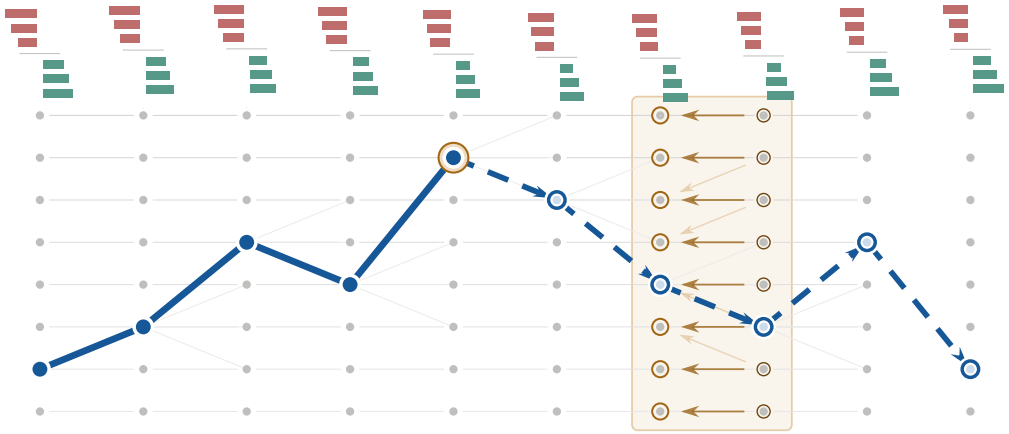
CHALMERS
UNIVERSITY OF TECHNOLOGY



A High-Frequency Algorithm for Battery Storage Trading on Continuous Intraday Electricity Markets

Replication, Optimization, and Empirical Comparison of the Rolling-Intrinsic Policy Across Three European Intraday Markets

Master's thesis in Engineering Mathematics and Computational Science

KARL HALLSTRÖM, NICLAS LINDMARK

DEPARTMENT OF MATHEMATICAL SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

A High-Frequency Algorithm for Battery Storage Trading on Continuous Intraday Electricity Markets

Replication, Optimization, and Empirical Comparison of the Rolling-Intrinsic Policy Across Three European Intraday Markets

KARL HALLSTRÖM, NICLAS LINDMARK

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Chalmers University of Technology
Gothenburg, Sweden 2026

A High-Frequency Algorithm for Battery Storage Trading on Continuous Intraday Electricity Markets
Replication, Optimization, and Empirical Comparison of the Rolling-Intrinsic Policy Across Three European Intraday Markets
KARL HALLSTRÖM, NICLAS LINDMARK

Supervisors: Michael Snellman, Flower
Carl Lindberg, Department of Mathematical Sciences
Examiner: Peter Helgesson, Department of Mathematical Sciences

Master's Thesis 2026
Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Schematic of one rolling-intrinsic dynamic-programming re-solve. The top row shows stylized order-book snapshots for successive contracts, with red asks above the mid-line and green bids below. The gray lattice is the discretized storage-state grid. Amber arrows show one full backward-pass step, where values in a later column are used to compute the preceding value column. The solid blue path is the realized state-of-charge history up to the current decision point; the dashed blue path is the newly planned trajectory for the remaining horizon.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2026

A High-Frequency Algorithm for Battery Storage Trading on Continuous Intraday Electricity Markets
Replication, Optimization, and Empirical Comparison of the Rolling-Intrinsic Policy Across Three European Intraday Markets
KARL HALLSTRÖM, NICLAS LINDMARK
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Battery storage earns revenue on continuous intraday electricity markets only with a trading policy that reoptimizes as the order book moves. This thesis replicates the rolling-intrinsic dynamic-programming (DP) policy of Schaurecker et al. and adapts it to Flower's operational setting. Schaurecker et al. study hourly products on the German EPEX market. We move the rolling intrinsic to quarter-hour contracts and backtest it on roughly a year of Nord Pool order-book data in each of three bidding zones: Sweden SE3, Sweden SE4, and Germany Amprion. The quarter-hour switch makes the DP backward pass roughly 64 times heavier than the hourly reference, because the number of stages, the action set, and the storage grid each grow fourfold. A C++/OpenMP kernel keeps event-resolution backtests tractable, reaching about 2.6 ms per re-solve on the SE3 benchmark. On a single 10 MWh, one-hour battery, re-solving on every order-book event earns roughly EUR 0.75M on Sweden SE3 and EUR 0.93M on Sweden SE4 over twelve-month windows, and EUR 1.08M on Germany Amprion over a longer fourteen-month window, and beats every fixed re-solve timeline tested in all three zones. Over the window common to all three zones, SE4 earns the most, ahead of Germany Amprion and then SE3; Germany's larger raw total reflects only its longer window. Set against the dominant ancillary-market alternative, the rolling intrinsic earns more on only 8% of days in Sweden SE3, 4% in Sweden SE4, and 10% in Germany Amprion. It is therefore a conditional trading policy for the battery, chosen day by day against ancillary opportunities, rather than a default replacement for ancillary-market participation.

Keywords: battery energy storage, continuous intraday market, rolling intrinsic, dynamic programming, order book, Nord Pool, quarter-hour contracts, high-performance computing, ancillary services.

Acknowledgements

This thesis was carried out at Flower in collaboration with the Department of Mathematical Sciences at Chalmers University of Technology.

We thank Michael Snellman, our supervisor at Flower, together with Marcus Evholt and Pablo Fernández de Salas, for proposing the problem, providing the data and operational context, and for many helpful discussions over the course of the project. We are also grateful to the wider Trading and Data Science team at Flower, whose feedback across many internal demo sessions helped shape the work.

We thank Carl Lindberg, our supervisor at the Department of Mathematical Sciences, and Peter Helgesson, our examiner, for their guidance and feedback.

Finally, we thank David Schaurecker, David Wozabal, Nils Löhndorf, and Thorsten Staake, whose work is the foundation of this thesis. The public repository they share alongside it was a valuable source of inspiration for our own implementation.

Karl Hallström and Niclas Lindmark
Gothenburg, May 2026

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

ACER	Agency for the Cooperation of Energy Regulators
aFRR	automatic Frequency Restoration Reserve
AVX	Advanced Vector Extensions
AVX2	Advanced Vector Extensions 2
BESS	Battery Energy Storage System
BLAS	Basic Linear Algebra Subprograms
CM	Capacity Market
DP	Dynamic Programming
EPEX	European Power Exchange
FCR	Frequency Containment Reserve
FMA	Fused Multiply-Add
FOK	Fill-or-Kill
GCC	GNU Compiler Collection
HPC	High-Performance Computing
IEA	International Energy Agency
IEEE	Institute of Electrical and Electronics Engineers
LAPACK	Linear Algebra Package
LOB	Limit Order Book
LP	Linear Program
MEF	Marginal Emission Factor
mFRR	manual Frequency Restoration Reserve
MILP	Mixed-Integer Linear Programming
MIP	Mixed-Integer Program
MKL	Math Kernel Library
MW	Megawatt
MWh	Megawatt-hour

NUMA	Non-Uniform Memory Access
OpenMP	Open Multi-Processing
P&L (or PnL)	Profit and Loss
PGO	Profile-Guided Optimization
PPA	Power Purchase Agreement
SDDP	Stochastic Dual Dynamic Programming
SE1	Swedish bidding zone 1 (northern Sweden)
SE2	Swedish bidding zone 2 (north-central Sweden)
SE3	Swedish bidding zone 3 (central Sweden)
SE4	Swedish bidding zone 4 (southern Sweden)
SIMD	Single Instruction, Multiple Data
SoC	State of Charge
TLB	Translation Lookaside Buffer
TSO	Transmission System Operator
ULP	Unit in the Last Place
VWAP	Volume-Weighted Average Price
XBID	Cross-Border Intraday

Nomenclature

Below is the nomenclature of indices, sets, parameters, variables, and functions used throughout this thesis. Generic textbook notation introduced only when reviewing linear programming, mixed-integer programming, and dynamic programming (for example the standard-form matrices of a linear program or the abstract Bellman primitives) is defined where it appears and is not repeated here.

Indices

t	Index for a tradable product, equivalently the delivery period and the DP stage
t^*	Decision time at which the intrinsic problem is (re-)solved
i, j	Indices for individual orders in an order book

Sets

$\mathcal{T}$	Set of tradable products, $\mathcal{T} = \{1, \dots, T\}$; the stages of the DP
$\mathcal{O}_t$	Order book of product t, $\mathcal{O}_t = \mathcal{O}_t^+ \cup \mathcal{O}_t^-$
$\mathcal{O}_t^+$	Active asks (sell orders) for product t
$\mathcal{O}_t^-$	Active bids (buy orders) for product t
G	Discretized storage grid, $G \subset [0, \bar{s}]$ with $\lvert G \rvert = m$ points
$\mathcal{X}_t$	Set of feasible actions k_t at stage t, $\mathcal{X}_t(s, f_t^0)$

Parameters

T	Number of tradable products (DP stages)
u	Minimum tradable lot, used as the action step (0.025 MWh by default)
κ	Action-step multiplier; the position f_t is discretized in steps of κu, $\kappa \in \mathbb{N}$

P_i	Limit price of order i
Q_i	Quantity (size) of order i
η^+	Charging efficiency, $\eta^+ \in (0, 1]$
η^-	Discharging efficiency, $\eta^- \in (0, 1]$
η_{RT}	Round-trip efficiency, $\eta_{\mathrm{RT}} = \eta^+\eta^-$
ν_{trade}	Trading cost per unit volume (€/MWh)
ν_{deg}	Battery degradation cost per unit volume (€/MWh)
ν	Combined per-unit trade cost, $\nu = \nu_{trade} + \nu_{deg}$
$\underline{f}$	Lower bound on the future position (discharge power limit)
$\bar{f}$	Upper bound on the future position (charge power limit)
$\bar{s}$	Storage capacity (upper bound on the storage level)
s_0	Initial storage level
f_t^0	Initial (current) future position of product t at a given solve
m	Number of storage grid points, $m = \lvert G \rvert$ (default 401)
ϕ	Spread-penalty parameter (unitless); $\phi = 0$ recovers the unpenalized rolling intrinsic
ϕ^*	Tuned spread penalty, located by Brent's method on a walk-forward window
$\delta_{t^*,t}$	Bid–ask spread of product t observed at the decision time t^*
ε	Relevance threshold on the VWAP move that triggers a re-solve (0.10 €/MWh)

Variables

q_i	Quantity executed against order i
k_i	Integer lot multiple for order i, $k_i \in \mathbb{N}_0$
k_t	Integer action (lot multiple) chosen at stage t
f_t	Net future position of product t
f_t^+	Total volume bought for product t
f_t^-	Total volume sold for product t
i_t	Charging flow for product t (energy injected into the battery)
w_t	Discharging flow for product t (energy withdrawn from the battery)
α_t	Binary charge/discharge mode selector for product t, $\alpha_t \in \{0, 1\}$
s_t	Storage level at the end of stage t (the DP state variable)

Functions

V_t	Value function (optimal value-to-go) at stage t, $V_t(s)$
$\tilde{V}_t$	Value function linearly interpolated at off-grid storage states
π_t	Per-stage cashflow of trading quantity q against the order book of product t, $\pi_t(q)$
$\hat{\pi}_t$	Spread-penalized per-stage cashflow, $\hat{\pi}_t(q) = \pi_t(q) - \phi\,\delta_{t^*,t}\,\lvert q\rvert$
S	Storage state-transition function $S(s_{t-1}, f_t)$

Thread scaling (Amdahl's law)

N	Number of parallel processors (threads) across which the DP kernel is distributed
s	Serial fraction of the workload, $s \in [0, 1]$ (distinct from the storage state s_t)
$S(N)$	Parallel speed-up on N processors relative to single-processor execution
S_∞	Asymptotic speed-up ceiling, $S_\infty = 1/s$

Emissions

E_{chg}	Energy charged into the battery, $E_{\text{chg}} = E_{\text{dis}}/\eta_{\text{RT}}$
E_{dis}	Energy discharged from the battery
MEF_{chg}	Marginal emission factor during charge hours
MEF_{dis}	Marginal emission factor during discharge hours
$\Delta\text{CO}_2^{\text{avoided}}$	Net CO_2 displacement: emissions avoided by discharge minus emissions caused by charge

Contents

List of Figures

List of Tables

1

Introduction

Schaurecker et al. [1] report that reoptimizing their rolling-intrinsic dynamic-programming (DP) policy on every order-book event earns roughly 58% more than once-an-hour reoptimization, on a full year of order-by-order data from the German continuous intraday market. That is a large gain for a policy whose mathematical lineage dates back to a gas-storage heuristic of the early 2000s [2, 3]. This thesis asks whether that gain holds when the policy moves out of the reference paper's hourly German EPEX setting into the operational reality of a Stockholm-based battery optimizer. That setting differs in four ways: three bidding zones studied here (Sweden SE3, Sweden SE4, and Germany Amprion), quarter-hourly delivery products, limit-order execution with partial fills, and existing ancillary-market participation that competes with intraday trading for the use of the same battery.

1.1 Energy markets and battery trading

European intraday electricity markets exist because variable renewable generation creates forecast errors that have to be absorbed between the day-ahead auction and physical delivery. Continuous-intraday trading volumes have grown substantially as wind and photovoltaic capacity has expanded across Europe [4]. The spread-narrowing literature reviewed in Section 2.2.2 documents how continuous trading both feeds on and compresses the resulting price spreads. The details of the continuous intraday market (its limit order book, gate-closure rules, and quarter-hourly delivery products) are developed in Section 2.1.1. The recent shift toward quarter-hourly delivery products is what motivates the high-frequency variant of the rolling-intrinsic policy: shorter delivery periods carry shorter-lived arbitrage opportunities.

Battery storage responds within milliseconds and uses the same hardware to both charge and discharge. The only constraint that binds in the long run is the cycle budget (Section 2.1.4). These two properties together make high-frequency intraday arbitrage an obvious application: each shallow round-trip earns a small profit and pays a small degradation cost [5, 6]. The policy's ability to react to a fast-moving order book is what separates profitable from unprofitable execution. The same two properties also make the policy unusually sensitive to execution realism: the profit on any one trade is small, so latency, partial fills, or slippage can easily erase it.

Several recent works adopt the rolling-intrinsic policy or close variants as the baseline for this problem. The approach begins with Gray and Khandelwal's [2, 3]

gas-storage heuristic. Lai et al. [7] and Löhndorf and Wozabal [8] formalized and benchmarked it in the operations-research literature, and Schaurecker et al. [1], Bertrand and Papavasiliou [9], and Boukas et al. [10] carried it over to continuous intraday electricity. The rolling intrinsic reoptimizes the battery's trades against the prices currently in the order book, without anticipating how those prices will change. This myopia keeps it conceptually simple and computationally tractable, yet it still performs within a few percent of the best policies proposed [9].

On a single 10 MWh / 1 h battery, the event-driven DP earns meaningful annual intraday revenue in all three studied bidding zones. On the window common to all three zones, SE4 earns the most, ahead of Germany Amprion and SE3. The event-driven policy beats every fixed timeline tested. The qualifications (the precise magnitudes, and the cases in which intraday arbitrage is the wrong allocation of the battery) are reported in Chapter 4 and synthesized in Chapter 5.

1.2 Flower's business model

Flower is a Stockholm-headquartered energy-asset operator and optimizer founded in 2020. The company operates a portfolio of flexible energy assets across the Nordic bidding zones and, more recently, the German market. Flower owns some of these outright and optimizes the rest for their owners. In each case, the company trades the asset's flexibility across electricity markets for higher revenue.

The current revenue mix is dominated by ancillary services: the primary, secondary, and tertiary frequency reserves (FCR, aFRR, mFRR). Continuous intraday trading participates but is not yet a primary strategy. That balance is shifting. Svenska Kraftnät publishes hourly marginal prices for the primary [11], secondary [12], and tertiary [13] frequency reserves through its Mimer portal. Across all three product families, clearing prices have compressed over 2024–2026. Flower's own market analysis [14] characterizes the high 2022–2024 clearing prices as in part an artifact of the energy crisis and the early-market immaturity of the Nordic battery fleet rather than a sustainable baseline. Operators in the region are shifting toward the day-ahead and intraday markets, and this thesis helps determine whether and when a battery should move capacity from ancillary services to intraday trading.

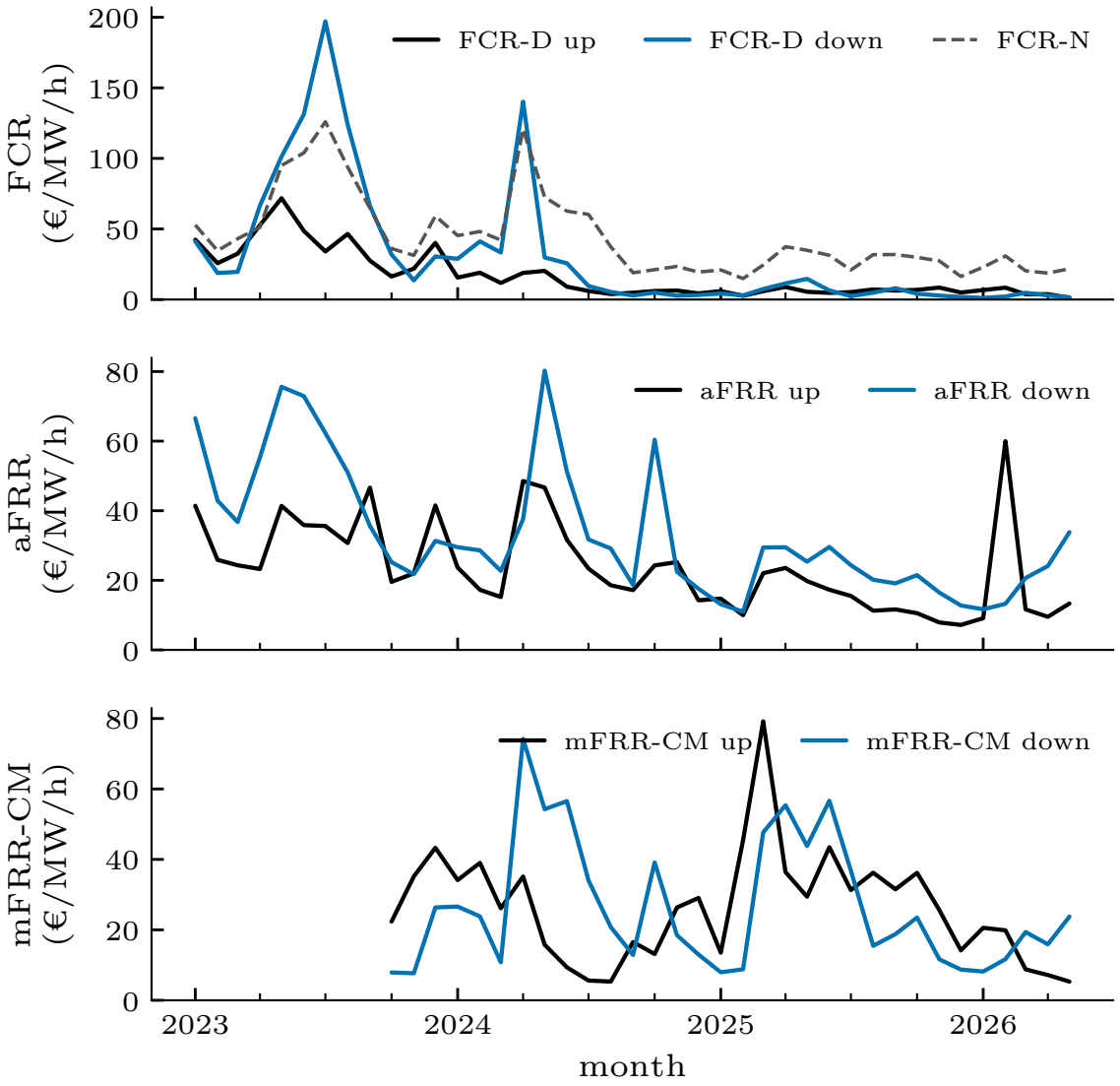


Figure 1.1: Monthly mean capacity-market clearing prices for the Swedish primary (FCR-N and FCR-D up/down, top), secondary (aFRR up/down, middle), and tertiary (mFRR-CM up/down, bottom) frequency reserves, January 2023 – April 2026. FCR prices are the Nordic single clearing price; aFRR and mFRR-CM prices are volume-weighted across the four Swedish bidding zones (SE1–SE4), with zero-volume hours excluded so they do not anchor the mean at zero. All three product families peak in 2023–2024 and compress over 2024–2026. Data source: Svenska Kraftnät's Mimer portal [11, 12, 13].

Each battery's flexibility is finite: one cycle budget, one state of charge at any moment, and one set of operating limits the trading software has to respect. The commercial question that motivates this thesis is therefore a selection question: given that the batteries in Flower's portfolio already earn ancillary revenue on the same hardware, when is it worth committing capacity to intraday arbitrage instead? The thesis develops the empirical material needed to answer that question: the realized intraday profit per zone, its sensitivity to decision timeline, and a day-level comparison against the dominant ancillary alternative. Chapter 5 draws the

practical recommendation.

1.3 Problem motivation

Verifying the rolling-intrinsic policy's reported gain is only the starting point. The rolling intrinsic is the standard baseline for this problem, yet its continuous-intraday evidence comes almost entirely from the German market, leaving open whether the gain holds anywhere else. This thesis adapts the policy to Flower's operational setting and asks if the rolling-intrinsic policy is still profitable. Four differences separate the reference paper's setting from Flower's. Each is a modeling choice, not merely an implementation detail, and each can change the answer.

The first difference is *geographic*. The reference paper is based on the German EPEX continuous intraday market, one of the deepest intraday books in Europe, and uses a contract structure in which each delivery product has its own rolling five-minute gate closure. The two Swedish bidding zones studied here, SE3 and SE4, sit on Nord Pool, whose books are thinner and gate closure is batched: all four quarter-hour contracts of a delivery hour close together at the top of the previous whole hour (Section 2.1.3). Most of the literature on this problem focuses on the German market. The closest Swedish analysis is Spodniak et al. [15], which characterizes aggregate market volumes rather than a trading strategy. Filling that gap for SE3 and SE4 is part of the motivation for this thesis.

The second difference is *granularity*. The reference paper trades hourly delivery products. Flower trades quarter-hourly. Because Nord Pool quotes its minimum order quantity in power terms (0.1 MW), the corresponding minimum energy lot shrinks by a factor of four when the delivery period drops from one hour to fifteen minutes (from 0.1 MWh per hourly contract to 0.025 MWh per quarter-hour contract). At the same time the number of tradable contracts in any given DP horizon quadruples, and the storage grid has to be four times finer to keep resolution comparable. Combined, the three changes inflate the DP backward-pass cost by a factor of $4^3 = 64$ relative to the hourly reference.

The third difference is *execution realism*. Real exchange connections have latency, and the orders posted on the market at the moment of a trading decision are not guaranteed to still be there by the time a submitted market order arrives. Both the reference paper and this thesis account for this by introducing a fixed technical delay between event arrival and trade execution, so the two policies share the same fixed-latency premise. The substantive difference is in the fill policy. The reference paper uses Fill-or-Kill, where the planned trade either executes in full at the original limit price or is canceled outright. This thesis instead uses a limit-order-style fill policy: it re-checks the order book after the delay, fills greedily against any orders still resting at the original limit price or better, and accepts the resulting partial fill, discarding only the unfilled remainder.

The fourth difference is the *ancillary opportunity cost*. The reference paper says nothing about how intraday arbitrage compares to running the same battery on ancillary services, because in its setting the battery does nothing but trade intraday.

For an operator with existing ancillary-market participation, this is the decisive commercial question, and a newly pressing one: as the ancillary prices documented in Section 1.2 fall, the revenue intraday has to beat shrinks with them.

1.4 Objective of the thesis

The objective of this thesis is to replicate, verify, and adapt the rolling-intrinsic DP of Schaurecker et al. [1] to Flower's operational setting, and to quantify what changes (and what does not) when the policy is applied outside the reference paper's setting.

1.5 Research questions

The empirical chapters of this thesis are organized around five research questions. Each is answered with either a number or a one-sentence verdict by the end of Chapter 4.

1. Does the rolling-intrinsic DP earn meaningful intraday revenue on Sweden SE3, Sweden SE4, and Germany Amprion quarter-hour order books at event-driven decision timeline?
2. How does the rolling-intrinsic DP compare to the MILP reference in solution quality and runtime, in particular at the quarter-hour granularity?
3. How does decision timeline affect realized profit?
4. Where does the DP's scaling break down (battery size, book thinness, state-grid resolution), and which implementation choices flip the policy from profitable to loss-making?
5. When, on a day-by-day basis, does intraday trading dominate the ancillary-market alternatives that a battery in the portfolio would otherwise be allocated to?

1.6 Scope and delimitations

The thesis covers a single battery in isolation. Portfolio coupling and fleet effects are out of scope. The empirical results are historical backtests against the recorded order books. Our simulated trades consume the matched orders from the book, so the same resting liquidity is never traded against twice. What we do not model is how other market participants would have updated their bids and asks in response to the policy's presence. The implications of this assumption for fleet-scale deployment are discussed in Chapter 5. Battery degradation is included in the optimizer through the linearized per-MWh penalty ν_{deg} but is not reported as a separate cost line. The Swedish window begins on 1 April 2025 (the date quarter-hour contracts went live on Nord Pool) and runs through March 2026. The German window covers January 2025 to February 2026. The policy's grid impact is quantified in Section 4.8, and

its wider environmental consequences are reviewed in Sections 2.2.3 and 2.2.4. The thesis does not perform a marginal-emissions analysis of the policy's trades.

1.7 Contributions relative to prior work

This thesis contributes the following relative to the reference paper and to the surrounding rolling-intrinsic literature.

1. **Multi-zone backtest.** First evaluation of the Schaurecker rolling-intrinsic DP on Nord Pool SE3 and SE4 in addition to Germany Amprion, on a year of real order-book data.
2. **Quarter-hour extension.** The first implementation of the policy at quarter-hour granularity, where the DP backward-pass cost scales $4^3 = 64\times$ over the hourly reference. The high-performance-computing measures required to make the resulting workload tractable are described in Chapter 3.
3. **Limit-order execution model.** Replacement of the reference paper's Fill-or-Kill fill policy with a limit-order-style rule that re-checks the order book after a fixed technical delay and accepts partial fills against orders still resting at the original limit price or better, on the same fixed-latency premise as the reference paper.
4. **DP-vs-MILP verification at quarter-hour granularity.** A comparison of the DP and a Gurobi MILP solver on identical order-book data, extending the reference paper's hourly DP-vs-MILP check to the more demanding quarter-hour case.
5. **Ancillary opportunity-cost analysis.** A day-level win-rate analysis of the intraday DP against the dominant ancillary alternative in SE3, SE4, and Germany Amprion, a comparison the reference paper does not make, since it models no alternative use of the battery.

1.8 Thesis outline

Chapter 2 establishes the microstructure of the continuous intraday market, surveys the impact of high-frequency battery trading on the battery, the market, and the wider system and environment, and develops the MILP and DP formulations of the rolling-intrinsic problem. Chapter 3 describes the order-book data pipeline, the C++/OpenMP DP kernel, and the execution-realism layer. Chapter 4 reports the empirical performance per market, the timeline sweep, the MILP comparison, the solver performance, and the ancillary opportunity-cost analysis. Chapter 5 synthesizes the findings into a practical recommendation for Flower's operational deployment of the policy.

2

Theory

In this chapter, four topics will be covered: the electricity market, the impact of battery trading on the electricity market, general mathematical concepts and finally the reference paper and modeling assumptions.

2.1 Electricity market background

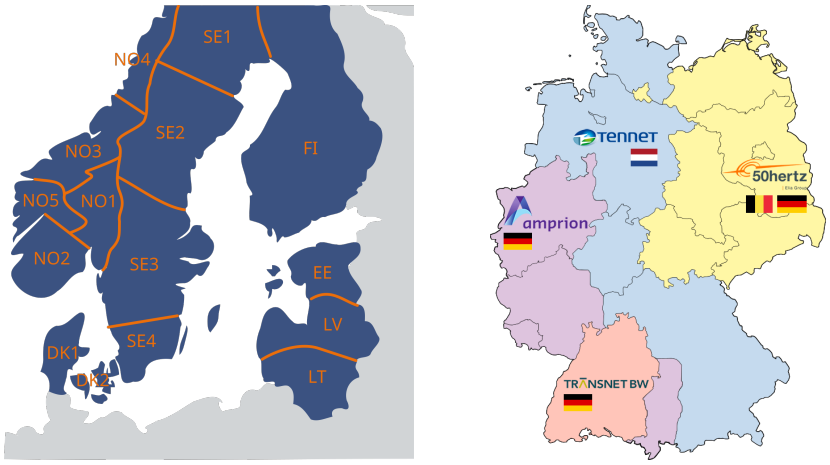
This section introduces the structural features of the European continuous intraday electricity market that shape every subsequent design choice in this thesis. The continuous intraday market is described first, alongside its relationship to the balancing reserve products. The section then highlights the differences between the German Amprion and Swedish SE3/SE4 zones that motivate studying both.

2.1.1 Intraday market and balancing markets

Electricity is a non-storable commodity that, in the absence of substantial storage capacity on the grid, has to be produced at the instant it is consumed. To organize the resulting balancing problem, the German and Swedish bidding zones studied in this thesis split wholesale trading (the bulk buying and selling of electricity between large market participants) into a sequence of markets that close progressively nearer to physical delivery. After these markets close, the Transmission System Operator (TSO) balances the grid in real time using balancing reserves. The day-ahead auction clears at 12:00 on day $D-1$, covers hourly and quarter-hourly delivery products, and is responsible for the bulk of physical wholesale volume [16]. The continuous intraday market opens shortly afterwards. Unlike the day-ahead, it matches bids and asks against each other as they arrive rather than at a single fixed time, and lets market participants update their positions as forecasts improve and unexpected events occur, up until a gate closure that varies by bidding zone, anywhere from a few minutes to over an hour before physical delivery. Any remaining imbalance at gate closure is covered physically by the TSO activating balancing reserves (Section 2.1.2), and each balance responsible party's net deviation over the delivery period is settled financially at the imbalance price.

This thesis focuses on the continuous intraday market, which is the segment best suited to a battery that wishes to monetize short-lived price fluctuations. Empirically, the study uses Nord Pool order-book data for three quarter-hourly bidding

areas: Sweden SE3, Sweden SE4, and Germany Amprion. Their geographic positions are shown in Figure 2.1.



(a) Nordic price zones *(b) German TSO control areas*

Figure 2.1: Geographic context for the three bidding areas in the empirical work. (a) Nordic price zones: Sweden is divided into four zones (SE1 in the north through SE4 in the south); this thesis uses **SE3** and **SE4**. (b) German transmission system operators (TSOs): the German grid is split across four control areas (Amprion, TenneT, 50Hertz, TransnetBW); this thesis uses the **Amprion** control area, which covers the south-west of Germany. Map sources: Hou710 (Wikimedia Commons, CC BY-SA) and Francis McLloyd (Wikimedia Commons, CC BY-SA 3.0).

These markets share the same basic trading mechanism. Trading is organized as a continuous double-sided market: each delivery product has its own limit order book, in which participants post bid and ask orders that are matched on price–time priority (best price first, then the earliest order at that price) [17]. The important market-specific differences are the gate-closure rule and the available liquidity; these differences are discussed separately in Section 2.1.3.

For the German Amprion market each quarter-hour contract can be traded up to five minutes before the start of its delivery interval, and orders submitted after that point are rejected [18]. Figure 2.2 illustrates this rolling gate-closure pattern for the four quarter-hour contracts of a single delivery hour. The Swedish SE3 and SE4 zones follow a different hourly-batched gate closure rule, shown in Figure 2.6.

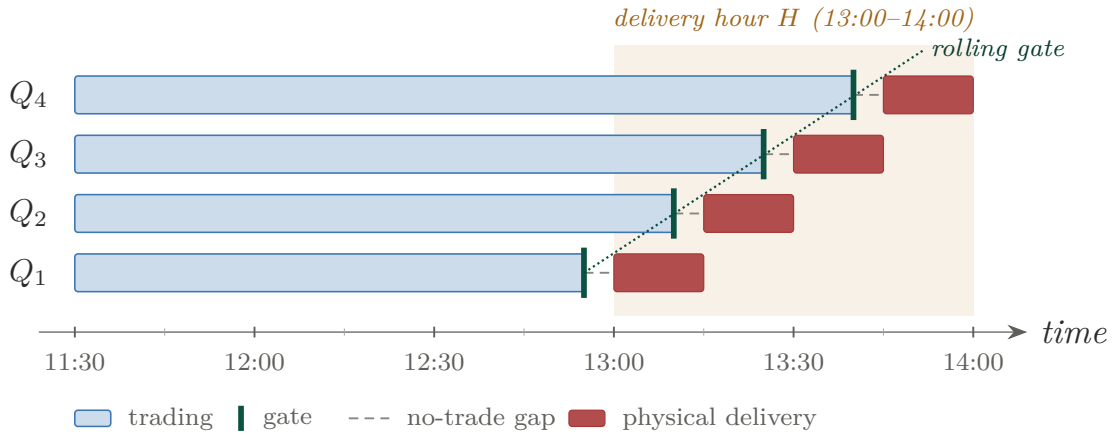


Figure 2.2: Gate closure in the German Amprion continuous intraday market. Each of the four quarter-hour contracts of delivery hour H has its own *rolling* gate that closes 5 minutes before that contract's delivery; the four green ticks march diagonally across H. Trading (blue) ends at the gate, leaving a 5-minute no-trade gap before physical delivery (red).

The information observed by a trader is a stream of limit order book snapshots, one per contract per event. Each snapshot consists of two stacks of orders, the asks $\mathcal{O}_t^+$ and the bids $\mathcal{O}_t^-$, where every order i is summarized by a limit price P_i (the lowest price an ask will accept, or the highest price a bid will pay) and a remaining quantity $Q_i > 0$. The best bid and the best ask bound the bid–ask spread, which is the prevailing transaction cost for any trader who needs to cross the book to execute immediately. Figure 2.3 shows an example snapshot. To buy immediately, a battery pays the best ask. To sell immediately, it accepts the best bid. A wider spread therefore makes each round trip more expensive, before any battery losses or fees are considered. Spreads also vary over the life of a contract: they are often wider far from delivery, when forecast uncertainty is larger and fewer participants are actively updating orders, and tend to tighten as delivery approaches and liquidity concentrates in the front-most contracts [19, 17].

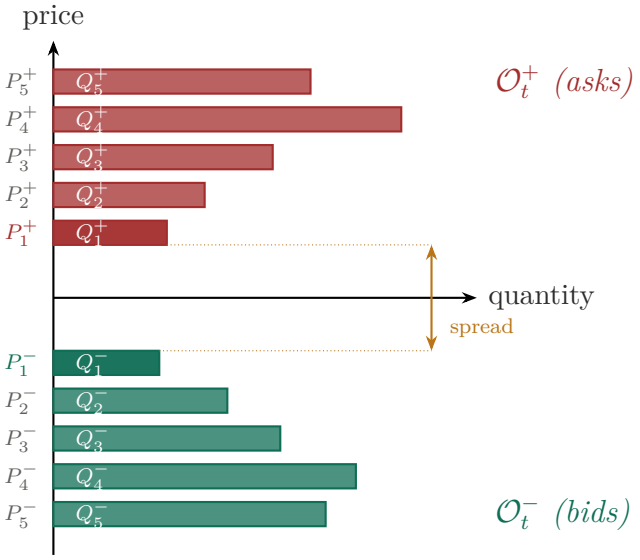


Figure 2.3: Snapshot of a limit order book for one contract t. Asks $\mathcal{O}_t^+$ (red) sit above the spread and bids $\mathcal{O}_t^-$ (green) sit below it; each rung is a (P_i, Q_i) pair. The best ask and best bid (highlighted) bound the spread (amber).

A market order to buy or sell a quantity Q_{order} is matched against the opposing stack from the inside out: the best price level is consumed first, the second-best next, and so on until either the order is fully filled or the stack is exhausted. The realized execution price of such a sweep is the volume-weighted average of the limit prices of the consumed orders, as illustrated in Figure 2.4. This walk-the-book mechanic is the source of the price-impact cost that any trading policy operating on the intraday market has to account for, and it motivates the explicit modeling of the order-book stacks in the MILP formulation introduced in Section 2.4.1.

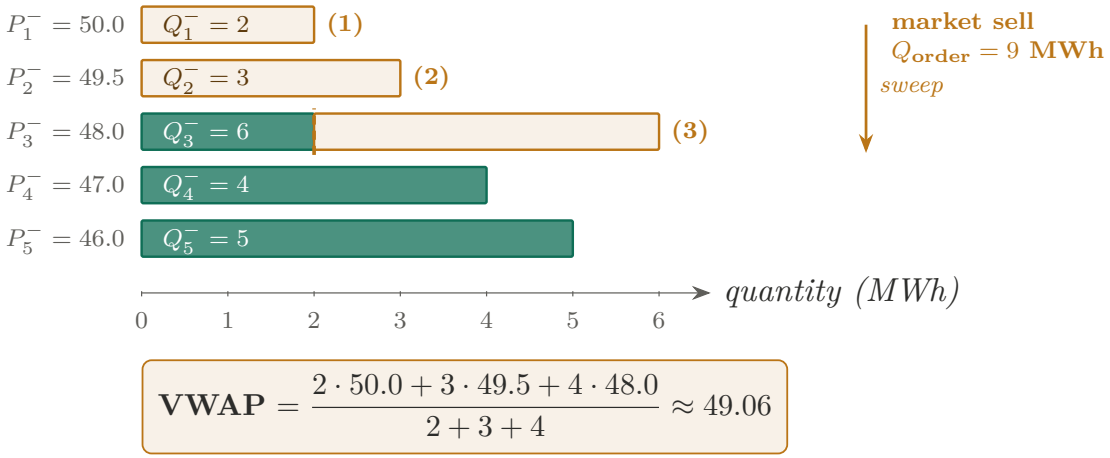


Figure 2.4: Example of a market sell order $Q_{\text{order}} = 9$ MWh sweeping the bid side of the LOB in Figure 2.3. The order consumes the best bid first and walks down the "ladder", fully exhausting bid 1 and 2 and partially filling bid 3. The realized proceeds are the volume-weighted average price (VWAP) over the consumed bids.

Beyond the order-matching mechanic, the exchange also imposes an explicit cap on the trading rate of any single participant of no more than 5,000 trades per hour [20].

2.1.2 FCR, aFRR, and mFRR

After the gate of a contract closes, the TSO becomes responsible for keeping the grid in instantaneous balance. To do so, it relies on a cascade of three balancing-reserve products [21, 22] that activate on different time scales and with different degrees of automation. The cascade is illustrated in Figure 2.5.

Frequency Containment Reserve (FCR), also called primary reserve, is activated automatically by local controllers in response to a deviation of the grid frequency from its nominal 50 Hz value. FCR responds within 30 seconds. Its role is to stop the deviation, not to fully restore the frequency. Automatic Frequency Restoration Reserve (aFRR), or secondary reserve, takes over within minutes and is dispatched by an automated signal sent from the TSO to qualified units. Its role is to bring the frequency back to nominal and to relieve the FCR-providing units so that they can return to their pre-disturbance setpoints. Manual Frequency Restoration Reserve (mFRR), or tertiary reserve, is dispatched manually by the TSO on a horizon of approximately 15 minutes and is used to free up the aFRR-providing units and to handle larger or longer-lived imbalances.

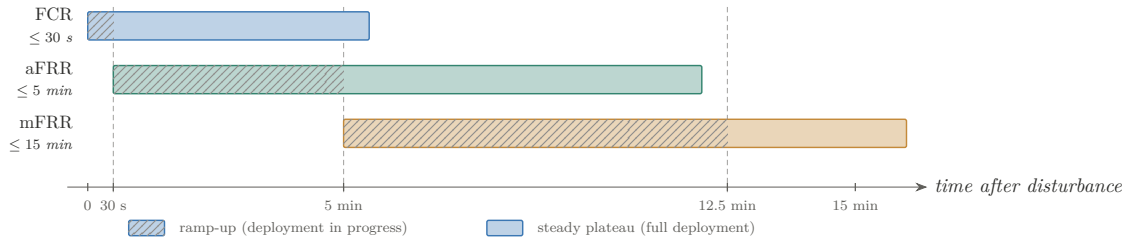


Figure 2.5: Cascade of balancing reserve products following a frequency disturbance. Frequency Containment Reserve (FCR, primary) deploys within 30 s and arrests the frequency drop; automatic Frequency Restoration Reserve (aFRR, secondary) takes over within minutes; manual FRR (mFRR, tertiary) restores reserves on a 15-min horizon.

A battery operator can earn revenue from balancing reserve products from two components [21, 22]. The first is a capacity fee paid for committing a given amount of power (MW) to be available during a defined delivery window. This capacity is procured through an auction held ahead of delivery, and the fee is paid whether or not the reserve is called. The second is an activation fee, settled per MWh of energy delivered when the TSO calls on the reserve. The capacity fee is steady revenue tied to availability. The activation fee scales with how often the reserve is called and how much energy is delivered on each call.

Consequently, for a battery operator, each of these reserve products is an additional revenue stream beyond intraday energy arbitrage. This thesis still focuses mainly on the continuous intraday market, both because the reference paper [1] does so

and because multi-market co-optimization introduces complexity beyond the scope of this thesis. The balancing context is summarized here primarily to clarify the design choices of the intraday market: short gate closure, quarter-hour granularity, and continuous matching are deliberate features that limit the volume of reserves the TSO has to activate after gate closure. Additionally, any strategy has to be compared to the revenue of the balancing reserve products.

2.1.3 Differences between Germany and Sweden

Although the German and Swedish continuous intraday markets share the broad design of a price–time-priority limit order book with hour and quarter-hour delivery products (plus a half-hour product on the German side), two structural differences shape how a battery operator can trade in each. These differences motivate treating SE3, SE4, and Germany Amprion as separate empirical markets rather than presenting one generic intraday result.

The first difference concerns gate closure. In the German Amprion zone, every contract has its own rolling gate that closes five minutes before its delivery (Figure 2.2). In the Swedish bidding zones SE3 and SE4 operated by Nord Pool, all four quarter-hour contracts of a delivery hour instead share a single batch gate at the top of the previous whole hour, as shown in Figure 2.6. The Swedish market uses variable waiting times of 60, 75, 90, and 105 minutes between gate closure and physical delivery for the first through fourth quarter-hour contracts. This also removes the close-to-delivery trading window where most trading happens in the German market.

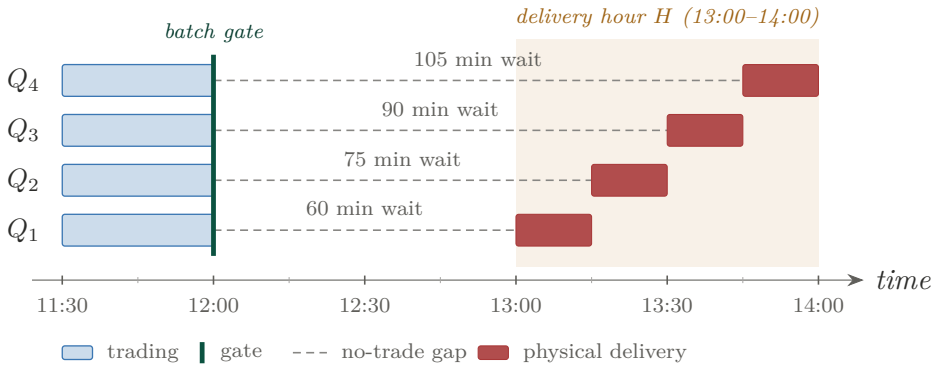


Figure 2.6: Gate closure in the Swedish (Nord Pool, SE3/SE4) continuous intraday market. In contrast to the German rolling gate (Figure 2.2), all four quarter-hour contracts of delivery hour H share a single *batch* gate at the top of the previous whole hour ($H-1 = 12{:}00$ here). The shared closure produces variable waiting times – 60, 75, 90, 105 minutes for Q_1 through Q_4 – between gate and physical delivery.

The second difference is liquidity. The German continuous intraday market is the most liquid intraday market in Europe by a substantial margin [18, 16]: order books for the quarter-hour contracts remain populated all the way to gate closure, and depth at the top of the order book typically supports multi-MW trades on most

contracts. The Swedish intraday market is much thinner [16]: order books are sparser, the time between trades is longer, and a battery trading against the book may move the price noticeably even on small volumes.

These two effects compound. Without a close-to-delivery trading window, the Swedish market offers fewer profitable trades to a rolling intrinsic policy (outlined in Section 2.4.1). The thinner liquidity also makes each trade more expensive: when a multi-MW order is larger than the volume offered at the best price, the remainder fills against orders at worse prices (Figure 2.4), pulling the realized average price away from the best quote.

2.1.4 Role of batteries in energy markets

Lithium-ion battery energy storage systems (BESS) have moved from a niche application to a mainstream grid asset over the past decade, propelled by the rapid cost decline of cells and by the growing share of variable renewable generation that the power system has to integrate. By 2024, costs had fallen to approximately USD 139/kWh at the cell or pack level and USD 250–400/kWh at the fully-installed system level [23]. From a power-system perspective, a battery is unusual in two ways. It is fast, with response times measured in milliseconds rather than minutes, and it is fully symmetric, in the sense that the same hardware can absorb and inject power within its rated capacity [23]. These two properties enable a battery to provide services that conventional thermal assets either cannot deliver or can only deliver at much higher cost. The revenue stack of a utility-scale battery typically combines three sources: wholesale arbitrage, balancing reserves, and local network services [23]. Wholesale arbitrage profits derive from price differences between charging and discharging the battery in the day-ahead auction and on the continuous intraday market [24, 25]. This thesis focuses on the intraday market. Balancing reserves (FCR, aFRR, and mFRR) pay a capacity fee for availability and an activation fee for delivered or absorbed energy [21, 22]. mFRR is the largest of these in SE3 and SE4 for many optimizers, while FCR and aFRR dominate in Germany [14]. Local network services such as congestion relief and black-start capability provide a smaller, location-dependent stream [23]. The operator must allocate the finite storage capacity and finite cycle budget across these competing revenue streams.

The cycle budget is the binding long-run constraint. Each cycle through the battery contributes to capacity degradation and brings the cell closer to its end-of-life replacement. Lithium-ion aging has both a calendar and a cycle component, with the relative weight depending on temperature, depth of discharge, and operating profile [5, 6]. Trading decisions therefore cannot be evaluated on instantaneous revenue alone: they must be net of a degradation cost. Following the reference paper [1], this thesis models the degradation cost as a flat per-MWh charge added to every executed trade. The resulting parameter is one of the inputs to the rolling intrinsic policy developed in the remainder of the thesis.

2.2 Impact of batteries and trading strategies

The trading policy developed in the remainder of this thesis has effects at three nested scales: on the battery itself, on the energy market it trades in, and on the wider system and environment around them. This section reviews what the literature says about each of these scales in turn. The frame of reference is a rolling-intrinsic battery operating on Germany Amprion and Sweden SE3/SE4. Because the Swedish setting differs from the German one in ways that bear directly on each of these scales, the Nordic-specific implications are gathered separately in Section 2.2.4 rather than threaded through the German literature.

2.2.1 Impact on the battery itself

Section 2.1.4 introduced the cycle budget as the binding long-run constraint on a grid-scale battery and the linear per-MWh degradation cost ν_{deg} as the simplest possible way of pricing it. The underlying physics is richer than this linearization suggests. The semi-empirical rainflow-counting model of Xu et al. [5] splits lithium-ion life loss into a calendar and a cycle component, with separate stress functions for state of charge, depth of discharge, C-rate (charging speed relative to capacity), and temperature, fitted to manufacturer datasheets. A companion paper by the same group [26] casts the cycle-aging cost as a piecewise-linear convex function so that it can drop directly into an LP or MILP without breaking linearity. This thesis follows the reference paper [1] in using a flat per-MWh degradation cost.

Empirically, the choice of trading strategy changes how fast the battery ages. Gräf et al. [6] report the only utility-scale measurement of this effect: on the 7.2 MW / 7.12 MWh Herdecke installation, day-ahead and intraday arbitrage degrade the pack noticeably faster than providing FCR. Even when the application is held fixed, the intra-container temperature spread alone causes annual capacity loss to vary by up to 0.97% within a single pack. In a combined lab and simulation study, Fares and Webber [25] find that arbitrage operation often retires lithium-ion cells through calendar aging before cycle limits bind.

Optimizers that internalize even a linearized version of the degradation cost close a measurable fraction of the gap left by trading strategies that do not. Collath et al. [27] report that a model-predictive-control digital twin running EPEX intraday arbitrage with combined calendar and cyclic degradation raises lifetime profit by up to 29% compared to an aging-blind reference scenario. Schade et al. [28] make the inverse observation: in a simplified arbitrage MILP, ignoring degradation entirely leaves ex-post hidden costs that can exceed gross revenue, turning seemingly profitable trades into losses. Together, these results justify the inclusion of the per-MWh penalty inside the intrinsic problem.

2.2.2 Impact on energy markets

A high-frequency trading policy does not just react to intraday prices. By posting on and crossing the order book, the battery it controls becomes part of the

activity that sets those prices. The empirical framework for studying that trading on the German continuous intraday market was established by Hagemann and Weber [17], whose four liquidity indicators – bid-ask spread, depth, resiliency, and traded volume – have since become the standard descriptors of the market state. Their data favor a profit-maximizing-trader model over a fundamental merit-order one (prices set by the most expensive generator running). On the econometric side, Kiesel and Paraschiv [29] document that quarter-hourly intraday prices respond to wind and photovoltaic forecast errors asymmetrically and with a merit-order-slope dependence, while Narajewski and Ziel [30] argue that the hourly continuous market is weak-form efficient under their ensemble forecasting test.

The rolling-intrinsic policy developed in the remainder of this thesis is one of several recently proposed alternatives in this design space. The reference paper [1] adapts the gas-storage rolling intrinsic to continuous intraday electricity on a full year of EPEX order-by-order data and reports that high-frequency reoptimization earns roughly 58% more than once-per-hour and 14% more than once-per-minute re-solving. This is the central quantitative justification for re-solving on every order-book event. Bertrand and Papavasiliou [9] parameterize an order-book threshold policy via REINFORCE (a policy-gradient reinforcement-learning algorithm) and report a 4.2% improvement over the rolling intrinsic on the last 165 days of 2015 in the German market. Boukas et al. [10] present a deep-reinforcement-learning baseline trained on real EPEX order-book features. More sophisticated policies beat the rolling intrinsic by only a few percent of realized profit. The gas-storage analysis of Löhndorf and Wozabal [8] provides a theoretical reason: for storage problems with this structure, the rolling intrinsic sits close to the dual-hedging upper bound.

A further implication of algorithmic trading activity is that it also affects the spreads it profits from. Hendershott, Jones, and Menkveld [31] document this for algorithmic trading in equities: liquidity provision narrows bid-ask spreads and reduces adverse selection while still earning the spread. Note, however, that this concerns algorithms which provide liquidity to the market, which is not a part of the implemented algorithm in this thesis. However, there are indications that algorithmic trading in general can reduce spreads. Balardy [19] studies the German continuous intraday market, showing that bid-ask spreads narrow with pre-gate-closure trading activity and book depth, with a characteristic U-shaped pattern over the life of each contract (wide far from delivery and near gate closure, narrowest in between). Lamp and Samano [24] reach a similar conclusion on California ISO data for the battery storage-specific case: each new grid-scale battery entry compresses temporal intraday wholesale price spreads, although the per-asset effect loses statistical significance after roughly five weeks as the market re-equilibrates. Whether algorithmic trading also reduces overall price variation is less clear, and the empirical evidence is confounded by concurrent structural changes. Following the 2018 launch of the Cross-Border Intraday (XBID) platform, which linked European intraday markets into a shared order book, Kath [32] reports at most a small *increase* in volatility on German quarter-hourly contracts. This change reflects the joint effect of deeper market integration and increased algorithmic participation, and the contribution of algorithmic trading alone cannot be isolated from these data. Algorithmic trading

therefore appears to narrow the bid-ask spread, but its effect on aggregate price variation cannot be identified from the available evidence. For a battery operator, the price differences that generate revenue are also the differences the trading shrinks.

At the system level, the rise of continuous intraday trading has rewritten how Germany balances its grid. Koch and Hirth [4] resolve the so-called German "balancing paradox" – rising renewables coinciding with falling reserve use – by attributing it to round-the-clock trading and to the growth in quarter-hourly intraday volumes, which together have shifted the absorption of forecast errors into the intraday market rather than into the TSO's reserves. They estimate that the shift to quarter-hourly products alone explains a 17% decrease in balancing energy, and that the total reserve requirement and reserve use have declined by 50% since 2011. The closely related effect of intentional opposite-imbalance positioning near gate closure, sometimes referred to as "passive balancing", is quantified by Koch and Maskos [33] at up to 5% reduction in balancing-energy demand and in high system balances. A fleet of high-frequency rolling-intrinsic batteries operates in this setting, but only as a price-taker on the intraday order book. The policy does not target balancing prices or imbalance positions, and it does not engage in passive balancing; it responds only to intraday liquidity. Its trades contribute to the aggregate forecast-error absorption Koch and Hirth document through the same channel as any other intraday participant, not through any balancing-specific mechanism. This indirect contribution motivates considering the thesis as a public-interest question, not only a private-profit one.

The welfare effects of profit maximizing battery trading (merchant storage) are less clear. Sioshansi [34] shows analytically and on wholesale market data from the eastern United States that merchant storage owners under-utilize capacity relative to the social optimum, because arbitrage transfers surplus from generators to consumers and so dilutes the storage owner's own profit. The social maximum is reached only under mixed ownership. Virasjoki et al. [35] reach a similar conclusion in a European bilevel model: consumers typically capture more storage-induced surplus than investors, but market power can suppress the optimal level of storage investment in the first place. The implication for high-frequency intraday arbitrage is that the practice can be privately profitable, but that alone does not tell us who actually benefits. No Germany- or Nordic-specific empirical study has quantified the redistribution that this thesis's policy would induce. These results assume competitive operation. Sioshansi [36] shows that a strategic operator with enough market power can improve its own arbitrage profit by withholding capacity, preserving the price spread that a competitive operator would compress. This caveat applies only at concentration levels well beyond any single trader in today's market: the recent wholesale concentration metrics reported by ACER [37] remain firmly below the threshold at which an individual participant could influence intraday clearing prices.

2.2.3 Impact on the environment

The intuition that storage automatically reduces emissions is not universally supported. Hittinger and Azevedo [38] show that, under historical United States grid

conditions, a 90%-round-trip-efficient bulk storage system performing pure arbitrage *increased* CO_2 emissions by 104 to 407 kg per MWh of delivered energy across the 20 eGRID subregions they study. The reason was that storage tended to charge from coal at night and discharge at peak hours, replacing gas generation. Craig, Jaramillo, and Hodge [39] subsequently show that the sign of this effect flips from positive to negative as the share of variable renewable generation grows. The assumption that "storage equals decarbonization" is therefore conditional on which generator is the most expensive one running when the storage charges and discharges.

The relevant quantity is the *marginal emissions factor* (MEF), the CO_2 intensity of the generator that responds to a marginal change in load at a given hour, as distinct from the average grid mix [38, 40]. The MEF varies sharply by zone and hour. The Swedish power-sector marginal is dominated by hydro and nuclear, with an annual mean in the 20–80 g CO_2/kWh range [41]; the German marginal swings from roughly 200 g/kWh under the midday solar shoulder to 500–550 g/kWh under the evening coal-and-gas peak [42, 43].

Additionally, previous research has tried to quantify the relationship between emissions and profits. Arciniegas and Hittinger [44] show what a policy lever can do: co-optimizing revenue against emissions enables 25 to 50% emission reductions at the cost of only 1 to 5% of revenue, through rescheduling alone. Even a modest CO_2 price closes most of the gap.

For the specific German context most directly relevant to this thesis, Kannangara and Davis [45] provide the quantitative reference. Their German-case MILP traces the trade-off between profit and emissions. Maximum-profit arbitrage raises system emissions by up to 7.5 t CO_2 per MWh of installed battery capacity per year, equivalent to roughly 12% of the battery's annual life-cycle emissions. About 60% of those induced emissions can be removed by giving up only 1.5 to 2.7% of net arbitrage profit. Fully CO_2-neutral operation costs about 7% of profit. This trade-off between profit and emissions is the main German finding for operational emissions from arbitrage. Whether and how it transfers to the Nordic case is taken up in Section 2.2.4.

Beyond operational emissions, the battery itself carries a lifecycle burden. Peters et al. [46] provide the meta-review used as the cell-production reference: roughly 110 kg CO_2-equivalent per kWh of cell capacity at the median, with a wide spread driven by electricity mix and cathode chemistry. The International Energy Agency's *Global Critical Minerals Outlook* [47] projects under its Net-Zero Emissions pathway a ninefold increase in lithium demand, a doubling of nickel and cobalt demand, and a fourfold increase in graphite demand by 2040. Refining capacity remains more than 85% concentrated in the top three producing countries. On the offsetting side, Kastner et al. [48] quantify the contribution of second-life batteries to European stationary storage: reusing 40% of end-of-life electric-vehicle batteries covers projected EU stationary demand by 2040 with a 1.5% reduction in primary-material demand over the 2020–2050 period (the 7.5% figure in the same paper is for vehicle-to-grid, not second-life). On lithium supply specifically, Chordia, Nordelöf, and Ellingsen [49] compare brine and spodumene extraction routes using a methodology close to the

Chalmers environmental-systems-analysis tradition [50] that informs this thesis.

Battery storage also has a real physical and social impact at the installation site. Baur et al. [51] report a German survey on the visual-impact acceptability of large stationary battery storage and find that acceptance is markedly higher in industrial and rural settings than in residential ones. Emmerich et al. [52] map public acceptance more broadly and identify trust in industry, perceived environmental benefit, and self-identity as the main mediators of the residual not-in-my-back-yard gap. On safety, the Electric Power Research Institute's community-siting report [53] documents that the worldwide utility-scale battery failure rate dropped by approximately 98% between 2018 and 2024 even as installed capacity expanded sharply.

2.2.4 Implications for the Nordic case

The literature reviewed in the three subsections above is overwhelmingly German, with a few studies based on French and United States systems. The Swedish bidding zones SE3 and SE4 differ from the Amprion benchmark in three ways that all bear on the impacts just discussed. Their intraday order books are substantially thinner. Their gate-closure rule is a single batch per delivery hour rather than the rolling five-minute German gate (Section 2.1.3). Their generation mix is almost carbon-free. These three differences are substantive, and hence algorithmic intraday trading by grid-scale batteries on SE3 and SE4 is one of the gaps in the literature that this thesis addresses.

On the market side, the closest Nordic analysis is Spodniak, Ollikka, and Honkapuro [15], who quantify how day-ahead, intraday, and balancing volumes in the Nord Pool area respond to wind and demand but do not include an explicit battery-trading-strategy model. The Hagemann-Weber liquidity indicators [17] have not been re-estimated on SE3 or SE4, and the rolling-intrinsic policy of the reference paper has never been backtested on Nord Pool order books. Cognéville, Deschatre, and Warin [54] provide the theoretical reason to expect that this matters: their order-book stochastic model shows that neglecting liquidity costs overstates battery revenue more in thin markets and more when many operators run the same strategy. The parallel SE3, SE4, and Amprion backtest reported later in this thesis therefore directly tests this gap.

On the environmental side, the Hittinger-Craig finding that the sign of the emissions effect depends on the marginal generator (Section 2.2.3) sits at its extreme on SE3 and SE4. With the Nordic marginal-emissions intensity among the lowest in Europe, the 7.5 t CO_2 per MWh-yr ceiling reported by Kannangara and Davis [45] for the German setting functions as a strict upper bound on what the same arbitrage policy could induce on the Nordic grid. The most plausible point estimate is close to CO_2-neutral regardless of trading frequency. By the same token, the abatement potential per profit-euro of switching to CO_2-aware operation is correspondingly small: there is little margin left to give up. In the absence of a Swedish study on battery-storage social acceptance, the German evidence from Baur and Emmerich serves as the closest available reference.

2.3 Mathematical concepts

The optimization problems studied in this thesis combine two ideas: mathematical programming, which models the battery and the market as a constrained optimization problem, and dynamic programming, which exploits the time-ordered structure of energy delivery products. This section gives the minimal terminology needed before the full rolling-intrinsic formulation is introduced.

2.3.1 Mixed-Integer Linear Programming (MILP)

A *linear program* (LP) maximizes a linear objective subject to linear equality and inequality constraints [55]. In standard form an LP can be written as

$$\begin{aligned} \max_{x \in \mathbb{R}^n} \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b, \\ & x \geq 0, \end{aligned} \tag{2.1}$$

where $c \in \mathbb{R}^n$ is the objective vector, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ encode the linear constraints, and the decision variables x are continuous and non-negative. In the battery setting the objective is typically trading revenue net of costs, and the constraints encode storage limits, charge and discharge limits, market-position limits, and energy balance.

A *mixed-integer linear program* (MILP) extends (2.1) by requiring some decision variables to take integer values:

$$\begin{aligned} \max_{x,\, y} \quad & c^\top x + d^\top y \\ \text{s.t.} \quad & Ax + By \leq b, \\ & x \in \mathbb{R}^n_{\geq 0}, \quad y \in \mathbb{Z}^k_{\geq 0}. \end{aligned} \tag{2.2}$$

The continuous block x behaves exactly as in the LP. The integer block y adds discrete choices. Dropping the integrality requirement $y \in \mathbb{Z}^k_{\geq 0}$ recovers an LP, usually called the *LP relaxation* of the MILP. The LP relaxation has the same objective and constraints but allows fractional values for y, so it is at least as easy to solve and provides an upper bound on the MILP optimum. MILP solvers exploit this bound through *branch-and-bound*: they recursively partition the integer search space and use the LP relaxation at each node to prune branches that cannot improve on the best integer solution found so far [55].

MILP variables are useful whenever the model must represent discrete choices. In this thesis they appear for two reasons. First, market orders can only be executed in multiples of the minimum tradable lot, so the executed volume cannot be treated as fully continuous. Second, the battery cannot charge and discharge the same product simultaneously, so the formulation needs a binary mode variable $\alpha \in \{0, 1\}$ to select one physical operating direction. These integer variables make the model more realistic but also harder to solve. MILP is NP-hard [55], with worst-case solver time growing exponentially in the number of integer variables, in contrast to the LP

relaxation, which is solved in polynomial time. A MILP solver may therefore need to explore many combinations of integer decisions before proving optimality, which is why the MILP is valuable as an exact reference model but too slow to call at every order-book event in a high-frequency backtest.

2.3.2 Dynamic Programming (DP)

Dynamic programming [56] is a way of solving sequential decision problems by breaking them into stages. At each stage, the controller observes a state, chooses an action, receives an immediate reward, and moves to a new state. The key object is the value function, which stores the best achievable future reward from each state. It satisfies the Bellman equation,

$$V_t(s) = \max_{a \in \mathcal{A}_t(s)} \Big\{ r_t(s,a) + V_{t+1}\Big(F_t(s,a)\Big) \Big\}, \tag{2.3}$$

where s is the current state, a is an admissible action, r_t is the immediate reward, F_t is the state-transition function, and V_{t+1} is the continuation value at the next stage. The recursion is solved backwards from a terminal condition, and an optimal trajectory is then reconstructed forwards from the initial state.

For battery trading, the natural state is the state of charge, the action is the net position chosen for a delivery product, and the stage index is the sequence of tradable products. The main computational benefit is that the dynamic program avoids repeatedly solving the full MILP. The trade-off is between approximation and computational efficiency: because the battery state is continuous, the value function has to be represented on a finite storage grid, and off-grid states must be approximated by interpolation.

2.3.3 Numerical and computational concepts

Four numerical and computational concepts recur throughout the rest of the thesis: discretization of a continuous state space, linear interpolation between grid points, Brent's method for one-dimensional optimization, and Amdahl's law for analyzing parallel speedup. The first three are standard numerical-analysis tools [57]; the last is a classical result in parallel computing [58]. We also use the standard $\mathcal{O}(\cdot)$ notation for computational complexity (e.g. a runtime of $\mathcal{O}(TmA)$ grows in proportion to the number of stages T, the number of storage-grid points m, and the number of feasible actions A per grid point). To mark infeasible states inside a maximization, we use the sentinel value $-\infty$, so that any feasible alternative wins the max in the Bellman equation (2.3). The implications of this convention under interpolation are discussed in Section 2.4.6.

2.3.3.1 Discretization

Discretization replaces a continuous domain $\mathcal{S} \subset \mathbb{R}$ with a finite grid

$$G = \{s_1, s_2, \ldots, s_m\} \subset \mathcal{S},$$

on which a function $f : \mathcal{S} \to \mathbb{R}$ is evaluated only at the grid points. For the equidistant grids used in this thesis,

$$s_i = s_{\min} + (i-1)\,h, \qquad h = \frac{s_{\max} - s_{\min}}{m-1}, \qquad i = 1, \ldots, m, \tag{2.4}$$

where h is the grid spacing. The storage cost is $\mathcal{O}(m)$ values and the cost of evaluating f on the grid is $\mathcal{O}(m)$ function calls. A finer grid (smaller h, larger m) reduces discretization error at the cost of additional memory and runtime.

2.3.3.2 Linear interpolation

When the system reaches a state $s \notin G$ that lies between two grid points s_i and s_{i+1}, the value at s is approximated by linear interpolation:

$$\tilde{f}(s) = (1-w)\,f(s_i) + w\,f(s_{i+1}), \qquad w = \frac{s - s_i}{s_{i+1} - s_i} \in [0,1]. \tag{2.5}$$

For functions $f \in C^2(\mathcal{S})$ the pointwise error of the piecewise-linear interpolant on a uniform grid of spacing h obeys the standard bound [57]

$$\left|f(s) - \tilde{f}(s)\right| \;\le\; \frac{h^2}{8} \max_{\xi \in [s_i, s_{i+1}]} \left|f''(\xi)\right|, \tag{2.6}$$

i.e. the interpolation error is second-order in the grid spacing. The DP implementation studied here uses linear interpolation because it is fast, simple, and consistent with the reference paper.

2.3.3.3 Brent's method

Brent's method [59, 60] is a derivative-free algorithm for locating an extremum of a scalar function $f : [a,b] \to \mathbb{R}$ on a bracket that contains the extremum. It combines two ingredients.

Golden-section search is a derivative-free bracket-shrinking method for a unimodal scalar function. Given a current bracket $[a,b]$ with interior point c, the next evaluation is placed at the point that divides the larger sub-interval in the golden-ratio proportion:

$$x = \begin{cases} c + (1 - 1/\varphi)\,(b-c), & \text{if } (c-a) < (b-c), \\ c - (1 - 1/\varphi)\,(c-a), & \text{otherwise,} \end{cases} \qquad \varphi = \tfrac{1+\sqrt{5}}{2}. \tag{2.7}$$

The bracket then shrinks by the fixed factor $1/\varphi \approx 0.618$ per iteration, giving linear convergence with exactly one new evaluation per step and no smoothness assumption on f.

Successive parabolic interpolation fits a quadratic through the three best points $(a, f(a))$, $(b, f(b))$, $(c, f(c))$ and steps to the analytic vertex of that quadratic:

$$u = b \; - \; \frac{1}{2} \frac{(b-a)^2\big[f(b)-f(c)\big] - (b-c)^2\big[f(b)-f(a)\big]}{(b-a)\big[f(b)-f(c)\big] - (b-c)\big[f(b)-f(a)\big]}. \tag{2.8}$$

This step converges superlinearly on smooth functions but can diverge or stagnate when the parabolic approximation is a poor fit.

Brent's method accepts the parabolic step (2.8) whenever (i) u lies inside the current bracket and (ii) the step is at most half the second-to-last step (so the bracket is provably shrinking). Otherwise it falls back to the golden-section step (2.7). The combination inherits the worst-case linear convergence of golden-section search and the superlinear local convergence of parabolic interpolation, and requires only function evaluations of f (no derivatives). It is therefore well suited to optimizing expensive black-box objectives such as the empirical reward of a backtested trading policy as a function of a single tuning parameter.

2.3.3.4 Amdahl's law

Amdahl's law [58, 61] bounds the speedup obtainable by distributing a fixed computation across N parallel processors. Let $s \in [0, 1]$ denote the fraction of the total work that is inherently serial, in the sense that it cannot be divided across processors, and let the remaining fraction $1 - s$ be perfectly parallelizable. The speedup relative to single-processor execution is then

$$S(N) \; = \; \frac{1}{s + (1-s)/N}.$$

As $N \to \infty$ the parallel term vanishes and the speedup saturates at the ceiling $S_\infty = 1/s$, so the serial fraction alone caps the attainable acceleration regardless of how many processors are added. The serial fraction s is not in general known in advance; it is estimated by fitting the measured speedup curve $S(N)$ to the expression above. A small estimated s locates the ceiling but does not by itself imply that a given configuration is near it, since the realized speedup at a finite, practically achievable N may lie well below S_∞. The law is applied in Section 3.6 to characterize the thread scaling of the DP kernel.

2.4 Reference paper and modeling assumptions

As outlined in the introduction, the purpose of this thesis is to replicate, verify, and adapt the model of Schaurecker et al. [1] to Flower's context. The modeling assumptions and formulations follow the reference paper closely. This section restates them for completeness before subsequent subsections introduce the extensions specific to our setup.

2.4.1 The intrinsic problem

We first present a simplified version of the MILP formulation for clarity, and then extend it to the full formulation and the rolling intrinsic algorithm. We call the simplified, linear problem the *intrinsic problem*. As outlined by [1], this strategy is widely applied in the literature on optimal storage and dispatch of natural gas [2, 3, 7, 62, 8]. The problem is set in a perfectly liquid futures market, trading $T \in \mathbb{N}$ futures of a storable commodity which trade for prices P_t. We assume that future contracts are for delivery in consecutive time periods $[0,1), [1,2), \ldots, [T-1,T)$, and that there are no overlapping contracts. This problem applies directly to a battery storage system trading futures for delivery in, for example, each quarter of the next day.

The battery operator maximizes income by deciding on what quantity q_t to buy or sell for each contract t in T. Volumes bought are transferred into the battery, while volumes sold are dispatched from the battery. Put differently, the battery operator chooses q_t to optimize the battery's storage level $(s_t)_{t=1}^T$ and the future positions $(f_t)_{t=1}^T$, which in turn maximizes the battery's income in the following relationship:

$$\begin{aligned} \max_{q_t} \quad & \sum_{t=1}^{T} P_t(-q_t) \\ \text{s.t.} \quad & f_t = f_t^0 + q_t, && \forall t = 1, \ldots, T \\ & f_t \in [\underline{f}, \bar{f}], && \forall t = 1, \ldots, T \\ & s_t = s_{t-1} + f_t, && \forall t = 1, \ldots, T \\ & s_t \in [0, \bar{s}], && \forall t = 1, \ldots, T, \end{aligned} \tag{2.9}$$

where s_0 is the initial storage level, f_0^0 is the initial future position, $\underline{f}$ and $\bar{f}$ are the lower and upper bounds on charging and discharging, and $\bar{s}$ is the upper bound on the storage level. A schematic of how the variables are related is shown in Figure 2.7.

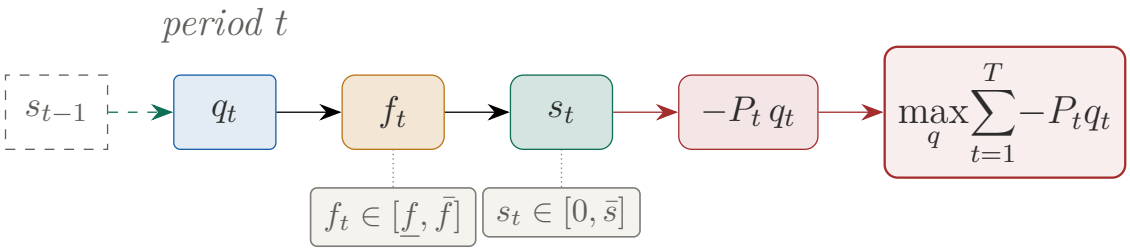


Figure 2.7: Schematic of one period of the MILP intrinsic problem (2.9). The decision q_t updates the future position f_t and the storage level s_t subject to the feasibility chips above each box; revenue accumulates as $-P_t q_t$ across all contracts. Green arrows mark the storage carry-over from one period to the next; the red arrow marks the cash-flow contribution to the objective.

The linear problem (2.9) is a simplification for the following reasons: it assumes a perfectly efficient storage system, in a perfectly liquid market, with no spreads,

and no cost of either operating the battery or trading futures. It is also myopic, optimizing only for the current period and current prices, without considering future prices or the future value of storage beyond the final available futures contract. The operator may realize more profit by waiting to trade a specific contract until the price moves to something more favorable, or by keeping storage in the battery beyond the available time horizon to capture future profits. These suboptimal properties motivate the name, taken from the intrinsic value of a financial option contract, which is the value of the option at the current time without considering the future value of the option.

Next comes an extended problem, intended to be more realistic about the mechanics of a continuous intraday energy market and a physical battery. This problem is still an intrinsic problem with the same myopic feature. The extension explicitly accounts for order book information, trading costs, charging and discharging efficiencies, and battery degradation. For an outline of limit order books, limit prices, and continuous intraday market mechanics, see Section 2.1.1.

For each contract $t \in T$, there is an order book $\mathcal{O}_t = \mathcal{O}_t^+ \cup \mathcal{O}_t^-$ with $\mathcal{O}_t^+$ containing all the currently active asks and $\mathcal{O}_t^-$ containing all the currently active bids for contract t. Each order $i \in \mathcal{O}_t$ has a limit price P_i and a quantity $Q_i > 0$. We introduce charge and discharge efficiencies $\eta^+, \eta^- \in (0, 1]$ because the battery cannot charge or discharge at 100% efficiency. We introduce a trading cost ν_{trade}, based on trading volume and incurred for each executed trade. We introduce a battery degradation cost ν_{deg}, incurred for each charge or discharge operation. This flat degradation cost is a major simplification, since a real battery degrades non-linearly and not only due to physical throughput (called cycling) [5, 6]. Batteries are typically replaced after a certain number of cycles [5, 6], so a linear degradation cost is a reasonable approximation. Both trading and degradation costs have the dimension €/MWh.

So far, every extension introduced on top of (2.9) preserves linearity, and the resulting problem can still be cast as a linear program. However, two characteristics of continuous intraday trading break this linearity and force a transition to a mixed-integer formulation.

The first source of non-linearity is that orders cannot be any arbitrary quantity. Instead, orders can only be matched in integer multiples of a minimum trading lot u, so the quantity executed against any order i must satisfy

$$q_i = k_i u, \qquad k_i \in \mathbb{N}_0, \tag{2.10}$$

which introduces integers to the optimization problem.

The second source of non-linearity arises from the interaction between sub-unit round-trip efficiencies and the possibility of negative prices. Consider a single product t together with two opposing orders $i \in \mathcal{O}_t^+$ and $j \in \mathcal{O}_t^-$. Suppose the operator buys a quantity q at the ask price P_i and immediately resells the energy that survives the round trip, $\eta^-(\eta^+ q)$, at the bid price P_j. Such a paired trade is profitable whenever

$$-qP_i + \eta^-(\eta^+ q)P_j > 0 \quad \Longleftrightarrow \quad (\eta^-\eta^+)P_j > P_i. \tag{2.11}$$

Since by construction $P_i > P_j$ for any matchable pair of ask and bid, inequality (2.11) can only be satisfied when both P_i and P_j are negative, in which case losing energy through the round trip is rewarded rather than penalized. Although such price configurations do occur in practice, the underlying trade is not physically realizable: a single battery cannot charge and discharge the same product simultaneously. The optimization model must therefore explicitly forbid these spurious arbitrage opportunities, which is enforced through binary indicator variables and is the second reason for adopting a MILP formulation.

With these extensions, the modified intrinsic problem becomes the following mixed-integer linear program:

$$
\begin{aligned}
\max_{f_t, s_t, \alpha_t, q_i, k_i} \quad & \sum_{t \in \mathcal{T}} \left(\sum_{i \in \mathcal{O}_t^-} (P_i - \nu) q_i - \sum_{i \in \mathcal{O}_t^+} (P_i + \nu) q_i \right) & \\
\text{s.t.} \quad & 0 \le q_i \le Q_i, & \forall i \in \mathcal{O}_t \\
& q_i = k_i u, & \forall i \in \mathcal{O}_t \\
& f_t^+ = \sum_{i \in \mathcal{O}_t^+} q_i, & \forall i \in \mathcal{O}_t^+ \\
& f_t^- = \sum_{i \in \mathcal{O}_t^-} q_i, & \forall i \in \mathcal{O}_t^- \\
& f_t = f_t^0 + f_t^+ - f_t^-, & \forall t \in \mathcal{T} \\
& f_t \in [\underline{f}, \bar{f}], & \forall t \in \mathcal{T} \\
& f_t = i_t - w_t, & \forall t \in \mathcal{T} \\
& i_t \in [0, \alpha_t \bar{f}], & \forall t \in \mathcal{T} \\
& w_t \in [0, -(1 - \alpha_t) \underline{f}], & \forall t \in \mathcal{T} \\
& s_t = s_{t-1} + \eta^+ i_t - \frac{1}{\eta^-} w_t, & \forall t \in \mathcal{T} \\
& s_t \in [0, \bar{s}], & \forall t \in \mathcal{T} \\
& \alpha_t \in \{0, 1\},\ k_i \in \mathbb{N}_0,\ \nu \in \mathbb{R}_{\ge 0}, & \forall t \in \mathcal{T}.
\end{aligned}
\tag{2.12}
$$

Here, $\nu := \nu_{trade} + \nu_{deg}$ is the combined per-unit cost incurred on every executed trade, lumping the trading fee and the linearized degradation cost introduced above. The variables f_t^+ and f_t^- aggregate, for each product t, the total volume bought and sold across the matched orders, and decompose the net future position f_t into its purchase and sale components. The pair (i_t, w_t) splits this net position into a non-negative charging flow i_t (energy injected into the battery) and a non-negative discharging flow w_t (energy withdrawn from the battery). The binary variable $\alpha_t \in \{0, 1\}$ acts as a mode selector: when $\alpha_t = 1$ the bounds force $w_t = 0$ and only charging is allowed, while $\alpha_t = 0$ forces $i_t = 0$ and only discharging is allowed. Finally, the storage update equation applies the charge efficiency η^+ to the inflow and the inverse of the discharge efficiency η^- to the outflow, so that round-trip losses are correctly attributed to the energy stored.

2.4.2 The rolling intrinsic algorithm

As discussed in the reference paper [1], the *rolling intrinsic* policy was originally introduced by Gray and Khandelwal [3] as a dynamic extension of the static intrinsic value formulated in (2.12). The idea is straightforward: rather than solving the intrinsic problem once and committing to its solution for the remainder of the trading horizon, the operator re-solves it repeatedly as the market evolves. Each re-run is initialized with the current storage state s_0 and the future positions $(f_t)_{t\in\mathcal{T}}$ accumulated in previous periods, and uses the order books observed at that moment to search for any opportunities to profitably rebalance the existing portfolio. The full procedure is summarized in Algorithm 1.

Although every individual decision produced by the rolling intrinsic remains myopic, the policy is a clear improvement over the static formulation, which never adapts to new information once an initial position has been taken. A useful consequence of the myopic structure is that the rolling intrinsic does not speculate on future price movements: it only commits to rebalancing trades that are already profitable at current prices, and therefore cannot accumulate paper losses from positions that fail to materialize. Together with its conceptual simplicity and relatively modest computational cost, these properties make the rolling intrinsic a natural baseline policy in the storage-DP literature [3, 7].

Algorithm 1 The rolling intrinsic

Require: Initial storage s_0, start time t_{start} and stop time t_{end} of the simulation, limit order book information $\mathcal{T}_{t^*}$ for all $t^* \in [t_{start}, t_{end}]$, solve frequency of the intrinsic, and other parameters $(\bar{f}, \underline{f}, \nu, \ldots)$.

Ensure: Final battery schedule over the simulation runtime and cumulated trade profits.

for each time instant $t^* \in [t_{start}, t_{end}]$ **do**
  Solve the intrinsic problem according to (2.12).
  Update all positions f_t, $\forall t \in \mathcal{T}_{t^*}$.
  Log profits/losses from trading.
end for

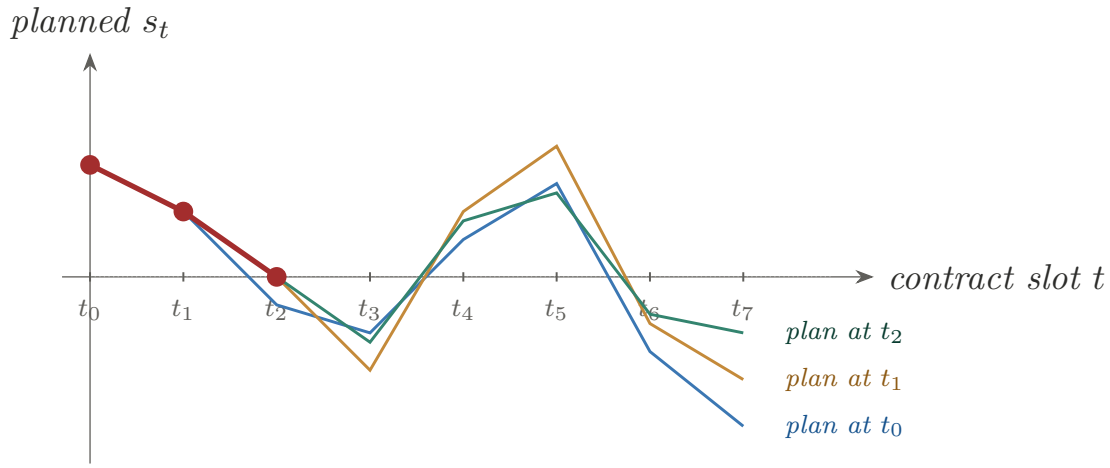


Figure 2.8: Rolling intrinsic algorithm: at each new LOB tick t_k the intrinsic problem (2.9) is re-solved on the remaining horizon, producing an updated plan (blue/amber/green). Only the immediate action is executed; the rest of the plan is discarded and replaced by the next re-solve. The realized trajectory (red) is the concatenation of first-period actions across re-solves.

2.4.3 DP formulation of the rolling intrinsic problem

The MILP formulation (2.12) is exact, but solving it from scratch every time the order book moves becomes prohibitive. As discussed in [1], this motivates an alternative formulation of the same problem as a sequence of dynamic programming (DP) subproblems. These can be solved by a customized DP algorithm that is in practice orders of magnitude faster than calling a general-purpose MILP solver (see Table 4.7 for the MILP-versus-DP timing comparison on our setup).

The reformulation rests on the observation that the products $t \in \mathcal{T}$ can be treated as the stages of a dynamic program, with the storage level s_t acting as the state variable that links one stage to the next. The decisions q_i, k_i, α_t, and f_t are the actions taken at stage t, and the resulting storage level enters the value function of stage $t+1$. Although in reality the decisions for all products are made simultaneously, this artificial sequencing is what unlocks the DP machinery and the associated speedup.

Without loss of generality, we assume $\mathcal{T} = \{1, \dots, T\}$, set the terminal value function $V_{T+1} \equiv 0$, and for $t = 1, \dots, T$ define

$$V_t(s_{t-1}) = \begin{cases} \max\limits_{f_t, s_t, \alpha_t, q_i, k_i} & \sum\limits_{i \in \mathcal{O}_t^-} (P_i - \nu) q_i - \sum\limits_{i \in \mathcal{O}_t^+} (P_i + \nu) q_i + V_{t+1}(s_t) \\ \text{s.t.} & \text{constraints of (2.12) restricted to product } t. \end{cases} \tag{2.13}$$

The objective collects the per-product cash flow at stage t and adds the value-to-go $V_{t+1}(s_t)$, while the constraints are exactly the ones already introduced for the MILP, restricted to a single product.

To turn (2.13) into something that can be solved efficiently, we discretize the action f_t in steps of $\kappa \cdot u$, where $\kappa \in \mathbb{N}$ and u is the minimum tradable lot. We further introduce

a function π_t that encodes the current order book of product t and returns the cost or revenue of buying or selling on the intraday market, including the combined per-unit cost $\nu = \nu_{trade} + \nu_{deg}$. Finally, we define the state transition function

$$S(s_{t-1}, f_t) = \begin{cases} s_{t-1} + \eta^+ f_t, & \text{if } f_t > 0, \\ s_{t-1} + \dfrac{f_t}{\eta^-}, & \text{otherwise,} \end{cases} \tag{2.14}$$

which applies the charging efficiency η^+ on inflows and the inverse of the discharging efficiency η^- on outflows, mirroring the storage update of the MILP. With these ingredients, (2.13) can be rewritten as

$$V_t(s_{t-1}) = \begin{cases} \max\limits_{k_t} \;\; \pi_t(f_t^0 - f_t) + V_{t+1}\Big(S(s_{t-1}, f_t)\Big) \\ \;\; \text{s.t.} \;\; f_t = f_t^0 + k_t u, \\ \qquad k_t \in \left[-\dfrac{\eta^- s_{t-1} + f_t^0}{u}, \; \dfrac{\bar{s} - s_{t-1}}{\eta^+ u} - \dfrac{f_t^0}{u} \right] \\ \qquad\quad \cap \left[\dfrac{\underline{f} - f_t^0}{u}, \; \dfrac{\bar{f} - f_t^0}{u} \right] \cap \kappa\mathbb{Z}. \end{cases} \tag{2.15}$$

where the first interval enforces the storage energy bounds and the second enforces the per-product power bounds. Given V_{t+1}, evaluating (2.15) reduces to looping over a small number of feasible k_t values and picking the best one, i.e., to a handful of arithmetic operations per stage.

The reformulation from (2.13) to (2.15) automatically takes care of the simultaneous charge–discharge issue discussed around (2.11): each stage commits to a single signed action f_t, so charging and discharging the same product at the same time is no longer expressible. By fixing the trading step-size u to a multiple of the minimum bid size, all feasible k_t automatically correspond to feasible position sizes on the exchange, satisfying the integrality constraint.

To solve (2.15) we proceed by backward induction in the standard DP fashion. Starting from the boundary condition $V_{T+1} \equiv 0$, we compute V_T, then V_{T-1}, and so on. A complication arises here because $\eta^{\pm} < 1$, which means that even when f_t is an integer multiple of κu, repeated injections and withdrawals translate into storage levels that do not sit on a regular grid. V_t has to be defined on the entire interval $[0, \bar{s}]$ rather than on a finite domain. To make this tractable, we discretize the storage interval to a finite grid $G \subset [0, \bar{s}]$ with $|G| = m$, evaluate the value function on G, and linearly interpolate for points $s \notin G$. Denoting the resulting approximation by $\tilde{V}_{t+1}$, for two neighboring grid points $s_i, s_{i+1} \in G$ and $s_i \le s_t \le s_{i+1}$ we set

$$\tilde{V}_{t+1}(s_t) = \frac{m(s_{i+1} - s_t)}{\bar{s}} V_{t+1}(s_i) + \frac{m(s_t - s_i)}{\bar{s}} V_{t+1}(s_{i+1}). \tag{2.16}$$

The choice of a linear interpolation is deliberate. In principle, more sophisticated schemes that exploit the curvature of the value function induced by the buy and sell price stacks could be used. The reference paper experiments with quadratic

interpolations and cubic splines and reports that these alternatives do not yield improvements over the linear scheme (2.16), while being computationally more expensive [1]. We also tested quadratic and cubic interpolations on our setup and confirmed this finding: neither yielded a measurable improvement over the linear scheme, and both came at a higher per-stage cost. The linear interpolant is retained throughout. Figure 2.9 shows the size of the approximation error on a coarse and a fine grid: piecewise-linear interpolation systematically underestimates the continuation value, particularly where the curvature is high.

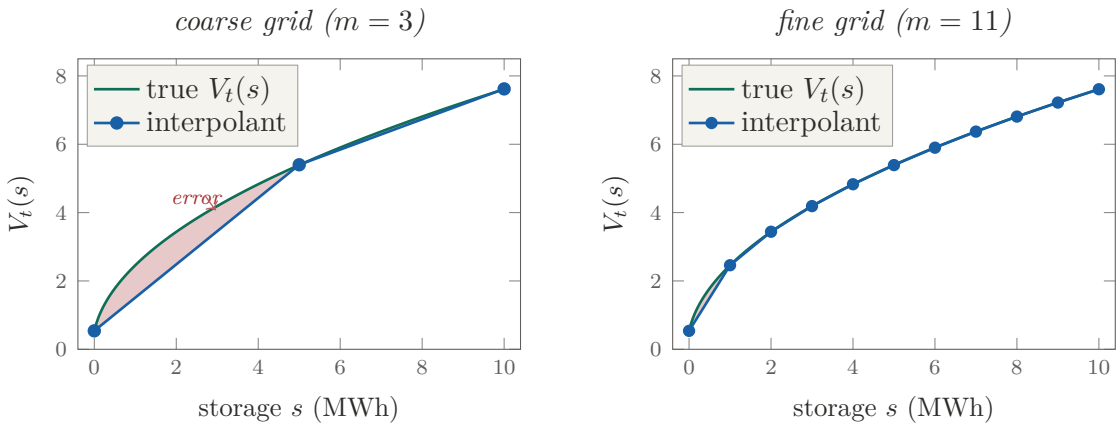


Figure 2.9: Piecewise-linear interpolation error of the value function on the same 10 MWh battery, comparing a coarse grid ($m = 3$, left) and a fine grid ($m = 11$, right). The shaded red region between the true $V_t(s)$ (green) and the interpolant (blue) is much larger on the left: coarse grids systematically underestimate the continuation value, particularly where curvature is high.

The linear interpolation (2.16) is not strictly necessary: [1] show in their Proposition 1 that, for every t, there exists a finite grid G_t that contains all storage states reachable at the beginning of period t. Choosing this grid to discretize the value functions removes the need for interpolation and makes the DP solution exact and equivalent to the MILP formulation. The price of exactness, however, is a very fine grid whenever there are many tradable products or whenever $\eta^{\pm} < 1$, which quickly becomes computationally heavy. We follow [1] in keeping a fixed, equidistant grid G across all stages and all calls of the intrinsic, accepting the resulting approximation error in exchange for a controllable runtime. In principle, the grid G could be chosen independently for every t and could vary between successive solves of the intrinsic. We use a constant equidistant grid throughout, i.e., $G_{t^*,t} \equiv G$. Non-uniform and adaptive grids are among the alternatives reviewed in Section 2.4.6.

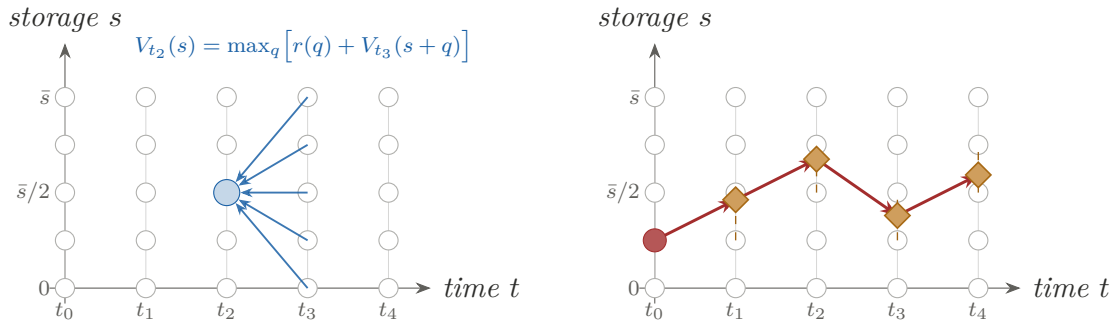
Algorithm 2 summarizes one solution run of the intrinsic problem at runtime t^*, where $\mathcal{X}_t(s, f_t^0)$ denotes the set of feasible actions k_t defined by the constraints of (2.15). It consists of two passes. In the backward pass, the optimal action and value $V_t(s)$ are computed at each storage grid point, for every stage t. In the forward pass, the initial storage level s_0 is taken as input, and the optimal trajectory is reconstructed using the value functions tabulated by the backward pass. If the

action lattice (integer multiples of κu) aligns with the storage grid G and $\eta^{\pm} = 1$, then the forward pass would be unnecessary, since every reachable storage state already lies on G and the backward pass has computed the optimal action and the resulting profits at each grid point. However, non-unit efficiencies cause the realized trajectory to traverse off-grid storage levels, requiring the forward pass. Both passes of Algorithm 2 are visualized in Figure 2.10.

Algorithm 2 DP at runtime t^*

Require: f_t^0, $\mathcal{O}_t$ for all tradable products $t \in \mathcal{T}$, other parameters $(\kappa, s_0, \nu, G, \ldots)$.
Ensure: Updated market positions f_t.

```
Set V_{T+1} ≡ 0.
Backwards Pass
for t = T, ..., t_0 do
    for each state s ∈ G do
        V_t(s) ← max_{k_t ∈ X_t(s, f_t^0)} { π_t(k_t u) + Ṽ_{t+1}( S(s, k_t u) ) }
    end for
end for
Forward Pass
for t = t_0, ..., T do
    f̂_t ← arg max_{k_t ∈ X_t(s, f_t^0)} { π(k_t u) + Ṽ_{t+1}( S(s_{t-1}, k_t u) ) }
    f_t^0 ← f̂_t
    s_t ← s_{t-1} + f̂_t
end for
```



(a) **(b)**

Figure 2.10: (a) Backward pass: $V_t(s)$ at each grid node is set to the best total return – immediate reward plus future value – achievable by any action q leading from (t, s) into a state on the next stage's grid; the highlighted blue node is the cell whose value is being computed, and the blue arrows are the candidate actions feeding it. (b) Forward pass with $\eta^{\pm} < 1$: the realized storage trajectory (red nodes, thick red arrows) lands between grid rows at off-grid positions (amber diamonds), so V_t at those off-grid states is queried by linear interpolation between the two enclosing rows.

2.4.4 DP scaling and computational considerations

The backward pass of Algorithm 2 runs in $\mathcal{O}(T \cdot m \cdot A)$ time, where T is the number of tradable contracts (the DP stages), $m = |G|$ is the number of storage grid points, and A is an upper bound on the cardinality of the feasible action set $\mathcal{X}_t(s, f_t^0)$ at any node, bounded above by $A \leq (\bar{f} - \underline{f})/(\kappa u)$. The forward pass visits each stage exactly once and therefore scales as $\mathcal{O}(T \cdot A)$, so the backward pass dominates the per-solve cost.

Relative to the configuration studied in the reference paper [1], three changes in our setting inflate this dominant backward-pass cost. First, as outlined in Section 2.1.1, we focus on quarter-hour contracts rather than hourly ones, which multiplies the number of stages T by four. Second, because the minimum trading size is denoted in power terms (0.1 MW), the per-contract minimum trading quantity falls from 0.1 MWh in the hourly case to 0.025 MWh in the quarter-hour case. With the same action step u this multiplies the action-set size A by four. Third, to keep the storage grid at a comparable discretization given the smaller minimum trading quantities, the number of grid points m has to quadruple. Combined, the three factors yield a $4^3 = 64$-fold increase in the backward-pass cost. Chapter 4 reports that further increases in m were necessary, raising the computational burden of backtests further.

2.4.5 Spread-penalty parameter for the rolling intrinsic

The rolling intrinsic is myopic by construction: each re-solve optimizes against the order book observed at the current moment, with no anticipation of how that order book will evolve. As documented in the reference paper [1], a practical consequence is that the policy tends to trade illiquid contracts *too early*, accepting wide spreads long before gate closure when waiting for the book to thicken would have yielded better fills. To counter this, [1] introduces a single scalar parameter $\phi \geq 0$ that discourages exactly this behavior by attaching a cost to trades on wide-spread products.

The mechanism is to replace the per-stage cash flow π_t in the objective function of (2.15) with a spread-penalized cash flow

$$\hat{\pi}_t(q) \;=\; \pi_t(q) \;-\; \phi \cdot \delta_{t^*,t} \cdot |q|, \tag{2.17}$$

where $q = f_t^0 - f_t$ is the trade size at stage t, $\delta_{t^*,t}$ is the bid–ask spread of contract t as observed at the current decision time t^*, and ϕ is a unitless scalar tuning parameter. The penalty term $\phi \cdot \delta_{t^*,t} \cdot |q|$ scales jointly with the spread of the targeted contract, the size of the planned trade, and the tuning parameter. Setting $\phi = 0$ recovers the unmodified rolling intrinsic.

Intuitively, ϕ shifts the planner's preference from *trade now* to *trade later*. Early trades on wide-spread contracts become more expensive in the planner's internal accounting, so the planner postpones them on the expectation that the same contract will be available at a tighter spread closer to gate closure. The choice of ϕ thus trades off two competing pressures: the original cash flow π_t rewards executing as soon as a profitable opportunity appears, while the penalty rewards waiting until the book is sufficiently liquid.

Because ϕ has no model-implied value, it must be tuned empirically. One natural choice, suggested by [1], is to treat the realized reward of the policy over a tuning window as a black-box function $f(\phi)$ and apply Brent's method (Section 2.3.3.3) to find its maximum. The choice of tuning window and the schedule by which the resulting ϕ^* is applied to subsequent trading are practical decisions outside the scope of this section.

2.4.6 Infeasibility handling

The DP formulation in (2.15) treats the storage envelope $s_t \in [0, \bar{s}]$ and the per-product power envelope $f_t \in [\underline{f}, \bar{f}]$ as hard constraints. These constraints interact with the pre-existing futures positions f_t^0 carried by the rolling intrinsic from one re-solve to the next, and with the non-unit efficiencies $\eta^{\pm} < 1$. Some grid points s at some stages t are simply unreachable: no admissible sequence of actions can take the battery from such a state to a feasible terminal state without violating a bound at an intermediate stage. The standard encoding is to set $V_t(s) = -\infty$ at every such grid point, so any infeasible action loses the max against any feasible alternative.

The $-\infty$ encoding does not cause problems inside the max itself, but it does under interpolation. Because $\eta^{\pm} < 1$, the realized storage state $S(s, k_t u)$ generally does not lie on the grid G, so the DP queries $\tilde{V}_{t+1}$ at off-grid arguments. If one enclosing grid point is infeasible and the other is finite, the linear-advance form $V_{t+1}(s_i) + w\,(V_{t+1}(s_{i+1}) - V_{t+1}(s_i))$ used by the kernel for performance (Section 3.6) evaluates to $(-\infty) + (+\infty) = \text{NaN}$. By the IEEE-754 floating-point standard, every comparison with NaN returns false, so the max silently picks an unpredictable action.

We believe the reference paper [1] addresses this by replacing $-\infty$ entries with finite values *before* any interpolation. We could not confirm this from the published paper, but an error message observed when running their binary suggests the implementation sweeps the V_t table left to right and then right to left, replacing each $-\infty$ entry with its nearest finite neighbor. The procedure runs in $\mathcal{O}(m)$ per stage and eliminates NaN propagation, at the cost of an optimism bias: the DP treats infeasible states as roughly worth their nearest feasible neighbor, so each backward step injects a small amount of optimism that accumulates over the horizon [63, 64].

We do not adopt nearest-finite filling in this thesis. In our own testing, the optimism bias it introduces yielded worse results than letting the NaN values propagate through the value table. Figure 2.11 shows the gap that filling would create: past the infeasibility cliff at $s^\star$, the true value function is $-\infty$, but nearest-finite filling replaces it with the value at the last feasible state.

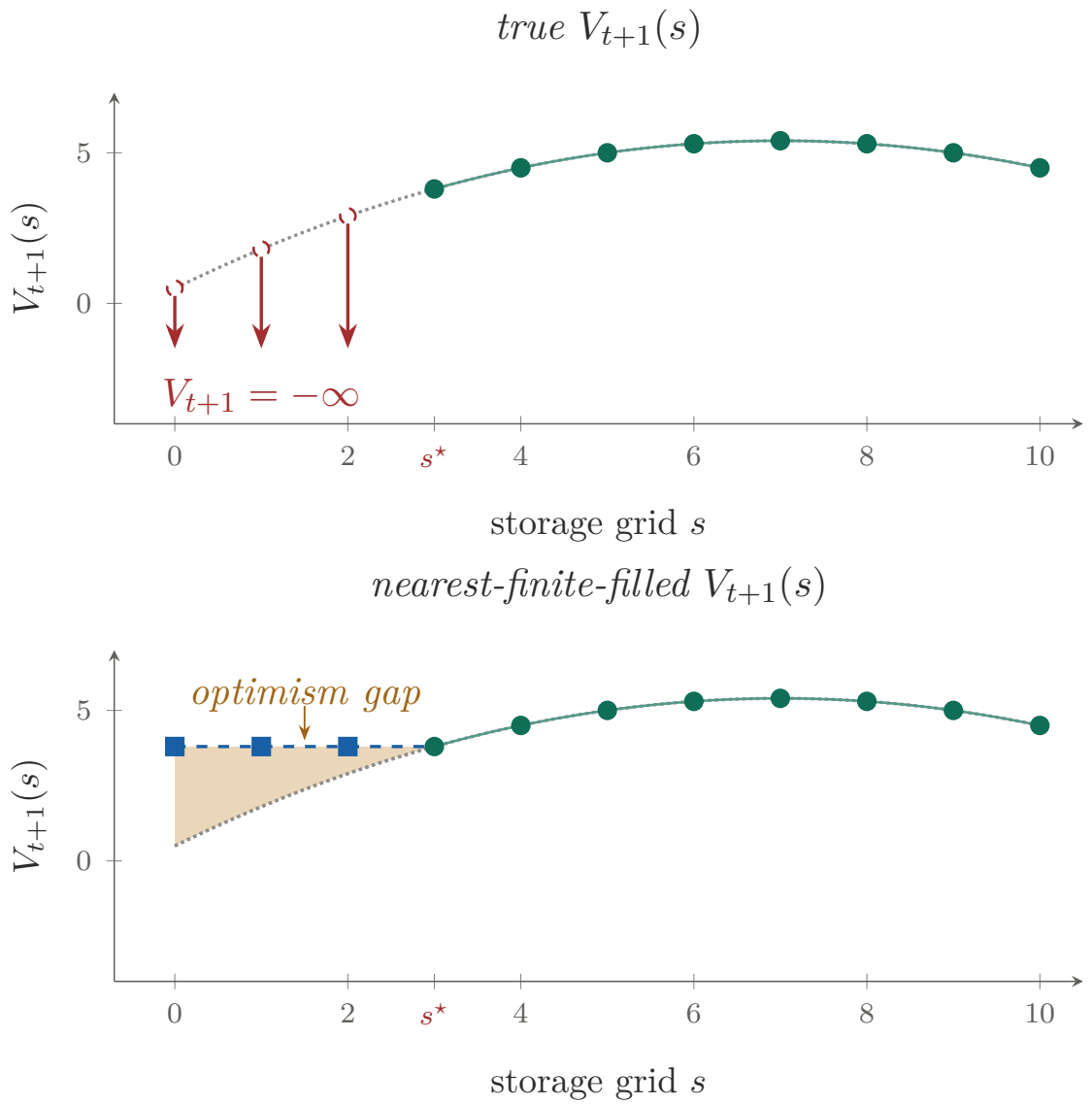


Figure 2.11: Optimism bias from nearest-finite filling on an 11-point storage grid. Here $s^\star$ is the first feasible storage cell; cells at $s < s^\star$ are infeasible. **Top:** the true V_{t+1} is finite on the feasible region (green) and $-\infty$ at every infeasible cell (red arrows). The dotted gray curve shows the counterfactual: what V_{t+1} would be at the infeasible cells if they were feasible. **Bottom:** nearest-finite filling replaces each infeasible entry with $V(s^\star)$ (blue squares). The shaded amber region is the resulting optimism gap, which sits above the counterfactual curve and is therefore doubly optimistic.

Several alternatives have been proposed. *Honest interpolation* [65] guards the interpolation routine to return $-\infty$ whenever an enclosing point is infeasible, at the cost of a smaller effective action set. *Soft constraints* [66] replace the hard bounds with smooth quadratic penalties added to the per-stage reward, at the cost of an optimum that can drift slightly outside the physical envelope. *Non-uniform grids* [64] place denser points near the storage boundaries where the infeasibility cliffs cluster. *Fea-*

sibility bookkeeping [63] propagates an explicit feasibility flag alongside each value, at the cost of doubling the memory footprint. Each represents a different point on the fidelity–cost frontier.

3

Methods

This chapter outlines the methods used to implement and study the rolling intrinsic policy. It starts with a description of the data, followed by the mathematical model and the key assumptions, and then a description of how the algorithm and the backtests were implemented, including the relevant scripts and how they relate to each other. The process used to create the results presented in chapter 4 is outlined next. The chapter closes with a discussion of the high-performance computing techniques developed to keep backtest runtimes tractable.

3.1 Data

In this section, the data is described. The raw data is first presented, followed by the precomputing process and the resulting precomputed data.

3.1.1 Raw data

The dataset comprises continuous intraday order-book messages from the Nord Pool exchange covering three bidding areas: the Swedish SE3 and SE4 price zones from April 1st 2025 through March 13th 2026, and the German market (Amprion control area) for the full calendar year of 2025 and through 24 February 2026. The present study is restricted to only the quarter-hourly contracts for all three areas.

Each file of the raw data contains a single delivery contract, meaning one fifteen-minute delivery slot on a specific calendar day. Within a calendar day there are ninety-six such files per area, one for every quarter-hour from 00:00 to 23:45. A contract is identified by its delivery date and by the index of the quarter-hour within the day.

The contents of a contract file are a chronologically ordered log of order-book revisions. Each row describes a single order at a particular revision and carries:

- the UTC time at which the revision was published (`revision_time`)
- an exchange-assigned monotone version number (`revision_number`)
- the side of the book (buy or sell)
- an order identifier that is stable for the lifetime of the order
- the limit price in EUR/MWh

- the resting quantity in MW
- two lifecycle timestamps recording when the order was first created and last updated.

A row whose quantity is zero records the cancellation of an order rather than a resting position.

The key structural property of this representation is that revisions are *sparse*: a revision contains only the orders whose state changed at that revision, not the entire book. If a single new buy order arrives at revision r and no other order is touched, the file records exactly one row tagged with `revision_number` $= r$. Recovering the full state of the book at revision r therefore requires replaying every revision up to r and maintaining a running snapshot keyed by order identifier: an order persists in the book until either a later revision modifies its price or quantity, or a later revision marks it as canceled. This is a compact and natural way of storing an order book that mutates near-continuously, but it has the operational implication that no consumer of the data can answer a point-in-time question about the book without first reconstructing it from the start of the contract. The pre-processing pipeline described in subsection 3.1.2 performs exactly this reconstruction once per contract and stores the result in a form that the backtest can subsequently replay event by event.

3.1.2 Precomputing pipeline and precomputed data

The raw data described above contains 4.1 billion order-book revision rows for SE3, 4.4 billion for SE4, and 7.8 billion for the German Amprion area, totaling 16.3 billion rows across the three datasets (see Appendix B for a full breakdown of dataset metrics). Replaying all of these revisions from scratch on every backtest run would be prohibitively slow: recovering the book state at any point in time requires processing every prior revision in sequence, and doing this repeatedly across 16.3 billion rows is not feasible within a research time frame. Instead, the raw data is processed once through a precomputing pipeline that restructures it into a form the backtest engine can consume directly. The pipeline serves a second, equally important purpose: it reduces the number of decision points the algorithm must act on from one per order-book revision to only those revisions where the market has moved enough to justify a re-solve. The pipeline consists of four sequential steps.

Step 1: Event extraction. The first step converts the raw revision log into a stream of individual order-book *events*. For each contract, revisions are replayed in order and the bid and ask snapshots are maintained as running dictionaries keyed by order identifier, exactly as described in subsection 3.1.1. Rather than storing the full snapshot at every revision, the pipeline takes the difference between consecutive snapshots and emits only what changed: each new order becomes an ADD event, each modified order a MODIFY event, and each canceled order a CANCEL event. The price and quantity of every event are stored as integers rather than floating-point numbers, by multiplying by fixed scale factors of 100 for price and 10 for quantity, so that one tick corresponds to 0.01 EUR/MWh and to 0.1 MW respectively. These scale factors match the granularity at which Nord Pool itself publishes order data: the

exchange does not quote quantities below 0.1 MW or prices below 0.01 EUR/MWh. Hence, the integer representation captures the raw data without loss of precision. Integer arithmetic is both faster than floating-point and free of rounding errors.

Step 2: Day-level packing. The second step organizes the events by calendar day. All events from every contract whose delivery date falls on a given day are merged into a single array, sorted by wall-clock time. The result is written to disk as a set of compact binary arrays, one file per day. The reason for this organization is that the backtest replays the market as a single continuous time series: it does not process one contract at a time but instead steps forward through real time, updating whichever contracts have an event at that moment.

Step 3: Relevance timeline. The third step computes a *relevance timeline* – the schedule of moments at which the rolling-intrinsic policy is re-solved. At every revision the pipeline computes, for each side of the book, the volume-weighted average price (VWAP) for a 1 MW trade: starting from the best available price, it fills quantity greedily across price levels until 1 MW is accumulated, and the average price paid is the VWAP. A revision is marked as *relevant* for a given contract if either the bid VWAP or the ask VWAP has moved by at least $\varepsilon = 0.10$ EUR/MWh compared to the previous relevant revision on that contract. We chose to use the VWAP as opposed to the best quoted price, which is used in the reference paper [1], because the best quoted price may reflect only a thin slice of liquidity, often from traders "testing the market" with very small quantities (see section B.2 for a quantification on the SE3 dataset). The union of relevant timestamps across all contracts forms the timeline: a list of moments at which at least one contract has seen a meaningful price move and a re-solve is appropriate.

Step 4: Manifest and metadata. The fourth and final step writes two index files that summarize the dataset at different granularities. A manifest indexes the dataset by delivery day, recording the path, event count, and time range of each day's binary file. A metadata table indexes the dataset by contract, recording the delivery start time and event range of each individual quarter-hour contract. Both files are loaded into memory at the start of every backtest run and serve as in-memory lookup tables thereafter, so that frequently asked questions – which file holds a given day, when does a given contract deliver – can be answered without scanning the bulk event data.

Beyond the precompute pipeline itself, the event-driven timeline is also downsampled into a set of fixed timelines for the timeline comparison reported in chapter 4. This is performed by a separate set of scripts: a general script living under `scripts/` and per-market Modal jobs under `modal/`, rather than by the precompute pipeline above. The timelines are produced at uniform intervals of 6 seconds, 1 minute, 5 minutes, 15 minutes, and 60 minutes, where each bucket retains only the last event-driven step that fell within it. A fixed timeline at interval Δt therefore represents a strategy that re-solves the DP at most once every Δt regardless of market activity, but does not re-solve within a quiet period even if Δt has elapsed without any relevant event.

3.2 Mathematical model and key assumptions

Both the MILP and the DP formulations of the intrinsic problem are implemented as standalone solvers. The mathematical model for each formulation is represented in Equations 2.12 and 2.15, respectively. Additionally, there are four modeling choices which differ from the reference paper: event relevance, latency, fill policy, and the overall profit calculation.

As described in Section 3.1.2, an event is considered relevant if either the bid VWAP or the ask VWAP has moved by at least $\varepsilon = 0.10$ EUR/MWh compared to the previous relevant revision on that contract. In the reference paper, any change in the best bid or best ask is considered relevant.

To create realistic backtests, a delay between the solver receiving an event and trades being executed is introduced. The reference paper models this delay as the sum of actual solver execution time and a fixed *technical delay*. Here, latency is modeled as a fixed technical delay only, on the grounds that execution latency on continuous intraday markets is itself variable and not predictable in advance [67]. Additionally, during this technical delay, the solver is "frozen" and cannot trade on any events that happen to arrive within this time frame.

Consequently, the introduction of latency necessitates an explicit fill policy. Latency introduces the possibility of targeted bids and asks being unavailable once an order is processed. The reference paper uses an all-or-none fill policy, where the order is either filled completely or not at all. Here, planned trades are re-checked after the delay, and filled greedily against any orders resting at the original limit price or better; partial fills are accepted and the unfilled remainder is discarded.

The fill is also recorded in a per-contract *consumed-liquidity overlay* keyed by resting-order identifier. The overlay is maintained inside the C++ book manager and subtracted from the packed book at every subsequent step until the contract is deactivated. A simulated buy depletes the asks it took; a simulated sell depletes the bids. The strategy therefore cannot re-trade a resting order it has already (partially) consumed, and large fills walking deep into the book leave a thinner book for the next re-solve. Two assumptions remain: the historical event stream is still replayed independently of our trades (i.e. other participants do not react to our footprint), and queue position within a price level is not modeled.

Finally, while degradation cost is included in the optimization objective to discourage excessive cycling, it is excluded from the reported profit figures. This differs from the reference paper, which subtracts degradation costs from profits. The motivation is practical: the reported figures reflect actual cash revenues and fees only, since degradation is a model-dependent estimate rather than a directly observable cost. Keeping it inside the optimizer is nonetheless necessary, because without a cycling deterrent the policy would chase razor-thin spreads through degradation-heavy trades, distorting the cycling profile in a way that could not later be reconciled with any *ex post* degradation accounting.

3.3 Implementation of algorithm and backtests

This section describes how the rolling intrinsic strategy is realized in software. The implementation splits naturally into two layers: a *solver stack* that performs the DP optimization problem, and a *backtest stack* that drives the solver through market time and records outcomes.

3.3.1 Solver stack

The solver stack returns an action for each active contract given a market snapshot. Its interface is a single function call that accepts four groups of inputs:

- a list of order-book snapshots, one per active contract sorted by delivery time
- battery parameters (capacity, charge/discharge rate limits, round-trip efficiencies)
- market parameters (time-step length, trading fee, degradation cost, minimum lot size)
- DP parameters controlling battery grid size, minimum action step, and terminal value.

As output, it returns the optimal trade for each contract, the resulting state-of-charge path, and the solver's estimate of the expected forward value of the current battery state.

The DP kernel is written in C++ and compiled into a binary extension module that Python can import and call directly, with no subprocess overhead. It implements the backward and forward pass described in Equation 2.15. The backward pass is processed in reverse order from the last active contract to the current step. For each stage t and for each grid point s_i, the solver enumerates all feasible actions q in multiples of u, looks up the cash flow from the order book, adds the discounted future value by linear interpolation of V_{t+1}, and stores the maximizing value and action. The forward pass then starts at the current state of charge and walks back through the stages, reading off the action stored at each visited grid point to produce the trade vector and the realized state-of-charge path returned to the caller.

The MILP formulation described in Equation 2.12 is implemented as a separate solver, with the MILP solved using Gurobi v11.0.3 for C++ [68]. MILP-specific settings (relative gap, thread count, warm-start behavior) live in a fifth configuration file, `milp.toml`, which is consumed only by this path. The settings used for the MILP-vs-DP comparison in subsection 4.1.4 follow the configuration of Schaurecker et al. [1]: a two-second wall-clock time limit per solve, a 10% relative MIP gap, and a warm start from the null action (i.e. the current battery schedule with no change). The overall structure of the solver stack – the three DP input groups (solver settings, order-book snapshots, and the per-call state from `BacktestState`), the Python bindings, the C++ DP kernel, and the MILP kernel – is summarized in Figure 3.1.

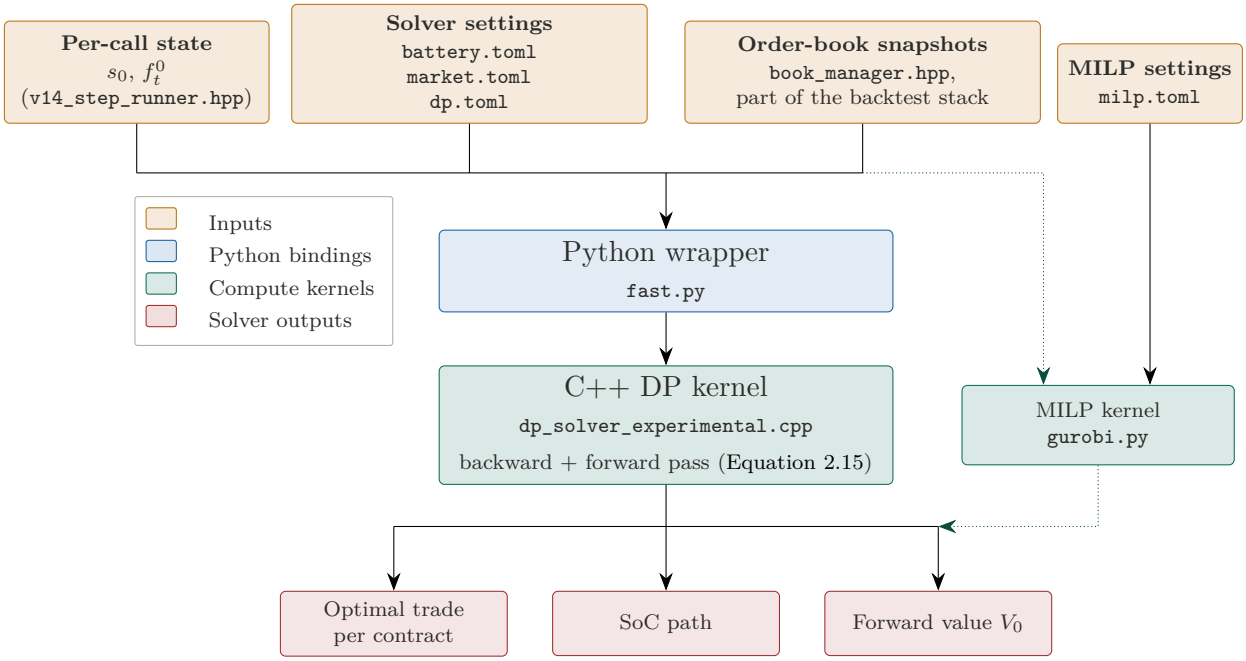


Figure 3.1: Software architecture of the solver stack, including filenames. The DP path receives three input groups: the static *solver settings* (`battery.toml`, `market.toml`, `dp.toml`); the runtime *order-book snapshots* (one per active contract, maintained by `book_manager.hpp` in the backtest stack); and the runtime *per-call state* (s_0 and f_t^0), supplied by the backtest stack's `BacktestState` object (declared in `v14_step_runner.hpp`). All three merge and enter the Python wrapper (`fast.py`), which imports the compiled `dp_solver_cpp` extension module and marshals the inputs across the language boundary into the C++ DP kernel (`dp_solver_experimental.cpp`). The MILP kernel (`gurobi.py`) is a separate entry point that consumes the same DP inputs (green dotted line off the merge) plus its own MILP-specific configuration file (`milp.toml`, solid arrow), and produces the same outputs (green dotted line into the output trunk). It is not invoked by the event-driven backtest loop. The color legend at the upper-left of the figure summarizes the role of each shade.

3.3.2 Backtest stack

The backtest stack drives the solver through a simulated continuous trading session covering a full calendar month and records every decision and trade. The stack has four layers: an execution and parallelism layer, a month-level runner, a step-level event loop, and a state-tracking layer.

3.3.2.1 Execution and parallelism

Backtests run in two modes: on Modal cloud compute or directly on a local machine. Both modes have the possibility to parallelize every month – one worker per calendar month, all months executing concurrently – so the per-month structure of the runner below is identical in either case. Each cloud worker is allocated 8 CPU

cores and 16 GB of RAM, of which the C++ kernel uses 6 OpenMP threads, with the remaining two cores reserved for the BLAS backend and the Python driver.

3.3.2.2 Month-level runner

At startup the month-level runner loads the manifest and contract-metadata tables produced by the precomputing pipeline (subsection 3.1.2), then reads the chosen timeline (event-driven or fixed) as a sorted list of nanosecond timestamps. The runner iterates through those timestamps in order, executing one *step* per timestamp. Order-book day files are loaded lazily: a day's binary event array is read from disk the first time a contract from that day becomes active, and released from memory once all contracts on that day have passed gate closure. This keeps memory use bounded by the size of the largest single day-file, and hence independent of the total monthly data volume.

3.3.2.3 Step-level event loop

At each timeline step the runner performs the following sequence of operations:

1. **Event application.** A book-replay module advances its cursor through the binary event array to the current timestamp, applying all incoming events and marking affected contracts as dirty. As noted in Section 3.1.2, an event represents the addition, modification, or cancellation of an order.

2. **Active-contract selection.** The runner queries the contract-metadata table for all contracts whose trading window is open at the current time and whose delivery has not yet begun.

3. **Book packing.** Dirty order books are converted into four dense NumPy arrays of shape $(N \times L)$ – bid prices, bid quantities, ask prices, ask quantities – where N is the number of currently active contracts and L is a fixed upper bound on the number of price levels kept per side. Only dirty contracts are repacked; non-dirty contracts reuse their packed arrays from the previous step. The output buffers are pre-allocated once and reused across steps, avoiding per-step memory allocation.

4. **Solve.** The packed arrays, together with the current state-of-charge and open positions, are passed to the solver stack. It returns the optimal action vector and the solver objective.

5. **Latency.** The execution timestamp is set to the current timestamp plus the fixed technical delay. The runner advances its book replica to that timestamp to obtain the book as it will look at execution time.

6. **FOK filter.** Each planned trade is re-checked against the execution-time book. Trades are filled greedily at the original limit price or better; partial fills are accepted and any unfilled remainder is discarded.

7. **State update.** The executed trades are recorded: open positions are updated, pending cash settlements are accrued, and the consumed quantity of each filled

resting order is added to the book manager's consumed-liquidity overlay. Contracts whose delivery window has closed are settled into the physical state-of-charge with degradation applied, then removed from the active position set and their overlay entries cleared.

8. **Output.** Each step's decision timestamp, solver objective, realized P&L, state of charge, active contract count, and solver wall time are appended to an in-memory buffer. A summary of aggregate statistics is written at the end of the month.

The data flow between the four layers, the eight steps of the inner loop, and their interactions with the contract-metadata table, the C++ DP kernel of Figure 3.1, and the input/output disk volumes is summarized in Figure 3.2.

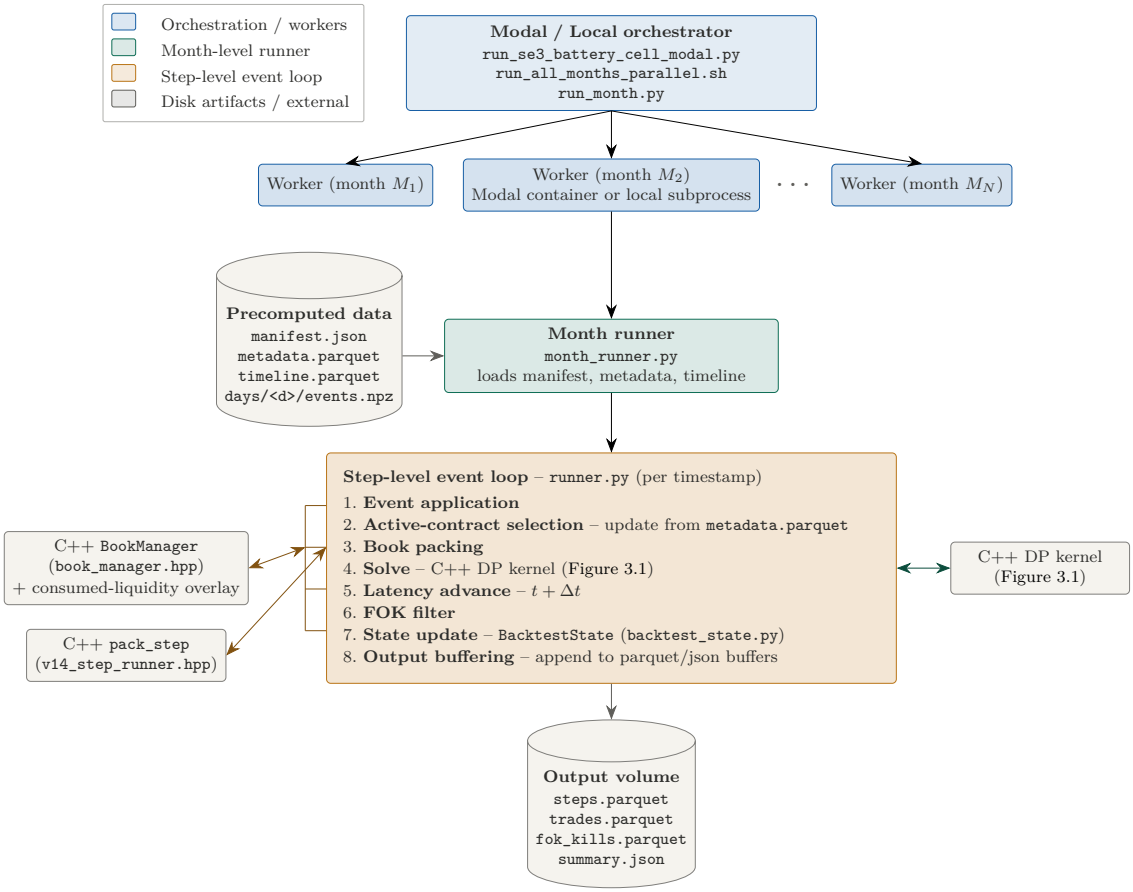


Figure 3.2: Software architecture of the backtest stack. Two interchangeable entry points – a Modal cloud orchestrator (`run_se3_battery_cell_modal.py`) and a local driver (`run_all_months_parallel.sh` together with `run_month.py`) – both spawn one per-month worker per calendar month in parallel, each worker running the same `month_runner.run_month_backtest` entry point. Within each worker the month runner consumes precomputed order-book day files lazily and iterates over the timeline, invoking the eight-step event loop implemented in `runner.py` once per timestamp. The loop calls the C++ DP kernel of Figure 3.1. Decision-level outputs are buffered in memory and written to `steps.parquet`, `trades.parquet`, `fok_kills.parquet`, and `summary.json` on the output volume (or local disk).

3.4 Settings and result artifact creation

The following parameters are held fixed across every backtest in this thesis: a charge and discharge efficiency of 95% in each direction, a fixed technical delay of $\Delta t = 200$ ms between event arrival and trade execution, and a zero terminal value. The efficiency and delay values follow the reference paper [1] and were reviewed by Flower as representative of their operational setting.

The remaining parameters take the following default values, which apply to every run except the battery-size sweep where they are varied:

- a 10 MWh battery with 10 MW charge and discharge limits (a one-hour duration)
- a DP grid of $m = 401$ uniformly spaced state-of-charge points giving a grid step of $10/400 = 0.025$ MWh
- an action step $u = 0.025$ MWh, equal to the grid step and to Nord Pool's 0.1 MW minimum order quantity expressed per quarter-hourly contract.

For the largest cells of the battery-size sweep, preserving this alignment at $u = 0.025$ MWh would demand a prohibitively large grid. The grid step is coarsened to 0.05 or 0.1 MWh on a cell-by-cell basis, trading some precision for tractable runtime.

The results in chapter 4 are produced by five families of backtest runs, each varying one axis of the default configuration above:

1. **Yearly event-driven DP.** The default battery on the event-driven timeline, run for each of the three markets considered: Sweden SE3 and Sweden SE4 over April 2025 to March 2026, and Germany Amprion over January 2025 to February 2026 (the longer German window reflects the operational data Flower has held longest in storage).
2. **Fixed-timeline sweep.** The same default battery and per-market reporting windows, on five fixed timelines (6 s, 1 min, 5 min, 15 min, 1 h) in addition to the event-driven baseline. Each timeline re-solves the DP at most once per interval, regardless of market activity (subsection 3.1.2).
3. **Spread-penalty ϕ-tuning.** A walk-forward tuning of the spread-penalty coefficient ϕ on the SE3 fixed-6 s policy with the default battery: ϕ^* is fit on each month and applied to the following month, yielding an 11-month tunable window over April 2025 to March 2026. Within each training month, ϕ^* is located by Brent's method (subsubsection 2.3.3.3) on the bracket $[0, 10]$ with initial seed $\phi_0 = 1$, terminated when the bracket shrinks below an absolute tolerance of 0.01 or after 15 function evaluations, whichever comes first.
4. **Battery-size sweep.** Eighteen battery configurations run on the SE3 6 s fixed timeline for April 2025 only (the event-driven timeline is intractable for the larger cells, and a single month was sufficient to expose the cell-to-cell trend): six energy capacities (10, 50, 100, 200, 500, 1,000 MWh) by three durations (0.5, 1.0, 2.0 h).
5. **MILP-vs-DP grid-size comparison.** Four 14-day intervals in SE3 (days 01–14 of April 2025, July 2025, October 2025, and February 2026), each solved

both with the rolling-intrinsic DP at three SoC-grid sizes ($m \in \{41, 201, 401\}$) and with the mixed-integer linear program via Gurobi against the same limit-order-book trajectory. The MILP serves as an exact-solver reference for the rolling-intrinsic problem (not as a perfect-foresight benchmark). Both solvers run on a common 60 s fixed decision timeline. The purpose is to investigate the reliability of DP profits as a function of grid size by quantifying how much the discretization error costs in realized P&L relative to the exact solver, and how that gap behaves across order-book regimes of differing liquidity. A short follow-up sweep at $m \in \{801, 1601, 3201\}$ on the same four windows extends the grid-size axis beyond the tabulated values to check whether DP rewards are still increasing.

3.5 Comparison benchmarks

The backtest runs described above report the profit of the rolling intrinsic on the continuous intraday market. To judge whether that profit is worth pursuing, it is set against two reference points: passive commitment of the same battery to each reserve capacity market in the zone, and two simple strategies that bid into the German intraday auction. This section describes how each benchmark is constructed; the comparisons themselves appear in Section 4.4.1.

3.5.1 Passive ancillary-market benchmark

The passive benchmark commits the full power of the 10 MWh / 10 MW battery to a single reserve capacity product for the whole reporting window and accumulates the capacity payment it would earn. Five products are priced for the Swedish zones (FCR-N, FCR-D up, FCR-D down, mFRR-CM up, mFRR-CM down) and four for Germany Amprion (aFRR-CM up, aFRR-CM down, mFRR-CM up, mFRR-CM down). Each product's cumulative revenue is compared against the cumulative profit of the event-driven rolling intrinsic on the same battery (Figures 4.8, 4.10, and 4.12).

The Swedish products clear hourly. Revenue in an hour is the marginal capacity price (EUR/MW) times an offered power of 10 MW times the one-hour interval, with an availability factor of one. The German aFRR and mFRR capacity products clear in daily four-hour blocks, so a block earns the marginal price times 10 MW times four hours, summed over the six blocks of a day. Hourly and block revenues are summed to daily totals and accumulated over the window. Swedish prices are taken from Mimer [11, 13] and German marginal capacity prices from regelleistung.net [22]; FCR clears across the whole Swedish synchronous area, so SE3 and SE4 use the same FCR prices. The Swedish window runs from May 2025 to March 2026 and the German window covers calendar 2025. The same priced revenues drive the daily-outperformance comparison of Section 3.5.3.

Two caveats apply. First, these revenues are capacity payments only: they exclude both the activation energy a delivered reserve would earn and the cycling and energy cost of being activated, so the figure is neither net of activation cost nor inclusive

of activation revenue. Second, offering 10 MW into a four-hour German block exceeds what a 10 MWh / 1 h battery can physically sustain, which would require 40 MWh; the German figures therefore use an upper-bound convention for cross-zone price comparison, and the availability factor of one assumes every bid clears at the marginal price.

3.5.2 Intraday-auction benchmark

The intraday-auction benchmark is reported for Germany Amprion only. Two price-taker strategies optimize the same 10 MWh / 10 MW battery against the auction's clearing prices over the 96 quarter-hourly products of each delivery day, each solved as a linear program with Gurobi on a 15-minute decision timeline.

The first strategy is a day-ahead-forecast bidder. It optimizes the daily schedule against a day-ahead price forecast; the resulting orders are then settled against the realized auction prices, so only the orders that clear at a profitable price execute (about 30% of the proposed volume). The second strategy is a perfect-foresight upper bound, which optimizes against the realized auction prices directly and gives the ceiling a price-taker could reach on the auction. Together they bracket the auction's value and locate where the rolling intrinsic on the continuous market sits relative to it.

Both strategies reset the state of charge to 5 MWh at the start of each delivery day, enforce a terminal state of charge, and use a round-trip efficiency of 0.95 per direction. Degradation (4 EUR/MWh) and a small throughput penalty enter the objective but are excluded from the reported profit, the same convention used for the rolling-intrinsic backtests (Section 3.2). The benchmark covers 352 delivery days of 2025.

3.5.3 Daily-outperformance comparison

The cumulative comparison of Section 3.5.1 accumulates each product over the whole window. A second comparison works day by day and asks on how many individual days the rolling intrinsic out-earns the best ancillary alternative. For each delivery day, the revenue of every product in the zone is computed as in Section 3.5.1. The highest-paying product that day (the per-day maximum across products) is the dominant ancillary, and the event-driven rolling intrinsic's profit for the same day is compared against that dominant revenue. The day counts as a win for the rolling intrinsic when its profit is the larger of the two. The rate of outperformance is the share of winning days over the window, reported both overall and split by which product was dominant (Figures 4.9, 4.11, and 4.13). The same per-day dominant revenue is the best-ancillary baseline behind the intraday-DP uplift reported in Section 4.6.3.

3.6 High-performance computing techniques

The rolling intrinsic policy of 2.15 is re-solved at every market event. On the SE3 benchmark this corresponds to roughly 1.8×10^5 DP solves per month, each carrying an inner loop of approximately 4×10^6 state–action evaluations. A naive C++ port of the original Python prototype was too slow to make a full-month grid sweep tractable within the timeframe of the thesis. The implementation went through fourteen numbered revisions (v1–v14.2) before stabilizing on the final v14.2 baseline. The resulting wall-time improvements are reported in section 4.5. The numbering jumps from v8.2 to v13 because v9–v12 were exploratory branches aimed at the infeasibility cascade discussed in subsection 2.4.6. These branches – non-uniform grids and related variants – either broke the bit-parity verification or yielded no measurable gain over v8, so the kernel work was continued from v8 into v13. Table 3.1 summarizes the chronology, and the remainder of this section documents the techniques behind each revision.

Version	Layer	Dominant change
Baseline	kernel	Naive C++ port of the Python prototype.
v1	kernel	Analytic state–action bound tightening.
v2	kernel	Hoisted timeout polling out of the inner loop.
v3	kernel	Flat-pointer indexing, hoisted SoC arithmetic, stage-level cashflow fast path.
v4	kernel	OpenMP parallelization of the backward row loop.
v5	kernel	Linear-advance interpolation; incremental position update.
v6	kernel	Persistent `DPStageCache`; partial recomputation by first-dirty stage.
v7	kernel	Branchless clamp via `vminsd`/`vmaxsd`; sentinel-padded value tables.
v8	pipeline	C++ `BookManager`; bulk `pack_dirty` replacing Python sorts.
v8.1 / v8.2	pipeline	FOK book lookup routed through the C++ `BookManager`.
v13	kernel	ΔV precompute; `schedule(static,16)`; OMP threshold lowered to ten stages. Final kernel.
v14.0	pipeline	C++ `pack_step`: snapshot construction and crossed-order stripping in C++.
v14.1	pipeline	C++ `BacktestState`; bulk position lookups; FMA fusion disabled in TU.
v14.2	pipeline	C++ `align_and_filter`; final baseline.

Table 3.1: Chronological evolution of the dynamic-programming solver from the naive C++ baseline to the final v14.2 implementation. Revisions baseline–v7 tighten the kernel itself. v8 onwards port the surrounding Python pipeline to C++, the one exception being v13, a final kernel pass that lands after the first pipeline revisions. v14.2 is the final end-to-end pipeline.

3.6.1 Toolchain and benchmarking environment

The solver is implemented in C++ and exposed to Python through `pybind11` [69]. The C++ translation units are compiled by GCC [70] with the flag set

```
-O3 -march=native -funroll-loops -flto -fopenmp,
```

which enables:

- aggressive inlining and loop unrolling (`-O3`, `-funroll-loops`)
- link-time optimization across the kernel and pipeline translation units (`-flto`)
- AVX2/FMA instruction selection on the AMD Zen3 hardware used for benchmarking [71] (`-march=native`)

- OpenMP threading [72] (`-fopenmp`).

Parallelism is provided by OpenMP through `libgomp`; the only other external libraries used on the C++ side are the C++17 standard library, `<chrono>` for wall-time instrumentation, and `<omp.h>`. No BLAS, LAPACK or Eigen are linked in, since the kernel performs only scalar arithmetic on small dense buffers.

Benchmark hygiene is enforced through four measures. Threads are pinned to a contiguous block of physical cores via `taskset`, and the OpenMP placement policy [73, 72] is fixed to `OMP_PROC_BIND=close` with `OMP_PLACES=cores`, preventing thread migration during a run. The thread counts of NumPy's [74] BLAS backends (OpenBLAS, MKL) are independently capped at one to eliminate cross-pool contention. Each variant is benchmarked over 2×10^5 steps of real market data, and competing variants are run in the same process via the same driver script so that they share machine state; variant ordering is rotated to detect ordering noise. We report the resulting numbers with an accepted noise band of approximately five percent.

3.6.2 Algorithmic and kernel-level optimizations

The discretized value function V_t of (2.15) is stored in memory as an array of m doubles, one per grid point; we refer to this array as the *value table* for stage t. Each revision was verified bit-identical to its predecessor on the per-solve objective and immediate action across 10^4 random solves. The most consequential changes are summarized below.

3.6.2.1 State–action bound tightening

For a fixed grid point s_i, the analytic feasible action interval $[k_{\mathrm{lo}}, k_{\mathrm{hi}}]$ implied by the storage and power envelopes is computed once per row, instead of being discovered iteratively by a per-iteration feasibility check inside the inner loop of the backward pass (see Algorithm 2). This skips the inner-loop iterations that would have been rejected, and removes the per-iteration check from those that remain.

3.6.2.2 Hoisted timeout polling

The solver enforces a hard wall-time budget per call. The naive implementation polled the system monotonic clock once every sixteen inner iterations, which, at a clock cost of approximately fifty nanoseconds, accounted for fifteen milliseconds of every solve. Hoisting the check from the inner loop to the outer loop reduces clock probes by roughly two orders of magnitude with negligible loss of timeout granularity.

3.6.2.3 Flat loops

Per-iteration helper-function indirection, type casts and bounds-clamp checks were replaced by raw pointer arithmetic. The reward and value tables are addressed through pointers that advance by a fixed stride per action, so each inner iteration

reduces to a pair of memory reads, an arithmetic update, and a running-argmax comparison. The reward arithmetic and state-of-charge update were inlined into the loop body, and a stage-level fast path skips the cash-flow-infeasibility branch on stages whose order book is dense enough to leave no $-\infty$ entries.

3.6.2.4 Linear-advance interpolation

Within the inner loop the interpolation argument $S(s, k_t u)$ that enters $\tilde{V}_{t+1}$ in Algorithm 2 is an affine function of k_t. The pre-existing implementation recomputed the bracketing grid index and interpolation weight (2.16) from scratch at every iteration; the linear-advance variant maintains them incrementally with a fixed stride per action, removing one multiplication and one helper call from each iteration. Numerical drift across the at most two hundred iterations (in the standard setting) of a single inner loop is bounded above by $\sim 10^{-13}$ EUR in propagated value-function terms, many orders of magnitude smaller than the value differences the running-argmax comparison actually distinguishes (section A.2).

3.6.2.5 V-difference precompute

For each backward stage we precompute $\Delta V_{t+1}[i] = V_{t+1}[i+1] - V_{t+1}[i]$ once per stage. The interpolation collapses to

$$\tilde{V}_{t+1}(s') \;=\; V_{t+1}[i] + w\,\Delta V_{t+1}[i],$$

which is a single fused multiply–add per inner iteration – the minimum compute possible for piecewise-linear interpolation. The $\mathcal{O}(m)$ precompute amortizes across the $\approx 70{,}400$ interpolations per stage (see section A.3) and yields a clear net saving.

3.6.2.6 Branchless clamping and sentinel-padded value tables

The bracketing grid index can fall just outside the valid range $[0, m-1]$ due to floating-point round-off at the storage envelope edges; a per-iteration boundary clamp brings it back inside. The clamp uses the AVX scalar min/max instructions `vminsd`/`vmaxsd` [71], which execute as fixed-latency arithmetic with no branch-prediction risk. To eliminate the right-edge boundary check inside the interpolation, the value tables are allocated with stride $m+1$ and the trailing sentinel $V[m] = V[m-1]$ is written at the end of every backward step. The interpolation then unconditionally reads two cells and returns the correct value at the boundary by construction.

3.6.2.7 Stage reuse cache

A single market event in the rolling intrinsic typically dirties only a handful of near-term contracts. If $t^\star$ denotes the index of the first dirty stage, then stages $t > t^\star$ inherit their value tables from the previous solve unchanged. A persistent `DPStageCache` keyed on the active contract list stores all value tables; on the next solve, only stages $0, \ldots, t^\star$ are recomputed. Cache invalidation is conservative – any change in the active contract list forces a full rebuild, since row indices shift – but

the typical dirty-stage index is small; the average solve recomputes roughly 7% of the horizon (section A.5).

3.6.3 Parallelism via OpenMP

Each backward step is embarrassingly parallel across grid points: every $V_t(s_i)$ reads only the previous-stage table V_{t+1} and writes a disjoint cell of V_t. The inner row loop is parallelized across threads using OpenMP, with the grid rows handed out in fixed contiguous blocks of sixteen iterations each — an assignment decided once at loop entry rather than dynamically at runtime. The block size is chosen to align with the cache hierarchy: an x86 cache line is 64 bytes and holds exactly eight doubles [75], so a sixteen-row block covers two whole cache lines, and no two threads ever write into the same line. This avoids *false sharing* — the case in which two threads writing to logically disjoint cells happen to share a cache line, forcing the line to ping-pong between cores under cache coherence and collapsing parallel throughput [61].

The thread scaling of the kernel is characterized by a thread-count sweep. The kernel is timed on a fixed representative workload at each thread count from one up to the full physical core count, and the speedup $S(N) = T(1)/T(N)$ relative to single-threaded execution is recorded at each. The resulting curve is fitted to Amdahl's law (subsubsection 2.3.3.4) with the serial fraction s as the single free parameter; the fitted s fixes both the speedup attained at any thread count and the asymptotic ceiling $1/s$. The serial fraction is then decomposed into its sources. A `libgomp` microbenchmark measures the per-region fork–join cost in isolation by executing an empty parallel region under the same thread count and loop schedule, and the share of the serial fraction it accounts for is computed directly. The remaining non-scaling time, which the microbenchmark does not explain, is attributed to memory-stall latency on the gather of the previous-stage value table.

The operating thread count is selected from the same sweep rather than fixed a priori. Each stage of the backward pass terminates in a synchronization barrier, so the per-stage barrier cost is independent of the thread count while the per-thread arithmetic decreases as threads are added. Once the per-thread workload falls below a threshold the barrier cost dominates and wall time rises with further threads, so the sweep attains an interior minimum near the physical core count. The thread count is fixed within this minimum region, which establishes that no reduction in wall time is available from a higher thread count. For further empirical findings shaping the OpenMP configuration, see section A.4.

3.6.4 Pipeline-level optimizations

The optimizations below port the Python pipeline that wraps the kernel – event replay, order-book snapshotting, packing into solver buffers, post-solve action filtering and trade execution – to C++. These components dominate the remaining wall time once the kernel revisions of section 3.6 are in place; moving them across the language boundary also reduces the number of Python-to-C++ marshalling events per step.

3.6.4.1 C++ `BookManager`

Steps 1, 3 and 5 of the event loop (Figure 3.2) all operate on the same per-contract order-book state, and are consolidated into a single C++ class `BookManager` (declared in `book_manager.hpp`) that holds that state in C++ memory for the duration of the timestamp. The class exposes two entry points:

- `apply_events`, invoked at steps 1 and 5 to advance the books to a target timestamp by consuming a packed NumPy array of order-book messages
- `pack_dirty`, invoked at step 3 to write the dirty contracts' books into the dense $(N \times L)$ solver buffer.

Each entry point replaces a Python loop – per event at steps 1 and 5, per price level per contract at step 3 – with a single cross-boundary call, eliminating both the interpreter cost of the loop body and the marshalling cost that would otherwise be paid on every event or level. Revisions v8.1 and v8.2 extend the same idea to the FOK filter of step 6. Its per-trade book lookups are served from the execution-time state that step 5 already leaves in the `BookManager`, rather than from a separate Python copy of the book, removing a redundant per-trade reconstruction.

3.6.4.2 Snapshot and crossed-order stripping

Snapshot construction and crossed-order stripping – the front half of step 3 in Figure 3.2 – are ported from Python to C++ via a single function `pack_step`, declared in `v14_step_runner.hpp`. The port preserves Python's order of operations exactly: the book is first truncated to the top twenty levels, and crossed orders are then stripped from the truncated view. Reversing the two produces a small but persistent trade drift on deep books.

3.6.4.3 C++ `BacktestState`

The bookkeeping object that tracks SoC, open positions and pending settlements was ported to C++. Bulk position lookups (one call per active contract list) replace per-contract Python list comprehensions. This translation unit is compiled with floating-point contraction turned off, so the compiler does not fuse the ported settlement arithmetic into single-rounded fused multiply–adds. The kernel leaves contraction on, but in the settlement path a fused multiply–add changes the last bit relative to the separately rounded Python reference. Disabling it here keeps the C++ settlement arithmetic bit-identical to that reference and preserves the parity bar maintained throughout the migration.

3.6.4.4 Action alignment and execution filtering

The Python routines that align the solver's planned actions to the live contract list and apply the execution filter were merged into a single C++ call returning a NumPy array; snapshots are reused in place rather than rebuilt as Python tuples.

3.6.5 Levers considered and rejected

Several standard HPC techniques were audited and dismissed on measured grounds rather than discarded a priori; we summarize them here for completeness.

`-ffast-math` [70] yielded a measured wall-time difference below one percent. The kernel's inner loop is a branch-based argmax rather than a sum-reduction, so `-fassociative-math` cannot vectorize it; FMA fusion is already enabled by default under `-O3 -march=native`; and `-ffinite-math-only` would silently invalidate the $-\infty$ infeasibility sentinel of 2.4.6.

NUMA pinning [76] is inapplicable: the benchmarking machine has only a single NUMA node, so cross-node binding is a no-op.

Transparent huge pages and software prefetch hints are inapplicable on this workload: the per-stage value table fits in L1 (≈ 3.2 kB), so pressure on the translation lookaside buffer (TLB) is negligible. The inner-loop access pattern is unit-stride within the clamped range (each iteration reads the next `double` in memory, eight bytes per access), which the Zen3 hardware prefetcher [71] already covers.

Custom arena allocators were considered for the temporary working buffers of the inner loop, but rejected because the C++ Standard Library's dynamic array already reuses its allocated memory once the buffers have grown to their steady-state size. There is essentially no allocation overhead left for an arena to remove.

Explicit SIMD intrinsics on the argmax loop were not pursued for three reasons:

- the loop is memory-latency-bound on the value-table gather rather than ALU-bound
- the trip count is data-dependent
- a vectorized gather–compare with masked lane management would risk one-ULP rounding drift that is hard to reconcile with the strict parity bar maintained throughout the migration.

Profile-guided optimization (PGO) [70] is identified as a remaining lever with non-negligible expected gain (roughly two to four percent). It was deferred outside the thesis window because the gain is speculative on a memory-latency-bound kernel and validating that the trained profile generalizes across months would have required a separate cross-month sweep.

3.7 Assessing impact on Flower, the environment, and the grid

The backtests produce realized P&L and operational metrics per zone. This section sets out how those outputs are turned into impact estimates at the three scales of Section 2.2: Flower's operations, the environment, and the grid. Each estimate names its data sources and assumptions; the resulting numbers are reported in Section 4.6 onward.

3.7.1 Per-battery economics

Per-battery economics were derived directly from each zone's annual backtest P&L. To frame the magnitude, a simple capex payback was computed as capex divided by annual P&L, on both a cell/pack and a fully-installed system-level basis. Installed costs were taken from the IEA 2024 battery survey [23], converted to euros and scaled to the default 10 MWh pack. The payback excludes opex, cycling degradation, ancillary revenue, and cost of capital.

3.7.2 Fleet economics

Operating many identical batteries raises two questions beyond the single-asset case: how many can trade before the exchange's order-rate limit binds, and how per-asset revenue scales as aggregate capacity grows. The first was answered by comparing the per-asset trade rate at each decision timeline against the exchange's order-rate limit [20]; the largest homogeneous fleet that stays under the cap is the cap divided by the per-asset rate. Revenue scaling was read from the battery-size sweep (Table 4.6), run for April 2025 on the SE3 fixed-6 s timeline. Because April was not an average month, its per-MWh revenue was annualized by April's share of the SE3 annual P&L rather than by a flat factor of twelve. The fleet aggregate falls between sweep points and was reached by interpolation; fleet revenue was then scaled from the single-asset annual figure by the interpolated per-MWh ratio.

3.7.3 Opportunity cost versus ancillary markets

The intraday-DP uplift over an always-on ancillary policy was computed against the per-day best-ancillary baseline of Section 3.5.3: the highest capacity revenue across the zone's products each day. The uplift was

$$\Delta = \sum_{t} \max\Big(0,\ \mathrm{DP}_t - \max_{p} \mathrm{anc}_{p,t}\Big),$$

the daily intraday P&L in excess of the best ancillary, summed over the days the DP exceeds it and annualized by the comparison window. For Germany it was computed twice: against the full aFRR+mFRR stack, and against the mFRR-CM subset a battery can readily contest.

3.7.4 Environmental impact

The net CO_2 displacement of a charge–discharge cycle was estimated as

$$\Delta \mathrm{CO}_2^{\mathrm{avoided}} = E_{\mathrm{dis}}\,\mathrm{MEF}_{\mathrm{dis}} - E_{\mathrm{chg}}\,\mathrm{MEF}_{\mathrm{chg}}, \qquad E_{\mathrm{chg}} = E_{\mathrm{dis}}/\eta_{\mathrm{RT}},$$

the emissions a discharge avoids less the emissions its charge causes, where a positive value is net climate-beneficial. Discharge volume was taken from each zone's main-run fixed-6 s throughput (Appendices C.1–C.3) on the 10 MWh battery, and charge volume followed from the round-trip efficiency $\eta_{\mathrm{RT}} = 0.95^2 = 0.9025$. Marginal emissions (Section 2.2.3) entered as a two-bucket proxy for the hourly profile [40]: one charge-hour factor (night and early afternoon) and one discharge-hour factor

(evening peak). Each estimate was swept over a plausible range of both factors to test whether the sign holds.

3.7.5 Grid impact

The single battery's footprint on the order book was measured as its annual traded volume ($E_{\text{chg}} + E_{\text{dis}}$, Section 3.7.4) divided by the zone's continuous-intraday volume, taken from EPEX for Germany [18] and from Nord Pool and ENTSO-E for Sweden [77, 16]. Order-book depth was read indirectly from the fill-or-kill (FOK) kill rate: with the fixed 200 ms execution delay, a planned trade is killed when the standing quote moves beyond its limit before execution, so the kill rate is a soft proxy for top-of-book depth and its volatility. Fleet-scale liquidity competition was read from the per-MWh slope of the battery-size sweep (Section 4.6.2).

4

Results

This chapter reports the empirical performance of the proposed high-frequency dynamic-programming (DP) trading strategy on three European bidding zones: Sweden SE3, Sweden SE4, and the German Amprion control area. For each market we quantify the yearly profit and operational behavior of the algorithm at its highest trading timeline (*event-driven*) and compare it against a family of fixed timelines (every 6 s, 1 min, 5 min, 15 min, 1 h).

Additionally, for SE3 we also study how performance is affected by a tuned spread penalty, how it scales with battery capacity and duration, and compare the rolling-intrinsic DP against a mixed-integer linear program (MILP) reference. Unless otherwise noted, results assume a 10 MWh / 1 h battery with the asset, fee and round-trip-efficiency parameters listed in Chapter 3.

The reporting period is April 2025 – March 2026 for the Swedish zones and January 2025 – February 2026 for Germany. March 2026 (SE3, SE4) and February 2026 (Germany) are incomplete months and are flagged as such throughout.

4.1 Sweden SE3

We begin with Sweden SE3 and use it as the canonical case to introduce each metric reported in this chapter. The equivalent SE4 and Germany sections then follow the same template.

4.1.1 Yearly High-Frequency DP Performance

This subsection covers the year-end profitability and operational footprint of the event-driven DP at its highest trading timeline, before turning to the per-timeline comparison.

4.1.1.1 Profitability

Over the twelve-month evaluation window the event-driven DP earns **€ 748,041** on a single 10 MWh / 1 h battery, corresponding to a time-averaged revenue rate of 8.98 €/MWh/h and a monthly capture of 6,234 €/MWh installed. Table 4.1 gives the monthly breakdown; profit is highest in spring and early autumn and lowest in November–December.

Table 4.1: Monthly performance of the event-driven DP on a 10 MWh / 1 h battery in SE3, April 2025 – March 2026. Revenue rate is normalized by installed energy capacity and by elapsed time.

Month	Profit (€)	Revenue (€/MWh/h)	Revenue (€/MWh/month)
Apr 2025	82,121	11.41	8,212
May 2025	88,621	11.91	8,862
Jun 2025	62,890	8.73	6,289
Jul 2025	61,040	8.20	6,104
Aug 2025	55,189	7.42	5,519
Sep 2025	80,467	11.18	8,047
Oct 2025	68,173	9.16	6,817
Nov 2025	50,056	6.95	5,006
Dec 2025	44,502	5.98	4,450
Jan 2026	76,532	10.29	7,653
Feb 2026	51,097	7.60	5,110
Mar 2026*	27,353	8.77	2,735
Year	**748,041**	**8.98**	**6,234**

* Incomplete month (13 days).

4.1.1.2 Operational metrics and trading pattern

The event-driven policy submits substantially more trades than any fixed-timeline baseline. Averaged over the evaluation period it issues 163.3 trades/hour. The exchange caps order entry at 5,000 trades/hour, so a homogeneous event-driven fleet reaches the cap at 30 identical batteries ($30 \times 163.3 = 4{,}899$ trades/h); a 31st battery would breach it. Table 4.2 compares the average trade rate across timelines.

Table 4.2: Average trades per hour per asset by decision timeline (SE3). The exchange's 5,000 trades/hour cap is reached at 30 identical event-driven batteries, or 52 on the fixed-6 s timeline.

Resolution	Avg. trades / h
Event-driven	163.3
Fixed 6 s	96.0
Fixed 1 min	42.4
Fixed 5 min	20.3
Fixed 15 min	11.4
Fixed 1 h	5.3

Beyond trade rates, total trades increase by a factor of approximately 31× between the hourly and event-driven timelines, totaling around 1,360,000 trades in the year. As the timeline coarsens, the FOK kill rate *rises* from 0.02% at event-driven to 1.71% on the hourly timeline, and the absolute number of partial fills drops from

2,733 to 100. Negative-P&L days are rare across all timelines (1–3 out of 347 trading days), with no timeline recording a streak longer than a single day. Table C.1 in Appendix C.1 details further operational metrics. The event-driven policy averages 6.28 cycles/day on the 10 MWh battery and the fixed-6 s baseline 5.09 cycles/day; throughput falls monotonically down to 1.96 cycles/day on the hourly timeline (Table 4.3).

Table 4.3: Annual mean cycles per day on the 10 MWh / 1 h SE3 battery, by decision timeline (Apr 2025 – Mar 2026, 347 trading days). One cycle is 10 MWh discharged.

Resolution	Avg. cycles / day
Event-driven	6.28
Fixed 6 s	5.09
Fixed 1 min	4.63
Fixed 5 min	4.21
Fixed 15 min	3.80
Fixed 1 h	1.96

Average daily trading pattern Additionally, Figure 4.1 shows the state-of-charge schedule of the event-driven DP across the year, together with marginal panels giving daily traded energy and the mean daily SoC profile. Trading follows a clear, repeatable diurnal pattern: it tends to charge in the early morning and again in the early afternoon, and discharges in the evening. The pattern is sharpest in spring and early autumn, and softens through November, December and January.

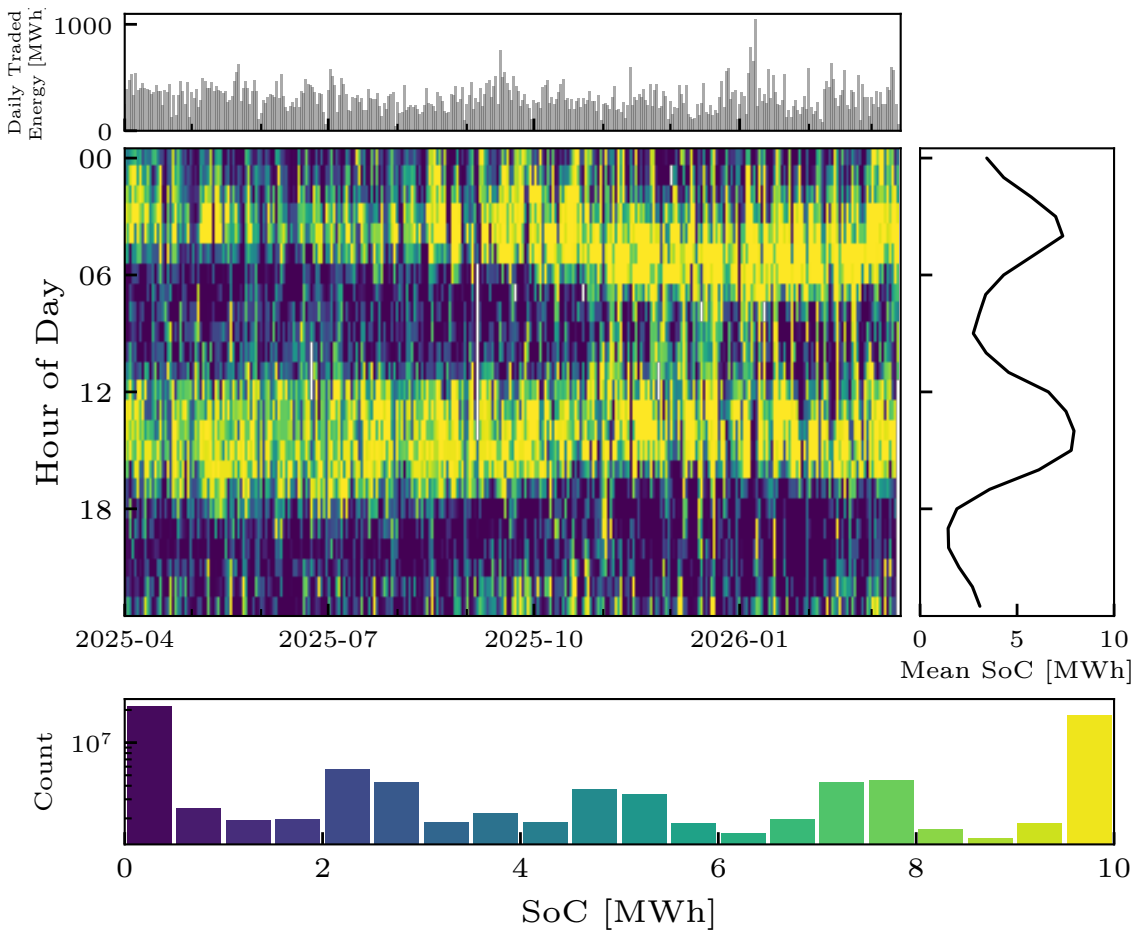


Figure 4.1: Yearly state-of-charge schedule of the event-driven DP in SE3 over Apr 2025 – Mar 2026 (main panel: date × hour-of-day, color = SoC at the start of the hour). Top marginal: daily traded MWh; right marginal: mean SoC profile across all days; bottom marginal: log-scale SoC histogram showing the policy parking the battery near the SoC bounds most of the time.

4.1.1.3 Fixed Timelines Performance

Coarsening the decision timeline degrades profit monotonically. Table 4.4 shows total annual profit at each fixed timeline; the event-driven policy collects €748 K, while the fixed timelines span €676 K (at 6 s) down to €439 K (at 1 h). The event-driven policy thus extracts roughly 11% more revenue than the next-best 6-second baseline and approximately 1.7 times the revenue of an hourly timeline strategy.

Table 4.4: SE3 annual profit (Apr 2025 – Mar 2026) at each decision timeline.

Resolution	Profit (k€)	Relative to event
Event-driven	748	1.00×
Fixed 6 s	676	0.90×
Fixed 1 min	583	0.78×
Fixed 5 min	522	0.70×
Fixed 15 min	478	0.64×
Fixed 1 h	439	0.59×

Figure 4.2 shows the corresponding cumulative-reward trajectories: the six curves separate from the start of the evaluation window and continue to diverge throughout the year.

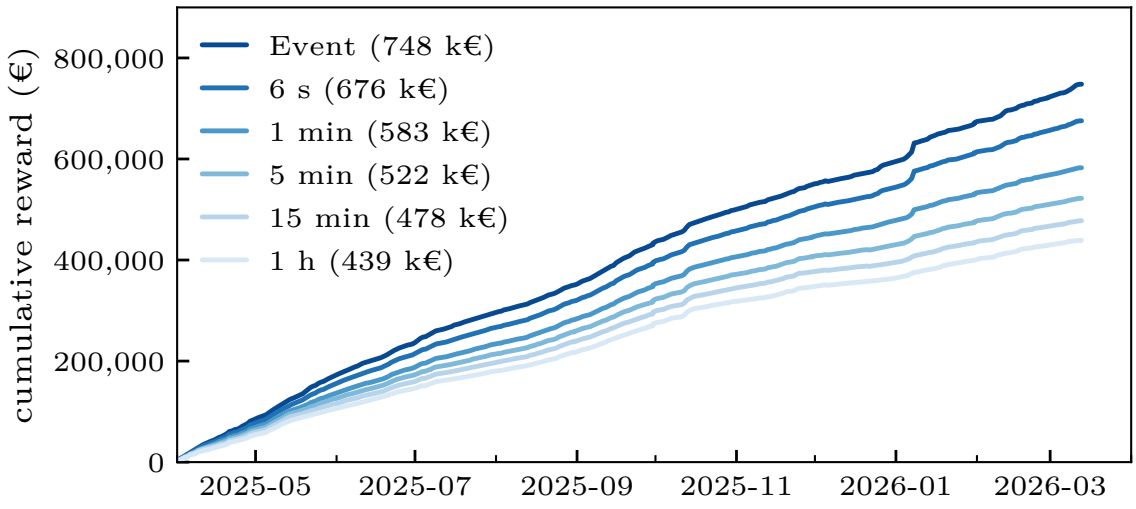


Figure 4.2: Cumulative reward of the rolling-intrinsic DP in SE3 over Apr 2025 – Mar 2026 at six decision timelines (Event, 6 s, 1 min, 5 min, 15 min, 1 h). Daily granularity; legend lists cumulative totals.

4.1.2 Spread-penalty fine-tuning

Applying the spread-penalty extension of subsection 2.4.5 to the SE3 fixed-6 s policy lifts annual profit by +6.21% over the $\phi = 0$ reference (€697.7 K vs. €656.9 K over the 11-month tunable window). Table 4.5 details the per-month gain. The tuning is positive in 9 of 11 months and clearly negative only in November 2025 (−19.4%).

Table 4.5: Walk-forward spread-penalty tuning on the SE3 fixed-6 s policy. ϕ^* is fit on the previous month and applied to the anchor month. Profit columns are in k€ for a single 10 MWh battery.

Anchor month	ϕ^*	Baseline (ϕ=0, k€)	ϕ-tuned (k€)	% change
2025-05	0.32	87.9	97.8	+11.2%
2025-06	0.30	65.7	71.7	+9.1%
2025-07	0.38	67.0	71.3	+6.4%
2025-08	0.62	60.6	66.2	+9.3%
2025-09	0.38	85.5	95.3	+11.4%
2025-10	0.57	67.3	69.6	+3.5%
2025-11	0.86	49.8	40.1	−19.4%
2025-12	0.22	44.6	44.4	−0.5%
2026-01	0.12	70.2	74.2	+5.8%
2026-02	0.18	37.3	42.2	+13.2%
2026-03	0.24	21.0	24.8	+18.1%
Total (11 months)		**656.9**	**697.7**	**+6.21%**

4.1.3 Battery size analysis

We sweep the battery installation across six energy capacities (10, 50, 100, 200, 500, 1,000 MWh) and three durations (0.5, 1.0, 2.0 h) and report normalized revenue (€ per MWh of installed energy) for April 2025 on the SE3 fixed-6 s timeline. Table 4.6 shows the result. Per-MWh revenue declines monotonically with both capacity and duration: the smallest 10 MWh / 0.5 h installation captures 12,870 €/$\text{MWh}_{\text{installed}}$, while a 1 GWh / 2 h installation captures only 3,055 €/$\text{MWh}_{\text{installed}}$.

Table 4.6: April 2025 revenue per MWh of installed energy (€/$\text{MWh}_{\text{installed}}$) for the SE3 fixed-6 s DP, swept over six capacities and three durations. Smaller installations capture more revenue per MWh of installed energy.

	Energy capacity					
Duration	10 MWh	50 MWh	100 MWh	200 MWh	500 MWh	1000 MWh
0.5 h	12,870	11,028	10,253	9,344	7,994	7,275
1.0 h	8,151	7,263	6,868	6,447	5,629	5,100
2.0 h	5,060	4,573	4,276	4,000	3,626	3,055

Per MWh of installed energy, shorter durations capture more revenue: the 0.5 h column is roughly 2.5× the 2 h column at every capacity.

4.1.4 Comparison between MILP and DP

Table 4.7 reports the MILP-versus-DP benchmark on four representative two-week windows in SE3. Both solvers run on a common 60 s fixed decision timeline; the

MILP uses a two-second time limit per solve, a 10% MIP gap, and a warm start from the null action.

The reward ordering is the same in every window: MILP > DP-401 > DP-201 > DP-41. The MILP–DP-401 gap ranges from €1,333 in Jul 2025 to €6,891 in Feb 2026, and the DP-401–DP-41 gap from €3,631 in Jul 2025 to €8,357 in Oct 2025. Per-solve cost ranges from $\sim$ 5–9 ms for the DP across the three grid sizes to $\sim$ 190–440 ms for the MILP. Aggregated over the $\sim$ 20,500 solves in a fortnight window this maps to 0.03–0.05 h of wall-clock for the DP variants and 1.09–2.50 h for the MILP. Traded volume moves in the opposite direction to reward across the DP grid sizes and the MILP. Within each window, volume is highest at DP-41 and falls monotonically through DP-201, DP-401 and the MILP. In Oct 2025 and Feb 2026 the MILP volume sits roughly half that of DP-401 (1,512 vs 3,123 MWh; 1,379 vs 2,883 MWh); in Apr 2025 and Jul 2025 the gap between MILP and DP-401 is smaller, on the order of 130–140 MWh.

Additional DP runs at $m \in \{801, 1601, 3201\}$ (not tabulated) produced rewards higher than DP-401 in each tested window, with the per-step reward increment shrinking as m grew.

Table 4.7: Performance comparison on four representative 14-day windows in SE3 between the rolling intrinsic solved using the MILP (Gurobi, 2 s time limit, 10% MIP gap) and the DP at three SOC-grid resolutions ($m \in \{41, 201, 401\}$). All runs use the 10 MWh / 1 h battery on the 60 s fixed decision timeline. *Sim-Time* is container wall-clock; *Solves* counts DP/MILP calls; *Vol* is the mean of buy- and sell-side traded volume. Format follows Schaurecker et al. Table 2 [1], with the cycles-per-day column omitted.

Window	Solver	Reward (€)	Sim-Time (h)	Solves	Vol (MWh)
Apr 2025	MILP	36,535	2.50	20,499	3,790
	DP-401	34,840	0.04	20,499	3,926
	DP-201	32,947	0.04	20,499	4,093
	DP- 41	30,717	0.03	20,499	4,353
Jul 2025	MILP	28,168	2.23	20,483	3,287
	DP-401	26,835	0.04	20,483	3,416
	DP-201	26,073	0.04	20,483	3,566
	DP- 41	23,204	0.03	20,483	3,854
Oct 2025	MILP	33,476	1.19	20,509	1,512
	DP-401	31,327	0.05	20,509	3,123
	DP-201	30,807	0.05	20,509	3,278
	DP- 41	22,970	0.04	20,509	3,602
Feb 2026	MILP	22,701	1.09	20,551	1,379
	DP-401	15,810	0.04	20,551	2,883
	DP-201	13,170	0.04	20,551	3,060
	DP- 41	11,758	0.03	20,551	3,319

Each window covers days 01–14 of the named month.

4.2 Sweden SE4

The same reporting template is applied as in SE3. However, there is no penalty tuning, no battery sizing sweep, and no comparison with the MILP.

4.2.1 Profitability

SE4 is more profitable than SE3. Over the same April 2025–March 2026 window the event-driven DP earns **€ 931,874** (Table 4.8), +25% above SE3. The seasonal pattern is similar (spring strong, late autumn weak) but more pronounced.

Table 4.8: Monthly performance of the event-driven DP on a 10 MWh / 1 h battery in SE4, April 2025 – March 2026.

Month	Profit (€)	Revenue (€/MWh/h)	Revenue (€/MWh/month)
Apr 2025	99,768	13.86	9,977
May 2025	102,747	13.81	10,275
Jun 2025	87,599	12.17	8,760
Jul 2025	78,557	10.56	7,856
Aug 2025	75,999	10.21	7,600
Sep 2025	97,326	13.52	9,733
Oct 2025	76,231	10.25	7,623
Nov 2025	73,269	10.18	7,327
Dec 2025	61,339	8.24	6,134
Jan 2026	80,187	10.78	8,019
Feb 2026	55,928	8.32	5,593
Mar 2026*	42,924	13.76	4,292
Year	**931,874**	**11.19**	**7,766**

* Incomplete month (13 days).

4.2.2 Operational metrics

Operational metrics for SE4 follow the same qualitative pattern as SE3 at somewhat higher trading volumes. Total trades range from 54,669 on the hourly timeline to 1,568,531 event-driven, a factor of approximately 29×. As the timeline coarsens, the FOK kill rate *rises* from 0.02% (event) to 2.17% (hourly) and the absolute partial-fill count drops from 2,952 to 120. Negative-P&L days are essentially absent across all timelines (1 event-driven, 0 at 6 s, 2 at 1 min, 0 on every coarser timeline, out of 347 trading days) with no timeline recording a streak longer than a single day. Table C.2 in Appendix C.2 details further operational metrics. Figure 4.3 shows the yearly state-of-charge schedule of the event-driven DP.

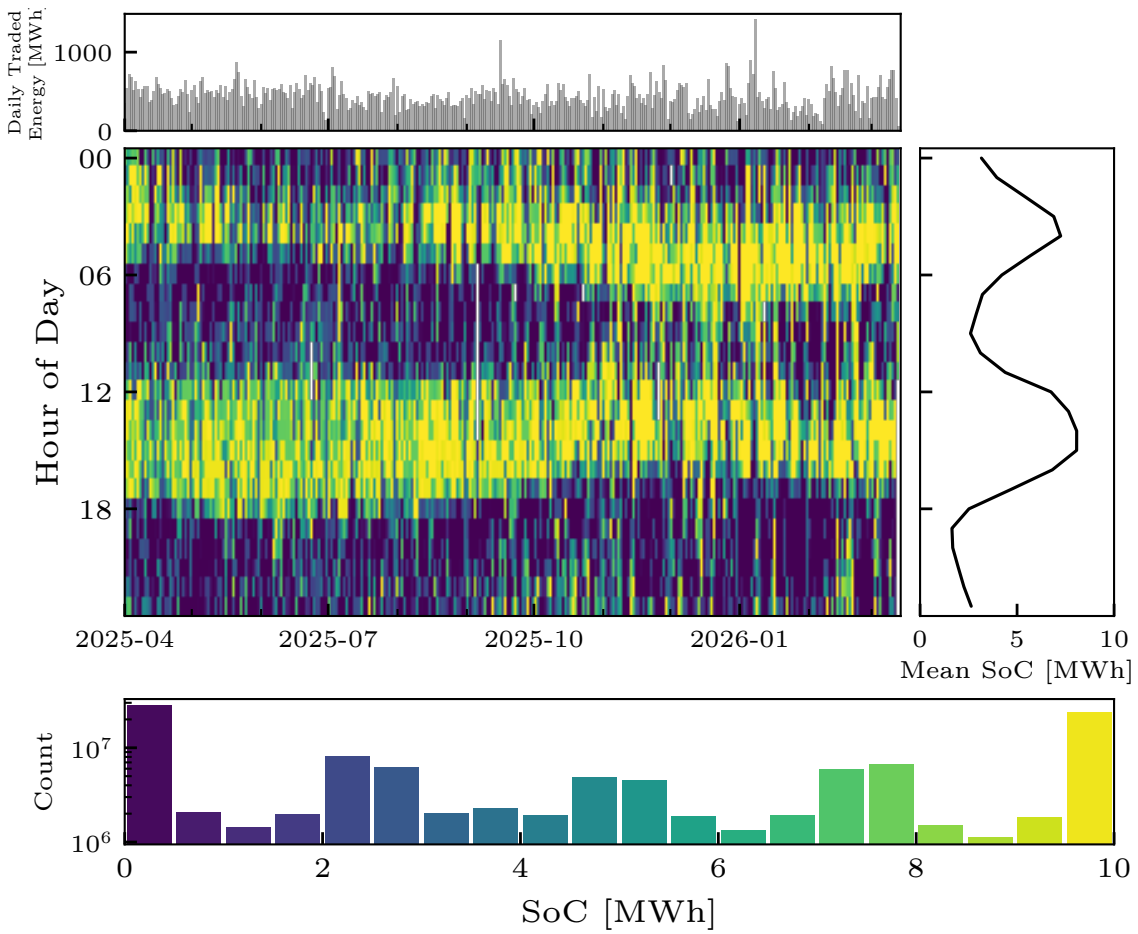


Figure 4.3: Yearly state-of-charge schedule of the event-driven DP in SE4 over Apr 2025 – Mar 2026, same layout as Figure 4.1.

4.2.3 Fixed Timelines Performance

Total profit by decision timeline: €932 K (event), €854 K (fixed 6 s), down to €617 K (fixed 1 h). The event-driven advantage in SE4 is slightly smaller in relative terms than in SE3 but larger in absolute terms. SE4's deeper book partially erodes the high-frequency edge while preserving more profitable spreads on every timeline.

Table 4.9: SE4 annual profit (Apr 2025–Mar 2026) at each decision timeline.

Resolution	Profit (k€)	Relative to event
Event-driven	932	1.00×
Fixed 6 s	854	0.92×
Fixed 1 min	757	0.81×
Fixed 5 min	699	0.75×
Fixed 15 min	659	0.71×
Fixed 1 h	617	0.66×

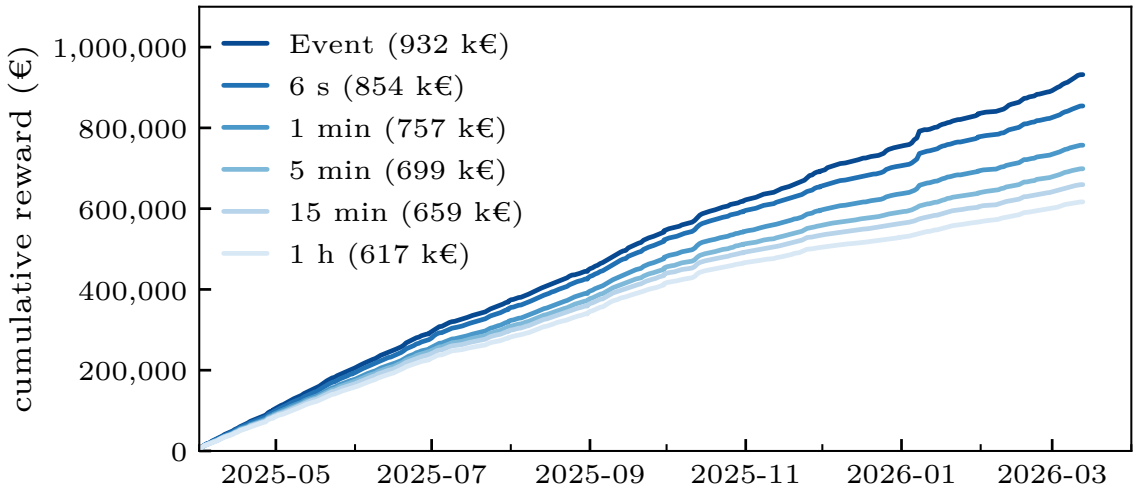


Figure 4.4: Cumulative reward of the rolling-intrinsic DP in SE4 over Apr 2025 – Mar 2026 at six decision timelines. Daily granularity; legend lists year-end cumulative totals. Same layout as Figure 4.2.

4.3 Germany Amprion

Germany Amprion is the only zone in the study with a rolling per-contract gate closure rather than a batched product auction. The same results as for SE4 are outlined below.

4.3.1 Profitability

Over its 14-month window (January 2025–February 2026) the event-driven DP earns **€ 1,081,490** on a single 10 MWh / 1 h battery in the German Amprion control area, a time-averaged 10.73 €/MWh/h. This is the highest raw total of the three zones, but only because the German window is the longest; on the window common to all three (April 2025–January 2026) SE4 earns more (Section 4.4). Table 4.10 gives the breakdown.

Table 4.10: Monthly performance of the event-driven DP on a 10 MWh / 1 h battery in the German Amprion control area, January 2025 – February 2026.

Month	Profit (€)	Revenue (€/MWh/h)	Revenue (€/MWh/month)
Jan 2025	92,964	12.50	9,296
Feb 2025	70,240	10.45	7,024
Mar 2025	96,404	12.96	9,640
Apr 2025	92,716	12.88	9,272
May 2025	113,806	15.30	11,381
Jun 2025	90,477	12.57	9,048
Jul 2025	92,951	12.49	9,295
Aug 2025	74,468	10.01	7,447
Sep 2025	85,291	11.85	8,529
Oct 2025	68,720	9.24	6,872
Nov 2025	48,848	6.78	4,885
Dec 2025	40,877	5.49	4,088
Jan 2026	78,374	10.53	7,837
Feb 2026*	35,354	6.14	3,535
14-month total	**1,081,490**	**10.73**	**7,725**

* Incomplete month (24 days).

4.3.2 Operational metrics

Germany Amprion is the most active of the three markets on every timeline. Total trades reach approximately 2,556,000 event-driven against 55,915 on the hourly timeline over the 14-month window, a factor of roughly 46×. As the timeline coarsens, the FOK kill rate *rises* from 0.03% at event-driven to 2.98% on the hourly timeline. The absolute partial-fill count drops from 8,231 at event-driven to 183 on the hourly timeline. Negative-P&L days are essentially absent (0–1 of 420 trading days on any timeline, no streak longer than a single day). Table C.3 in Appendix C.3 details further operational metrics. Figure 4.5 shows the corresponding state-of-charge schedule.

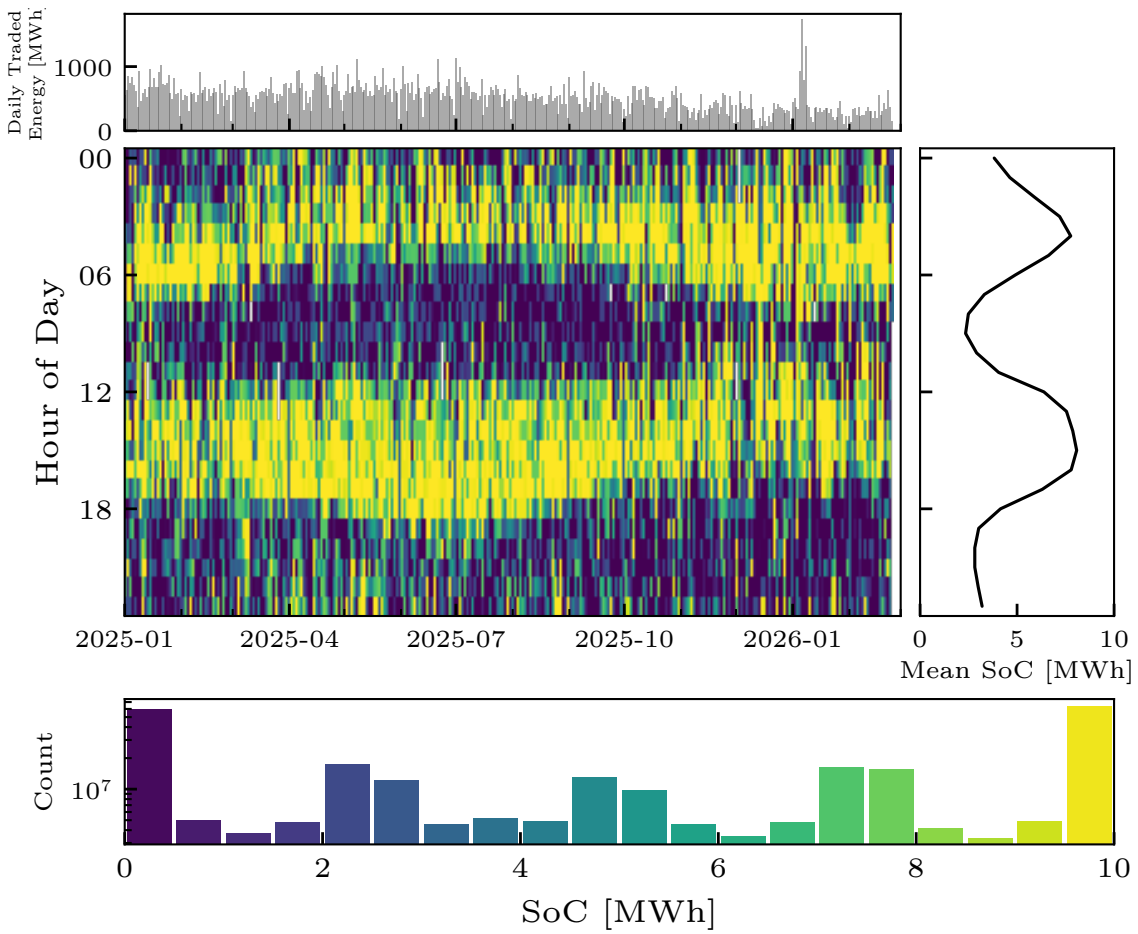


Figure 4.5: Yearly state-of-charge schedule of the event-driven DP in Germany Amprion. Layout as Figure 4.1; the window covers January 2025 – February 2026.

4.3.3 Fixed Timelines Performance

Total profit by decision timeline is reported in Table 4.11, summed over the same 14-month window. Unlike SE3 and SE4, Germany's intermediate timelines are not strictly monotonic in profit: the fixed-5 min run earns marginally more than the fixed-1 min run (885.6 k€ vs. 863.8 k€).

Table 4.11: Germany Amprion profit at each decision timeline, summed over the full 14-month window (Jan 2025–Feb 2026).

Resolution	Profit (k€)	Relative to event
Event-driven	1,082	1.00×
Fixed 6 s	945	0.87×
Fixed 1 min	864	0.80×
Fixed 5 min	886	0.82×
Fixed 15 min	823	0.76×
Fixed 1 h	812	0.75×

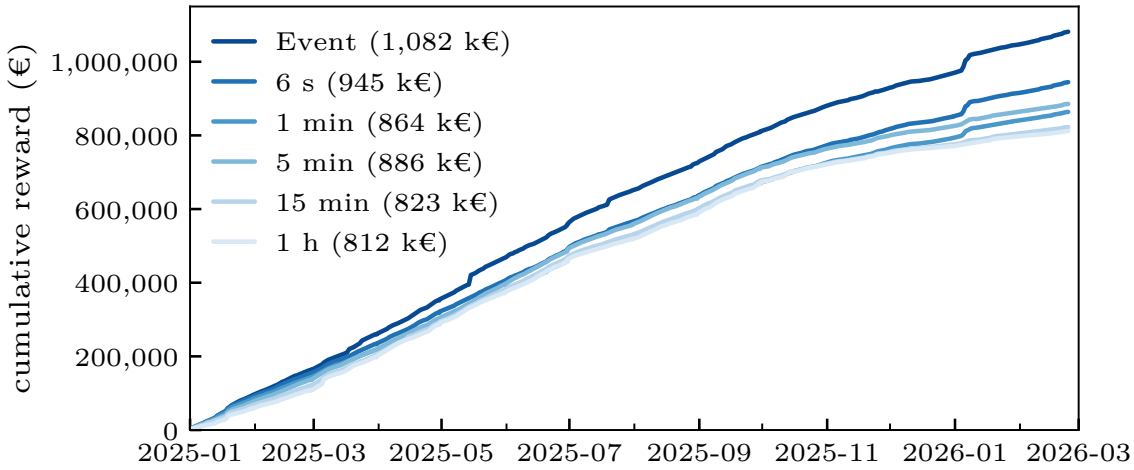


Figure 4.6: Cumulative reward of the rolling-intrinsic DP in Germany Amprion at six decision timelines, January 2025 – February 2026.

4.4 Cross-market summary

Comparing the three zones requires a common window: they cover different spans, so only April 2025–January 2026 (ten complete months) is comparable across all three. On that window the event-driven ordering is **SE4 (€ 833 k) > Germany Amprion (€ 787 k) > SE3 (€ 670 k)** (Table 4.12), and the time-averaged revenue rate agrees: SE4 (11.19 €/MWh/h) > Germany (10.73) > SE3 (8.98). Germany earns the highest raw total (Table 4.10) only because its window is four months longer; over a matched window SE4 is highest. The event-driven edge over the next-best 6-second baseline sits in a narrow band of +9%–+15% across the three zones, with the hourly gap ranging from +33% (Germany) to +70% (SE3). All markets show a multimodal distribution of charge states, evenly distributed across the SoC.

Table 4.12: Cross-market comparison. The event-driven profit row is the total over the common April 2025–January 2026 window (ten complete months), the only span comparable across all three zones; SE4 is highest. The event-vs.-fixed uplift rows are within-zone ratios over each zone's full window and are unaffected by window length. Per-zone full-window totals at every timeline are in Tables 4.4, 4.9 and 4.11.

Metric	SE3	SE4	Germany Amprion
Profit, event timeline, common window (k€)	670	833	787
Time-averaged rate (€/MWh/h, full window)	8.98	11.19	10.73
Event vs. fixed-6 s uplift	+11%	+9%	+15%
Event vs. fixed-1 h uplift	+70%	+51%	+33%

Figure 4.7 plots each zone's profit relative to its own event-driven timeline. Anchoring each zone at its own event total cancels the differing window lengths and makes the decay shapes comparable across zones. The event-driven timeline is best

in every zone, and SE3 and SE4 decline monotonically as the timeline is coarsened. Germany has the largest 6 s gap of the three (+15%, versus +11% in SE3 and +9% in SE4) but the smallest hourly gap (+33%, versus +70% in SE3 and +51% in SE4): its deeper book preserves more of the profit only on the slowest timelines. Its line therefore stays highest at the coarse end, holding 0.75 of event profit at 1 h versus 0.59 in SE3. The small 1 min–5 min inversion discussed in Section 4.3 shows as a slight uptick. Absolute, matched-window totals are in Table 4.12.

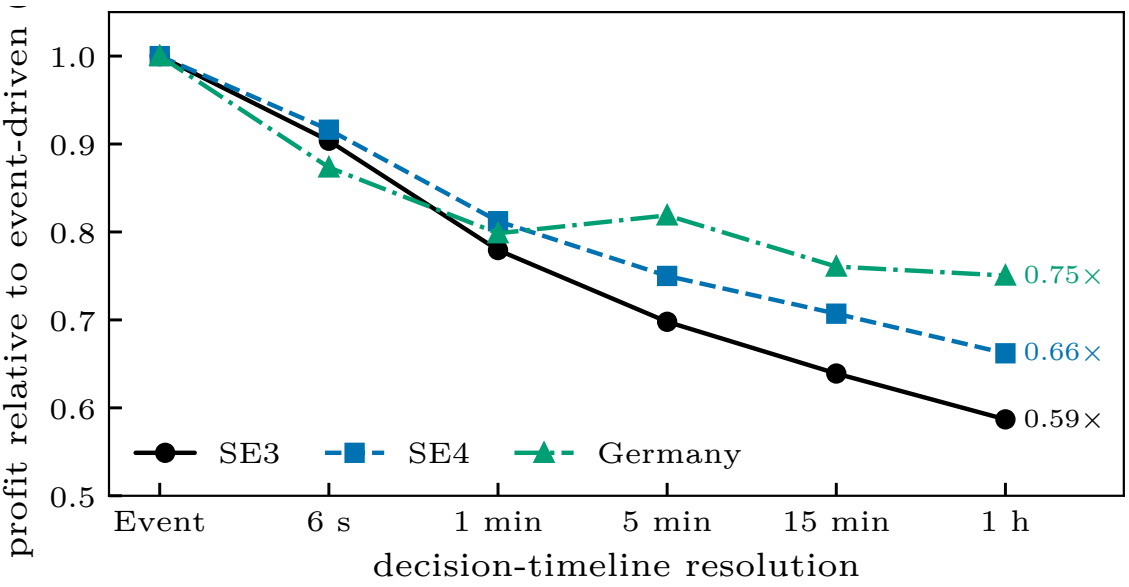


Figure 4.7: Profit at each decision timeline relative to the event-driven timeline (event = 1.00) for SE3, SE4 and Germany Amprion, one line per zone. Plotting the ratio makes the three zones comparable despite their different window lengths; these are the "Relative to event" columns of Tables 4.4, 4.9 and 4.11. End-of-line labels give the 1 h ratio. SE3 and SE4 are monotone in timeline; Germany has the 1 min–5 min inversion discussed in Section 4.3. The matched-window absolute comparison is in Table 4.12.

4.4.1 Competing markets

For each zone the cumulative revenue of the event-driven intraday DP on a 10 MWh battery is compared against passive commitment of the same capacity to each available ancillary capacity market in the zone. For Germany Amprion two intraday-auction benchmarks are also reported: a perfect-foresight upper bound and a day-ahead-forecast bidder, both run on the intraday-auction price book at a 15-minute (quarter-hourly) decision interval. Ancillary revenues are capacity-market only; activation energy is excluded.

4.4.1.1 SE3

Over May 2025–March 2026 (Figure 4.8), the event-driven intraday DP is more profitable than FCR-D but less profitable than the other ancillary markets. The SE3 intraday DP cumulates to €0.67 M. Ancillary totals: mFRR-CM-up €2.39 M,

FCR-N €1.58 M, mFRR-CM-down €0.98 M, FCR-D-down €0.50 M, FCR-D-up €0.46 M.

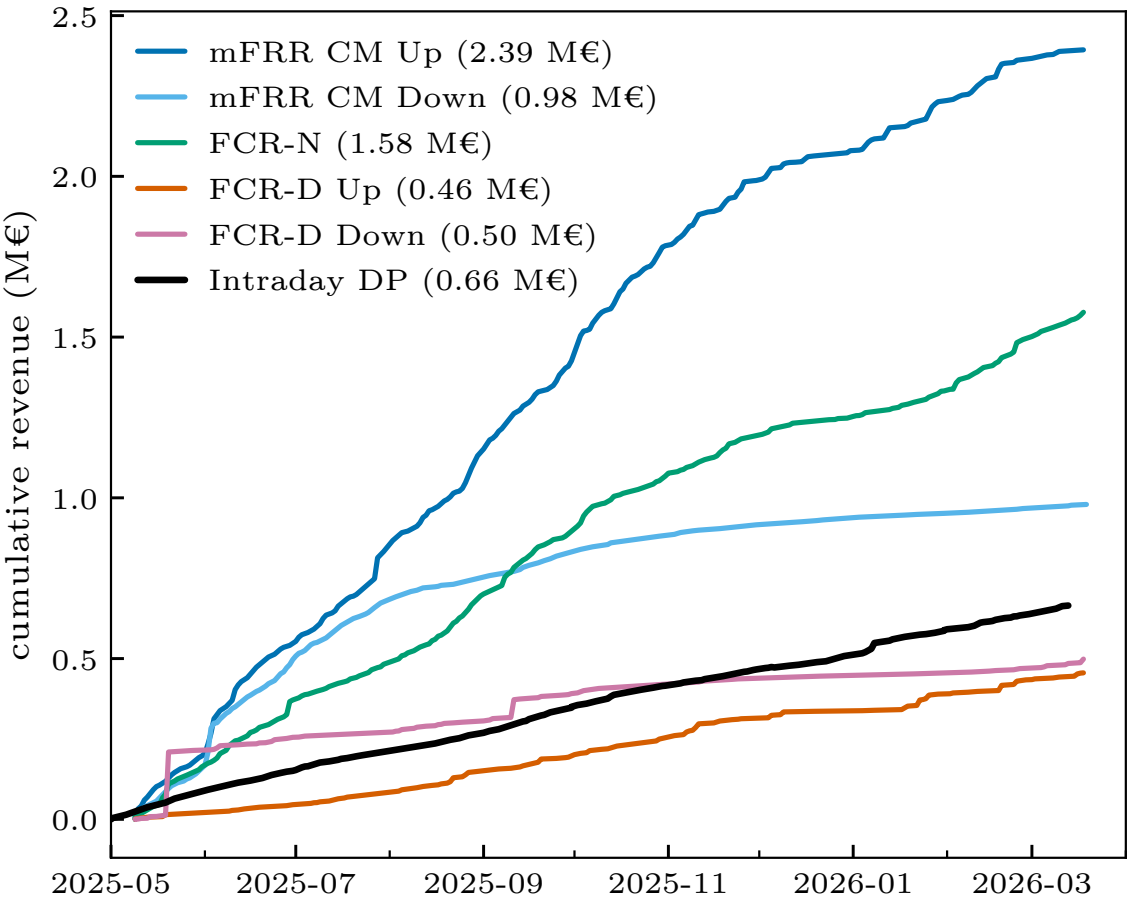


Figure 4.8: Cumulative revenue, SE3, May 2025–March 2026, 10 MWh battery. Five passive ancillary trajectories and the event-driven intraday DP. Legend reports end-of-window totals.

Daily outperformance. Across 309 days (2025-05-09–2026-03-13, Figure 4.9) the SE3 intraday DP outperforms the daily highest-paying ancillary on 26 days (8%). The wins fall on FCR-N-dominant days (14 of 102) and mFRR-dominant days (12 of 197); on the 10 FCR-D-dominant days the DP never wins.

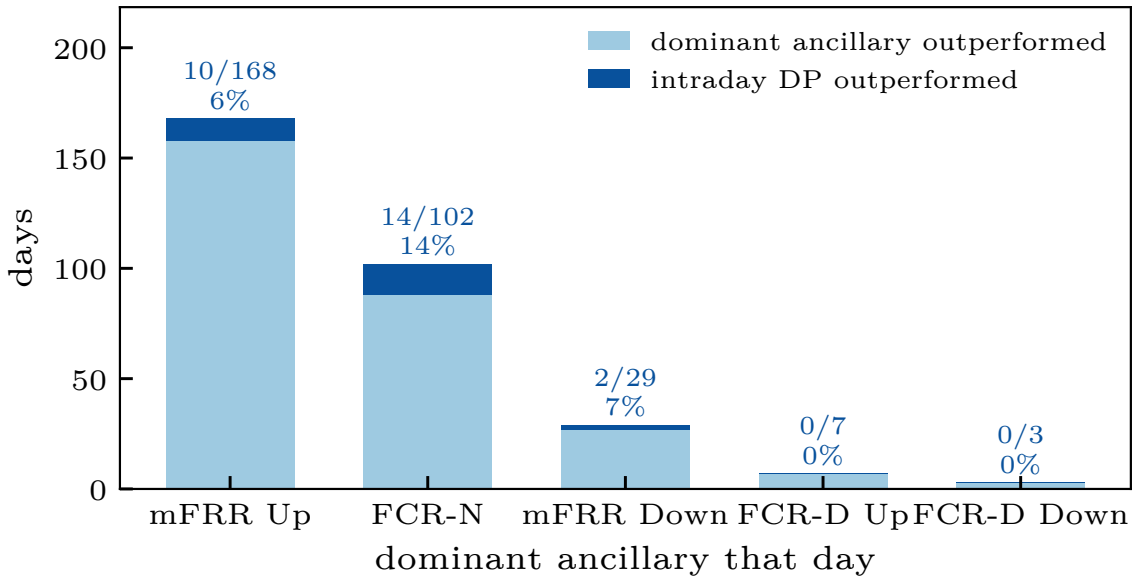


Figure 4.9: SE3 event-driven intraday DP daily P&L compared against the daily highest-paying ancillary for the same day, 2025-05-09 – 2026-03-13 (309 days). Layout as Figure 4.11.

4.4.1.2 SE4

The same pattern is repeated for SE4. Over the window (Figure 4.10), the SE4 intraday DP cumulates to €0.83 M. Ancillary totals: mFRR-CM-up €3.21 M, FCR-N €1.58 M, mFRR-CM-down €1.06 M, FCR-D-down €0.50 M, FCR-D-up €0.46 M.

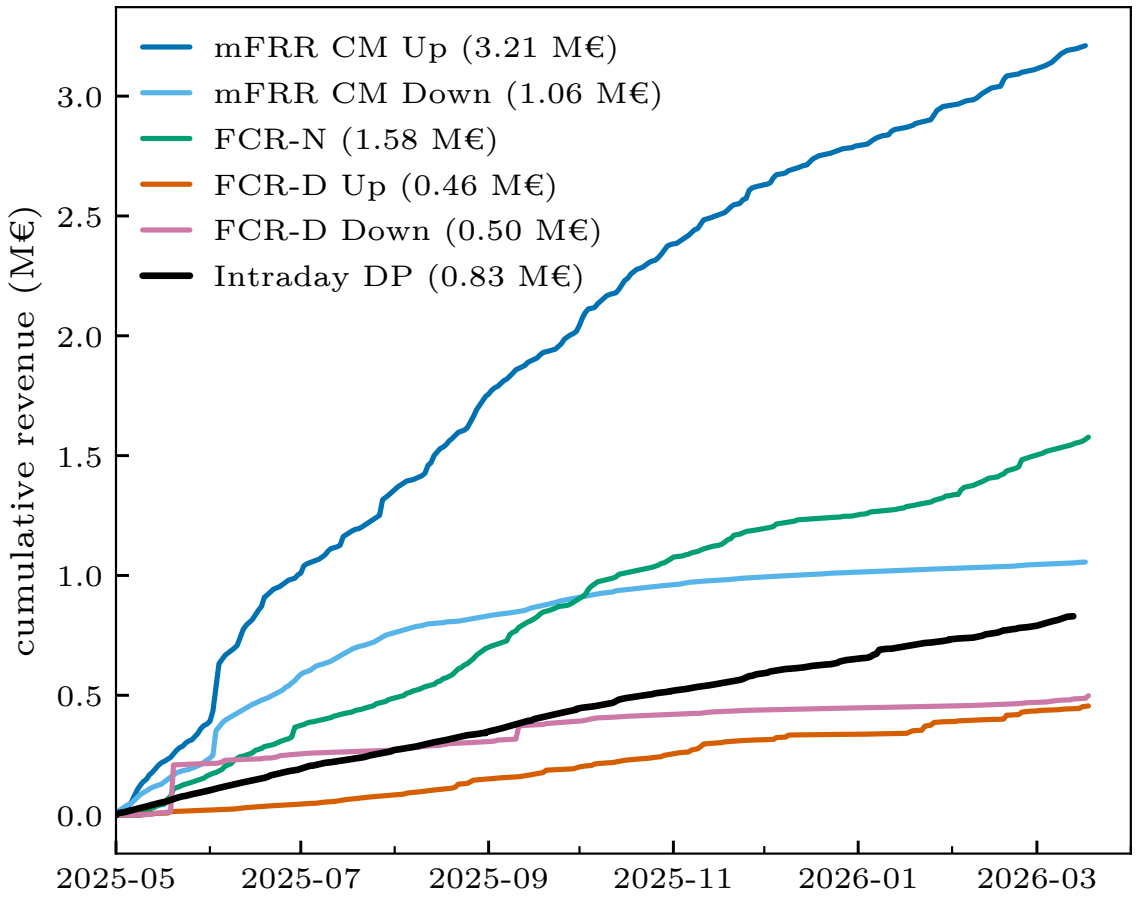


Figure 4.10: Cumulative revenue, SE4, May 2025–March 2026, 10 MWh battery. Layout as Figure 4.8.

Daily outperformance. Across 309 days (2025-05-09–2026-03-13, Figure 4.11) the SE4 intraday DP outperforms the daily highest-paying ancillary on 13 days (4%). The denser SE4 ancillary stack leaves fewer winning days than SE3: 10 of 237 mFRR-dominant days and 3 of 65 FCR-N-dominant days, none on FCR-D-dominant days.

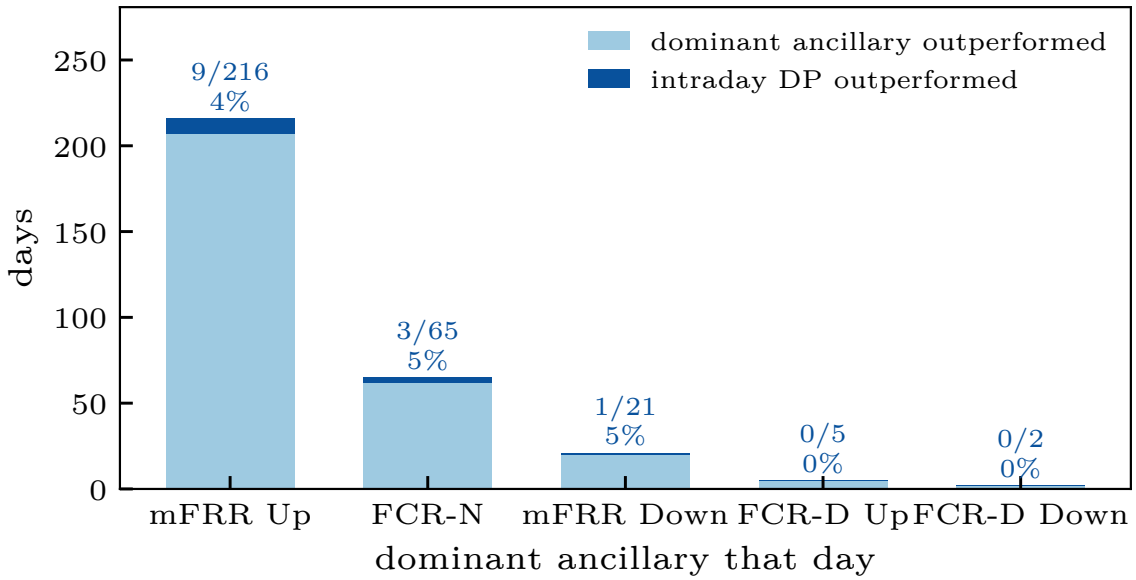


Figure 4.11: SE4 event-driven intraday DP daily P&L compared against the daily highest-paying ancillary for the same day, 2025-05-09 – 2026-03-13 (309 days). Stacked bar per dominant ancillary; the numerator/denominator above each bar is the DP outperformance count over the total day count for that dominant ancillary.

4.4.1.3 Germany Amprion

Over January 2025–February 2026 (Figure 4.12), the Germany Amprion event-driven intraday DP cumulates to €1.08 M, the intraday-auction perfect-foresight upper bound to €0.70 M, and the day-ahead-forecast bidder to €0.16 M. Passive capacity totals: aFRR-CM-up €2.43 M, aFRR-CM-down €2.18 M, mFRR-CM-down €1.32 M, mFRR-CM-up €0.66 M. Capacity totals use the Germany marginal clearing price at 10 MW offered per 4-hour product block; the two auction benchmarks cover 352 delivery days of 2025, a narrower window than the 14-month event-driven DP series shown alongside.

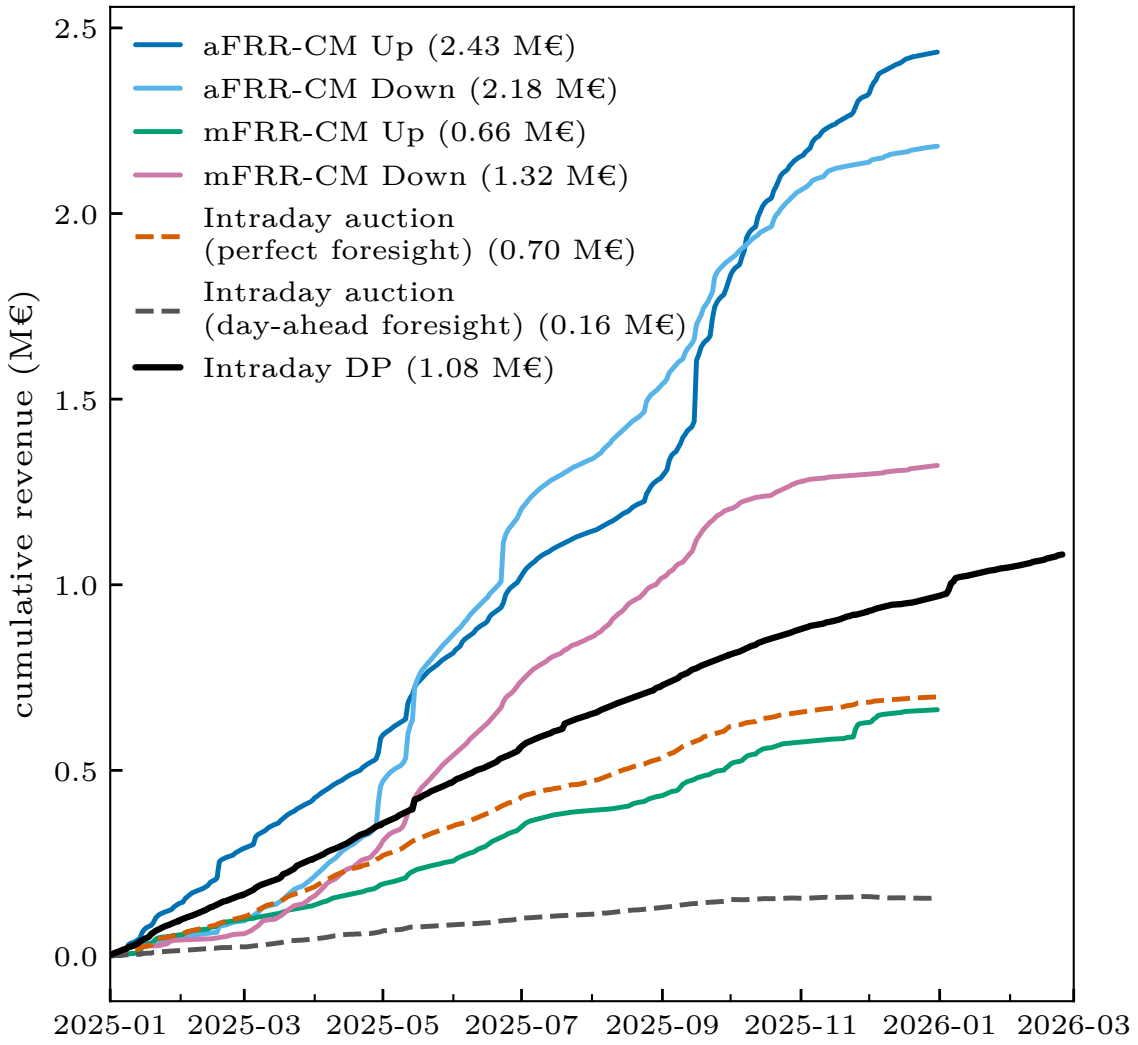


Figure 4.12: Cumulative revenue, Germany Amprion, January 2025–February 2026, 10 MWh battery. Event-driven intraday DP is the canonical main-run series (Section 4.3); perfect-foresight and day-ahead-forecast benchmarks bid into the intraday auction.

Daily outperformance. Across the 365 days of calendar 2025 (Figure 4.13) the Germany Amprion intraday DP outperforms the daily highest-paying ancillary on 35 days (10%). Most wins fall on aFRR-dominant days (32 of 316); on the 49 days an mFRR product tops the stack the DP wins 3.

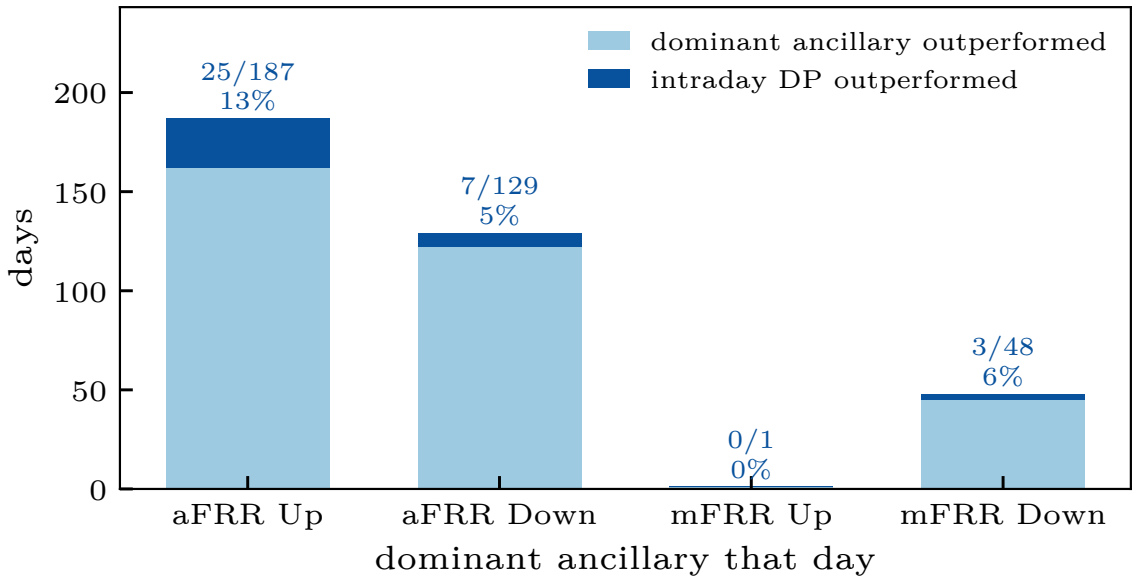


Figure 4.13: Germany Amprion event-driven intraday DP daily P&L compared against the daily highest-paying ancillary for the same day, calendar 2025 (365 days). Layout as Figure 4.11.

4.5 Solver performance

The optimization effort documented in section 3.6 reduced the per-step wall time from ~ 40 ms in a naive C++ port of the Python prototype to 2.639 ms in the final v14.2 baseline on the 2026-03 SE3 month-long benchmark. This is an end-to-end speedup of approximately 15×. Of the v14.2 wall time, roughly 85% is spent in the DP kernel and the remaining 15% in the surrounding Python pipeline. On the busier first half of October 2025 the same configuration reproduces a wall time of 3.030 ms per step, a 1.51× speedup over the v7 reference; the corresponding non-solver speedup on the busier month is 3.06×. Inside the kernel, the libgomp fork–join overhead measures 0.634 μs per parallel region, accounting for less than 2% of kernel wall time.

The bit-identical verification protocol of section 3.6 held across all kernel-only revisions: realized P&L is bit-identical between v7 and v13. The pipeline-level v8 port introduced a small change in same-timestamp event-tie handling between the Python replay and the C++ `BookManager`, producing a persistent drift of -0.7% in realized P&L from v8 onwards.

4.6 Impact on Flower

This section discusses the impact of the DP algorithm on Flower's operations based on the results above.

4.6.1 Per-battery economics

On a single 10 MWh / 1 h battery the event-driven DP earns **€748,041**/yr in SE3, **€931,874**/yr in SE4 (both 12-month windows), and **€1,081,490** in Germany Amprion over the 14-month window (Table 4.12); the Germany figure annualizes to roughly €927 k/yr, which is comparable to SE4 on a like-for-like basis.

The IEA 2024 battery survey reports a 2023 lithium-ion price of roughly USD 139/kWh at cell/pack level and USD 250–400/kWh at fully-installed system level [23]. Converting at 0.92 EUR/USD and scaling to the 10 MWh pack gives a cell/pack capex of €1.28 M and a system-level midpoint of €2.99 M (range €2.30–€3.68 M). Annualized, these revenues imply simple paybacks of 1.71 yr (cell/pack) to 4.00 yr (system) in SE3, 1.37 to 3.21 yr in SE4, and 1.38 to 3.23 yr in Germany. Even on the system-level basis the battery recovers its installed cost well within a typical pack service life in all three zones.

4.6.2 Fleet economics

On the event-driven timeline the per-asset rate is 163.3 trades/h, so a homogeneous fleet reaches the exchange's order-rate limit of 5,000 trades/h [20] at 30 identical batteries ($30 \times 163.3 = 4{,}899$ trades/h); the 31st would breach it. A fleet of up to 30 batteries can therefore run event-driven. Beyond 30 it must slow to the fixed-6 s timeline, where the per-asset rate falls to 96 trades/h and the cap is not reached until 52 batteries ($52 \times 96 = 4{,}992$ trades/h), at a per-asset revenue cost of 10% in SE3 (€748 K to €676 K) and 8% in SE4 (€932 K to €854 K; Table 4.12).

Per-asset revenue falls as aggregate capacity grows. April 2025 captured 11.0% of the SE3 annual P&L (€82,121 of €748,041), so the sweep's April rates annualize by ×9.11: a single 10 MWh battery yields ∼€74 K/MWh/yr, falling to ∼€63 K at 100 MWh, ∼€52 K at 500 MWh, and ∼€47 K at 1 GWh, a factor of 1.6 over the full 100× capacity range (Table 4.6). By interpolation the 30-battery / 300 MWh aggregate sits near €56 K/MWh/yr, a ∼ 24% discount on the single-asset figure. Thirty batteries stay under the order-rate cap on the event-driven timeline, so the fleet earns $30 \times$ €748 K $\times$ (56/74) $\approx$ **€17.0 M/yr** in SE3. A fleet larger than 30 would have to drop to the fixed-6 s timeline to stay under the cap, forgoing the 10% event-driven uplift per asset on top of this liquidity discount.

4.6.3 Opportunity cost versus ancillary markets

Passive ancillary out-earns the event-driven intraday DP on most days in every zone (Section 4.4.1), so the figure that matters for an ancillary-default fleet is the uplift the DP adds on the days it pays more. Table 4.13 reports that uplift over the per-day best-ancillary baseline (Section 3.5.3).

Table 4.13: Intraday-DP uplift over the per-day best-ancillary baseline. *Best-anc* is the daily maximum across all products in the zone; per-product totals are aligned with Section 4.4.1. *Germany (full)* uses the full aFRR+mFRR stack; *Germany (mFRR only)* restricts to the contestable mFRR-CM subset. Swedish windows cover 2025-05-09 to 2026-03-13; the German window covers calendar 2025.

Zone	DP-only (€ M)	Best-anc (€ M)	DP uplift (€ K/yr)	DP wins (days)
SE3	0.63	3.13	40	26/309
SE4	0.79	3.64	26	13/309
Germany (full)	0.95	3.08	34	35/365
Germany (mFRR only)	0.95	1.51	159	122/365

The uplift is **€ 40 K/yr** per battery in SE3 and **€ 26 K/yr** in SE4. In Germany it is **€ 34 K/yr** against the full aFRR+mFRR stack, rising to **€ 159 K/yr** against the mFRR-CM products a battery can readily contest [22, 14].

4.7 Impact on the environment

The event-driven DP charges in lower-price windows (night and early afternoon) and discharges into higher-price evening periods (Figures 4.1–4.5), so a cycle's net effect depends on how much dirtier the discharge-hour marginal is than the charge-hour marginal, net of the $\sim$10% round-trip loss (Section 3.7.4). From the zone marginal-intensity ranges of Section 2.2.3, the charge- and discharge-hour factors are taken as 30 and 40 g/kWh for Sweden and 280 and 500 g/kWh for Germany, applied to each zone's main-run fixed-6 s throughput (Table 4.14).

Under these assumptions a cycle is net climate-beneficial in all three zones, but by very different margins. The Swedish benefit is small, $+126\,\mathrm{tCO_2/yr}$ in SE3 and $+134\,\mathrm{tCO_2/yr}$ in SE4, because the marginal barely changes between charge and discharge hours. Germany is far larger at $+3{,}705\,\mathrm{tCO_2/yr}$. Per profit-euro the German battery avoids $\sim 3.4\,\mathrm{kg\,CO_2/EUR}$, against ~ 0.17 in SE3 and ~ 0.14 in SE4 (Table 4.14).

The German sign is stable: sweeping $\mathrm{MEF_{chg}} \in [200, 350]$ and $\mathrm{MEF_{dis}} \in [400, 550]$ g/kWh keeps it net-beneficial ($+240$ to $+6{,}400\,\mathrm{tCO_2/yr}$). The Swedish sign is not: the narrow marginal gap lets the sweep ($[20, 40]$ / $[30, 50]$) run from -270 to $+520\,\mathrm{tCO_2/yr}$, flipping net-harmful at the boundary.

Table 4.14: Annual CO_2 displacement per zone for a single 10 MWh / 1 h battery under the two-bucket assumptions of Section 3.7.4. Cycles/day are taken directly from each zone's main-run fixed-6 s annual mean (5.09 SE3, 5.43 SE4, 5.35 Germany; see Appendices C.1–C.3). Two-bucket MEFs are as stated above.

Zone	E_{chg} (MWh/yr)	E_{dis} (MWh/yr)	MEF_{chg} (g/kWh)	MEF_{dis} (g/kWh)	avoided (t/yr)	kg/€
SE3	20,587	18,579	30	40	+126	0.17
SE4	21,963	19,820	30	40	+134	0.14
DE Amp.	21,638	19,528	280	500	+3,705	3.43

4.8 Impact on the grid

A single 10 MWh battery on the event-driven DP submits 1.36 M trades/yr in SE3, 1.57 M in SE4, and 2.56 M in Germany Amprion (Tables 4.2, C.2, C.3), an annual traded volume of ~ 39, ~ 42, and ~ 41 GWh ($E_{chg} + E_{dis}$). Against zone-level continuous-intraday volume this is a clear price-taker: roughly 0.04% of the ~ 91.3 TWh German book [18] and 0.13–0.16% of the smaller ~ 7.2 TWh Swedish book [77, 16]. The single battery does not move the book.

A 30-battery fleet is not a price-taker. Its per-MWh revenue falls as aggregate capacity grows, by $\sim 15\%$ at 10 batteries, $\sim 24\%$ at 30, $\sim 30\%$ at 50, and $\sim 37\%$ at 100 (holding duration at 1 h, Section 4.6.2). This is direct evidence of liquidity depletion: as deployment grows the strategy competes with itself for the same order-book depth and the realized per-MWh spread compresses.

The FOK kill rate is an independent depth proxy: with a 200 ms execution delay, a planned trade is killed when the standing quote has moved beyond its limit by execution time. At the event-driven timeline it is 0.02% in SE3 and SE4 and 0.03% in Germany, and the gap widens as the timeline coarsens (hourly 1.71%, 2.17%, 2.98%; Tables C.1–C.3), consistent with shallower, faster-moving top-of-book depth in the busier German market. Taken together, the sizing slope and the kill rate place a soft ceiling near a $\sim 30\%$ per-MWh penalty: around 500 MWh aggregate in SE3 (roughly 50 default batteries), and proportionally less in Germany.

5

Conclusion

This thesis set out to test whether the high-frequency rolling-intrinsic storage strategy of Schaurecker et al. [1] can be adapted to Flower's operational setting: quarter-hour continuous intraday order books and the Swedish markets SE3 and SE4. The answer is yes, with three important qualifications. First, the event-driven DP produces material intraday revenue in all three studied markets, and in general, the results of the reference paper translate well. Second, the commercial value of that revenue depends strongly on market selection and timing: the algorithm is not always the best use of the battery once ancillary-market opportunity costs are included. Third, the broader impact is mixed. For Flower, the intraday DP outperforms the highest-paying ancillary alternative on 8%, 4% and 10% of days in SE3, SE4 and Germany Amprion respectively, so its commercial value depends on whether those days can be forecast at ancillary-auction time, and whether these trends persist. For the grid, a single battery is a price-taker (0.04% of the German book, 0.13–0.16% of SE3), and whether its trades dampen or amplify residual imbalance is not measured here. For the environment, a charge–discharge cycle is net carbon-beneficial in the central case, small in Sweden (+126 and +134 tCO_2/yr) and larger in Germany (+3,705 tCO_2/yr), though this rests on a two-bucket marginal-emissions proxy.

5.1 Main findings

The empirical results show a clear and stable value of high-frequency decision making. Over the evaluation windows of this thesis (April 2025–March 2026 in Sweden, January 2025–February 2026 in Germany), the event-driven DP earns €748 k in SE3 and €932 k in SE4 over 12 months, and €1.082 M in Germany Amprion over 14 months (annualized: ∼€927 k/yr) for a single 10 MWh / 1 h battery. Relative to the next-best fixed-6 s timeline, the event-driven timeline increases profit by 11% in SE3, 9% in SE4, and 15% in Germany. Against an hourly timeline, the gains are larger still: 70%, 51%, and 33%, respectively. The benefit is therefore not only that the DP finds profitable temporal spreads, but that it reacts quickly enough to capture spreads before they disappear from the order book. This advantage is structural rather than concentrated in a few volatile months: the cumulative-reward trajectories at the six decision timelines separate from the start of the evaluation window and continue to diverge throughout the year for all markets.

On the window common to all three zones (April 2025–January 2026, the only span comparable across all three) the cross-market ordering is SE4 (€833 k) >

Germany Amprion (€787 k) > SE3 (€670 k), and the time-averaged revenue rate agrees (SE4 11.19 > Germany 10.73 > SE3 8.98 €/MWh/h). Germany earns the highest raw total only because its window is four months longer. What is consistent across all three zones is the event-driven uplift over every fixed timeline.

Several market differences could explain the cross-market profit gap, and the design here does not isolate which one dominates. Three are plausible. First, market efficiency: the German continuous market is weak-form efficient [30], while the efficiency of SE3 and SE4 is unmeasured, and a less efficient market would leave more arbitrage for the rolling intrinsic to collect. Second, volatility: larger price swings should widen the temporal spreads the policy earns on, but we did not quantify realized volatility per zone. Third, market design: Germany Amprion has a rolling per-contract gate closure that keeps contracts tradable to within five minutes of delivery, where liquidity concentrates, likely the single largest structural difference between the markets. Liquidity interacts with all three rather than acting as a clean single cause: over the comparable window the less liquid SE4 earns more than Germany Amprion, so order-book depth alone does not order the zones. The Swedish zones, by contrast, share a batched gate closure for the four quarter-hour contracts in an hour, which limits the last moments at which trading occurs. Within Sweden, the higher trading volumes and larger absolute profits observed in SE4 relative to SE3 are consistent with SE4's denser order book and possibly wider imbalance-driven temporal spreads because of imports. The substantially higher order kill rates observed in Germany Amprion at every timeline are also consistent with this market-structure story: denser books and shorter-lived quotes under a rolling per-contract gate closure (Figure 2.2) leave a larger share of limit orders stale by the time they reach the exchange. The difference in the number of negative P&L days likely comes from deeper liquidity as well. In theory, losses should occur because of two reasons: either the interpolated value functions are too optimistic, or a partial fill leaves the SoC on a path that will lead to infeasibility unless a loss-making trade corrects it. With deeper liquidity, both partial fills and reconciliatory trades should be less impactful.

The reason our results are significantly higher than the reference paper comes down to three factors. First, we explicitly omitted battery degradation costs from our overall P&L, as per Flower's request — the cost is still incurred physically, but it is not subtracted from the reported revenue. Second, the policy operates at a substantially higher cycling. Third, quarter-hour contracts allow for four times more profitable opportunities than the hourly products used in the reference.

The state-of-charge schedules show that the DP follows a diurnal SoC path, as shown by the yearly schedules for SE3 (Figure 4.1), SE4 (Figure 4.3), and Germany Amprion (Figure 4.5). Prices typically dip during low-demand morning and midday hours and peak in the evening as residential demand rises, and the policy correspondingly charges in the early morning and early afternoon and discharges in the evening. The seasonal modulation, a sharper pattern in spring and early autumn and a softer one in November and December, tracks the inter-contract spread availability discussed above and may be consistent with wind- and solar-driven widening of spreads in high-renewable months. The hypothesis is that with higher unpre-

dictability of renewable generation, the prices creating the diurnal SoC path are less likely to occur.

5.2 Usability of penalty parameter

The penalty parameter ϕ yielded an expected increase in profits of 6.21%. However, there was significant overfitting for the penalty tuned on October 2025 and used in November 2025, yielding a loss of 19.4%. This presents a significant risk in using the analyzed tuning schedule, one that is not surprising given its structure. For example, this schedule makes the market behavior on October 1 impact the penalty on November 30, where market conditions may be very different. Our proposed response is to introduce a tuning schedule in which the penalty is more dynamic and based on events closer in time to the application period. When experimenting with a smaller dataset, consisting of five minute resolution with only the five best asks and bids, a rolling penalty constructed as the exponential moving average of ten daily penalties, each fit on its own preceding thirty-day window, profits were increased by 7.4%. While the increase is small, a rolling penalty could reduce downside risk. Also, the reward can be evaluated on more data points: 365 per year compared to 11 for the original schedule. Furthermore, there could be other, more advanced methods to tune a more effective penalty, including adaptive risk-measure tuning on a rolling window of realized performance [78], microstructure-aware penalties whose effective cost floor scales with order-book depth rather than calendar-day windows [79], and probabilistic spread-forecast gating that trades only when the forecast bid-ask spread is below the penalty threshold [80].

5.3 Battery size analysis

Battery sizing results show that the strategy is liquidity-limited. In SE3 (April 2025, fixed-6 s timeline), revenue per MWh of installed energy declines monotonically with both battery capacity and duration. Small, high-power installations earn the most per installed MWh because their trades can be executed near the top of the order book, while larger installations must walk deeper into the book and accept worse prices. Shorter durations dominate on a per-MWh basis – the 0.5 h column is roughly 2.5× the 2 h column at every capacity, because a 0.5 h cell has twice the power per MWh and completes more arbitrage cycles per day. While not explicitly tested, we believe that this decline will be less pronounced for more liquid markets, such as Germany Amprion. Then, the bigger battery will have deeper order books to trade on, which will mitigate the decrease in per MWh profitability. Additionally, since continuous-intraday volumes have grown substantially with rising renewable penetration [4], this decrease is likely to become less pronounced as the market itself becomes more liquid into the future.

5.4 Comparison to MILP

Two findings stand out from the head-to-head benchmark in subsection 4.1.4, run at a common decision timeline so that the two solvers are compared on equal footing rather than at their respective natural rates.

First, the MILP outperforms the DP in every window, and within the DP the reward grows monotonically with the grid size m. This is the ordering one would expect from the mathematics: the MILP solves each rolling-intrinsic instance exactly in the SoC dimension, and a finer DP grid reduces the value-function interpolation error. Coarser DP grids also overtrade relative to the finer grids and to the MILP, consistent with value-function noise generating spurious round-trips that move volume without adding profit. This clean ordering is a more theoretically expected outcome than the one reported in the reference paper [1], where the coarser DP variants occasionally exceeded the MILP in single two-week windows. We do not reproduce that inversion on the SE3 data, and we read its absence as a sign that the comparison here behaves as the approximation theory would predict. Additionally, we did see a significant reduction in profits when using a coarser grid, again in contrast with the reference paper. This may be due to a smarter infeasibility handling, or, as we saw with preliminary backtests on the German market, a difference because of the lower liquidity of the SE market.

Second, an exploratory continuation of the grid-size sweep beyond m=401 showed a further positive trend in reward with diminishing returns, which means the DP has not saturated at the grid sizes used elsewhere in the thesis. The implication for a production deployment is that a finer SoC grid would extract additional value, moving the DP closer to the MILP upper bound. The qualifier is that a finer grid increases per-solve time, and once per-solve time grows the policy reacts to fewer order-book updates and exposes its planned orders to a longer interval during which the intended order can expire or be matched against other participants. In other words, raising m trades a smaller value-function approximation error against a larger partial-fill risk.

Whether the additional precision is worth the slower reaction is genuinely empirical, and the answer depends on the local event rate, the order-book depth, and the available compute budget on the production host. Mapping the optimal grid size against these factors is left as future work.

5.5 High performance computing techniques

The optimization effort yielded a substantial reduction in per-step wall time and, in doing so, made the month-long backtests tractable despite the increased computational burden imposed by quarter-hour contracts and the fine state-of-charge grid. Two aspects of the solver performance reported in section 4.5 merit further comment.

5.5.1 Pipeline ports scale with workload

The solver stack splits into two halves: the DP kernel itself — the C++ core that solves the dynamic program and accounts for roughly 85% of per-step wall time — and the surrounding Python pipeline that feeds it, comprising event replay, order-book snapshotting, packing of the solver buffers, and post-solve action filtering. The two were optimized in separate phases: an early kernel-focused phase, culminating in the OpenMP parallelization of section 3.6, and a later phase (the v8 and v14 revisions) that ported the pipeline hot-paths from Python to C++. Because the kernel dominates wall time, it is reasonable to ask whether this second phase was redundant — whether reducing the remaining 15% was worth pursuing once the kernel was already fast.

The benchmark evidence in section 4.5 indicates that it was. On the busier first half of October 2025 the kernel speed-up over the v7 reference is 1.51×, whereas the non-kernel pipeline speed-up over the same reference is 3.06× — roughly twice as large. The pipeline ports therefore deliver their biggest gains precisely when the event rate is highest, which is the regime that governs the cost of a month-long backtest. The two phases are complementary rather than overlapping: the kernel phase sets the floor on per-solve cost, while the pipeline phase keeps the cost of moving data into and out of the kernel from dominating once the workload climbs.

5.5.2 Kernel parallelization gains are bounded

A further question is whether the kernel could be accelerated through increased thread-level parallelism alone. The thread-scaling analysis of section A.4 indicates that it cannot. Fitting the measured scaling curve to Amdahl's law places the serial fraction at roughly nine percent, which sets a theoretical ceiling of about 11× over single-threaded execution. At the six threads actually used, the kernel realizes only about 4× of this ceiling, so a further factor of two remains available in principle. In practice it is unreachable, for two independent reasons.

First, the per-stage workload is too small to divide further. The value table spans only $m \approx 401$ grid points, and the threads must synchronize at a barrier at every one of the ~ 92 stages of a solve. Across six to eight threads each thread already receives only about fifty grid points per stage. Beyond that point the per-stage barrier cost overtakes the per-thread arithmetic, and adding threads makes the kernel *slower* rather than faster — at sixteen threads the runtime collapses by a factor of several hundred relative to eight (section A.4). Second, the portion that does not scale is memory-bound rather than compute-bound: the same analysis attributes only about fifteen percent of the serial fraction to fork–join overhead (consistent with the sub-two-percent figure measured directly in section 4.5). The remainder is dominated by L3 load latency on the value-table gather, which more threads cannot relieve because they would merely contend for the same memory.

The practical consequence is that increased thread-level parallelism offers no further headroom. Any remaining gains lie in the single-threaded kernel work and in the surrounding pipeline, or in an algorithmic restructuring of the value-table gather

to reduce its memory-latency cost, rather than in a higher thread count. Concrete levers along these lines are deferred to subsection 5.9.2.

5.6 Practical implications

From Flower's perspective, the clearest product implication is that the intraday DP should be treated as a conditional trading policy, not as a default replacement for ancillary-market participation. The comparison against ancillary revenues, now computed for all three zones (Section 4.4.1), shows that intraday trading beats the dominant daily ancillary alternative on only a minority of days: 26 of 309 days in SE3 (8%), 13 of 309 in SE4 (4%), and 35 of 365 in Germany Amprion (10%). The win pattern is structured rather than random. In the Swedish zones the DP wins most often on days when the mid-paying FCR-N tops the stack rather than the higher-paying mFRR-CM-up, and never on the few days the lowest-paying FCR-D products dominate. Germany Amprion has the highest daily win-rate: its aFRR products out-earn the DP many times over in cumulative terms, yet the DP still beats the best-paying alternative on 10% of individual days, concentrated on the aFRR-up-dominant days (25 of 187). A production scheduler should therefore decide between intraday and ancillary allocation day by day, or even contract block by contract block, based on expected ancillary clearing prices and current intraday liquidity.

How to effectively predict when the intraday DP beats the ancillary alternative is outside the scope of this thesis. Considerable exploratory work was nonetheless done to classify market conditions, compare scenarios, and evaluate the predictability of profits, without yielding a usable predictor. That effort, and its two stable findings — the role of solar- and load-forecast revisions in driving the DP's intraday edge — are documented in Appendix D.

However, despite the current moderate practical usefulness, we believe two market developments will make the intraday DP more valuable in the future: the increase in continuous-intraday volumes and the decrease in ancillary market prices. The within-Sweden comparison shows the volume effect directly: SE4's denser order book earns more than the thinner SE3, on both total profit and per-MWh/h terms, so more depth gives the policy more profitable trades to find. Furthermore, a decrease in ancillary market prices will naturally make the intraday DP more attractive. This is already particularly true for Germany Amprion, where it is outperformed primarily by the aFRR benchmark. Access to the aFRR market is constrained by entry barriers [81], and the procured volume is small and fixed relative to the growing battery fleet [82]. Hence, most actors will not have this as an option.

The implementation results show that such a policy is technically feasible. Across the SE3 March 2026 benchmark, the full pipeline reaches 2.639 ms per step. This does not by itself prove production readiness, but it demonstrates that the algorithmic idea can be evaluated on the event-driven timeline on realistic data volumes and that latency budgets in the low-millisecond range are not obviously out of reach.

The event-driven high trade rate has implications for immediate and future production deployment. At fleet scale the exchange's order-rate limit of 5,000 trades/h is the

binding constraint on the timeline. The event-driven timeline runs at 163.3 trades/h per asset, so a homogeneous fleet reaches the cap at 30 identical batteries and the 31st would breach it (Section 4.6.2). A fleet larger than 30 must run the fixed-6 s timeline, which at 96 trades/h per asset stays under the cap up to 52 batteries, making it the realistic production timeline at scale. Switching to it forgoes the event-driven uplift: about 9–11% per asset in the Swedish zones and 15% in Germany Amprion. Batching orders together or running the DP for several assets at once could push the event-driven ceiling above 30. The worst case, in which every battery runs the intraday DP at once, may not be realistic, so the aggregate rate could stay under the cap and let a larger fleet run event-driven.

5.7 Limitations

Several limitations should temper the interpretation of the reported profits. First, the backtests assume that the historical order book is not affected by the strategy's own trades, apart from the specific order being removed. This is reasonable for small volumes but becomes less defensible as battery size or fleet size grows, especially in the thinner Swedish books, where added battery capacity itself compresses the temporal spreads the policy trades on [24]; the liquidity-degradation slope of the sizing sweep (Section 4.6.2) is the empirical warning. Second, reported profits exclude battery degradation. The numbers represent market cash flow before asset-wear costs, not full economic profit; ignoring degradation can erode arbitrage profit, sometimes substantially [27, 28]. Third, the execution model uses a fixed technical delay and a partial-fill policy, both of which are more realistic than frictionless execution but still simplify the behavior of a live exchange connection. Specifically, it has been noted that the real execution delay from the exchange can vary significantly, and can sometimes be several seconds [67]. Note however that this delay presumably affects all actors on the exchange the same, reducing its impact. Finally, the comparison to ancillary markets, while now computed for all three zones, treats the ancillary alternative as a passive full-period capacity commitment at the marginal clearing price and excludes activation energy, as well as buyback and sellback costs. It is a rough opportunity-cost benchmark rather than a like-for-like comparison, and a fleet-wide dispatch rule would need to value the two revenue streams jointly within a single day. Beyond the profit figures, the per-battery paybacks rest on 2023 installed-cost figures that have varied by roughly a factor of two year to year [23], and the per-MWh fleet revenue assumes each battery's power rating scales with its energy rating, so longer-duration packs would be power-limited rather than energy-limited.

5.8 Impact on Flower, the environment and the grid

As a standalone revenue stream the intraday DP is self-financing on a single battery: annual P&L of €0.75–€0.93 M (Germany annualized; the 14-month raw total is €1.08 M) on a 10 MWh / 1 h battery implies simple paybacks of roughly 1.4 to 4

years across the three zones studied, even on a fully-installed system-level capex basis. This gross figure is not the operationally relevant one: most of it overlaps with ancillary revenue the battery would earn anyway (Section 4.6.3).

Scaled to a homogeneous fleet of 30 batteries, the most that fit under the exchange's order-rate limit on the event-driven timeline, the intraday DP earns roughly €17.0 M/yr in SE3, but the scaling is sublinear. It stays below 30× the single-asset figure because per-asset revenue erodes by about a quarter: all 30 batteries draw on the same order-book depth. The order-rate cap sets how large the fleet can grow before it must slow to the fixed-6 s timeline and forgo the event-driven uplift; within that size, the binding constraint on aggregate fleet value is shared order-book liquidity.

The intraday DP is not Flower's primary use of the battery: passive ancillary outearns it on most days in every zone, so its operational contribution is the uplift it adds on the days it pays more, €40 K/yr per battery in SE3 and €26 K/yr in SE4. The case is strongest in Germany, where aFRR is the dominant ancillary service. Because aFRR requires prequalification, the baseline a battery can readily contest is the mFRR products, against which the intraday DP adds €159 K/yr. Its broader commercial value lies less in standalone intraday revenue than in the additional use cases the same rolling-intrinsic engine supports: buybacks of ancillary commitments, joint optimization across markets, and compliance-aware asset steering. Carrying the engine into production is therefore a one-time investment rather than a per-extension rebuild.

A charge–discharge cycle is carbon-beneficial in the central case in every zone, but its size and even its sign depend on the regime. Where both charge and discharge hours are coal-marginal the round-trip loss dominates and a cycle can add CO_2 [38]; where renewables set the charge-hour marginal the benefit is clear. Germany sits in the transitional regime, solar rising and coal retiring, so its +3,705 tCO_2/yr estimate will drift year to year, while the Swedish benefit stays small because the marginal barely moves across the day. We conclude that the direct displacement is real but modest, and that the larger climate contribution is likely second-order: bidding into the lowest-price windows raises the revenue floor for variable renewables and underwrites capacity a price-taker backtest cannot quantify [36]. Both rest on a two-bucket marginal-emissions proxy; a defensible all-hours figure would use an hourly marginal profile [40]. Set against the battery's embodied CO_2 (of order 1,000 t for a 10 MWh pack at the cell-production median of Section 2.2.3 [46]), the German operational displacement repays that within months, the Swedish over several years.

On the grid, a single battery is a price-taker — well under 0.2% of the intraday book in every zone — so it neither moves prices nor changes balancing in any measurable way. At fleet scale the batteries compete for the same depth, and the spread compression that is welfare-positive while the fleet is small turns ambiguous once the fleet is large enough to set the marginal price [38, 36]; the empirical slopes put that turning point beyond ∼100 MWh aggregate in SE3. We do not measure whether the DP's trades dampen or amplify residual imbalance: that would need a sign-aligned comparison against settled balancing volumes, and the trades are

conditioned on the same solar- and load-forecast revisions that drive imbalance cost (Appendix D). The fleet-on-fleet effect against other algorithmic participants is left open, since a static replay cannot model the closed loop.

5.9 Potential improvements and future work

Overall, the thesis shows that high-frequency rolling-intrinsic battery trading earns material revenue and runs at a tractable cost on real European intraday order books. The DP is not, however, the right choice on most days: ancillary capacity markets pay more on the majority of days in every studied zone. Its place is alongside ancillary participation, taking over on the days when intraday spreads beat the dominant ancillary product. The rest of this section covers what is still missing to deploy that combination in production, what can be improved on the algorithm side, and what additional use cases the same engine could support.

5.9.1 Production Implementation

This thesis benchmarks the DP in offline simulation across three zones. Putting it in front of a live exchange would necessitate additional work in several areas. The first piece is a forecaster of intraday-vs-ancillary value. The DP only wins on a minority of days in every zone (Section 4.4.1), so the allocation rule has to predict those days ahead of ancillary auction close.

Second, the assumed technical delay should be validated against measured round-trips, since the real delay can vary [67] rather than be a fixed value used in the backtests, and the partial-fill rule needs stress-testing on high-volatility days. The exchange also caps trades at 5,000 trades/h [20], which the current backtests ignore. The simple workaround is to use the 6 s timeline, but a more structural fix is to raise the event-driven threshold dynamically when the recent trade rate approaches the cap, or introduce a trade rate penalty, which may keep most of the event-driven profit without breaching the limit. Cycling limits also need to be considered: the event-driven and fixed-6 s policies average 6.28 and 5.09 cycles/day on the SE3 10 MWh battery (Table 4.3). Each asset has its own throughput limit set by the owner, so a cycling penalty needs to be specified per asset.

On the numerical side, the share of infeasible states on thin books should be reduced further. Non-uniform SoC grids were tested in this thesis but did not deliver the expected improvement. The remaining candidates are explicit feasibility bookkeeping and adaptive refinement near the SoC boundaries.

5.9.2 Improvements in algorithm performance

This subsection collects changes to the algorithm itself, separate from the production-deployment work above. Each item below targets a different source of leftover profit, presented in roughly the order in which we expect them to matter: the terminal value of state of charge at the end of the optimization horizon, additional contract

types and the day-ahead market, the event-driven trigger, the penalty structure, and posting limit orders rather than only taking the book.

5.9.2.1 Terminal value

The DP solves a finite-horizon program that runs until the last delivered intraday contract. At that horizon the residual state-of-charge is valued through a terminal-value function, which in the thesis implementation is simply set to zero. With a zero terminal value the DP gets no credit for SoC held past the horizon, so it tends to drain the battery at the end of the optimization window even when the prices it sells into are below what the next day would have paid. This is the most likely place where the DP currently leaves profit unrecovered.

The most direct fix, and the one we expect to recover the most leftover profit, is to build the terminal value from the day-ahead clearing price for each remaining contract, then a price forecast for the following horizon, then an average price thereafter. The construction is still deterministic, but it gives the DP a sensible default value for residual SoC and removes the end-of-horizon drain. Beyond this, the alternatives below are more structured options, in roughly increasing order of implementation effort:

1. **Multi-day rollout with discounting.** Extend the horizon over several days, applying a per-day discount factor that reflects the increasing forecast uncertainty further out. A one-day horizon hides any price the DP could earn the next day, so it has no reason to hold SoC across the boundary even when the next day would pay more. A two- or three-day horizon fixes this.
2. **Stochastic rollout instead of deterministic.** Sample many price paths from a calibrated price model and average the resulting terminal values across paths. A deterministic construction undervalues holding state-of-charge into a volatile horizon, because it cannot see the option value of being able to act differently in each future scenario. Sampling restores that optionality and weights end-of-horizon decisions by realistic uncertainty rather than a single point estimate.
3. **Forward-curve-implied terminal value.** Read the terminal value directly from the term structure of forward and futures contracts, so residual SoC at the horizon is worth whatever the nearest forward contract clears at. The advantage is that the boundary condition is grounded in observable hedging instruments rather than a constructed forecast. The downside is that short-dated power forward markets can be illiquid, so the implied value may not be available at the relevant horizon for every contract.
4. **SDDP / piecewise-linear value-function fitting.** Use stochastic dual dynamic programming, or an equivalent Benders-cut scheme, to fit a piecewise-linear convex approximation of the storage value function from sampled future price scenarios. The method has been used successfully for the related hydrothermal scheduling problem, where reservoir storage plays the same coupling role that SoC does here [8].
5. **Stationary infinite-horizon value function.** Solve once for a stationary value

function on the SoC state under an assumed recurring price process, and use it as the terminal condition every day [56]. This removes the end-of-day discontinuity completely, since there is no longer a boundary to assign a value to. The binding assumption is that the price process is approximately stationary; seasonality and longer-term trends mean any single fit would need periodic refitting.

6. **Fitted value iteration with a learned V_T.** Train a model (a neural network or a random forest) to predict the terminal value V_T as a function of state of charge, time of day, recent prices, and other observed features [63]. The model is fit on simulated trajectories from the existing DP or on historical realized P&L, and replaces a forecast-based construction with a data-driven boundary that adapts to whichever regime the market is in.

5.9.2.2 Other contract granularities and day-ahead

The contract set used in this thesis is only quarter-hour (15-minute) intraday contracts. The other granularities available differ by zone. Germany Amprion has 1-hour, 30-minute and 15-minute contracts all trading in parallel. Sweden SE3 and SE4 only have 1-hour and 15-minute contracts. Adding the hourly contracts (and the half-hourly contract in Germany) is the most direct extension of the present formulation. The cross-contract part of the optimization already exists; what changes is the size of the contract set and the bookkeeping for which contracts a given quarter-hour position can be unwound on.

The day-ahead market is a different problem. It is a single daily auction, not a continuous order book, so it cannot be incorporated into the rolling-intrinsic loop as another contract. A reasonable integration is to treat the day-ahead clearing as the initial SoC schedule for the day and let the intraday DP rebalance around it as new information arrives. This is the same structural change required for the buyback-and-restore extension in subsubsection 5.9.3.1.

5.9.2.3 Improving the event-driven nature of the policy

The event-driven timeline fires the DP whenever the best bid or the best ask moves by more than 0.1€/MWh in either direction. The first event that crosses the threshold triggers a solve, and that solve blocks any further trigger inside its 200ms execution latency, so subsequent events arriving in that window are not separately acted on. A smarter rule could improve on this in two ways. It could condition the threshold on order-book state, raising it in calm markets where most events are small and would not change the optimal action, and lowering it when prices are moving rapidly. It could also wait briefly before triggering when more information is likely to arrive in the latency window, so the solve uses the best available book rather than the one that happened to cross the threshold first. Both changes can unlock profit that the first-event-wins rule currently leaves on the table.

5.9.2.4 More penalties

Two penalty components shape the formulation. The spread penalty ϕ depends on trade size and spread, so a proposed trade is penalized in proportion to its size, which deters the policy from trading deep into thin books. A flat per-trade degradation cost acts as a profit threshold: trades whose expected profit does not exceed it are not executed. Penalty tuning is delicate even for ϕ alone, since a value fit on one window can overfit and underperform on others.

The rolling penalty proposed earlier is one structural improvement. It constructs ϕ for each day as an exponential moving average of ten daily penalties, where each daily penalty is fit on its own preceding thirty-day window. The value used on a given day therefore reflects recent market conditions rather than a single calendar-year schedule. Beyond this, a penalty can be made conditional on factors the optimizer currently treats uniformly: proximity to gate closure (to deter speculative early trades), the specific contract or market (to reflect liquidity differences), as well as a number-of-trades penalty and an explicit cycling penalty.

5.9.2.5 Posting limit orders

The market-taker policy executes every trade against existing bids and asks, paying the full half-spread on each side. Posting limit orders instead (acting as a market maker on selected contracts) would let the policy capture the half-spread on the filled portion of its volume. The cost is fill uncertainty and adverse-selection risk. Two ingredients are missing before this is realistic. The first is a credible fill model: an estimate of the probability that a posted limit at a given distance from the top of the book is filled within the holding period, conditional on the current order-book state. The second is an explicit treatment of adverse selection. The trades that do fill will be precisely those where the market has moved against the posted price [79]. A reasonable starting point is to enable limit posting on the most liquid contracts in Germany Amprion, where order arrival rates are highest and the empirical fill model can be calibrated on a meaningful sample. Additionally, this could be combined with an engine that measures the full profit from the DP algorithm given that a certain limit order is filled. This way, the algorithm can execute a strategy it previously could not, something we believe could significantly increase profits.

5.9.3 Additional usecases

The same rolling-intrinsic engine can be used for purposes beyond pure intraday arbitrage. Four of those uses are sketched below.

5.9.3.1 Buybacks and restoration

The same rolling-intrinsic engine can value the decision to retire a commitment made on another market. The motivating case is a battery that has cleared an ancillary capacity obligation for some upcoming delivery period. Given the intraday prices subsequently observed, it might do better to buy back the obligation and trade the intraday spread instead. The DP can handle this with a modified boundary

condition. An outstanding obligation imposes a fixed-direction power flow during the obligation window. That flow enters the SoC dynamics as a known disturbance. The DP then chooses between paying the current buyback price and honoring the obligation later. A similar logic applies after a partial fill on the intraday side. Rather than treating the residual as a fixed disturbance, the DP can be re-formulated to actively unwind it on the most favorable subsequent contract.

5.9.3.2 Interactions with other markets

The DP can be extended into a joint optimizer that decides intraday participation alongside day-ahead bids and ancillary capacity commitments. Each additional market introduces its own decision variables but shares the SoC trajectory as the coupling state. The existing engine therefore carries the structural part of the problem. What is missing is a calibrated price model for the auction-based markets, and a way to value activation calls on ancillary obligations probabilistically. Both are open problems in the literature rather than mechanical changes to the DP. The direct effect of such a formulation is to replace the current treatment of ancillary participation as a fixed outside option (Section 4.4.1) with an endogenous choice. The same machinery also provides an optimal way to fulfill commitments made on other markets and to hit a target SoC at a target time. This is done by modifying the backward-pass values in the DP at the constrained timestamps, so SoC outcomes that do not match the commitment are penalized or ruled out and the policy is steered toward the required trajectory.

5.9.3.3 Environmentally aware trading

A CO_2-price term can be added to the objective, similar to the spread penalty. Charging during high-emission intervals then carries an additional penalty, and discharging during high-emission intervals carries a reward. With a real-time grid-intensity feed (already available for most European bidding zones) [40] this turns the DP into a directly carbon-aware optimizer. Only the objective function changes, the state space, action space, and solver are untouched, and the trade-off between revenue and avoided emissions is set by the chosen CO_2 price. At high enough prices the policy will favor charging windows that correlate with renewable surplus. Those windows are also typically the lowest-price intervals of the day in renewable-heavy zones, so the revenue cost of carbon awareness shrinks in exactly the regimes where renewables drive the price floor. Wrapping this in a green-certificate or PPA structure, where verified low-carbon charging clears a separate revenue stream, would convert the carbon term from a soft preference into a hard commercial line item.

5.9.3.4 Asset steering and compliance

Flower's assets operate under contractual constraints set by the owner. Typical examples are minimum and maximum SoC bands, must-charge windows before contracted obligations, and exclusions on specific markets or hours. Each of these maps onto a small modification of the DP. SoC bands and must-charge targets are enforced by setting the backward-pass value outside the allowed range to negative

infinity (a hard bound) or to a large finite penalty (a soft one) at the relevant timestamps. Market or hour exclusions are handled by restricting the action set at those timestamps. The same engine then handles both the trading and the contract compliance. Instead of running a separate rule engine on top of the trading policy, the constraints are part of the optimization, and the trade plan the DP returns is contract-compliant by construction. The practical pay-off for Flower is that asset-owner agreements differ across portfolios, and a single configurable optimizer is simpler to operate and audit than a stack of per-asset post-processing rules.

5.9.3.5 Bid submission prices

Ancillary markets, the day-ahead market and the intraday auction market all require submission of bids to participate. In theory the auctions are constructed so that each participant is encouraged to bid their marginal cost of production or delivery. However, in practice, it may not be clear what this cost is, or participation in a specific market is part of a broader strategy, making it optimal for the participant to bid any price that gives access to that market.

As the results show, the DP generates relatively stable and predictable profits and is, additionally, always available to an asset owner, making it suitable as a benchmark for the opportunity cost of bidding on other markets. Hence, the asset owner can use the expected DP profits as a reservation price (floor) for bids on other markets, which will both increase profits when the bid sets the clearing price, and reduce the risk of unprofitable participation.

Bibliography

[1] D. Schaurecker, D. Wozabal, N. Löhndorf, and T. Staake, "Maximizing battery storage profits via high-frequency intraday trading." arXiv preprint arXiv:2504.06932, version 3 (August 2025), Apr. 2025.

[2] J. Gray and P. Khandelwal, "Realistic gas storage models II: Trading strategies," *Commodities Now*, pp. 1–5, Sept. 2004.

[3] J. Gray and P. Khandelwal, "Towards a realistic gas storage model," *Commodities Now*, vol. 7, pp. 1–4, 2004.

[4] C. Koch and L. Hirth, "Short-term electricity trading for system balancing: An empirical analysis of the role of intraday trading in balancing Germany's electricity system," *Renewable and Sustainable Energy Reviews*, vol. 113, p. 109275, 2019.

[5] B. Xu, A. Oudalov, A. Ulbig, G. Andersson, and D. S. Kirschen, "Modeling of lithium-ion battery degradation for cell life assessment," *IEEE Transactions on Smart Grid*, vol. 9, no. 2, pp. 1131–1140, 2018.

[6] D. Gräf, J. Marschewski, L. Ibing, D. Huckebrink, M. Fiebrandt, G. Hanau, and V. Bertsch, "What drives capacity degradation in utility-scale battery energy storage systems? The impact of operating strategy and temperature in different grid applications," *Journal of Energy Storage*, vol. 47, p. 103533, 2022.

[7] G. Lai, F. Margot, and N. Secomandi, "An approximate dynamic programming approach to benchmark practice-based heuristics for natural gas storage valuation," *Operations Research*, vol. 58, no. 3, pp. 564–582, 2010.

[8] N. Löhndorf and D. Wozabal, "Gas storage valuation in incomplete markets," *European Journal of Operational Research*, vol. 288, no. 1, pp. 318–330, 2021.

[9] G. Bertrand and A. Papavasiliou, "Adaptive trading in continuous intraday electricity markets for a storage unit," *IEEE Transactions on Power Systems*, vol. 35, no. 3, pp. 2339–2350, 2020.

[10] I. Boukas, D. Ernst, T. Théate, A. Bolland, A. Huynen, M. Buchwald, C. Wynants, and B. Cornélusse, "A deep reinforcement learning framework for continuous intraday market bidding," *Machine Learning*, vol. 110, no. 9, pp. 2335–2387, 2021.

[11] Svenska Kraftnät, "Mimer: Primary regulation (FCR) market data." Online database, Svenska Kraftnät, 2026. Hourly marginal clearing prices (EUR/MW) and volumes (MW) for FCR-N, FCR-D up and FCR-D down across the SE1, SE2, SE3, SE4 and DK2 bidding zones; historical series from January 2021; downloadable as CSV or Excel. Accessed May 2026.

[12] Svenska Kraftnät, "Mimer: Automatic frequency restoration reserve (aFRR) market data." Online database, Svenska Kraftnät, 2026. Hourly marginal clearing prices (EUR/MW) and volumes (MW) for aFRR up and aFRR down across the Swedish bidding zones; downloadable as CSV or Excel. Accessed May 2026.

[13] Svenska Kraftnät, "Mimer: Manual frequency restoration reserve capacity market (mFRR-CM) data." Online database, Svenska Kraftnät, 2026. Marginal clearing prices (EUR/MW) and volumes (MW) for mFRR capacity-market up and down products across the Swedish bidding zones; downloadable as CSV or Excel. Accessed May 2026.

[14] Flower Infrastructure Technologies AB, "Why are prices in ancillary services markets falling?." Market Insight, Flower Infrastructure Technologies AB, 2026. Accessed May 2026; references prequalified-capacity data through April 2026.

[15] P. Spodniak, K. Ollikka, and S. Honkapuro, "The impact of wind power and electricity demand on the relevance of different short-term electricity markets: The Nordic case," *Applied Energy*, vol. 283, p. 116063, 2021.

[16] ENTSO-E, "Market monitoring report 2024," tech. rep., European Network of Transmission System Operators for Electricity, 2024.

[17] S. Hagemann and C. Weber, "An empirical analysis of liquidity and its determinants in The German intraday market for electricity," Tech. Rep. 17/2013, EWL Working Paper, University of Duisburg-Essen, 2013.

[18] EPEX SPOT, "Annual market review 2024," 2024. Accessed 2026.

[19] C. Balardy, "An empirical analysis of the bid-ask spread in the continuous intraday trading of the German power market," *The Energy Journal*, vol. 43, no. 3, 2022.

[20] M. Snellman and M. Evholt. Personal communication, February 2026. Data Scientists, Flower Infrastructure Technologies AB.

[21] Svenska Kraftnät, "Balancing market outlook 2030," tech. rep., Svenska Kraftnät, Sundbyberg, 2024. Overview of the Swedish balancing reserve products (FCR-N, FCR-D, aFRR, mFRR), their roles in frequency control, and outlook to 2030.

[22] 50Hertz Transmission, Amprion, TenneT TSO, and TransnetBW, "Description of concepts for balancing and the balancing markets in germany," tech. rep., regelleistung.net, June 2025. Public description published jointly by the four German TSOs under the EU Electricity Balancing Guideline (EB GL); covers FCR, aFRR, and mFRR roles, response times, and dispatch mechanisms in

the German balancing market. Implements BNetzA decisions BK6-15-158 and BK6-15-159.

[23] International Energy Agency, "Batteries and secure energy transitions," tech. rep., IEA, Paris, 2024.

[24] S. Lamp and M. Samano, "Large-scale battery storage, short-term market outcomes, and arbitrage," *Energy Economics*, vol. 107, p. 105786, 2022.

[25] R. L. Fares and M. E. Webber, "What are the tradeoffs between battery energy storage cycle life and calendar life in the energy arbitrage application?," *Journal of Energy Storage*, vol. 16, pp. 37–45, 2018.

[26] B. Xu, J. Zhao, T. Zheng, E. Litvinov, and D. S. Kirschen, "Factoring the cycle aging cost of batteries participating in electricity markets," *IEEE Transactions on Power Systems*, vol. 33, no. 2, pp. 2248–2259, 2018.

[27] N. Collath, B. Tepe, S. Englberger, A. Jossen, and H. Hesse, "Increasing the lifetime profitability of battery energy storage systems through aging aware operation," *Applied Energy*, vol. 348, p. 121531, 2023.

[28] B. Schade, T. Boomsma, and S. Pineda, "Battery degradation: Impact on economic dispatch," *Energy Storage*, vol. 6, no. 3, p. e588, 2024.

[29] R. Kiesel and F. Paraschiv, "Econometric analysis of 15-minute intraday electricity prices," *Energy Economics*, vol. 64, pp. 77–90, 2017.

[30] M. Narajewski and F. Ziel, "Ensemble forecasting for intraday electricity prices: Simulating trajectories," *Applied Energy*, vol. 279, p. 115801, 2020.

[31] T. Hendershott, C. M. Jones, and A. J. Menkveld, "Does algorithmic trading improve liquidity?," *The Journal of Finance*, vol. 66, no. 1, pp. 1–33, 2011.

[32] C. Kath, "Modeling intraday markets under the new advances of the cross-border intraday project (XBID): Evidence from the German intraday market," *Energies*, vol. 12, no. 22, p. 4339, 2019.

[33] C. Koch and P. Maskos, "Passive balancing through intraday trading: Whether interactions between short-term trading and balancing stabilize Germany's electricity system," *International Journal of Energy Economics and Policy*, vol. 10, no. 2, pp. 101–112, 2020.

[34] R. Sioshansi, "Welfare impacts of electricity storage and the implications of ownership structure," *The Energy Journal*, vol. 31, no. 2, pp. 173–198, 2010.

[35] V. Virasjoki, A. S. Siddiqui, B. Zakeri, and A. Salo, "Utility-scale energy storage in an imperfectly competitive power sector," *Energy Economics*, vol. 88, p. 104716, 2020.

[36] R. Sioshansi, "When energy storage reduces social welfare," *Energy Economics*, vol. 41, pp. 106–116, 2014.

[37] ACER, "Market monitoring report 2024 – electricity wholesale markets," tech. rep., Agency for the Cooperation of Energy Regulators, 2024.

[38] E. Hittinger and I. M. L. Azevedo, "Bulk energy storage increases United States electricity system emissions," *Environmental Science & Technology*, vol. 49, no. 5, pp. 3203–3210, 2015.

[39] M. T. Craig, P. Jaramillo, and B.-M. Hodge, "Carbon dioxide emissions effects of grid-scale electricity storage in a decarbonizing power system," *Environmental Research Letters*, vol. 13, no. 1, p. 014008, 2018.

[40] B. Tranberg, O. Corradi, B. Lajoie, T. Gibon, I. Staffell, and G. B. Andresen, "Real-time carbon accounting method for the european electricity markets," *Energy Strategy Reviews*, vol. 26, p. 100367, 2019.

[41] Energimyndigheten, "Energiläget 2024," tech. rep., Swedish Energy Agency (Energimyndigheten), Eskilstuna, 2024. Annual statistical compendium of the Swedish energy system, including power-sector CO_2 intensity.

[42] Agora Energiewende, "Die energiewende in deutschland: Stand der dinge 2024," tech. rep., Agora Energiewende, Berlin, 2025. Annual review of the German electricity transition; merit-order and hourly-emission profile.

[43] Umweltbundesamt, "Strommix in deutschland: Spezifische emissionsfaktoren des deutschen strommix," tech. rep., Umweltbundesamt (UBA), Dessau-Roßlau, 2024. Official emission-factor tables for German power generation.

[44] L. M. Arciniegas and E. Hittinger, "Tradeoffs between revenue and emissions in energy storage operation," *Energy*, vol. 143, pp. 1–11, 2018.

[45] K. Kannangara and J. Davis, "Optimal economic and environmental arbitrage of grid-scale batteries with a degradation-aware model," *Journal of Energy Storage*, 2024. ScienceDirect S2590174524000321.

[46] J. F. Peters, M. Baumann, B. Zimmermann, J. Braun, and M. Weil, "The environmental impact of Li-ion batteries and the role of key parameters – a review," *Renewable and Sustainable Energy Reviews*, vol. 67, pp. 491–506, 2017.

[47] International Energy Agency, "Global critical minerals outlook 2024," tech. rep., International Energy Agency, Paris, 2024.

[48] L. Kastner, T. Wagner, V. Bach, and M. Finkbeiner, "On the potential of vehicle-to-grid and second-life batteries to provide energy and material security," *Nature Communications*, vol. 15, p. 4179, 2024.

[49] M. Chordia, A. Nordelöf, and L. A.-W. Ellingsen, "Life cycle environmental impacts of current and future battery-grade lithium supply from brine and spodumene," *Resources, Conservation and Recycling*, vol. 174, p. 105725, 2021.

[50] H. Baumann and A.-M. Tillman, *The Hitch Hiker's Guide to LCA: An Orientation in Life Cycle Assessment Methodology and Application.* Lund, Sweden: Studentlitteratur, 2004. Canonical textbook for the Chalmers environmental-systems-analysis tradition; authors are at the Chalmers Department of Environmental Systems Analysis.

[51] D. Baur, M. J. Baumann, P. Stuhm, and M. Weil, "Societal acceptability of large stationary battery storage systems," *Energy Technology*, vol. 11, no. 7, p. 2201454, 2023.

[52] P. Emmerich, A.-G. Hülemeier, D. Jendryczko, M. J. Baumann, M. Weil, and D. Baur, "Public acceptance of emerging energy technologies in context of the German energy transition," *Energy, Sustainability and Society*, vol. 11, no. 1, p. 23, 2021.

[53] Electric Power Research Institute, "Community-based siting and permitting for grid-scale BESS," Tech. Rep. 3002031624, Electric Power Research Institute, 2024.

[54] E. Cognéville, T. Deschatre, and X. Warin, "Battery valuation on electricity intraday markets with liquidity costs." arXiv preprint arXiv:2412.15959, 2024.

[55] L. A. Wolsey, *Integer Programming*. Wiley, 2 ed., 2020.

[56] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, vol. 1. Athena Scientific, 4 ed., 2017.

[57] A. Quarteroni, R. Sacco, and F. Saleri, *Numerical Mathematics*. Springer, 2 ed., 2014.

[58] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18–20, 1967, Spring Joint Computer Conference (AFIPS '67)*, pp. 483–485, 1967.

[59] R. P. Brent, *Algorithms for Minimization Without Derivatives*. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[60] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 3 ed., 2007.

[61] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, 6 ed., 2017.

[62] G. Lai, M. X. Wang, S. Kekre, A. Scheller-Wolf, and N. Secomandi, "Valuation of storage at a liquefied natural gas terminal," *Operations Research*, vol. 59, no. 3, pp. 602–616, 2011.

[63] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley, 2 ed., 2011.

[64] R. Munos and A. Moore, "Variable resolution discretization in optimal control," *Machine Learning*, vol. 49, pp. 291–323, 2002.

[65] D. P. Bertsekas, *Dynamic Programming and Optimal Control: Approximate Dynamic Programming*, vol. 2. Athena Scientific, 4 ed., 2012.

[66] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 2 ed., 2006.

[67] M. Snellman and M. Evholt. Personal communication, February 2026. Data Scientists, Flower Infrastructure Technologies AB.

[68] Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, 2024.

[69] W. Jakob, J. Rhinelander, and D. Moldovan, *pybind11 – Seamless operability between C++11 and Python*, 2017. `https://github.com/pybind/pybind11`.

[70] Free Software Foundation, *Using the GNU Compiler Collection (GCC): Optimize Options*, 2024. GCC online documentation. `https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html`.

[71] Advanced Micro Devices, Inc., *Software Optimization Guide for AMD Family 19h Processors*, 2021. Revision 3.05. `https://developer.amd.com/resources/developer-guides-manuals/`.

[72] OpenMP Architecture Review Board, *OpenMP Application Programming Interface, Version 5.2*, 2021. `https://www.openmp.org/spec-html/5.2/openmp.html`.

[73] B. Chapman, G. Jost, and R. van der Pas, *Using OpenMP: Portable Shared Memory Parallel Programming*. Cambridge, MA: MIT Press, 2007.

[74] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[75] Intel Corporation, *Intel® 64 and IA-32 Architectures Optimization Reference Manual*, 2024. Order Number 248966-049.

[76] U. Drepper, "What Every Programmer Should Know About Memory," tech. rep., Red Hat, Inc., 2007. Originally published as a series of articles on LWN.net. `https://people.freebsd.org/~lstewart/articles/cpumemory.pdf`.

[77] Nord Pool, "Annual market statistics 2024." Nord Pool Group, Oslo, 2025. Day-ahead and intraday volumes and trade counts for the Nordic and Baltic power markets.

[78] R. Bruneel, M. Schuurmans, and P. Patrinos, "Optimal intraday power trading for single-price balancing markets: An adaptive risk-averse strategy using mixture models," *Applied Energy*, vol. 389, 2025.

[79] R. Almgren and N. Chriss, "Optimal execution of portfolio transactions," *Journal of Risk*, vol. 3, pp. 5–39, 2001.

[80] J. Chen, S. Lerch, M. Schienle, T. Serafin, and R. Weron, "Probabilistic intraday electricity price forecasting using generative machine learning," 2025.

[81] M. Merten, *Participation of Battery Storage Systems in the Automatic Frequency Restoration Reserve Market Based on Machine Learning*. PhD thesis, RWTH Aachen University, 2020.

[82] Modo Energy, "German ancillary service saturation: Lessons from great britain." https://modoenergy.com/research/october-2025-germany-afrr-saturation-bess-frequency-services, October 2025. Industry research note.

A

Supplementary figures and calculations

This appendix collects auxiliary illustrations and quantitative arguments that did not fit naturally into the main text but may aid the reader in building intuition for the methods described in the theory and methods chapters.

A.1 Forward-pass interpolation

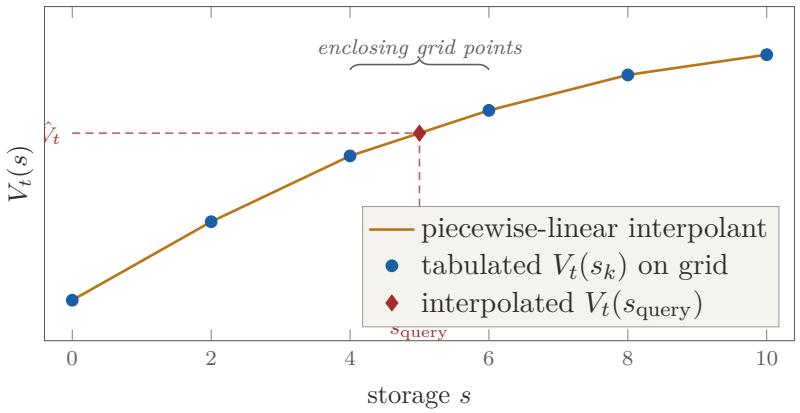


Figure A.1: Forward pass with linear interpolation in the dynamic-programming formulation of the rolling intrinsic. Tabulated grid points (blue) define V_t at discrete storage levels; off-grid queries (amber diamonds) are obtained by interpolating between the two enclosing rows.

A.2 Drift bound for linear-advance interpolation

This section substantiates the drift bound stated in the Linear-advance interpolation subsubsection of section 3.6. Two quantities are compared: an upper bound on the accumulated round-off drift in the running interpolation weight after $N = 200$ inner-loop iterations, and a lower bound on the smallest difference the running argmax of (2.15) can meaningfully distinguish. The first is the error introduced by the linear-advance scheme; the second is the precision floor below which no error can change a trading decision.

A.2.0.0.1 Accumulated drift in the interpolation weight. The optimized inner loop maintains the interpolation weight $w \in [0, 1)$ of (2.16) by a single floating-point addition per iteration, $w \mathrel{+}= \Delta/h$, where $\Delta = \eta^{+}\, u$ (or u/η^{-}) is the per-action stride in the post-transition storage state and $h = \bar{s}/(m-1)$ is the grid step. Each addition is rounded to the nearest IEEE 754 double, introducing a per-step error bounded by $\mathrm{ULP}(|w|) \le 2^{-52}$ since $|w| < 1$. Linear accumulation across the at most $N = 200$ iterations of a single inner loop therefore bounds the drift in w above by

$$\Delta w_{\max} \;\le\; N \cdot 2^{-52} \;\approx\; 4 \times 10^{-14}. \tag{A.1}$$

Propagated through the linear interpolation (2.16), the resulting drift in the interpolated value $\tilde{V}_{t+1}$ is bounded by

$$|\Delta \tilde{V}_{t+1}| \;\le\; \Delta w_{\max} \cdot |V_{t+1}(s_{i+1}) - V_{t+1}(s_i)|. \tag{A.2}$$

For our setting, the per-grid-step difference $|V_{t+1}(s_{i+1}) - V_{t+1}(s_i)|$ is approximately the marginal value of one grid step of storage, of order $h \cdot p \approx 0.025 \cdot 100 \approx 2.5$ EUR at typical electricity prices, and bounded above by a value a few times larger. Substituting into (A.2) gives

$$|\Delta \tilde{V}_{t+1}| \;\lesssim\; 4 \times 10^{-14} \cdot \mathcal{O}(10)\ \text{EUR} \;\approx\; 10^{-13}\ \text{EUR}. \tag{A.3}$$

A.2.0.0.2 Argmax comparison floor. The argmax in (2.15) compares $\pi_t(k_t u) + \tilde{V}_{t+1}(S(s, k_t u))$ between adjacent actions k_t and $k_t + 1$. The reward π_t is computed by sweeping the order book at integer-tick prices: Nord Pool quotes prices at a minimum tick of 0.01 EUR/MWh and quantities at a minimum of 0.1 MW (subsection 3.1.2, Step 1), so the smallest possible difference between two adjacent actions' rewards is

$$\Delta\pi_{\min} \;=\; u \cdot (\text{price tick}) \;=\; 0.025\ \text{MWh} \cdot 0.01\ \text{EUR/MWh} \;=\; 2.5 \times 10^{-4}\ \text{EUR}. \tag{A.4}$$

This is the smallest difference that two argmax candidates can have and still be ordered correctly; any disturbance smaller than $\Delta\pi_{\min}$ is sub-tick noise that cannot change which action wins the maximum.

A.2.0.0.3 Conclusion. Combining (A.3) and (A.4), the worst-case drift introduced by the linear-advance scheme is approximately

$$\frac{|\Delta \tilde{V}_{t+1}|}{\Delta\pi_{\min}} \;\approx\; \frac{10^{-13}}{2.5 \times 10^{-4}} \;\approx\; 4 \times 10^{-10},$$

i.e. nine orders of magnitude smaller than the smallest difference the argmax comparison can resolve. The drift is therefore guaranteed never to change the optimal action selected at any inner-loop iteration, which justifies the bit-identical verification reported in section 3.6.

A.3 Interpolations per stage in the backward pass

This section derives the figure of ≈ 70,400 interpolations per stage quoted in the V-difference precompute subsubsection of section 3.6. The quantity is the total number of inner-loop iterations performed within one stage of the backward pass and is the base over which the $\mathcal{O}(m)$ precompute amortizes.

For one stage of the backward pass, the outer loop walks all m storage grid points and the inner loop walks the feasible actions $k_t \in \mathcal{X}_t(s, f_t^0)$ at each grid point. The total number of inner-loop iterations per stage is therefore

$$N_{\text{stage}} \;=\; \sum_{i=0}^{m-1} |\mathcal{X}_t(s_i, f_t^0)| \;=\; m \cdot \overline{N_{\text{feas}}}, \tag{A.5}$$

where $\overline{N_{\text{feas}}}$ is the average feasible-action count per grid point. With the default battery of section 3.6 ($m = 401$, action step $u = 0.025$ MWh), the maximum action range is $(\bar{f} - \underline{f})/(\kappa u) = 5/0.025 = 200$ actions per grid point; the state–action bound tightening of section 3.6 reduces this only at grid points near the storage envelope.

The average $\overline{N_{\text{feas}}}$ was estimated empirically by instrumenting the $v4$ kernel during the OpenMP-scaling investigation and counting the inner-loop entries summed over one full backward pass. The resulting figure is $\overline{N_{\text{feas}}} \approx 176$, i.e. approximately 88% of the theoretical maximum of 200. Substituting into (A.5) gives

$$N_{\text{stage}} \;\approx\; 401 \cdot 176 \;=\; 70{,}576 \;\approx\; 70{,}400. \tag{A.6}$$

Three independent decompositions of the same per-stage workload corroborate this figure. With 8 OpenMP threads, the work per thread per stage is $(m/8) \cdot \overline{N_{\text{feas}}} \approx 50 \cdot 176 = 8{,}800$ evaluations, so the per-stage total across threads is $8 \cdot 8{,}800 = 70{,}400$. With 16 threads, the per-thread work drops to $25 \cdot 176 = 4{,}400$ evaluations and the per-stage total reconstructs as $16 \cdot 4{,}400 = 70{,}400$. All three decompositions agree to within the rounding of the empirical $\overline{N_{\text{feas}}}$, which is the source of the small discrepancy between 70,576 and the rounded headline figure of 70,400.

A.4 Empirical findings shaping the OpenMP configuration

Two empirical findings shaped the OpenMP configuration described in section 3.6. First, parallel efficiency degrades catastrophically once the per-thread workload drops below approximately fifty grid points: at sixteen threads the runtime collapses by a factor of several hundred relative to eight threads, because the per-stage barrier cost overtakes the per-thread arithmetic. Second, while eight threads are slightly faster than six on an isolated microbenchmark, six threads paired with a six-core `taskset` pin maximize stability on the shared lab machine. Amdahl decomposition of the thread-scaling curve estimates a serial fraction of approximately nine percent, of which a libgomp microbenchmark attributes only fifteen percent to

the per-stage fork–join cost; the remaining non-scaling portion is dominated by L3 load latency on the value-table gather, which is not removable without algorithmic restructuring.

A.5 Stage-reuse cache hit rate

This section documents the empirical basis for the $\approx 7\%$ recomputation figure quoted for the stage-reuse cache in section 3.6. The relevant quantity is the typical maximum dirty stage index $t^\star$ across a representative workload: stages $[0, t^\star]$ are recomputed on each solve and stages $(t^\star, T-1]$ are reused from cache, so the fraction of the horizon recomputed is $(t^\star + 1)/T$.

For the standard SE3 workload the horizon contains $T = 92$ active contracts per solve. Instrumentation of the v6 kernel during the introduction of the cache showed that the typical market event dirties one or two near-term contracts, placing the maximum dirty stage index at $t^\star \approx 5$. The resulting fraction recomputed is

$$\frac{t^\star + 1}{T} \;\approx\; \frac{6}{92} \;\approx\; 6.5\%, \tag{A.7}$$

so roughly 93.5% of the value-table work is served from cache on the average solve. The savings are reported in the project's internal v6 session summary as a $\sim 14\times$ speedup per event, with full-rebuild solves (triggered by changes in the active contract list rather than by ordinary market events) occurring at a substantially lower rate and not included in this average.

B

Dataset metrics

This appendix collects descriptive statistics of the three intraday order-book datasets used in this thesis. Section B.1 reports raw-row and decision-step counts; Section B.2 reports the size distribution of the best-quoted bid and ask, which motivates the choice in subsection 3.1.2 to key the relevance timeline on a 1 MW VWAP rather than on the best quoted price.

B.1 Raw rows and decision-step counts

Table B.1 summarizes key characteristics of the three market datasets used in this thesis. Raw revision rows are the total number of rows across all contract files before any pre-processing. The remaining rows report the number of decision steps produced by the precomputing pipeline: the event-driven timeline retains one step for every revision at which the 1-MW VWAP moved by at least 0.10 EUR/MWh on at least one active contract; the fixed timelines sub-sample those steps to uniform spacings of 6 seconds, 1 minute, 5 minutes, 15 minutes, and 60 minutes.

Metric	SE3	SE4	Germany
Raw revision rows	4,073,175,734	4,397,498,558	7,783,974,656
Event-driven steps	88,500,668	109,401,059	265,059,645
Fixed 6 s steps	4,854,772	4,918,977	4,138,680
Fixed 1 min steps	499,638	500,166	414,290
Fixed 5 min steps	100,282	100,279	82,888
Fixed 15 min steps	33,480	33,469	27,655
Fixed 60 min steps	8,397	8,394	6,931

Table B.1: Key metrics for the three intraday order-book datasets. Raw revision rows are counted before pre-processing. Decision steps are the number of solver invocations per full dataset under each timeline variant.

B.2 Top-of-book size distribution

This section quantifies the size distribution of best-bid and best-ask quotes in the SE3 order-book data, motivating the choice in subsection 3.1.2 to define the rele-

vance timeline on a 1 MW volume-weighted average price (VWAP) rather than on the best quoted price.

A sample of sixty contracts was drawn at uniform spacing across the SE3 dataset. For each contract, the bid and ask books were reconstructed from the start of the contract by replaying every revision in chronological order; after each revision the size at the best price on each side was recorded as the sum of resting quantities of all orders sharing that best price. The procedure produced approximately 3.2×10^6 top-of-book observations per side. Table B.2 reports the resulting size distribution.

Top-of-book size	Best bid (%)	Best ask (%)
$\leq$ 0.1 MW (exchange minimum)	10.76	11.37
$\leq$ 0.2 MW	16.63	17.32
$\leq$ 0.3 MW	20.26	19.90
$\leq$ 0.5 MW	25.45	26.11
$\leq$ 1.0 MW	34.36	34.60
$\leq$ 2.0 MW	45.22	44.74
$\leq$ 5.0 MW	76.41	76.76
$\leq$ 10.0 MW	90.36	93.50
$>$ 10.0 MW	9.65	6.50

Table B.2: Cumulative distribution of best-bid and best-ask sizes in the SE3 dataset, computed from sixty sampled contracts and approximately 3.2×10^6 top-of-book observations per side.

Roughly one in ten top-of-book quotes carries the 0.1 MW exchange-minimum size, and approximately one in three quotes on either side is at or below 1 MW. A non-trivial share of top-of-book quotes therefore reflects a single small order rather than a representative depth of liquidity, and a strategy keyed to the best quoted price would react to price moves driven by very small orders that a 1 MW trade could not realistically clear at that price. The 1 MW VWAP is the smallest target volume that consistently averages across more than the very top of the book, and is used as the reference quantity for the relevance threshold in subsection 3.1.2.

C

Further results

This appendix collects per-market operational-metric tables that supplement the headline profit and trade-rate figures in chapter 4. For each zone we report total solves, total trades, FOK kill counts and rates, partial-fill counts and partial-fill share of kills, the share of negative-objective solves, and the number of negative-PnL days. All figures are aggregated over the full evaluation window of the corresponding section in chapter 4.

C.1 Sweden SE3

Table C.1: Operational metrics for the SE3 portfolio by decision-timeline resolution (Apr 2025 – Mar 2026, 10 MWh / 1 h battery). Kill rate is FOK kills as a share of executed solves; partial-fill share is the share of FOK kills that were partial rather than full rejections.

Metric	Event	6 s	1 min	5 min	15 min	1 h
Total solves (executed)	88,500,668	4,854,772	499,638	100,282	33,480	8,397
Total trades	1,359,756	799,031	353,351	168,903	94,949	44,101
Avg. trades / h	163.3	96.0	42.4	20.3	11.4	5.3
Avg. cycles / day	6.28	5.09	4.63	4.21	3.80	1.96
Total cycles (yr)	2,181	1,766	1,608	1,462	1,319	681
FOK kills	20,495	3,037	824	332	215	144
Kill rate (%)	0.02	0.06	0.16	0.33	0.64	1.71
Partial fills	2,733	723	307	153	115	100
Partial-fill share of kills (%)	13.3	23.8	37.3	46.1	53.5	69.4
Neg. objective solves (%)	0.39	3.83	12.33	18.89	19.66	15.35
Negative-PnL days	1/347	1/347	1/347	3/347	3/347	2/347
Max negative streak (days)	1	1	1	1	1	1

C.2 Sweden SE4

Table C.2: Operational metrics for the SE4 portfolio by decision-timeline resolution (Apr 2025 – Mar 2026, 10 MWh / 1 h battery). Column definitions follow Table C.1.

Metric	Event	6 s	1 min	5 min	15 min	1 h
Total solves (executed)	109,401,059	4,918,977	500,166	100,279	33,469	8,394
Total trades	1,568,531	924,189	427,231	205,494	116,565	54,669
Avg. trades / h	188.3	111.0	51.3	24.7	14.0	6.6
FOK kills	24,412	3,787	1,105	433	279	182
Kill rate (%)	0.02	0.08	0.22	0.43	0.83	2.17
Partial fills	2,952	800	342	181	137	120
Partial-fill share of kills (%)	12.1	21.1	31.0	41.8	49.1	65.9
Neg. objective solves (%)	0.35	4.00	13.39	20.02	20.91	15.87
Negative-PnL days	1/347	0/347	2/347	0/347	0/347	0/347
Max negative streak (days)	1	0	1	0	0	0

C.3 Germany Amprion

Table C.3: Operational metrics for the Germany Amprion portfolio by decision-timeline resolution. Aggregated over the 14-month window Jan 2025 – Feb 2026 (420 trading days, 10 MWh / 1 h battery). Column definitions follow Table C.1.

Metric	Event	6 s	1 min	5 min	15 min	1 h
Total solves (executed)	265,059,645	6,063,618	606,994	121,446	40,519	10,158
Total trades	2,556,492	1,127,912	472,493	229,635	123,911	55,915
Avg. trades / h	253.6	111.9	46.9	22.8	12.3	5.5
FOK kills	90,764	13,269	3,340	1,634	873	303
Kill rate (%)	0.03	0.22	0.55	1.35	2.15	2.98
Partial fills	8,231	2,865	1,156	791	498	183
Partial-fill share of kills (%)	9.1	21.6	34.6	48.4	57.0	60.4
Neg. objective solves (%)	0.24	3.67	10.71	16.75	18.81	17.45
Negative-PnL days	1/420	0/420	0/420	1/420	1/420	0/420
Max negative streak (days)	1	0	0	1	1	0

D

Ex-post analysis of intraday profit drivers

Beyond the headline backtests, a substantial exploratory effort was made to explain *when* the event-driven intraday DP is profitable and when it beats the slower benchmarks — both the ancillary capacity markets of Section 4.4.1 and a day-ahead-forecast bidder into the intraday auction. The aim was a predictive signal that a production scheduler could use to decide, ahead of delivery, whether to commit the battery to intraday trading. This appendix records that effort and its two stable findings. The analysis is restricted to Germany Amprion, for which hourly system-level forecast data from SMARD is available.

The results in this section come from a much earlier and simpler stage of the project, and must be read with that in mind. The intraday P&L plotted here was produced by an early dynamic-programming solver — a coarse 51-bin state-of-charge grid — run at a *five-minute* decision resolution on a *reduced* order-book dataset that retained only the five best bid and five best ask levels at each step. This is far simpler than the final event-driven C++ DP of Chapter 4, which runs at full event resolution on the complete order book. The figures below are therefore *qualitative* evidence of a relationship only: the absolute euro values are not comparable to the headline results, and the analysis was never repeated on the final algorithm and dataset.

D.1 Forecast revisions drive the intraday edge

Two relationships survived scrutiny, and both point in the same direction. Using the SMARD D-1 forecast series for German solar generation and total load, a forecast *revision* is defined as the lag-one change in the forecast ordered by delivery hour, $\text{revision}(T) = \text{forecast}(T) - \text{forecast}(T-1)$. Figure D.1 plots the hourly intraday DP P&L against the solar revision and Figure D.2 against the total-load revision. In both, the large DP P&L values concentrate on hours where the forecast was revised *downward* (negative revision) and collapse toward zero as the revision approaches zero, staying near zero for upward revisions.

The interpretation is consistent with the spread mechanism discussed in the main text. A high-frequency intraday policy earns its advantage precisely when new

information arrives between forecast vintages — a downward solar revision or a load-forecast surprise — because those revisions move intraday prices and widen inter-contract spreads after the day-ahead auction has cleared. The same concentration appears in the DP's edge over a day-ahead-forecast auction bidder (the DP-minus-auction gap, not shown here). When forecasts are stable there is little intraday opportunity left to capture and the DP's P&L collapses toward that of the slower benchmark. This is suggestive rather than predictive: the relationship is clear in aggregate but, as the broader modeling effort of Section D.2 found, too noisy at the daily level to drive an ahead-of-delivery commitment rule on its own.

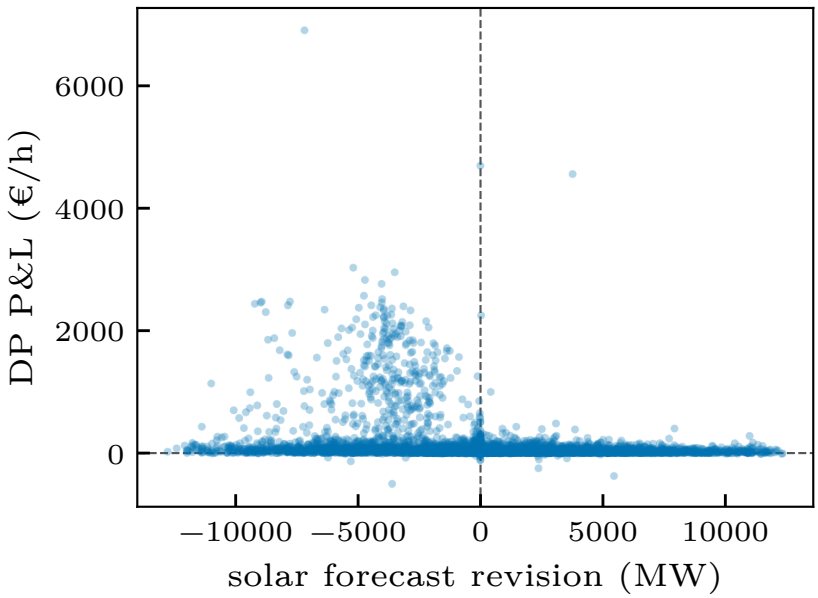


Figure D.1: Germany Amprion hourly intraday DP P&L (€/h) against the solar generation forecast revision, revision(T) = forecast(T) − forecast$(T-1)$, the lag-one change in the D-1 forecast ordered by delivery hour (calendar 2025, 8,446 hours). Profit concentrates on hours with a downward (negative) revision and collapses toward zero when the forecast is stable. *Early prototype: a 51-bin DP at five-minute resolution on a five-level order book, not the final event-driven backtest; the relationship is qualitative.* Data recovered from the original `forecast_analysis` dataset.

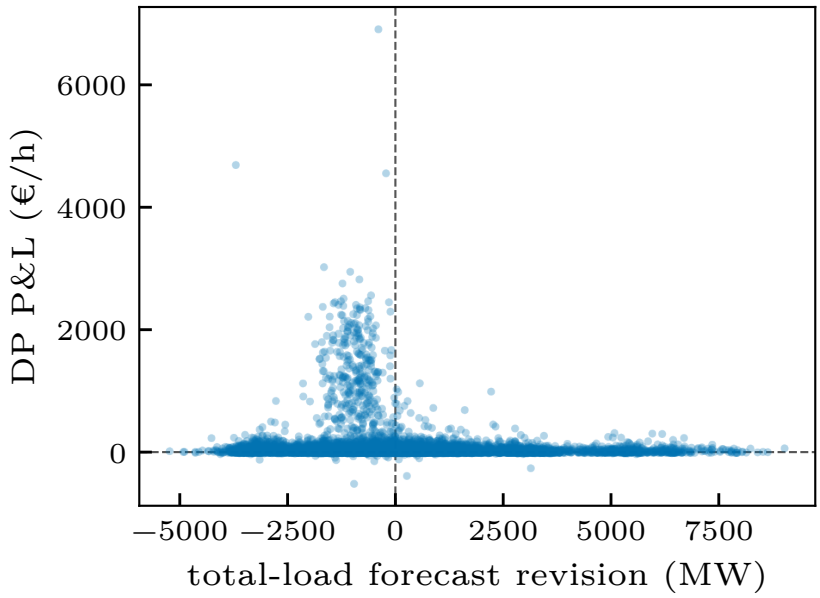


Figure D.2: As Figure D.1, but against the total-load forecast revision. The DP's profit is again largest on hours with the largest (downward) load-forecast revisions and shrinks to near zero when the load forecast is stable. *The same early-prototype caveat applies: a* 51*-bin DP at five-minute resolution on a five-level order book, qualitative only.*

D.2 Methods that did not yield a clear signal

Three families of explanatory analysis were tried. First, each day was labeled by market regime along five axes — a fuel-price regime, a balancing-stress regime, a congestion regime, a system-tightness regime, and a cross-border export regime — and intraday profitability was compared across regimes. Second, an unsupervised clustering of the joint daily regime features was used to look for profit-homogeneous clusters. Third, supervised ex-post models (random forest and gradient-boosted trees) were trained to predict two targets: daily DP profitability, and the daily gap between the DP and an intraday-auction forecast bidder. None of these produced a stable, low-variance predictor: cross-validated skill was weak and feature pruning discarded most candidate predictors as noise. We do not claim a usable profitability classifier; the practical recommendation of Section 5.9.1 — a day-by-day market-selection layer — rests on the comparative results of Section 4.4.1, not on a learned predictor.

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY