})| \le \psi(t), \end{cases} \tag{24}$$

creating a no-trade region where the cost of crossing the spread outweighs the marginal benefit of adjusting inventory. For numerical implementation, this discontinuous policy is replaced by a smooth approximation to enable stable gradient-based learning.

5 Data, validation, and performance

5.1 Choosing model parameters

In Section 3, we introduced the structure of the optimal trading model and its parameters. The primary aim of this study is to develop a scalable agent that solves a high-dimensional stochastic control problem, not to replicate the full complexity of real-world markets. Nonetheless, to meaningfully assess the model's performance, we test whether it can operate effectively in a stylized but empirically informed market environment. This section outlines how the model parameters are specified, based on empirical data, reported market characteristics, and well-founded assumptions.

The remainder of this paper adopts the perspective of an onshore wind power producer in Germany with an installed capacity of 200 MW, trading hourly products with delivery at 12:00 CET. A trading horizon of six hours prior to delivery is assumed, consistent with the findings of Féron et al. (2021), who observe that liquidity in the German intraday market typically begins to emerge only five to six hours before delivery. This horizon is discretized into $N = 144$ equidistant time steps of 150 seconds. At this resolution, it is assumed that the producer can both execute trades and receive updates, either directly or indirectly related to the generation position, at a comparable frequency.

The parameter values obtained in this section are listed in Table 3.

Initial Position and Price

An initial position of $X_0 = 0$ MWh and an initial price $P_0 = 80$ EUR/MWh are assumed.

Permanent Impact Coefficient and Price Drift

The permanent price impact coefficient is calibrated using the empirical approach of Glas et al. (2020), applied to signed trade data from the German intraday electricity market. This yields an estimated value of $\nu = 0.042$ EUR/(MWh)^2, statistically significant at the one percent level. Consistent with Glas et al. (2020), the mid price drift is assumed to be zero.

Half-Spread

Féron et al. (2021) analyze the German intraday electricity market and find that the bid-ask spread ranges from 2 to 18 EUR/MWh across different delivery times. Based on this empirical evidence, we assume a spread of 10 EUR/MWh in our model, corresponding to a half-spread of $\psi = 5$ EUR/MWh, while also examining $\psi = 10$.

Temporary impact coefficient

We set the temporary impact coefficient in our model to $\gamma = 0.01$ EUR/MWh^2, consistent with recent simulation-based estimates for the intraday electricity market. In particular, Chatziandreou Karbach (2025) estimate time-varying instantaneous temporary impact coefficients by simulating virtual market orders that sweep through statistically reconstructed snapshots of the German EPEX SPOT intraday order book. They assume a linear relationship between order size and the resulting price change per MWh, with the slope representing the temporary impact coefficient. These estimates range between 0.008 and 0.015 EUR/MWh across the trading horizon. Our choice of $\gamma = 0.01$ EUR/MWh^2 is consistent with the mid-range of these model-based impact levels.

Price volatility

A phenomenon commonly mentioned in the empirical literature on commodity markets is the *Samuelson effect*, which refers to the characteristic that price volatility increases as the delivery time approaches due to rising trading activity and the incorporation of real-time information. Féron et al. (2021) empirically observed this effect in the German intraday electricity market, where the volatility ranges approximately from 3 to 35 EUR/(MWh) $\cdot h^{1/2}$, depending on the time to delivery and the delivery hour.

Following Féron et al. (2021) and consistent with the Samuelson effect, this paper assumes that the volatility of the price process increases linearly from 6.3 to 10.95 EUR/(MWh $\cdot h^{1/2}$) over the six-hour trading horizon. This corresponds to a variance increasing from 40 to 120 EUR^2/(MWh$^2 \cdot h$), as specified in (30). These values fall within the empirical range reported by Féron et al. (2021), and are selected to reflect a realistic yet numerically stable volatility profile for simulation over the six-hour trading period.

Generation forecast volatility and drift

Bielecki et al. (2010) find that the volatility of generation forecasts decreases with the forecast horizon. Empirically, they observe a volatility of approximately 9% and 20% of installed wind power capacity for 1-hour and 6-hour forecast horizons, respectively. With our installed capacity of 200 MW, this corresponds to 18 MWh and 40 MWh. However, to be more conservative with respect to short-term uncertainty, we assume a higher volatility of 15% of installed capacity for the 1-hour horizon, corresponding to 30 MWh.

These values corresponds to variances of 900 MWh for the 1-hour and 1600 MWh for the 6-hour forecast horizon. Based on this, we assume that the generation forecast variance in (30) decreases linearly between these two values.

The generation forecast drift μ_G is assumed to be zero.

Price-forecast correlation

Based on empirical findings in Féron et al. (2021), the correlation between market price increments and renewable forecast increments in the German intraday market ranges from approximately -0.08 to -0.01 for delivery at 12 p.m. during winter months. This paper therefore assumes a reasonable value for this correlation to be $\rho = -0.06$.

Imbalance penalty coefficient

The imbalance penalty is modeled as a quadratic function with an assumed coefficient of $\eta = 10$ EUR/MW^2, reflecting the increasing marginal cost of deviations from the target. We consider this to be a reasonable baseline estimate, but also conduct a sensitivity analysis using a higher value in Section 5.3 to assess the robustness of policy performance under stricter delivery conditions.

5.2 Validation under Smooth and Nonsmooth Trading Costs

Before evaluating the agent in empirically calibrated settings, we verify that it recovers known optimal controls and generalizes to settings without closed-form solutions. At each time step t_i, the action $q(t_i)$ is computed in feedback form using the trained policy $\bar{q}^{\theta}(t_i, \boldsymbol{y}_{t_i})$ produced by the LSTM architecture of Figure 1.

Smooth case ($\psi = 0$)

Figure 2 illustrates that, without a spread, the learned policy closely replicates the analytical benchmark: the control, value process, and state trajectories overlap across simulated paths. Minor deviations in $Y^{\theta}(t)$ are expected because the network reconstructs the value process by integrating its learned derivatives rather than predicting $Y(t)$ directly, which can accumulate small numerical error over time. What matters for trading performance, however, is the feedback control $q^{\theta}(t)$, and its close alignment with the analytical solution shows that the solver identifies the correct control structure.

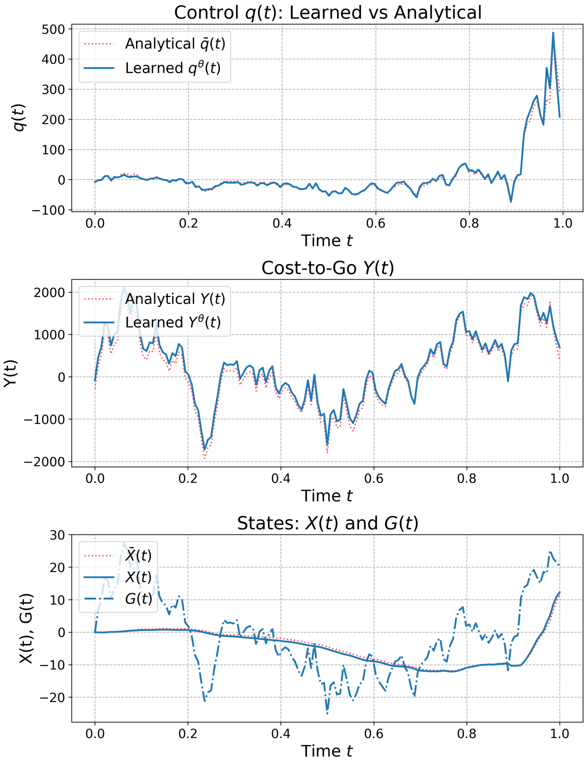


Figure 2: Comparison between learned and analytical solution in the absence of bid–ask spread ($\psi = 0$ EUR/MWh). Solid blue lines denote the neural network approximation, while red dotted lines show the corresponding analytical benchmark. Each panel displays the evolution of control $q(t)$, value function $Y(t)$, and state trajectories $X(t)$ and $G(t)$.

This agreement is also reflected in realized performance. As shown in Figure 3, the empirical cost distributions of the analytical and learned strategies are nearly identical over 100,000 forward simulations. The small deviation from the continuous-time benchmark arises from discrete implementation, where each control is held constant over $[t_i, t_{i+1})$ and thus reacts more slowly than the idealized continuous formulation. Together, these results confirm that the model accurately recovers the analytical solution in the smooth case.

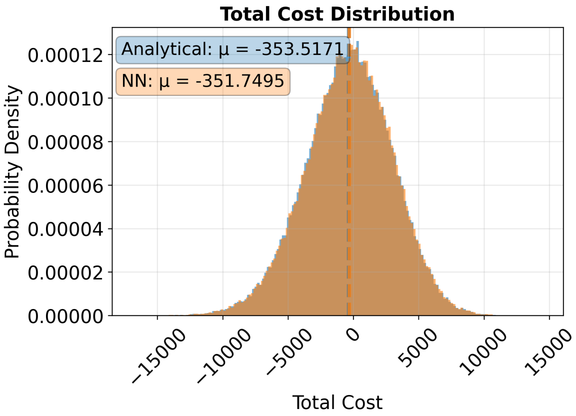


Figure 3: Empirical cost distribution across 100,000 simulation runs for the Analytical agent (blue) and the Deep BDSE agent (orange). Results correspond to the base-case configuration with parameters in Table 3.

We now turn to the more challenging case where trading frictions induce nonsmoothness in the optimal control.

Nonsmooth case ($\psi > 0$)

In the presence of a spread, the optimal control becomes piecewise and sparse: execution occurs only when the marginal value of trading exceeds the spread cost. This introduces challenges that preclude closed-form analytical solutions, resulting in a policy where $q^\theta(t) = 0$ most of the time, with nonzero trades triggered only at distinct points in time.

This behavior aligns with the Hamiltonian minimization rule: trading is optimal only when the marginal execution value (24) satisfies $|\Lambda| > \psi$. As shown in the top panel of Figure 4, trading bursts appear only occasionally, often closer to the terminal time. The learned policy captures this structure effectively, identifying when the benefit of trading outweighs the transaction cost.

The associated state dynamics become stepwise: as shown in the lower panel, $X(t)$ remains constant for extended periods and adjusts in discrete jumps when execution occurs. This reflects a threshold mechanism in which inactivity is optimal until marginal gains justify trading. Close to delivery, imbalances are typically reduced but not eliminated. Large final trades are discouraged by temporary and permanent price impact, and the spread makes small corrective transactions expensive. Since the terminal production level $G(T)$ is uncertain at the time of the final trade, exact matching $X(T) = G(T)$ is not attainable. The learned control accounts for these effects and tolerates residual imbalance when further adjustment is not worth the cost.

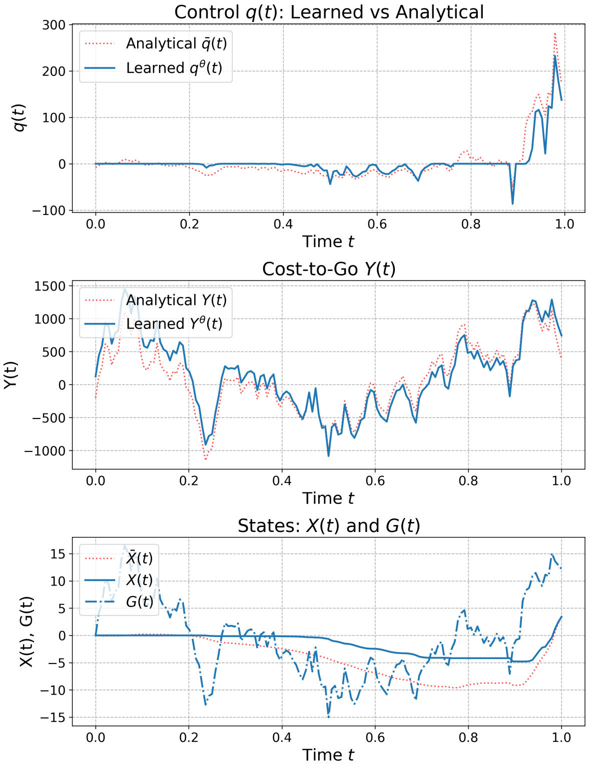


Figure 4: Comparison between learned and analytical control strategies under a bid–ask spread of $\psi = 5$ EUR/MWh. Solid lines represent the learned NN strategy, while dotted lines show the corresponding analytical optimal control (assuming zero-spread).

5.3 Performance on the Continuous Intraday Electricity Market

This section evaluates the performance of the neural network policy for optimal intraday trading on the German continuous electricity market. Performance is benchmarked against TWAP and the analytical control introduced in Section 4.3. TWAP is chosen since it is widely used and performs well in comparable execution settings (Kath Ziel, 2020), while VWAP cannot be applied here due to its dependence on unmodeled market volume profiles.

We start by comparing the strategies across different bid–ask spreads while holding all other parameters at their baseline values in Table 3. Since no analytical closed-form exists when spreads are present, the analytical policy corresponds to the zero–spread case. We then examine sensitivity to the imbalance penalty in an analogous manner, varying the penalty parameter while keeping all remaining parameters fixed to isolate its impact on execution performance.

Performance under Bid–Ask Spreads

We compare policy performance across different bid-ask spreads while keeping all remaining parameters fixed at their baseline values in Table 3. Table 1 reports results for the zero-spread case, the base-case half-spread of $\psi = 5$ EUR/MWh, and a high-spread setting of $\psi = 10$ EUR/MWh.

Table 1: Performance across different bid–ask spread levels ψ. Mean cost values shown with variance in parentheses.

Half-Spread ψ	**NN**	**Analytical**	**TWAP**
0	-351.75 (3312.41)	-353.52 (3311.57)	-3.67 (3295.40)
5	-114.93 (3291.21)	-96.84 (3292.31)	252.39 (3293.31)
10	86.01 (3252.44)	180.12 (3275.58)	508.46 (3293.78)

At $\psi = 0$, the NN policy closely approximates the analytical benchmark. Once spreads become positive, the NN policy consistently outperforms both the analytical control and TWAP. For instance, at $\psi = 5$, the NN achieves a mean cost of -114.93 EUR compared to -96.84 EUR for the analytical strategy and 252.39 EUR for TWAP. This margin widens when $\psi = 10$, where all strategies incur positive costs due to friction effects, but the NN remains the most cost-efficient. Differences in variance remain small across strategies, indicating that the performance gap is driven primarily by mean shifts rather than dispersion. Figure 5 illustrates cost distributions for spread $\psi = 5$.

Figure 6 shows how the NN adapts its trading intensity across spread regimes. At $\psi = 0$, control values are dispersed away from zero, whereas for 5 and 10, they cluster tightly around zero. This reflects learned selectivity in the presence of execution frictions: the NN refrains from trading unless the expected benefit exceeds spread costs, consistent with the performance trends in Table 1.

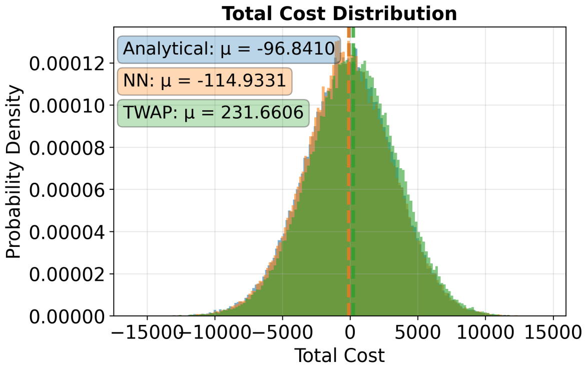


Figure 5: Empirical cost distribution across 1,000,000 simulation runs for the three different policies. Results are based on the base-case parameter set in Table 3.

Performance across different imbalance penalties

Falling into imbalance poses a significant financial risk for electricity producers, potentially leading to penalties or unfavorable settlement prices. Accurately managing this risk is therefore critical for effective trading. In this section, we compare the base-case scenario with an imbalance penalty of $\eta = 10$ EUR/MWh^2 to a stricter scenario with $\eta = 50$ EUR/MWh^2, in order to evaluate whether the NN policy can effectively minimize terminal inventory and maintain strong performance under heightened delivery requirements.

Table 2 presents the performance results under the two imbalance-penalty levels introduced above. The base-case scenario displays the same result as in previous sections, with the NN policy outperforming the two other policies. However, in the scenario with $\eta = 50$ EUR/MWh^2, the analytical policy outperforms the NN policy, achieving a mean total cost of 508.87 EUR compared to 573.89 EUR for the NN policy. The NN policy only marginally outperforms the TWAP strategy, which incurs a mean cost of 576.39 EUR. These results are somewhat surprising, as the NN policy is generally expected to outperform simple benchmarks.

Table 2: Performance across different imbalance penalty levels η. Mean values shown with variance in parentheses.

Penalty η	**NN**	**Analytical**	**TWAP**
10	-114.93 (3291.21)	-96.84 (3292.31)	252.39 (3293.31)
50	573.89 (3294.49)	508.87 (3292.31)	576.29 (3293.31)

Figure 7, which shows the state trajectories for both the analytical policy $\hat{X}(t)$ and the NN-based policy $X(t)$ under the two penalty levels, illustrates how the imbalance-penalty parameter influences the extent to which the policies attempt to match the generation forecast $G(t)$ at gate closure. In Figure 7 (A), corresponding to the base-case scenario with $\eta = 10$, neither the NN policy nor the analytical policy aligns closely with the generation forecast at the end of the horizon. This behavior can be attributed to the relatively low penalty level, which allows for some terminal deviation without incurring significant cost.

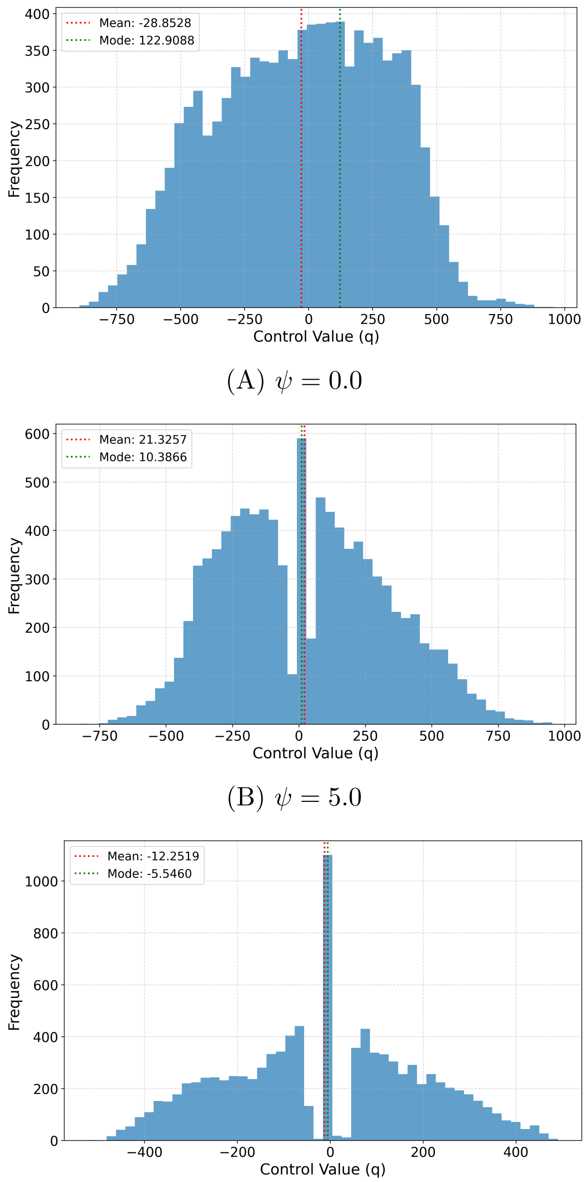
In contrast, Figure 7 (B), where $\eta = 50$ EUR/MWh^2, shows that both the NN and analytical policies more closely match the generation forecast at gate closure, driven by the stronger incentive to minimize imbalance. Despite this alignment, the NN policy performs worse than the analytical benchmark in terms of total mean cost.

6 Conclusion

This paper formulates optimal execution in continuous intraday electricity markets under forecast uncertainty, price impact, and bid–ask spreads within an Almgren–Chriss–type stochastic control framework. Once spreads are included, the associated Hamilton–Jacobi–Bellman equation is no longer available in closed form beyond restrictive cases, highlighting the need for numerical methods that extend classical execution models without relying on grid-based discretization.

We address this challenge using a deep BSDE approach based on a forward–backward representation of the value function, from which a deterministic feedback policy is obtained via Hamiltonian minimization using learned value gradients. A key feature of the approach is that it remains fully model-based while accommodating nonsmooth transaction costs, thereby extending the scope of analytically tractable execution models rather than replacing them with heuristic or model-free alternatives. Training enforces BSDE consistency, and incorporates an physics-informed residual penalty together with a direct objective term.

Numerically, the learned policy closely matches the analytical benchmark in the zero-spread case and exhibits sparse, threshold-based trading behavior once spreads are introduced. In empirically calibrated German continuous intraday market settings, it outperforms TWAP and the zero-spread analytical benchmark across the spread regimes studied, while under substantially higher imbalance penalties the analytical benchmark attains lower mean cost. Overall, the results indicate that deep BSDE methods provide a structured numerical extension of classical execution theory to more realistic intraday trading environments, while also highlighting sensitivity to calibration and training design in nonsmooth control problems.

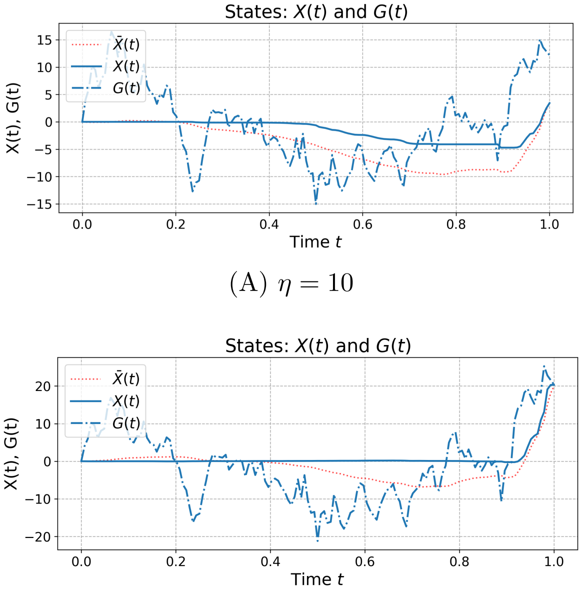


(A) $\psi = 0.0$

(B) $\psi = 5.0$

(C) $\psi = 10.0$

Figure 6: Distribution of control values $q(t)$ produced by the NN under varying half-spread levels ψ. Each histogram aggregates control values across all trajectories and time steps. Red dashed lines indicate the mean, green dashed lines the mode.



(A) $\eta = 10$

(B) $\eta = 50$

Figure 7: State trajectories for the analytical policy $\hat{X}(t)$, the NN policy $X(t)$, and the generation forecast $G(t)$ under two imbalance-penalty levels: (A) $\eta = 10$ and (B) $\eta = 50$. The higher penalty in panel (B) induces both control policies to align more closely with the generation forecast toward the end of the horizon, reflecting the stronger incentive to avoid imbalance costs.

References

Aïd R, Gruet P, Pham H (2016) An optimal trading problem in intraday electricity markets. *Mathematics and Financial Economics* 10(1):49–85, URL http://dx.doi.org/10.1007/s11579-015-0150-8.

Almgren R (2012) Optimal trading with stochastic liquidity and volatility. *SIAM J. Financial Math.* 3:163–181, URL http://dx.doi.org/10.1137/090763470.

Almgren R, Chriss N (2001) Optimal execution of portfolio transactions. *The Journal of Risk* 3(2):5–39, URL https://www.smallake.kr/wp-content/uploads/2016/03/optliq.pdf.

Almgren RF (2003) Optimal execution with nonlinear impact functions and trading-enhanced risk. *Applied Mathematical Finance* 10(1):1–18, URL http://dx.doi.org/10.1080/135048602100056.

Angel JJ, Harris LE, Spatt CS (2011) Equity trading in the 21st century. *The Quarterly Journal of Finance* 1(1):1–53, URL http://dx.doi.org/10.1142/S2010139211000012.

Ata B, Harrison JM, Si N (2025) Drift control of high-dimensional reflected Brownian motion: A computational method based on neural networks. *Stochastic Systems* 15(2):111–146, URL http://dx.doi.org/10.1287/stsy.2023.0044.

Beck C, E W, Jentzen A (2019) Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations. *Journal of Nonlinear Science* 29(4):1563–1619, ISSN 1432-1467, URL http://dx.doi.org/10.1007/s00332-018-9525-3.

Bertsimas D, Lo AW (1998) Optimal control of execution costs. *Journal of Financial Markets* 1(1):1–50, ISSN 1386-4181, URL http://dx.doi.org/https://doi.org/10.1016/S1386-4181(97)00012-8.

Bielecki MF, Kemper JJ, Acker TL (2010) A methodology for comprehensive characterization of errors in wind power forecasting. *Proceedings of ASME 2010 4th International Conference on Energy Sustainability*, volume 2, 867–876, URL http://dx.doi.org/10.1115/ES2010-90381.

Bouchaud JP (2010) Price impact. Cont R, ed., *Encyclopedia of Quantitative Finance* (John Wiley & Sons), URL http://dx.doi.org/10.1002/9780470061602.eqf18006.

Breuer A, Burghof HP, Stitz J (2011) Order dynamics in a high-frequency trading environment. *SSRN Electronic Journal* URL http://dx.doi.org/10.2139/ssrn.1927276.

Bulthuis B, Concha J, Leung T, Ward B (2017) Optimal execution of limit and market orders with trade director, speed limiter, and fill uncertainty. *International Journal of Financial Engineering* 4(02n03):1750020, URL http://dx.doi.org/10.1142/S2424786317500207.

Bush J (2023) Trading in the continuous intraday market: how does it work? URL https://modoenergy.com/research/continuous-intraday-trading-wholesale-epex-n2ex-volume-battery-energy-storage-prices, accessed: 2025-05-22.

Cartea A, Jaimungal S (2016) Incorporating order-flow into optimal execution. *Mathematics and Financial Economics* 10(4):339–364, URL http://dx.doi.org/10.1007/s11579-016-0162-z.

Cartea Á, Jaimungal S, Penalva J (2015) *Algorithmic and High-Frequency Trading* (Cambridge University Press), ISBN 9781107091146, URL https://books.google.no/books?id=5dMmCgAAQBAJ.

Chatziandreou K, Karbach S (2025) Optimal execution in intraday energy markets under Hawkes processes with transient impact. URL https://arxiv.org/abs/2504.10282.

Cheng X, Di Giacinto M, Wang TH (2019) Optimal execution with dynamic risk adjustment. *Journal of the Operational Research Society* 70(10):1662–1677, URL http://dx.doi.org/10.1080/01605682.2019.1644143.

Cheng X, Giacinto MD, and THW (2017) Optimal execution with uncertain order fills in Almgren–Chriss framework. *Quantitative Finance* 17(1):55–69, URL http://dx.doi.org/10.1080/14697688.2016.1185531.

Ciccone M, Gallieri M, Masci J, Osendorfer C, Gomez FJ (2018) NAIS-Net: Stable deep networks from non-autonomous differential equations. *32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.* (Neural Information Processing Systems Foundation), URL https://proceedings.neurips.cc/paper_files/paper/2018/file/7bd28f15a49d5e5848d6ec70e584e625-Paper.pdf.

E W, Han J, Jentzen A (2017) Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in mathematics and statistics* 5(4):349–380, URL http://dx.doi.org/10.1007/s40304-017-0117-6.

E W, Han J, Jentzen A (2022) Algorithms for solving high dimensional PDEs: from nonlinear Monte Carlo to machine learning. *Nonlinearity* 35(1):278–310, ISSN 0951-7715, URL http://dx.doi.org/10.1088/1361-6544/ac337f.

EPEX SPOT (2021) New record volume traded on EPEX SPOT in 2020. URL https://www.epexspot.com/en/news/new-record-volume-traded-epex-spot-2020, accessed: 2025-12-19.

EPEX SPOT (2025) Annual power trading results 2024. Technical report, EPEX SPOT, URL https://www.epexspot.com/sites/default/files/download_center_files/2025-01-28_EPEX%20SPOT_Annual%20Power%20Trading%20Results%202024_finaldraft_0.pdf, accessed: 2025-12-19.

Féron O, Tankov P, Tinsi L (2021) Price formation and optimal trading in intraday electricity markets. Lasaulce S, Mertikopoulos P, Orda A, eds., *Network Games, Control and Optimization*, 294–305

(Springer International Publishing), URL http://dx.doi.org/10.1007/978-3-030-87473-5_26.

Fleming M (2003) Measuring treasury market liquidity. *Economic Policy Review* 9:83–108, URL http://dx.doi.org/10.2139/ssrn.276289.

Forsyth P, Kennedy J, Tse S, Windcliff H (2012) Optimal trade execution: A mean quadratic variation approach. *Journal of Economic Dynamics and Control* 36(12):1971 – 1991, URL http://dx.doi.org/10.1016/j.jedc.2012.05.007, cited by: 84.

Frey S, Sandås P (2017) The impact of iceberg orders in limit order books. *The Quarterly Journal of Finance* 07(03):1750007, URL http://dx.doi.org/10.1142/S2010139217500070.

Gatheral J, Schied A (2012) Optimal trade execution under Geometric Brownian motion in the Almgren and Chriss framework. *International Journal of Theoretical and Applied Finance* 14:353–368, URL http://dx.doi.org/10.1142/S0219024911006577.

Glas S, Kiesel R, Kolkmann S, Kremer M, Luckner N, Ostmeier L, Urban K, Weber C (2020) Intraday renewable electricity trading: advanced modeling and numerical optimal control. *Journal of Mathematics in Industry* 10(3), URL http://dx.doi.org/10.1186/s13362-020-0071-x.

Güler B, Laignelet A, Parpas P (2019) Towards robust and stable deep learning algorithms for forward backward stochastic differential equations. URL https://arxiv.org/abs/1910.11623.

Han J, Jentzen A, E W (2018) Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences* 115(34):8505–8510, URL http://dx.doi.org/10.1073/pnas.1718942115.

Harris L (2003) *Trading and Exchanges: Market Microstructure for Practitioners* (Oxford University Press).

Hendricks D, Wilcox D (2014) A reinforcement learning extension to the almgren-chriss model for optimal trade execution. *IEEE/IAFE Conference on Computational Intelligence for Financial Engineering, Proceedings (CIFEr)* URL http://dx.doi.org/10.1109/CIFEr.2014.6924109.

Henry-Labordere P (2017) Deep primal-dual algorithm for BSDEs: Applications of machine learning to CVA and IM. *SSRN Electronic Journal* URL http://dx.doi.org/10.2139/ssrn.3071506.

Huré C, Pham H, Warin X (2020) Deep backward schemes for high-dimensional nonlinear PDEs. *Mathematics of Computation* 89(324):1547–1579, URL http://dx.doi.org/doi.org/10.1090/mcom/3514.

Intercontinental Exchange (2025) ICE: Transforming what's possible. URL https://www.ice.com/, accessed: 2025-12-19.

Ji S, Peng S, Peng Y, Zhang X (2022) Solving stochastic optimal control problem via stochastic maximum principle with deep learning method. *Journal of Scientific Computing* 93(1):30, URL http://dx.doi.org/10.1007/s10915-022-01979-5.

Katarzyna M, Nitka W, Weron T (2019) Day-ahead vs. intraday—forecasting the price spread to maximize economic benefits. *Energies* 12:631, URL http://dx.doi.org/10.3390/en12040631.

Kath C, Ziel F (2020) Optimal order execution in intraday markets: Minimizing costs in trade trajectories. URL https://arxiv.org/abs/2009.07892.

Kendall A, Gal Y, Cipolla R (2018) Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7482–7491 (IEEE).

Kwok Y, Lau K (2005) Optimal execution strategy of liquidation. *Journal of Industrial and Management Optimization* 2, URL http://dx.doi.org/10.2139/ssrn.717401.

Lin S, Beling P (2020) An end-to-end optimal trade execution framework based on proximal policy optimization. 4498–4504, URL http://dx.doi.org/10.24963/ijcai.2020/627.

Marzo M, Ritelli D, Zagaglia P (2011) Optimal trading execution with nonlinear market impact: An alternative solution method. URL https://arxiv.org/abs/1111.6826.

Nevmyvaka Y, Feng Y, Kearns M (2006) Reinforcement learning for optimized trade execution. volume 2006, 673–680, URL http://dx.doi.org/10.1145/1143844.1143929.

Ning B, Lin FHT, Jaimungal S (2021) Double deep q-learning for optimal execution. *Applied Mathematical Finance* 28(4):361–380, URL http://dx.doi.org/10.1080/1350486X.2022.2077783.

Nord Pool (2025) Nord Pool: Europe's leading power market. URL https://www.nordpoolgroup.com/, accessed: 2025-12-192.

Ozenbas D, Pagano MS, Schwartz RA, Weber BW (2022) *Liquidity, Trading, and Price Determination in Equity Markets: A Finance Course Application* (Cham: Springer International Publishing), ISBN 978-3-030-74817-3, URL http://dx.doi.org/10.1007/978-3-030-74817-3_2.

Pereira M, Wang Z, Exarchos I, Theodorou E (2019) Learning deep stochastic optimal control policies using forward-backward SDEs. URL http://dx.doi.org/10.15607/RSS.2019.XV.070.

Raissi M, Perdikaris P, Karniadakis GE (2019) Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* 378:686–707, URL http://dx.doi.org/10.1016/j.jcp.2018.10.045.

Schied A (2013) Robust strategies for optimal order execution in the almgren-chriss framework. *Applied Mathematical Finance* 20(3):264 – 286, URL http://dx.doi.org/10.1080/1350486X.2012.683963.

Shilova A, Delliaux T, Preux P, Raffin B (2024) Learning HJB viscosity solutions with PINNs for continuous-time reinforcement learning. *Inria Research Report* (RR-9541), available at https://inria.hal.science/hal-04445160v1.

Sirignano J, Spiliopoulos K (2018) DGM: A deep learning algorithm for solving partial differential equations. *Journal of computational physics* 375:1339–1364.

Tan Z, Tankov P (2018) Optimal trading policies for wind energy producer. *SIAM Journal on Financial Mathematics* 9(1):315–346, URL `http://dx.doi.org/10.1137/16M1093069`.

Vaes J, Hauser R (2022) Optimal trade execution with uncertain volume target. *Journal of Computational Finance* ISSN 1755-2850, URL `http://dx.doi.org/10.21314/jcf.2022.018`.

Yong J, Zhou XY (1999) *Stochastic Controls: Hamiltonian Systems and HJB Equations*, volume 43 of *Applications of Mathematics* (Springer), ISBN 978-0387987234.

7 Analytical solution

To benchmark the neural network-based control policy, we consider the closed-form solution derived in Aïd et al. (2016) for the HJB equation (29) under the simplified model assumptions. We postulate a quadratic ansatz for the value function:

$$\begin{aligned} v(t,\boldsymbol{y}) =& A(\tau)(X-G)^2 + B(\tau)P^2 + F(\tau)(X-G)P + K(\tau), \\ \tau =& T-t, \end{aligned} \tag{25}$$

with initial condition $v(T,\boldsymbol{y}) = \frac{\eta}{2}(X-G)^2$, implying:

$$A(0) = \frac{\eta}{2}, \quad B(0) = 0, \quad F(0) = 0, \quad K(0) = 0. \tag{26}$$

To determine the coefficient functions $A(\tau)$, $B(\tau)$, $F(\tau)$, $K(\tau)$, the ansatz is substituted into the HJB equation, and terms are matched. This yields a system of ODEs that must be satisfied to solve the PDE:

$$\begin{aligned} A'(\tau) &= -\frac{1}{4\gamma}\left(-2A(\tau) + \nu F(\tau)\right)^2, \\ B'(\tau) &= -\frac{1}{4\gamma}\left(2\nu B(\tau) - F(\tau) + 1\right)^2, \\ F'(\tau) &= -\frac{1}{2\gamma}\left(-2A(\tau) + \nu F(\tau)\right)\left(2\nu B(\tau) - F(\tau) + 1\right), \\ K'(\tau) &= \Big(\sigma_P^2 B(\tau) + \sigma_G^2 A(\tau) + \rho\sigma_P\sigma_G F(\tau)\Big), \end{aligned} \tag{27}$$

Solving this ODE system explicitly, the coefficients become:

$$\begin{aligned} A(\tau) &= \frac{\eta\left(\frac{\nu}{2}\tau + \gamma\right)}{(\eta+\nu)\tau + 2\gamma}, \\ B(\tau) &= -\frac{\tau}{2\left[(\eta+\nu)\tau + 2\gamma\right]}, \\ F(\tau) &= -\frac{\eta\tau}{(\eta+\nu)\tau + 2\gamma}, \\ K(\tau) &= \gamma\frac{\sigma_P^2 + \sigma_G^2\eta^2 - 2\rho\sigma_P\sigma_G\eta}{(\eta+\nu)^2}\ln\left(1 + \frac{(\eta+\nu)\tau}{2\gamma}\right) \\ &\quad + \frac{\sigma_G^2\eta\nu + 2\rho\sigma_P\sigma_G\eta - \sigma_P^2}{2(\eta+\nu)}\tau. \end{aligned}$$

Inserting these coefficients in (25) and using the resulting v in (22), we obtain the following closed-form expression for the optimal control:

$$\bar{q}(t,\boldsymbol{y}) = \frac{\eta\,(G-X) - P}{(\eta+\nu)(T-t) + 2\gamma}. \tag{28}$$

and inserting this minimizer in (12) gives the resulting HJB equation:

$$
\begin{aligned}
0 = {} & \partial_t v(t, \boldsymbol{y}) \\
& + \frac{1}{2}\sigma_P^2(t)\, \partial_{PP}^2 v + \frac{1}{2}\sigma_G^2(t)\, \partial_{GG}^2 v + \rho\, \sigma_P(t)\sigma_G(t)\, \partial_{PG}^2 v \\
& - \frac{1}{4\gamma}\left(P + \partial_X v + \nu\, \partial_P v\right)^2,
\end{aligned} \tag{29}
$$

In this case, the optimal control $\bar{q}$ is linear and Lipschitz continuous in the state variables, uniformly in time. Given any initial state (X_t, P_t, G_t) at time t, the forward dynamics admit a unique strong solution under the feedback control $\bar{q}$. Moreover, the process satisfies the integrability condition $\mathbb{E}[\int_t^T |\bar{q}_s|^2 ds] < \infty$, which ensures that $\bar{q} \in \mathcal{A}_t$. Together with the smoothness of the value function $v(t, \boldsymbol{y}) \in C^{1,2}$, these conditions guarantee the applicability of the verification theorem, thereby establishing that $v = V$ and that $\bar{q}$ is indeed the optimal control. This closed-form solution serves as a benchmark for evaluating the performance of the neural network-based control policy $q^\theta(t, \boldsymbol{y})$ in more general settings.

7.1 Time-Dependent Volatility

We now generalize the analytical solution by allowing time-varying volatilities in both the price and generation dynamics. Specifically, we assume the volatilities evolve linearly in time-to-maturity:

$$
\sigma_P^2(\tau) = \alpha_P \tau + \beta_P, \quad \sigma_G^2(\tau) = \alpha_G \tau + \beta_G. \tag{30}
$$

This models increasing uncertainty in the price and decreasing uncertainty in generation as delivery approaches, which is consistent with empirical observations in electricity markets.

Under this extension, the coefficient functions $A(\tau)$, $B(\tau)$, $F(\tau)$ remain unchanged from the constant volatility case. However, the term $K(\tau)$ satisfies the ODE:

$$
K'(\tau) = B(\tau)\, \sigma_P^2(\tau) + A(\tau)\, \sigma_G^2(\tau) + \rho\, F(\tau)\, \sigma_P(\tau)\, \sigma_G(\tau). \tag{31}
$$

which has the solution:

$$
\begin{aligned}
K(\tau) = \int_0^\tau \Big[& B(s)\, (\alpha_P s + \beta_P) + A(s)\, (\alpha_G s + \beta_G) \\
& + \rho\, F(s)\, \sqrt{\alpha_P s + \beta_P}\, \sqrt{\alpha_G s + \beta_G} \Big] ds.
\end{aligned} \tag{32}
$$

In general, the integral expression for $K(\tau)$ does not admit a closed-form solution due to the nonlinearity introduced by the square roots, and must be evaluated numerically for comparison with the learned value function.

As in the constant volatility case, the value function $v(t, \boldsymbol{y})$ constructed via the same quadratic ansatz satisfies the terminal condition and admits continuous derivatives. Since the optimal control derived from it remains admissible and the verification theorem still applies, it follows that $v = V$ and the associated $\bar{q}$ remains optimal under time-dependent volatility.

7.2 Optimal Control with Bid-Ask Spread

We now relax the assumption of zero bid–ask spread by letting $\psi(t) > 0$. The spread introduces an additional term in the execution price:

$$\tilde{P}(t, \boldsymbol{y}, q) = P + \psi(t)\,\mathrm{sign}(q) + \gamma q,$$

leading to a nonsmooth term in the Hamiltonian:

$$H(t, \boldsymbol{y}, q, \nabla v) = \gamma q^2 + (P + \partial_X v + \nu\,\partial_P v)\,q + \psi(t)|q|.$$

Define the *marginal execution value*:

$$\Lambda(t, \boldsymbol{y}) := P + \partial_X v(t, \boldsymbol{y}) + \nu\,\partial_P v(t, \boldsymbol{y}), \tag{33}$$

so that the Hamiltonian becomes:

$$H(t, \boldsymbol{y}, q, \nabla v) = \gamma q^2 + \Lambda(t, \boldsymbol{y})q + \psi(t)|q|.$$

Minimizing over $q \in \mathbb{R}$ yields the piecewise-optimal control:

$$\bar{q}(t, \boldsymbol{y}) = \begin{cases} -\frac{1}{2\gamma}\left(\Lambda(t, \boldsymbol{y}) + \psi(t)\right), & \text{if } \Lambda(t, \boldsymbol{y}) > \psi(t), \\ -\frac{1}{2\gamma}\left(\Lambda(t, \boldsymbol{y}) - \psi(t)\right), & \text{if } \Lambda(t, \boldsymbol{y}) < -\psi(t), \\ 0, & \text{otherwise.} \end{cases} \tag{34}$$

Looking at the marginal execution value (33), the spot price P reflects the immediate trading revenue, $\partial_X v$ measures the marginal value of adjusting the inventory, and $\nu\,\partial_P v$ accounts for the expected impact of permanent price changes on future value. Their sum determines whether the agent finds it beneficial to trade, relative to the cost of crossing the spread. This defines a no-trade region $|\Lambda(t, \boldsymbol{y})| \leq \psi(t)$, where the marginal benefit of trading does not exceed the cost of crossing the spread.

7.2.1 Neural approximation of optimal control.

Since the control policy must remain differentiable for gradient-based training, we adopt a smoothed approximation using sigmoid transitions:

$$\begin{aligned} \bar{q}^{\theta}_{\varepsilon}(t, \boldsymbol{y}) = \; -\frac{1}{2\gamma}\Big[&\left(\Lambda^\theta + \psi\right) \mathrm{sigmoid}_\varepsilon(\Lambda^\theta - \psi) \\ &+ \left(\Lambda^\theta - \psi\right) \mathrm{sigmoid}_\varepsilon(-\Lambda^\theta - \psi)\Big], \end{aligned} \tag{35}$$

where $\Lambda^\theta = P + \partial_X Y^\theta + \nu\partial_P Y^\theta$, and $\mathrm{sigmoid}_\varepsilon(z) = \frac{1}{1+e^{-z/\varepsilon}}$ is a scaled sigmoid function with smoothness parameter ε. As $\varepsilon \to 0$, the smoothed policy $\bar{q}^\theta_\varepsilon$ converges pointwise to the discontinuous piecewise-optimal control $\bar{q}$ given in (24).

Figure 8 illustrates the behavior of the control policy as a function of the marginal execution

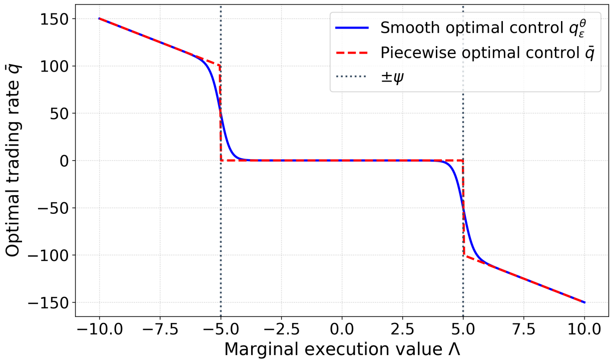


Figure 8: Comparison between the smooth neural-network-based control policy $\bar{q}_\varepsilon^\theta(t, \boldsymbol{y})$ and the piecewise-optimal control $\bar{q}(t, \boldsymbol{y})$. The smoothed policy approximates the no-trade zone $|\Lambda| \leq \psi$ using sigmoid transitions. The spread used here is $\psi = 5$, with temporary impact $\gamma = 0.05$, and smoothing parameter $\varepsilon = 0.2$.

value Λ. For $|\Lambda| \leq \psi = 5$, the piecewise-optimal control $\bar{q}$ drops sharply to zero, defining a no-trade region where the cost of crossing the spread outweighs the execution benefit. Outside this region, the optimal trading rate increases linearly with Λ, with slope determined by γ. The smoothed approximation $\bar{q}_\varepsilon^\theta$ transitions gradually between regimes. At the threshold $|\Lambda| = \psi$, the control becomes $\bar{q} = \pm\Lambda/\gamma$, which yields exactly ± 100 with the parameters $\psi = 5$, $\gamma = 0.05$.

7.3 Epsilon Scheduling

To progressively approximate the viscosity solution, we implement a non-adaptive scheduler for the smoothing parameter ε. Starting from a stable initialization ε_0, the scheduler updates ε every N_u training steps by applying a fixed decay factor $k_\varepsilon < 1$:

$$\varepsilon_{n+1} = \begin{cases} k_\varepsilon \cdot \varepsilon_n, & \text{if } (n+1) \mod N_u = 0, \\ \varepsilon_n, & \text{otherwise.} \end{cases}$$

where n is the current training epoch, $k_\varepsilon \in (0, 1)$ is a fixed decay factor, and N_u the interval between updates. This discretized decay scheme gradually reduces the smoothing parameter ε, allowing the network to first learn a smooth approximation and then progressively converge toward the true nonsmooth optimal policy. This approach follows the methodology proposed by Shilova et al. (2024), who demonstrate that PINNs trained with a scheduled decrease in the smoothing parameter can recover the unique viscosity solution to HJB equations arising in stochastic control problems with non-differentiable cost structures.

The piecewise control $\bar{q}(t, \boldsymbol{y})$ derived above minimizes the Hamiltonian pointwise but is discontinuous due to the spread term. Although this violates classical differentiability, the corresponding value function $v(t, \boldsymbol{y})$ remains continuous and satisfies the HJB equation in the viscosity sense. The control $\bar{q}$ is measurable, adapted, and satisfies the integrability conditions imposed on admissible controls, i.e., $\bar{q} \in \mathcal{A}$. By the vertification theorem, this ensures that $v = V$ and that the feedback control $\bar{q}$ remains globally optimal despite the nonsmooth structure of the cost function.

8 Estimated Parameters

8.1 Permanent Impact Calibration

The estimation of the permanent impact coefficient used in Section 5.1 is based on signed trade data for hourly products from the German intraday electricity market on EPEX SPOT. The dataset spans from 2024.11.29 to 2025.06.26.

The coefficient in the permanent impact function $g(t, q(t, \boldsymbol{y}))$ is estimated following the regression based approach of Glas et al. (2020), which builds on related methodologies from equity markets proposed by Cartea Jaimungal (2016). Glas et al. (2020) assume a time invariant linear permanent impact function and estimate ν by analyzing the long term price difference between two mid price proxies regressed against net order flow.[3]

The estimation rests on the assumption that the DAM auction price provides an unbiased estimate of the fundamental value at the start of continuous trading. Accordingly, the opening intraday mid price should lie close to the DAM price. In contrast, VWAP, which is treated as an unbiased proxy for the average mid price throughout the trading session, may reflect the cumulative effect of net order flow and thus deviate from the DAM price by the end of the session.

Taking the difference $\text{VWAP}_d - \text{DA}_d$ for each delivery day d gives a measure of the intraday price impact due to trading. After removing the 5% most extreme deviations in absolute value, this difference is regressed against the corresponding net order flow rate, defined as total net traded volume divided by the trading window length. The resulting scatterplot and fitted regression are shown in Figure 9.

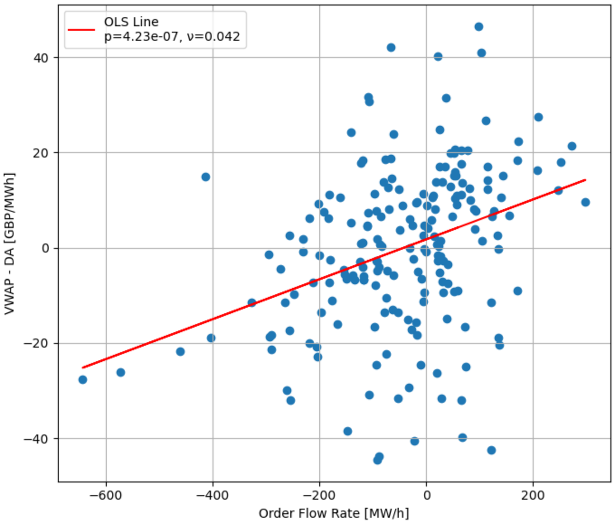


Figure 9: Scatterplot of price deviation $\text{VWAP}_d - \text{DA}_d$ versus net order flow rate for the hourly product with delivery at 12:00 CET. Each point corresponds to a delivery day between December 2024 and the end of June. Outliers (top 5% in absolute deviation) are removed prior to regression. The red line shows the fitted OLS regression according to 36, yielding an estimated coefficient $\nu = 0.042$ EUR/(MWh)^2.

[3] Net order flow over a period is computed as buyer initiated traded volume minus seller initiated traded volume.

Table 3: Model calibration for hourly German intraday products

Symbol	**Name**	**Unit**	**Value**
N	Time steps	–	144
P_0	Initial price	EUR/MWh	80
X_0	Initial position	MWh	0
ψ	Half-spread	EUR/MWh	5
γ	Temporary impact	EUR/(MWh)^2	0.01
ν	Permanent impact	EUR/(MWh)^2	0.042
μ_P	Price drift	EUR/MWh/h	0
μ_G	Generation drift	MWh/h	0
$\sigma_P^2(t)$	Price variance	–	increasing (40→120)
$\sigma_G^2(t)$	Generation variance	–	decreasing (1600→900)
ρ	Correlation	–	–0.06
η	Imbalance penalty	EUR/MWh^2	10

The regression model is

$$\mathrm{VWAP}_d - \mathrm{DA}_d = \nu \cdot \left(\frac{\mathrm{NOF}_d}{\mathrm{horizon}_d} \right) + \varepsilon_d, \tag{36}$$

where VWAP_d is the volume weighted average intraday price for day d, DA_d is the corresponding DAM auction price, and NOF_d is the net order flow in MWh. The term $\mathrm{horizon}_d$ denotes the number of hours the product was actively traded that day. The slope ν captures the permanent price impact per unit of average net order flow rate, while the residual ε_d accounts for unexplained variation.

Applying this regression to the 12:00 delivery product over the sample period yields an estimate of $\nu = 0.042$ EUR/(MWh)^2, statistically significant at the 1% level. In line with Glas et al. (2020), the mid price drift μ_P is set to zero once ν has been calibrated.

9 Performance Optimization

All experiments are conducted on NTNU's IDUN high-performance computing cluster, utilizing four NVIDIA H100 GPUs in parallel. The training framework is implemented using PyTorch with support for fully data-parallel training via `torch.nn.parallel.DistributedDataParallel`, enabling efficient scaling across devices.

To maximize the diversity and coverage of the state space, the model is trained on a large batch of independent stochastic simulation paths. Specifically, each training step uses a total batch size of 16,384 trajectories, distributed evenly across the four GPUs (4,096 simulations per device). This configuration balances memory constraints with the goal of exposing the model to a broad range of input states, enhancing generalization to unseen scenarios.

By simulating large batches in parallel, the method efficiently captures the stochasticity of the dynamics and improves robustness of the learned control policy. Larger batch sizes also reduce the variance of gradient estimates and lead to more stable convergence during training.

In addition to speed and scalability, this setup ensures that the learned neural control policy generalizes well across paths, state realizations, and discretization levels. The simulation-based nature of the FBSDE framework benefits substantially from high-throughput sampling, making distributed training an important component for achieving both computational efficiency and policy accuracy.