Appendices

Appendix 3.1. List of symbols, indices and units used in the model formulation.

Table 3.1a. Sets and indices.

Symbol	Meaning	Unit	Used in
n	Node in the physical network.	Dimensionless	Nodal benchmark, redispatch, nodal demand and inflow allocation
i, j	Endpoint nodes of a transmission line.	Dimensionless	DC power flow equation
z	Norwegian bidding zone.	Dimensionless	Zonal market clearing, aggregation and scenario comparison
t	Hourly time step.	Hour	All time-dependent equations
m	Month.	Month	Demand scaling and monthly demand allocation
w	Rolling-horizon window or week, depending on the equation context.	Dimensionless / week	Rolling horizon and hydrological inflow processing
d	Implemented day.	Day	Redispatch storage-target tracking
ℓ	Transmission line.	Dimensionless	DC power flow and line-capacity constraints
k	Reservoir turbine segment.	Dimensionless	Segmented reservoir generation
q	Terminal-value storage segment.	Dimensionless	Terminal water-value approximation
s	Demand sector or hydrological scenario, depending on context.	Dimensionless	Demand construction and scenario design
N_z	Set of nodes located in bidding zone z.	Dimensionless	Spatial aggregation and allocation equations
T_m	Set of hours belonging to month m.	Hour set	Hourly demand-profile normalisation
$A_{n,\ell}$	Network incidence coefficient linking node n and transmission line ℓ.	Dimensionless	Nodal power-balance equation

Description: This table defines the main sets and indices used throughout the mathematical formulation of the model. These indices identify the spatial, temporal and technical dimensions of the optimization problem, including nodes, bidding zones, hours, months, rolling-horizon windows, transmission lines, turbine segments and terminal-value segments. They provide the common notation used in hydropower, demand, transmission and redispatch equations.

Source: Own compilation based on the model formulation developed in Chapter 3 and implemented in the Python/PyPSA modelling framework.

Table 3.1b. Hydropower and storage notation.

Symbol	Meaning	Unit	Used in
$S_{n,t}$	Reservoir storage at node n and hour t.	MWh-equivalent; aggregated results may be GWh or TWh	Reservoir water balance
S^{max}_{n}	Maximum storage capacity at node n.	MWh-equivalent; aggregated results may be GWh or TWh	Storage bounds
$I^{res}_{n,t}$	Reservoir inflow allocated to node n at hour t.	MWh-equivalent per hour	Reservoir balance
$I^{ror}_{n,t}$	Run-of-river inflow allocated to node n at hour t.	MWh-equivalent per hour	Run-of-river balance
$I_{z,t}$	Total hydrological inflow in bidding zone z at hour t.	MWh-equivalent per hour	Inflow processing
$I_{n,t}$	Total hydrological inflow allocated to node n at hour t.	MWh-equivalent per hour	Inflow allocation
$G^{res}_{n,t}$	Reservoir hydropower generation at node n and hour t.	MW, interpreted as MWh over one hour in balances	Reservoir generation and power balance
$G^{ror}_{n,t}$	Run-of-river generation at node n and hour t.	MW, interpreted as MWh over one hour in balances	Run-of-river generation and power balance
$g^{res}_{n,k,t}$	Generation from reservoir turbine segment k at node n and hour t.	MW	Segmented turbine offers
$G^{max}_{n,k}$	Maximum generation capacity of turbine segment k at node n.	MW	Segment capacity constraint
Cap^{res}_{n}	Installed reservoir hydropower capacity at node n.	MW	Inflow split between reservoir and run-of-river components
Cap^{ror}_{n}	Installed run-of-river hydropower capacity at node n.	MW	Inflow split between reservoir and run-of-river components
$Share^{hydro}_{n}$	Share of zonal hydropower capacity assigned to node n.	Dimensionless	Allocation of zonal inflows to nodes

Symbol	Meaning	Unit	Used in
$Spill^{res}_{n,t}$	Reservoir spill at node n and hour t.	MWh-equivalent per hour	Reservoir water balance and spill analysis
$Spill^{ror}_{n,t}$	Run-of-river spill at node n and hour t.	MWh-equivalent per hour	Run-of-river balance and spill analysis

Description: This table reports the main variables used to represent reservoir and run-of-river hydropower in the model. It includes storage levels, inflows, generation, spill, turbine-segment dispatch and capacity limits. The variables are expressed in energy-equivalent terms where they refer to water availability or stored water, reflecting the fact that the model represents hydropower operation within an electricity-system optimization framework rather than as a physical hydrological routing model.

Source: Own compilation based on the model formulation developed in Chapter 3 and implemented in the Python/PyPSA modelling framework.

Table 3.1c. Demand-construction notation.

Symbol	Meaning	Unit	Used in
α_m	Hydro-compatible scaling factor for month m.	Dimensionless	Demand scaling for the hydro-only model
$HydroProduction_m$	Hydropower production in month m.	MWh	Hydro-compatible demand scaling
$TotalElectricityProduction_m$	Total electricity production in month m.	MWh	Hydro-compatible demand scaling
$D^{observed}_{z,m,s}$	Observed electricity consumption in bidding zone z, month m and sector s.	MWh	SSB-based demand construction
$D^{hydro}_{z,m,s}$	Hydro-compatible demand after scaling.	MWh	Demand scaling
$D^{res}_{n,m}$	Monthly residential demand is allocated to node n.	MWh	Residential demand allocation
$D^{ind}_{n,m}$	Monthly industrial demand is allocated to node n.	MWh	Industrial demand allocation
$D_{n,t}$	Total hourly demand at node n and hour t.	MW, interpreted as MWh over one hour in balances	Nodal power balance
w^{res}_t	Residential hourly load-shape coefficient.	Dimensionless	Hourly demand disaggregation
w^{ind}_t	Industrial hourly load-shape coefficient.	Dimensionless	Hourly demand disaggregation
Pop_n	Population assigned to node n.	Number of inhabitants	Residential demand allocation
Ind_n	Industrial activity indicator assigned to node n.	Number or index of enterprises	Industrial demand allocation

Description: This table summarizes the notation used for electricity-demand construction. It includes observed zonal demand, hydro-compatible scaled demand, residential and industrial nodal demand, hourly nodal demand, load-shape coefficients, and the socio-economic allocation indicators used to distribute demand across nodes. The table clarifies how monthly zonal electricity-consumption data are transformed into hourly nodal demand profiles while preserving aggregate monthly energy totals.

Source: Own compilation based on the model formulation developed in Chapter 3 and implemented in the Python/PyPSA modelling framework.

Table 3.1d. Transmission, load shedding and price notation.

Symbol	Meaning	Unit	Used in
$F_{\ell,t}$	Active power flow on transmission line ell at hour t.	MW	DC power-flow equation
F^{max}_{ℓ}	Thermal transfer capacity of line ell.	MW	Transmission constraint
b_{ℓ}	Susceptance of transmission line ell.	MW/radian or per-unit equivalent, depending on implementation	DC power-flow equation
$\theta_{i,t}$	Voltage angle at node i and hour t.	Radians	DC power-flow equation
$\theta_{j,t}$	Voltage angle at node j and hour t.	Radians	DC power-flow equation
$LS_{n,t}$	Load shedding at node n and hour t.	MW, interpreted as MWh over one hour	Power balance and adequacy-stress analysis
$p_{n,t}$	Nodal electricity price at node n and hour t.	EUR/MWh	Nodal price formation
$p_{z,t}$	Zonal electricity price in bidding zone z and hour t.	EUR/MWh	Zonal market clearing
$\lambda_{n,t}$	Dual variable of the nodal power-balance constraint.	EUR/MWh	Price formation
$\lambda_{z,t}$	Dual variable of the zonal power-balance constraint.	EUR/MWh	Zonal price formation

Description: This table defines the variables and parameters used in the transmission-network and price-formation equations. It includes active power flows, line-capacity limits, line susceptance, voltage angles, incidence coefficients, nodal prices, zonal prices and dual variables. These elements are central to the DC-OPF formulation, as they determine how electricity flows through the network, when congestion becomes binding, and how locational or zonal prices are formed.

Source: Own compilation based on the model formulation developed in Chapter 3 and implemented in the Python/PyPSA modelling framework.

Table 3.1e. Redispatch notation.

Symbol	Meaning	Unit	Used in
$G^{phys}_{n,t}$	Realised physical generation at node n and hour t after redispatch.	MW	Nodal redispatch stage
$G^{target}_{n,t}$	Nodalised reference schedule derived from the zonal market stage.	MW	Redispatch adjustment equation
$RD^{+}_{n,t}$	Upward redispatch at node n and hour t.	MW	Redispatch correction
$RD^{-}_{n,t}$	Downward redispatch at node n and hour t.	MW	Redispatch correction
$S^{phys}_{z,d,end}$	Realised end-of-day storage in zone z after redispatch.	MWh-equivalent; aggregated results may be GWh or TWh	Storage-target tracking
$S^{target}_{z,d,end}$	End-of-day storage target from the zonal market stage.	MWh-equivalent; aggregated results may be GWh or TWh	Storage-target tracking
$Dev^{+}_{z,d}$	Positive deviation between realised redispatch storage and zonal storage target.	MWh-equivalent	Storage-deviation penalty
$Dev^{-}_{z,d}$	Negative deviation between realized redispatch storage and zonal storage target.	MWh-equivalent	Storage-deviation penalty
$C^{redispatch}_{t}$	Cost or penalty associated with redispatch adjustments at time t.	EUR	Redispatch objective component

Description: This table presents the notation used for the nodal redispatch stage of zonal market design. It defines physical generation after redispatch, nodalized target schedules derived from zonal market clearing, upward and downward redispatch variables, and deviations between zonal storage targets and realized redispatch storage levels. These variables are used to measure the corrections required to transform the zonal market schedule into a physically feasible nodal dispatch.

Source: Own compilation based on the model formulation developed in Chapter 3 and implemented in the Python/PyPSA modelling framework.

Table 3.1f. Cost, calibration and terminal-value notation.

Symbol	Meaning	Unit	Used in
C^{res}_{t}	Cost associated with reservoir hydropower generation at time t.	EUR	General optimization objective
C^{ror}_{t}	Cost associated with run-of-river generation at time t.	EUR	General optimization objective
C^{spill}_{t}	Penalty or cost associated with spilled water at time t.	EUR	General optimization objective
C^{shed}_{t}	High penalty associated with load shedding at time t.	EUR	General optimization objective

Symbol	Meaning	Unit	Used in
$V^{terminal}_{t}$	Terminal value of water remaining in storage at the end of the optimization window.	EUR	General optimization objective
$V^{terminal}_{n,w}$	Terminal value assigned to storage at node n in rolling-horizon window w.	EUR	Terminal water-value approximation
$v_{n,q,w}$	Marginal terminal value of storage segment q at node n in window w.	EUR/MWh	Piecewise-linear terminal value
$S^{terminal}_{n,q,w}$	Storage assigned to terminal-value segment q at node n in window w.	MWh-equivalent	Piecewise-linear terminal value
$Shortfall_{n}$	Final storage shortfall variable at node n.	MWh-equivalent	Soft final storage requirement

Description: This table reports the main cost, calibration and end-of-horizon variables used in the optimization framework. It includes generation-cost components, spill and load-shedding penalties, terminal water values, marginal terminal-value coefficients, storage assigned to terminal-value segments, and final-storage shortfall variables. These parameters help ensure that reservoir use remains consistent with the rolling-horizon formulation and that end-of-window storage is not undervalued.

Source: Own compilation based on the model formulation developed in Chapter 3 and implemented in the Python/PyPSA modelling framework.

Appendix 3.2.1. Detailed data sources used in the modelling framework.

Table 3.2.1a. Summary of data sources, variables and preprocessing steps used in the modelling framework.

Input category	**Exact source**	**Source link**	**Reference year / coverage**	**Spatial resolution**	**Temporal resolution**	**Variable(s) used**	**Transformation applied**	**Linked appendix figure/table**
Bidding zones	Statnett, "Why we have bidding zones"	Statnett source page	Official market structure used as the institutional reference for 2023/2025	NO1–NO5 bidding zones	Static	Norwegian bidding-zone structure and market-area boundaries	Approximate bidding-zone polygons were manually constructed in GeoJSON format from publicly available Statnett maps and used as the common spatial framework.	Appendix 3.3. Norwegian bidding zones and model spatial representation
Electrical nodes	Satheeskumar (2020), "Open source model of the Nordic power system for EU project Spine"; Nordic 490 model	DiVA/KTH thesis record	Static open-source Nordic network representation	Electrical bus / node level	Static	Bus identifier, voltage level, geographical coordinates	Duplicate or near-duplicate buses were identified and consolidated. The remaining nodes were assigned to bidding zones through a point-in-polygon procedure.	Appendix 3.4. Electrical nodes used in the modelling framework
Transmission lines	Satheeskumar (2020), Nordic 490 transmission-line dataset	DiVA/KTH thesis record	Static open-source Nordic network representation	Line level, connecting electrical nodes	Static	Line endpoints, voltage class, reactance, circuit information where available	Line endpoints were matched to the cleaned nodal set. Disconnected fragments were reconnected to the main network through nearest-node links. Voltage classes were converted into stylised thermal capacities.	Appendix 3.8. Transmission network representation

Input category	Exact source	Source link	Reference year / coverage	Spatial resolution	Temporal resolution	Variable(s) used	Transformation applied	Linked appendix figure/table
Hydropower assets	Energy-Modelling-Toolkit / JRC Hydro-power database	JRC hydro-power database	Static plant-level database; extraction used for the thesis model	Plant level, later aggregated to modelling nodes	Static	Plant location, installed capacity, technology classification, reservoir/run-of-river information where available, storage-related fields where available	Plants below 5 MW were excluded. Remaining plants were assigned to the nearest electrical node. Reservoir and run-of-river capacities were then aggregated at nodal level.	Appendix 3.5. Hydropower plants and capacity assignment
Electricity consumption	Statistics Norway / SSB StatBank Table 14092, "Electricity consumption by consumer group and price area (MWh)"	SSB StatBank Table 14092	2023 monthly values used as empirical reference	Bidding zone and consumer group	Monthly	Electricity consumption by price area and consumer group, especially residential and industrial demand	Monthly zonal demand was separated into residential and industrial components before being spatially allocated to nodes and temporally disaggregated to hourly profiles.	Appendix Table 3.2.2b — Observed electricity-consumption inputs; Appendix 3.6 — spatial allocation maps; Appendix 3.7 — hourly load-shape templates
Electricity balance / hydro scaling factor	Statistics Norway / SSB StatBank Table 14091, "Electricity balance (MWh)"	SSB StatBank Table 14091	2023 monthly or annual values, depending on the extraction used for calibration	National level	Monthly / annual aggregation	Total electricity production and hydropower production	A hydro-compatibility scaling factor was computed as the ratio between hydropower production and total electricity production, in order to align demand with the hydro-only supply representation.	Appendix Table 3.2.2a — Monthly hydro-compatible scaling factor
Hydrological inflows	Norwegian Water Resources and Energy Directorate / NVE, "Hydrologiske data til kraftsituasjonsrapporten"	NVE hydrological data page	2023 baseline inflows; sensitivity scenarios derived from this base	Bidding zone, then allocated to nodes	Weekly, converted to hourly	Usable inflow ("nyttbart tilsig"), expressed in GWh/week	Weekly zonal inflows were converted into hourly inflow series. They were then allocated to nodes and split between reservoir and run-of-river components according to installed hydropower capacity shares.	Discussed in Section 3.4.3
Population grid	Statistics Norway / Geonorge, "Befolkning på rutenett 1000 m 2023"	Data.norge population dataset	2023	1 km × 1 km grid	Static / annual	Resident population per grid cell	Grid-cell centroids were assigned to the nearest modelling node within the bidding-zone framework. Population	Appendix Table 3.2.2d; Appendix 3.6

Input category	Exact source	Source link	Reference year / coverage	Spatial resolution	Temporal resolution	Variable(s) used	Transformation applied	Linked appendix figure/table
							values were aggregated at nodal level and used as allocation weights for residential demand.	
Enterprise / industrial activity grid	Statistics Norway / Geonorge, "Bedrifter på rutenett"	Data.norge enterprise dataset	2023 extraction from the annual grid dataset	1 km × 1 km grid	Static / annual	Number or density of enterprises / industrial establishments per grid cell	Grid-cell centroids were assigned to the nearest modelling node. Enterprise indicators were aggregated at nodal level and used as allocation weights for industrial demand.	Appendix Table 3.2.2d; Appendix 3.6
Hourly load-shape templates	Model-specific stylised sectoral load profiles constructed for the thesis	No external source	Model-specific	Sector level: residential and industrial	Hourly, with weekday/weekend distinction	Relative hourly load coefficients	Profiles were normalised by month so that hourly values preserve the original monthly energy totals for each sector and node.	Appendix 3.7 — Hourly load-shape templates
Hydrological sensitivity scenarios	Derived from NVE 2023 usable inflow data	NVE hydrological data page	Base 2023; -10%, -15%, +10%, +15% sensitivity cases	Bidding zone, then node level	Weekly, converted to hourly	Scenario-specific inflow values	Baseline inflow series were uniformly scaled to construct dry and wet sensitivity scenarios.	Discussed in Sections 3.1 and 3.4.3
Rolling-horizon robustness outputs	Model-generated diagnostic outputs from alternative rolling-horizon runs	No external source	Model outputs generated for the thesis	System, bidding-zone and/or nodal level depending on the indicator	Hourly and aggregated outputs	Storage, generation, spill, load shedding, prices and congestion indicators	Alternative rolling-horizon settings were compared against the retained 24/48 specification to assess robustness.	Appendix 3.9 — Internal validation and diagnostic checks

Description: This appendix summarizes the main datasets used in the modelling framework. For each input category, it reports the exact source, reference year, spatial and temporal resolution, variables used, and transformations applied before integration into the model. The table is intended to document the data-processing workflow underlying the construction of the nodal hydro-economic model, including bidding zones, network topology, hydropower assets, demand, inflows, spatial allocation data, and model-generated diagnostic outputs. The table is intended to make each input traceable from its original source to its transformed model representation.

Source: Own compilation based on Statnett, Satheeskumar (2020), the JRC Hydro-power database, SSB StatBank Tables 14091 and 14092, NVE hydrological data, Statistics Norway population and enterprise grid datasets, and model-generated diagnostic outputs.

***Note:** *Model-generated diagnostic outputs do not have an external data source; they are produced by the Python/PyPSA simulations and are included to document internal model consistency.*

Appendix 3.2.2. Additional raw data and intermediate calculations used for demand construction.

Table 3.2.2a. Monthly hydro-compatible scaling factor used for demand construction.

Month	Total electricity generation (GWh)	Hydropower generation (GWh)	Hydro-compatible scaling factor
2023M01	14,139.5	12,247.3	0.866
2023M02	13,781.3	12,043.6	0.874
2023M03	14,299.9	12,830.9	0.897
2023M04	12,003.4	10,648.9	0.887
2023M05	11,122.7	9,830.1	0.884
2023M06	11,333.2	10,444.3	0.922
2023M07	11,480.2	10,596.6	0.923
2023M08	12,198.7	11,225.7	0.920
2023M09	12,003.9	10,707.2	0.892
2023M10	13,364.5	11,775.3	0.881
2023M11	14,283.4	12,716.2	0.890
2023M12	13,964.6	12,261.7	0.878

Description: This table reports the monthly scaling factor used to adapt observed electricity consumption to the hydro-only modelling framework. Total electricity generation and hydropower generation are expressed in GWh. The scaling factor is dimensionless and is calculated as hydropower generation divided by total electricity generation.

Source: Statistics Norway electricity balance data, SSB StatBank Table 14091; author's calculations.

Table 3.2.2b. Observed electricity consumption inputs by bidding zone and consumer group.

Consumer group / zone	2023M01	2023M02	2023M03	2023M04	2023M05	2023M06	2023M07	2023M08	2023M09	2023M10	2023M11	2023M12
Non-household consumption (primary, secondary and tertiary industries)												
NO1 Sørøst-Norge	1,752.2	1,519.6	1,699.3	1,271.8	1,134.7	1,084.6	969.6	1,092.7	1,140.1	1,413.7	1,621.0	1,842.0
NO2 Sørvest-Norge	2,185.9	1,931.0	2,191.4	1,950.7	1,932.7	1,855.2	1,855.6	1,902.0	1,832.2	2,033.9	2,149.6	2,245.8
NO3 Midt-Norge	1,866.2	1,671.1	1,859.6	1,600.9	1,573.6	1,437.8	1,542.4	1,660.3	1,568.2	1,789.9	1,839.6	1,928.7
NO4 Nord-Norge	1,306.0	1,210.4	1,350.1	1,219.6	1,079.0	1,071.6	1,088.1	1,123.7	1,139.6	1,286.0	1,332.6	1,294.6
NO5 Vest-Norge	1,077.4	957.7	1,073.5	936.6	900.2	849.1	899.5	831.2	705.9	993.9	1,047.8	1,111.5
Household consumption												
NO1 Sørøst-Norge	1,910.3	1,607.5	1,760.3	1,285.3	955.9	683.2	667.1	749.1	825.9	1,361.1	1,816.2	2,244.7
NO2 Sørvest-Norge	1,045.0	876.9	1,003.1	758.4	609.6	425.2	449.0	463.7	486.0	779.9	1,011.3	1,215.7
NO3 Midt-Norge	709.7	621.2	711.6	520.0	447.0	328.4	298.7	321.1	370.0	578.2	738.2	943.7

Consumer group / zone	2023M01	2023M02	2023M03	2023M04	2023M05	2023M06	2023M07	2023M08	2023M09	2023M10	2023M11	2023M12
NO4 Nord-Norge	640.9	476.5	554.9	403.3	349.1	259.7	209.7	213.5	286.7	452.7	546.5	610.6
NO5 Vest-Norge	443.4	373.8	426.3	311.5	270.5	181.3	179.8	198.4	223.4	339.5	417.1	499.3

Description: This table reports observed monthly electricity consumption before hydro-compatible scaling. Non-household consumption aggregates primary, secondary and tertiary industries, while household consumption is shown separately. Values are expressed in GWh per month.
Source: Statistics Norway electricity consumption data by price area and consumer group, SSB StatBank Table 14092; author's aggregation.

*__Note:__ *Non-household consumption aggregates industrial and other non-residential consumer groups used in the model demand construction.*

Table 3.2.2c. Hydro-compatible demand intensities used for nodal allocation.

Consumer group / zone	2023M01	2023M02	2023M03	2023M04	2023M05	2023M06	2023M07	2023M08	2023M09	2023M10	2023M11	2023M12
Non-household intensity (MWh per enterprise/month)												
NO1 Sørøst-Norge	5.177	4.530	5.201	3.849	3.421	3.410	3.053	3.430	3.469	4.249	4.922	5.517
NO2 Sørvest-Norge	12.130	10.811	12.597	11.087	10.943	10.953	10.972	11.213	10.470	11.480	12.260	12.633
NO3 Midt-Norge	16.581	14.980	17.115	14.568	14.265	13.592	14.603	15.672	14.348	16.177	16.799	17.371
NO4 Nord-Norge	18.075	16.903	19.357	17.289	15.238	15.780	16.048	16.524	16.242	18.105	18.958	18.164
NO5 Vest-Norge	16.769	15.041	17.309	14.932	14.296	14.062	14.920	13.745	11.314	15.737	16.763	17.538

Consumer group / zone	2023M01	2023M02	2023M03	2023M04	2023M05	2023M06	2023M07	2023M08	2023M09	2023M10	2023M11	2023M12
Household intensity (MWh per inhabitant/month)												
NO1 Sørøst-Norge	0.708	0.601	0.676	0.488	0.361	0.269	0.263	0.295	0.315	0.513	0.692	0.843
NO2 Sørvest-Norge	0.672	0.569	0.668	0.500	0.400	0.291	0.308	0.317	0.322	0.510	0.669	0.793
NO3 Midt-Norge	0.741	0.655	0.770	0.556	0.476	0.365	0.332	0.356	0.398	0.614	0.792	0.999
NO4 Nord-Norge	1.145	0.859	1.027	0.738	0.636	0.494	0.399	0.405	0.527	0.822	1.003	1.106
NO5 Vest-Norge	0.771	0.656	0.768	0.555	0.480	0.336	0.333	0.367	0.400	0.601	0.746	0.881

Description: This table reports the demand intensities obtained after applying the monthly hydro-compatible scaling factor. Non-household intensity is expressed in MWh per enterprise per month; household intensity is expressed in MWh per inhabitant per month. These intensities are later combined with node-level enterprise and population counts to construct monthly nodal demand.

Source: Calculations from SSB StatBank Tables 14091 and 14092 and the spatial allocation keys shown in Appendix Table 3.2.2d.

Table 3.2.2d. Spatial allocation keys used for nodal demand distribution.

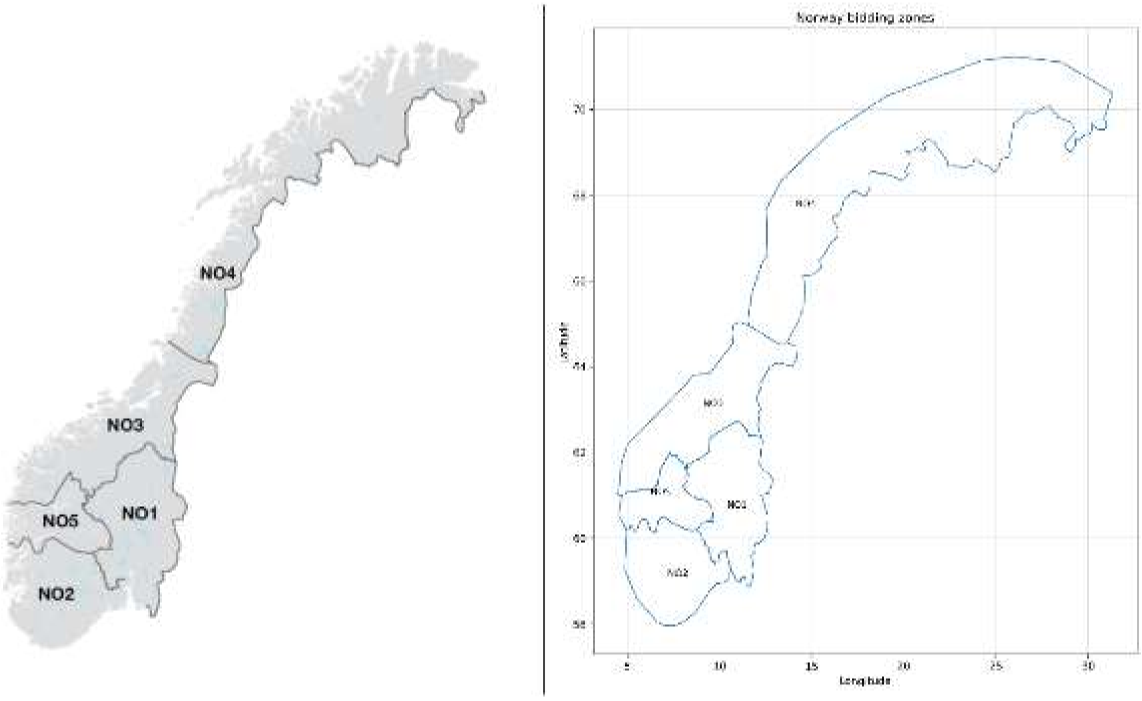
Zone	Population count	Population share	Enterprise / economic-activity count	Enterprise share
NO1	2,337,656	42.50%	293,166	44.01%
NO2	1,346,408	24.48%	156,093	23.44%
NO3	829,264	15.08%	97,489	14.64%
NO4	484,908	8.82%	62,582	9.40%
NO5	497,822	9.05%	55,648	8.35%
Unassigned / none	4,849	0.09%	1,087	0.16%
SUM	5,500,907	100.00%	666,065	100.00%

Description: This table reports the population and enterprise/economic-activity counts used as allocation keys for distributing demand across the model geography. Population shares are used for household demand, while enterprise shares are used for non-household demand. The unassigned row is retained for traceability.

Source: Author's spatial preprocessing based on Statistics Norway population grid data and enterprise grid data, aggregated to the five Norwegian bidding zones.

Appendix 3.3. Norwegian bidding zones and spatial representation used in the model.

Figure 3.3a. Official Norwegian bidding zones and thesis-specific spatial representation used in the model.

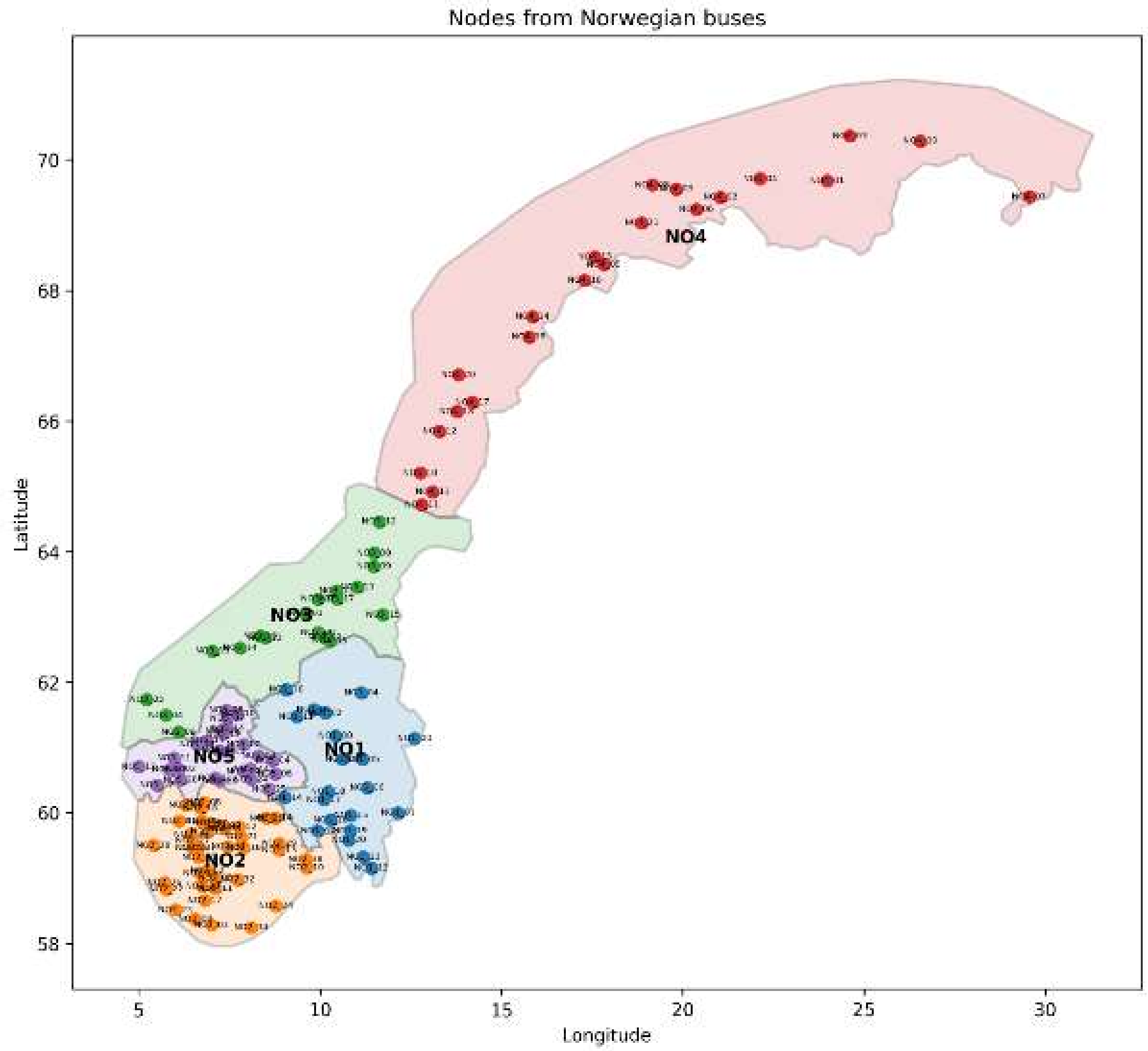


Description: The left-hand map presents the official Norwegian bidding zones published by Statnett from *Why we have bidding zones* (Statnett, 2025). The right-hand map is the model's representation, inspired by the official zonal division and used as the spatial basis for the modelling framework developed in this thesis.

Source: Left-hand map: Statnett, "Why we have bidding zones" (2025). Right-hand map: author's own GeoJSON-based spatial reconstruction inspired by Statnett's official bidding-zone representation.

Appendix 3.4. Spatial node representation used in the modelling framework.

Figure 3.4a. Thesis-specific representation of the Norwegian nodes used in the modelling framework.

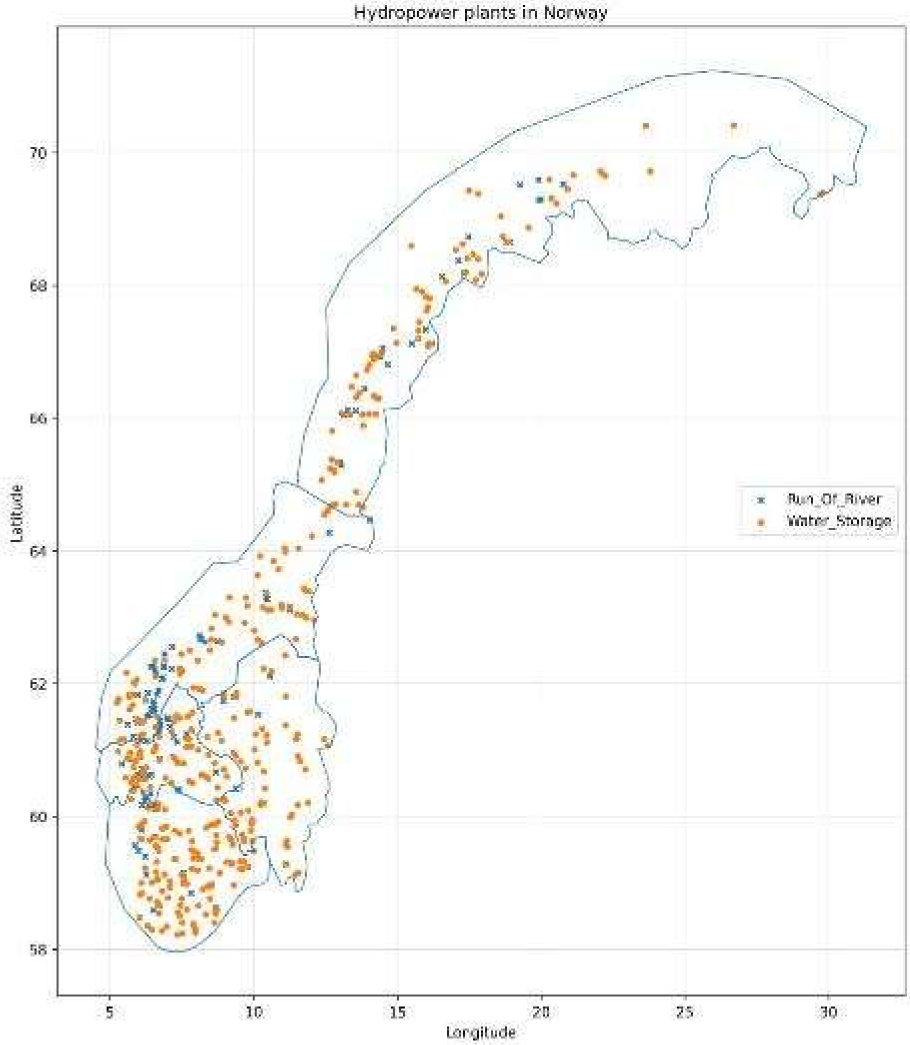


Description: The figure shows the representation of the Norwegian nodes used in the model, grouped within the five official bidding zones. The underlying spatial structure is derived from open-source material provided in the *spatial-disaggregation* repository by Iasonask, itself associated with Satheeskumar (2020). The figure is reproduced and adapted for the purposes of the present thesis. Nodes are grouped by Norwegian bidding zone after the point-in-polygon assignment procedure.

Source: Author's processing based on the Nordic 490 / spatial-disaggregation open-source network material associated with Satheeskumar (2020).

Appendix 3.5. Hydropower plants used for node-capacity assessment.

Figure 3.5a. Hydropower plants with installed capacity of at least 5 MW plotted on the Norwegian bidding zones.

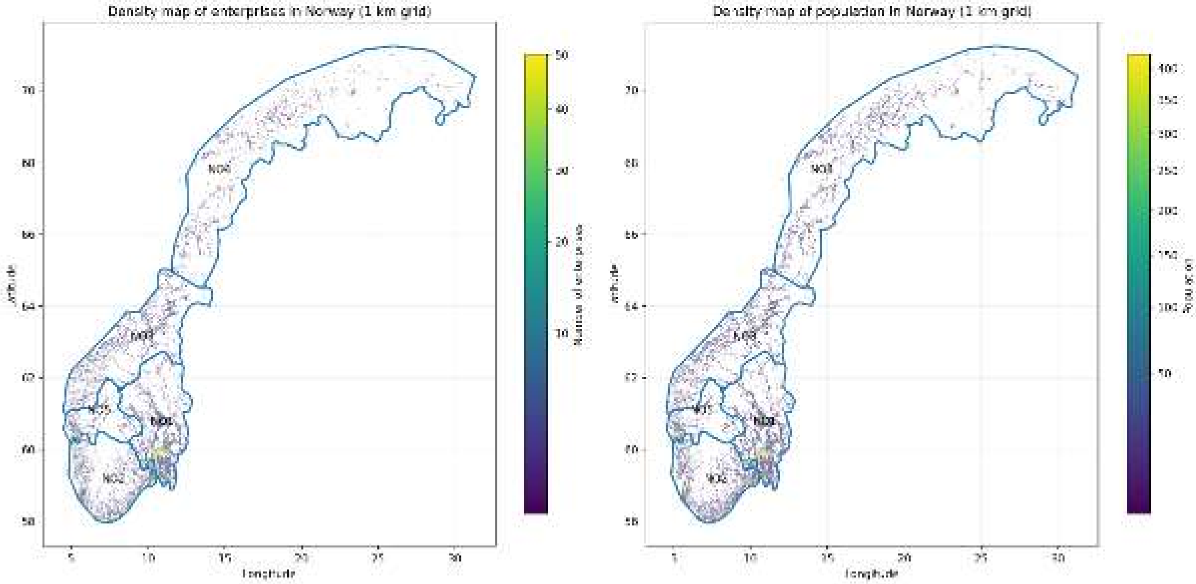


Description: The figure shows the model's representation of hydropower plants with an installed capacity greater than or equal to 5 MW, based on data from the *JRC Hydro-power plants database*. Plants are classified into run-of-river and water-storage technologies and plotted against the Norwegian bidding-zone structure. These plant-level data are used in this thesis to assess and assign generation capacity at the node level.

Source: Author's processing based on the JRC Hydro-power plants database / Energy-Modelling-Toolkit hydro-power database.

Appendix 3.6. Comparative density maps of population and enterprises in Norway.

Figure 3.6a. Comparative density maps of enterprises and population in Norway (1 km grid).

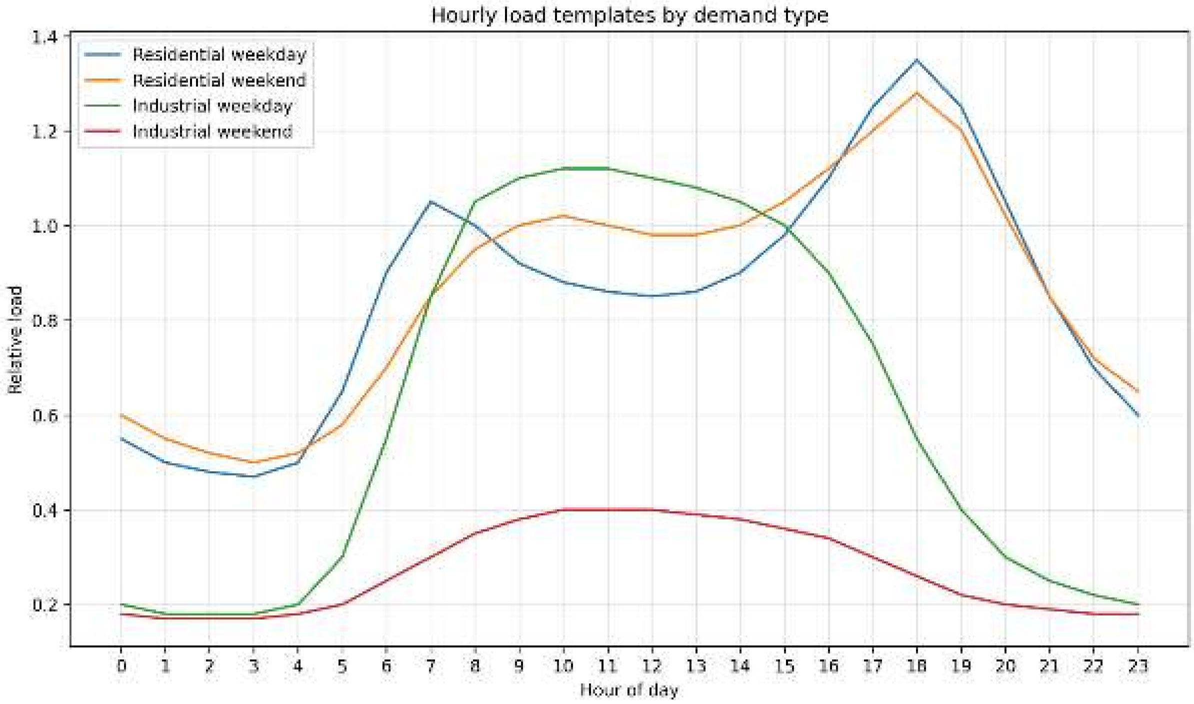


Description: The left-hand panel shows enterprise density based on the Bedrifter på rutenett dataset, while the right-hand panel shows population density based on the Befolkning på rutenett 1000 m 2023 dataset. Both figures are the representations developed in Python from open datasets published on data.norge.no. These spatial indicators are used in the model to inform the geographical allocation of demand. The maps are based on 1 km grid cells and are used as spatial proxies for residential and non-household demand allocation.

Source: Author’s processing based on Statistics Norway, “Befolkning på rutenett 1000 m 2023”, and Statistics Norway enterprise grid data, “Bedrifter på rutenett”.

Appendix 3.7. Hourly load templates used for demand disaggregation.

Figure 3.7a. Hourly load templates by demand type.

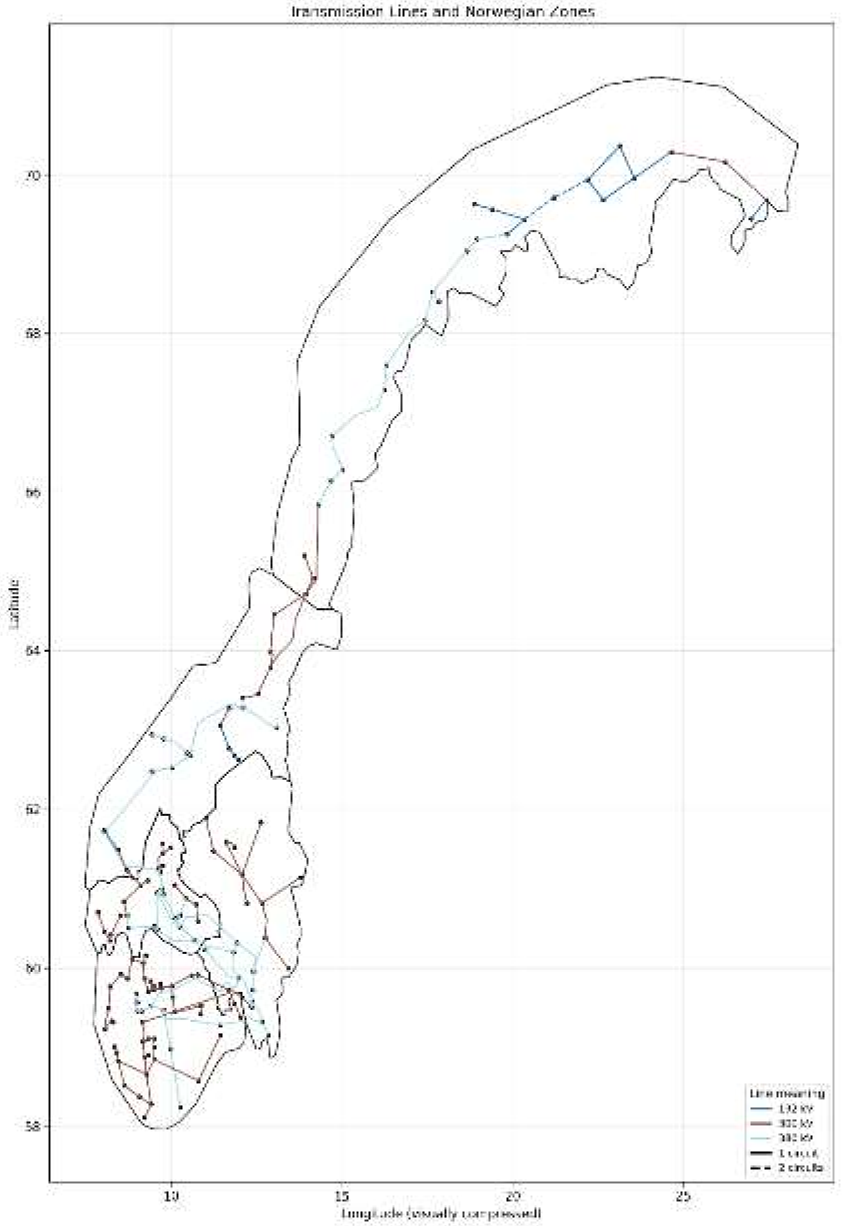


Description: The figure shows the stylized hourly load templates used in the thesis to disaggregate demand over the day for residential and industrial sectors, with a distinction between weekdays and weekends. Values are expressed as relative load and therefore represent normalized demand shapes rather than absolute consumption levels.

Source: Author’s model-specific stylized load-shape templates.

Appendix 3.8. Transmission network representation used in the modelling framework.

Figure 3.8a. Processed transmission network representation of Norway used in the modelling framework.



Description: The figure presents the representation of the Norwegian transmission network used in the model. Transmission lines are differentiated by voltage class (132 kV, 300 kV, and 380 kV) and by the number of circuits. The network is shown in stylised form for modelling purposes and should therefore be interpreted as a model-consistent transmission topology rather than as an exact engineering map of the real system. The underlying network structure is derived from open-source material provided in the *spatial-disaggregation* repository by Iasonask, itself associated with Satheeskumar (2020).

Source: Author's processing based on the Nordic 490 / spatial-disaggregation transmission-network material associated with Satheeskumar (2020).

***Note:** *Line capacities are stylized corridor capacities derived from voltage classes. The figure should therefore be interpreted as a model-consistent network representation rather than as an exact engineering map of the Norwegian transmission system.*

Appendix 3.9. Model validation and diagnostic checks.

Table 3.9a. Internal consistency checks applied to the modelling framework.

Diagnostic area	Main check	Purpose
Demand construction	Monthly nodal demand sums back to zonal totals	Verifies that spatial and temporal disaggregation preserves aggregate energy.
Reservoir balance	Storage equals previous storage plus inflow minus turbine release and spill	Confirms water conservation.
Run-of-river balance	Inflow equals generation plus spill	Confirms that run-of-river inflow is not stored implicitly.
Transmission	Line flows remain within thermal limits	Checks physical feasibility of the network solution.
Rolling horizon	End-of-day storage is correctly carried to the next window	Ensures chronological continuity.
Redispatch	Nodal redispatch solution satisfies full physical network constraints	Validates that the zonal schedule has been corrected into a feasible physical dispatch.
Storage targets	Redispatch storage deviations are recorded and penalised	Ensures that redispatch does not freely redefine reservoir management.
Terminal values	Longer look-ahead and terminal-value diagnostics are compared ex post	Evaluates whether the end-of-window treatment materially distorts results.

Description: This appendix presents the internal validation and diagnostic checks carried out to assess the consistency of the modelling framework. These checks verify whether the main components of the model behave as expected under the assumptions described in Chapter 3. They cover hourly nodal demand construction, reservoir and run-of-river water balances, storage limits, transmission constraints, rolling-horizon state carry-over, and the link between zonal market schedules and nodal redispatch outcomes. The purpose of this appendix is not to validate the model against observed historical market outcomes, but to ensure the internal coherence of the simulated system and to document the main consistency tests performed before interpreting the scenario results.

Source: Author's internal validation framework based on the Python/PyPSA model implementation and simulation outputs.

Appendix 3.10. Terminal water-value and rolling-horizon consistency checks.

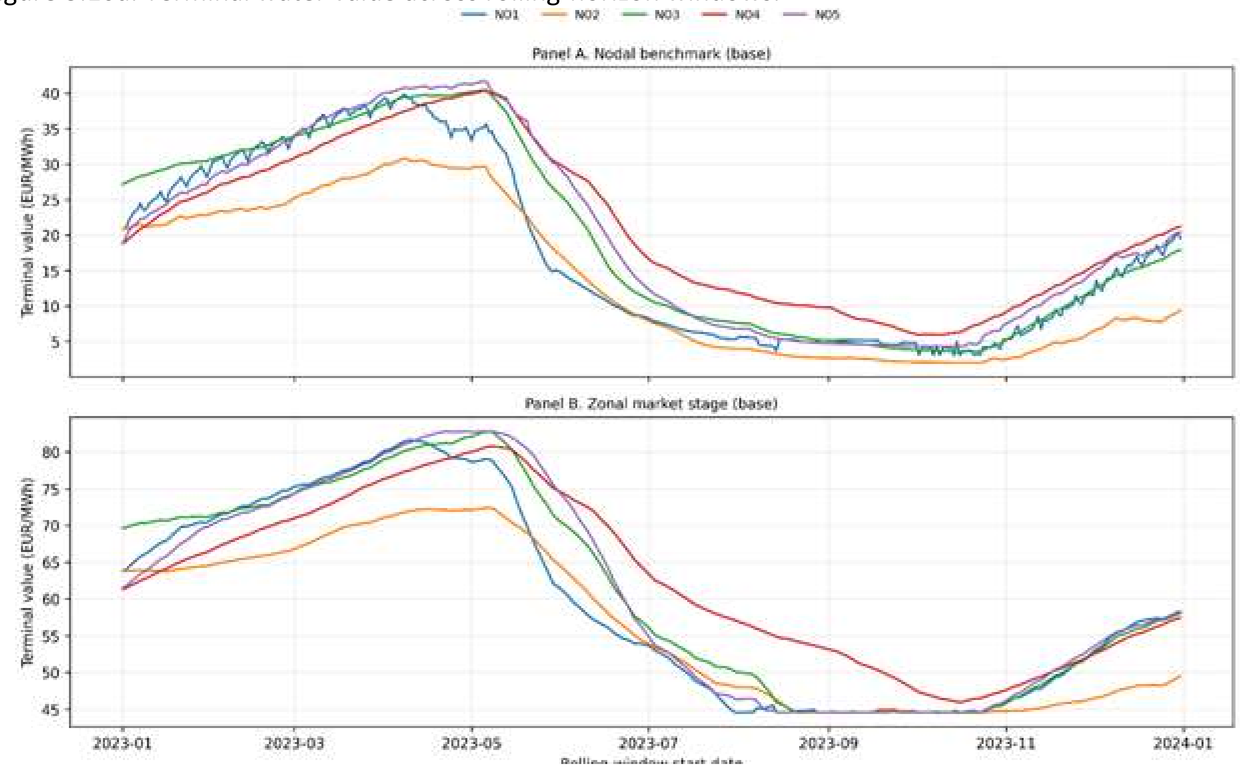
Table 3.10a. Terminal water-value construction checks.

Design	Scenario	Base min (EUR/MWh)	Base max (EUR/MWh)	Segment multipliers	Descending multipliers	Negative values	Non-finite values	Above max	Fill logic	Inflow logic	Load logic
Nodal benchmark	-15%	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	Yes	Yes
Nodal benchmark	-10%	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	Yes	Yes
Nodal benchmark	base	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	Yes	Yes
Nodal benchmark	+10%	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	Yes	Yes
Nodal benchmark	+15%	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	Yes	Yes
Zonal market stage	-15%	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	—	—
Zonal market stage	-10%	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	—	—
Zonal market stage	base	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	—	—
Zonal market stage	+10%	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	—	—
Zonal market stage	+15%	2.000	85.00	1.35, 1.05, 0.75, 0.45	Yes	0.000	0.000	0.000	Yes	—	—

Description: This table summarizes the checks applied to the terminal water-value formulation. It examines whether terminal values remain within the imposed lower and upper bounds, whether any negative or non-finite values occur, and whether the segment multipliers used in the terminal-value function decrease as intended. The results support the internal consistency of the formulation. No negative, non-finite or above-bound values are identified, and the segment multipliers remain correctly ordered across the available scenarios and market-design configurations.

Source: Author's calculations from the Python/PyPSA rolling-horizon simulations.

Figure 3.10a. Terminal water value across rolling-horizon windows.



Description: This figure presents the evolution of terminal water values across rolling horizon windows in the base scenario. Panel A refers to the nodal benchmark, while Panel B refers to the zonal market stage. The results show that terminal water values vary over time and across zones. They are therefore not fixed parameters but change with system conditions and with the spatial structure of the market representation. This supports the interpretation that the model assigns changing opportunity values to stored water throughout the simulation. The x-axis reports rolling horizon windows over the simulation year. The y-axis reports terminal water values in EUR/MWh.

Source: Author's calculations from rolling-horizon simulation outputs.

Table 3.10b. Terminal water-value diagnostics by zone and scenario..

Design	Scenario	Zone	Windows	Mean TWV (EUR/MWh)	Min TWV	Max TWV	Mean fill ratio	Corr. TWV-fill	Missing	Negative	Non-finite	Above max
Nodal benchmark	-15%	NO1	365.0	20.15	3.910	40.89	0.629	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	-15%	NO2	365.0	16.03	2.770	32.17	0.717	-1.000	0.000	0.000	0.000	0.000
Nodal benchmark	-15%	NO3	365.0	22.36	6.390	41.38	0.596	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	-15%	NO4	365.0	24.55	12.15	41.28	0.553	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	-15%	NO5	365.0	22.55	5.550	43.10	0.584	-0.998	0.000	0.000	0.000	0.000
Nodal benchmark	-10%	NO1	365.0	19.12	3.400	40.50	0.649	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	-10%	NO2	365.0	15.07	2.290	31.72	0.736	-1.000	0.000	0.000	0.000	0.000
Nodal benchmark	-10%	NO3	365.0	21.26	5.150	41.07	0.618	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	-10%	NO4	365.0	23.34	9.940	41.00	0.577	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	-10%	NO5	365.0	21.46	4.580	42.65	0.607	-0.998	0.000	0.000	0.000	0.000
Nodal benchmark	base	NO1	365.0	17.72	3.010	39.86	0.677	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	base	NO2	365.0	13.67	2.000	30.85	0.762	-1.000	0.000	0.000	0.000	0.000
Nodal benchmark	base	NO3	365.0	19.59	3.500	40.49	0.651	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	base	NO4	365.0	21.13	5.910	40.43	0.620	-1.000	0.000	0.000	0.000	0.000
Nodal benchmark	base	NO5	365.0	20.13	4.240	41.73	0.634	-0.999	0.000	0.000	0.000	0.000

Design	Scenario	Zone	Windows	Mean TWV (EUR/MWh)	Min TWV	Max TWV	Mean fill ratio	Corr. TWV-fill	Missing	Negative	Non-finite	Above max
Nodal benchmark	+10%	NO1	365.0	16.77	2.840	39.21	0.696	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	+10%	NO2	365.0	12.73	2.000	29.97	0.781	-1.000	0.000	0.000	0.000	0.000
Nodal benchmark	+10%	NO3	365.0	18.41	2.880	39.63	0.674	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	+10%	NO4	365.0	19.42	3.930	39.81	0.654	-1.000	0.000	0.000	0.000	0.000
Nodal benchmark	+10%	NO5	365.0	19.20	4.030	40.88	0.651	-0.998	0.000	0.000	0.000	0.000
Nodal benchmark	+15%	NO1	365.0	16.40	2.770	38.95	0.703	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	+15%	NO2	365.0	12.34	2.000	29.54	0.788	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	+15%	NO3	365.0	17.91	2.600	38.99	0.683	-0.999	0.000	0.000	0.000	0.000
Nodal benchmark	+15%	NO4	365.0	18.79	3.470	39.45	0.666	-1.000	0.000	0.000	0.000	0.000
Nodal benchmark	+15%	NO5	365.0	18.84	4.020	40.47	0.657	-0.998	0.000	0.000	0.000	0.000
Zonal market stage	-15%	NO1	365.0	63.25	46.68	82.81	0.632	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	-15%	NO2	365.0	59.59	45.32	74.33	0.704	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	-15%	NO3	365.0	65.06	50.09	82.85	0.597	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	-15%	NO4	365.0	66.60	55.23	82.18	0.567	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	-15%	NO5	365.0	64.34	47.88	83.02	0.611	-1.000	0.000	0.000	0.000	0.000

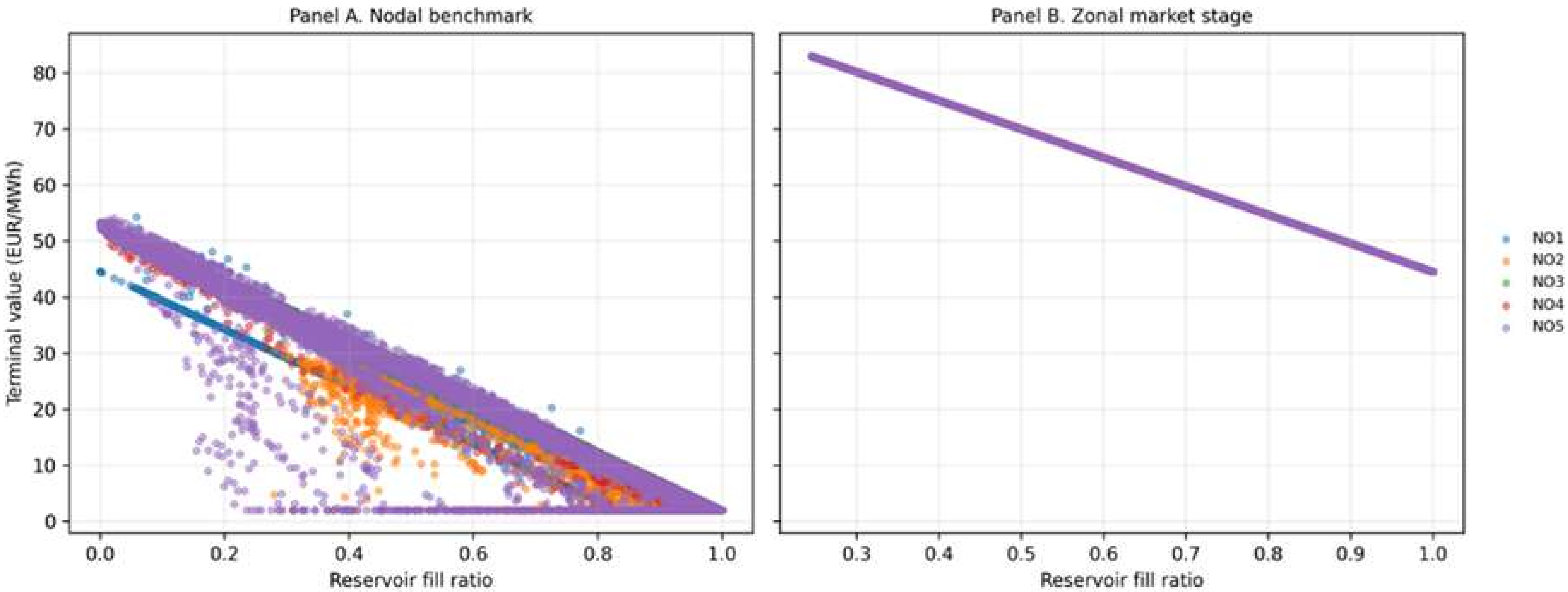
Design	Scenario	Zone	Windows	Mean TWV (EUR/MWh)	Min TWV	Max TWV	Mean fill ratio	Corr. TWV-fill	Missing	Negative	Non-finite	Above max
Zonal market stage	-10%	NO1	365.0	61.30	44.50	82.45	0.671	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	-10%	NO2	365.0	58.14	44.50	73.66	0.732	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	-10%	NO3	365.0	63.11	44.50	82.83	0.635	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	-10%	NO4	365.0	65.30	52.49	81.76	0.592	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	-10%	NO5	365.0	62.63	44.50	82.91	0.644	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	base	NO1	365.0	59.58	44.50	81.63	0.704	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	base	NO2	365.0	56.48	44.50	72.40	0.765	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	base	NO3	365.0	61.22	44.50	82.77	0.672	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	base	NO4	365.0	62.48	45.93	80.79	0.647	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	base	NO5	365.0	60.96	44.50	82.80	0.677	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+10%	NO1	365.0	58.35	44.50	80.51	0.728	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+10%	NO2	365.0	55.46	44.50	71.65	0.785	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+10%	NO3	365.0	59.98	44.50	81.47	0.697	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+10%	NO4	365.0	60.61	44.50	79.99	0.684	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+10%	NO5	365.0	60.18	44.50	82.79	0.693	-1.000	0.000	0.000	0.000	0.000

Design	Scenario	Zone	Windows	Mean TWV (EUR/MWh)	Min TWV	Max TWV	Mean fill ratio	Corr. TWV-fill	Missing	Negative	Non-finite	Above max
Zonal market stage	+15%	NO1	365.0	57.89	44.50	80.06	0.737	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+15%	NO2	365.0	55.08	44.50	71.34	0.793	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+15%	NO3	365.0	59.51	44.50	80.72	0.706	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+15%	NO4	365.0	60.01	44.50	79.65	0.696	-1.000	0.000	0.000	0.000	0.000
Zonal market stage	+15%	NO5	365.0	59.86	44.50	82.78	0.699	-1.000	0.000	0.000	0.000	0.000

Description: This table reports terminal water-value diagnostics by market design, scenario and zone. It includes the number of rolling-horizon windows, mean, minimum and maximum terminal water values, mean reservoir fill ratio, the correlation between terminal water values and fill ratios, and checks for missing, negative or non-finite values. The table is used to verify that terminal water values remain numerically consistent and vary in line with the storage-scarcity logic of the model. Mean, minimum and maximum terminal values are expressed in EUR/MWh. Fill ratios are dimensionless.

Source: Author’s calculations from rolling-horizon simulation outputs.

Figure 3.10b. Relationship between terminal water values and reservoir fill ratio.



Description: This figure presents the relationship between reservoir fill ratios and terminal water values. The observed downward relationship is consistent with the expected economic logic of water values. When reservoirs are relatively low, stored water is scarcer and is assigned a higher terminal value. When reservoirs are fuller, the marginal value of additional stored water decreases. The figure therefore supports the coherence of the implemented terminal-value formula, while remaining a diagnostic check rather than a validation of true long-term stochastic water values.

Source: Author's calculations from rolling-horizon simulation outputs.

Table 3.10c. Storage carry-over and rolling-horizon boundary checks.

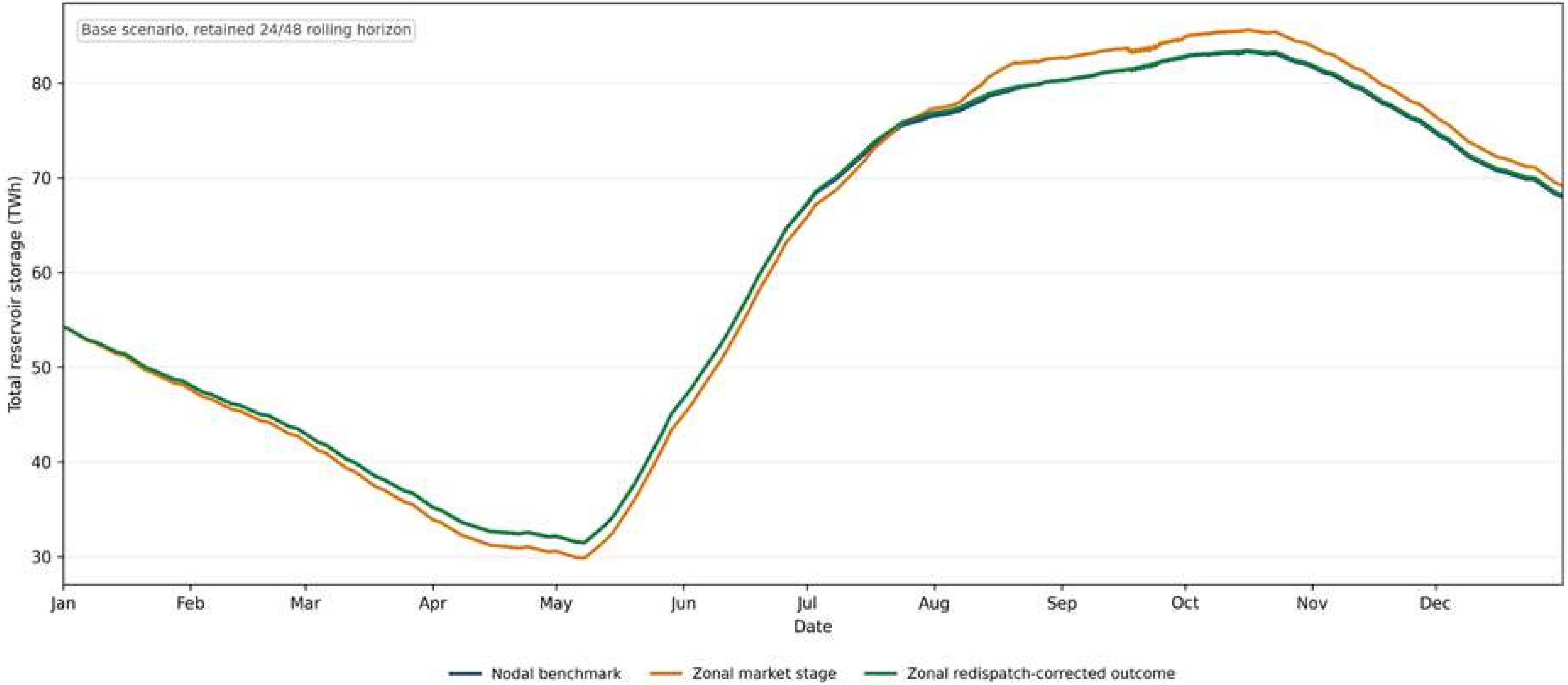
Design	Scenario	Windows	Mean carry-over error (MWh)	Max abs. carry-over error (MWh)	Errors > 1e-6	Mean boundary change (MWh)	Mean non-boundary change (MWh)	Boundary-change ratio	Max boundary change (MWh)
Nodal benchmark	-15%	365.0	0.000	0.000	0.000	9,525	11,015	0.865	33,596
Nodal benchmark	-10%	365.0	0.000	0.000	0.000	9,495	11,202	0.848	35,700
Nodal benchmark	base	365.0	0.000	0.000	0.000	9,235	11,640	0.793	40,075
Nodal benchmark	+10%	365.0	0.000	0.000	0.000	9,174	12,112	0.757	44,193
Nodal benchmark	+15%	365.0	0.000	0.000	0.000	9,178	12,339	0.744	46,360
Zonal market stage	-15%	365.0	0.000	0.000	0.000	9,877	11,544	0.856	33,691
Zonal market stage	-10%	365.0	0.000	0.000	0.000	10,367	12,057	0.860	35,876
Zonal market stage	base	365.0	0.000	0.000	0.000	11,350	13,392	0.847	51,771
Zonal market stage	+10%	365.0	0.000	0.000	0.000	12,853	14,917	0.862	113,179
Zonal market stage	+15%	365.0	0.000	0.000	0.000	13,835	15,751	0.878	125,981

Description: This table reports storage carry-over and rolling-horizon boundary checks for the available simulations. It verifies whether reservoir storage is transferred consistently from one rolling-horizon window to the next, and whether storage changes at window boundaries differ substantially from ordinary within-window changes. The results show zero mean carry-over errors, zero maximum absolute carry-over errors and no errors above the numerical tolerance across all scenarios and market-design configurations. This supports the chronological consistency of the rolling-horizon implementation. Carry-over errors are expressed in MWh. Boundary jumps compare storage changes at rolling-horizon transitions with ordinary within-window changes.

Source: Author’s calculations from rolling-horizon simulation outputs.

***Note:** *Zero or near-zero carry-over errors indicate chronological continuity between successive rolling-horizon windows.*

Figure 3.10c. Reservoir storage trajectory around rolling-horizon boundaries.



Description: This figure presents the full-year trajectory of total reservoir storage under the base hydrological scenario and the retained 24/48 rolling-horizon specification. The three curves compare the nodal benchmark, the zonal market stage, and the zonal redispatch-corrected outcome. The purpose of the figure is to verify that reservoir storage evolves smoothly over the full simulation year and that the rolling-horizon implementation does not generate obvious artificial discontinuities or repeated end-of-window depletion. The trajectories follow the expected seasonal pattern of reservoir management, with winter and spring drawdown, summer refill, and late-year decline. This supports the reasonableness of the rolling-horizon and terminal water-value treatment, while not constituting a full validation of long-term stochastic water values. The x-axis reports the simulation date in 2023. The y-axis reports total reservoir storage in TWh.

Source: Author’s calculations from rolling-horizon simulation outputs.

Appendix 3.11.1. Core calibration and rolling-horizon settings.

***Note:** *The listings are selected excerpts intended to document the main modelling choices. They do not reproduce the complete executable scripts, but they show the code components directly linked to the methodology and results discussed in the thesis.*

Listing 3.11a. Main calibration parameters used in the Python implementation.

```
# Time window
START_TIMESTAMP = pd.Timestamp("2023-01-01 00:00:00")
END_TIMESTAMP_EXCLUSIVE = pd.Timestamp("2023-12-31 23:00:00")

# Rolling-horizon settings
USE_ROLLING_HORIZON = True
ROLLING_LOOKAHEAD_HOURS = 48
ROLLING_STEP_HOURS = 24

# Economic assumptions
LOAD_SHEDDING_COST = 200.0
SPILL_COST = 5.0
HYDRO_TURBINE_VARIABLE_COST = 0.5
TERMINAL_STORAGE_SHORTFALL_COST = 150.0

# Endogenous terminal water-value approximation
USE_ENDOGENOUS_TERMINAL_VALUE = True
TERMINAL_VALUE_NUM_SEGMENTS = 4
TERMINAL_VALUE_SEGMENT_MULTIPLIERS = [1.35, 1.05, 0.75, 0.45]
TERMINAL_VALUE_BASE_MIN = 2.0
TERMINAL_VALUE_SCARCITY_COEF = 85.0
TERMINAL_VALUE_BASE_MAX = 85.0
TERMINAL_VALUE_FILL_WEIGHT = 0.60
TERMINAL_VALUE_LOAD_WEIGHT = 0.40
TERMINAL_VALUE_INFLOW_WEIGHT = 0.50

# Reservoir initial filling by zone (%)
INITIAL_SOC_BY_ZONE = {
  "NO1": 0.622,
  "NO2": 0.621,
  "NO3": 0.508,
  "NO4": 0.670,
  "NO5": 0.666,
}

# Minimum final storage by zone (%)
FINAL_SOC_MIN_BY_ZONE = {
  "NO1": 0.15,
  "NO2": 0.15,
  "NO3": 0.15,
  "NO4": 0.15,
  "NO5": 0.15,
}

# Inflow split calibration
ROR_SPLIT_BOOST = 3.0
```

Description: This excerpt reports the main numerical settings used in the simulations. The model covers the year 2023 and uses a 48-hour optimization window with a 24-hour implementation step. The cost parameters define the penalties associated with load shedding, spill and final storage shortfalls. Initial reservoir levels are specified by bidding zone, while the terminal water-value parameters control the reward assigned to water remaining in storage at the end of each optimization window.

Source: Author's Python/PyPSA implementation.

***Note:** *Cost parameters are modelling assumptions used for comparative simulation purposes and should not be interpreted as calibrated estimates of actual Norwegian market costs.*

Appendix 3.11.2. Splitting total hydrological inflow between reservoir and run-of-river hydropower.

Listing 3.11b. Hydrological inflow split at nodal level.

```
inflow_total_wide = (
    inflows.pivot(index="timestamp", columns="node_id", values="node_inflow_hour_mwh")
    .reindex(index=snapshots, columns=nodes["node_id"])
    .fillna(0.0)
)

nodes["effective_ror_cap_mw"] = ROR_SPLIT_BOOST * nodes["ror_capacity_MW"]
nodes["effective_total_hydro_cap_mw"] = (
    nodes["effective_ror_cap_mw"] + nodes["water_storage_capacity_MW"]
)

nodes["ror_share"] = np.where(
    nodes["effective_total_hydro_cap_mw"] > 0,
    nodes["effective_ror_cap_mw"] / nodes["effective_total_hydro_cap_mw"],
    0.0
)

nodes["res_share"] = np.where(
    nodes["effective_total_hydro_cap_mw"] > 0,
    nodes["water_storage_capacity_MW"] / nodes["effective_total_hydro_cap_mw"],
    0.0
)

ror_share = (
    nodes.set_index("node_id")["ror_share"]
    .reindex(inflow_total_wide.columns)
    .fillna(0.0)
)

res_share = (
    nodes.set_index("node_id")["res_share"]
    .reindex(inflow_total_wide.columns)
    .fillna(0.0)
)

inflow_ror_wide = inflow_total_wide.multiply(ror_share, axis=1)
inflow_res_wide = inflow_total_wide.multiply(res_share, axis=1)

split_check = (
    inflow_ror_wide + inflow_res_wide - inflow_total_wide
).abs().max().max()
```

```
print(f"Max abs diff inflow split reconstruction: {split_check}")
```

Description: This excerpt shows how total hourly nodal inflow is divided between reservoir and run-of-river hydropower. The split is based on effective capacity shares. A run-of-river boost factor is applied to avoid underrepresenting contemporaneous run-of-river availability in the model. The final reconstruction check verifies that the split preserves the original total inflow at nodal level.
Source: Author's Python/PyPSA implementation.

Appendix 3.11.3. Construction of the physical nodal electricity network.

Listing 3.11c. Nodal buses, lines and loads.

```
n = pypsa.Network()
n.set_snapshots(snapshots)

carrier_names = [
    "AC", "water", "water_ror",
    "hydro_store", "hydro_turbine", "hydro_inflow",
    "hydro_spill", "spill_sink",
    "ror_inflow", "ror_turbine", "ror_spill",
    "load_shedding"
]
n.add("Carrier", carrier_names)

# Electricity buses
n.add(
    "Bus",
    nodes["node_id"].values,
    v_nom=nodes["bus_voltage_kV"].astype(float).values,
    x=nodes["lon"].astype(float).values,
    y=nodes["lat"].astype(float).values,
    carrier=["AC"] * len(nodes)
)

# Transmission lines
n.add(
    "Line",
    lines["line_name"].values,
    bus0=lines["bus0"].values,
    bus1=lines["bus1"].values,
    x=lines["x"].astype(float).values,
    r=np.full(len(lines), MIN_RESISTANCE),
    s_nom=lines["s_nom"].astype(float).values,
    carrier=["AC"] * len(lines)
)

# Nodal electricity demand
load_p_set = (
    demand.pivot(index="timestamp", columns="node_id", values="Demand")
    .reindex(index=snapshots, columns=nodes["node_id"])
    .fillna(0.0)
)

n.add(
    "Load",
```

```
    nodes["node_id"].values,
    bus=nodes["node_id"].values,
    p_set=load_p_set.to_numpy()
)
```

Description: This excerpt implements the physical nodal network used in the nodal benchmark. Electrical buses correspond to the cleaned modelling nodes, while transmission lines connect those nodes through a lossless DC representation. Demand is treated as an exogenous hourly nodal load. This implementation corresponds to the physical benchmark in which reservoir dispatch, run-of-river generation, spill, load shedding, storage and line flows are optimized on the full nodal system
Source: Author's Python/PyPSA implementation.

Appendix 3.11.4. Reservoir hydropower representation.

Listing 3.11d. Reservoir stores, turbine links, inflow generators and spill links.

```
hydro_nodes = nodes.loc[
    (nodes["water_storage_capacity_MW"] > 0) |
    (nodes["node_storage_capacity_MWh"] > 0)
].copy()

hydro_nodes["turbine_p_nom_mw"] = (
    hydro_nodes["water_storage_capacity_MW"].clip(lower=MIN_P_NOM)
)

hydro_nodes["reservoir_e_nom_mwh"] = (
    hydro_nodes["node_storage_capacity_MWh"].clip(lower=MIN_E_NOM)
)

hydro_nodes["soc_initial_fraction"] = hydro_nodes["zone"].map(INITIAL_SOC_BY_ZONE)
hydro_nodes["soc_final_min_fraction"] = hydro_nodes["zone"].map(FINAL_SOC_MIN_BY_ZONE)

hydro_nodes["e_initial_mwh"] = (
    hydro_nodes["soc_initial_fraction"] * hydro_nodes["reservoir_e_nom_mwh"]
)

hydro_nodes["e_final_min_mwh"] = (
    hydro_nodes["soc_final_min_fraction"] * hydro_nodes["reservoir_e_nom_mwh"]
)

# Water buses
hydro_nodes["water_bus"] = hydro_nodes["node_id"].apply(lambda x: f"WATER_{x}")
n.add(
    "Bus",
    hydro_nodes["water_bus"].values,
    carrier=["water"] * len(hydro_nodes)
)

# Reservoir stores
res_names = [f"RES_{nid}" for nid in hydro_nodes["node_id"].values]
n.add(
    "Store",
    res_names,
    bus=hydro_nodes["water_bus"].values,
    e_nom=hydro_nodes["reservoir_e_nom_mwh"].astype(float).values,
    e_initial=hydro_nodes["e_initial_mwh"].astype(float).values,
    e_cyclic=[False] * len(hydro_nodes),
```

```
    standing_loss=np.full(len(hydro_nodes), STANDING_LOSS_STORE),
    carrier=["hydro_store"] * len(hydro_nodes)
)

# Segmented turbine links
segment_rows = []

for row in hydro_nodes.itertuples(index=False):
    total_p_nom = float(row.turbine_p_nom_mw)

    for seg_label, seg_share in zip(TURBINE_SEGMENT_LABELS, TURBINE_SEGMENT_SHARES):
        segment_rows.append({
            "segment_name": f"TURB_{row.node_id}_{seg_label}",
            "node_id": row.node_id,
            "zone": row.zone,
            "water_bus": row.water_bus,
            "electric_bus": row.node_id,
            "segment_label": seg_label,
            "p_nom_mw": total_p_nom * float(seg_share)
        })

turb_segments = pd.DataFrame(segment_rows)
turb_names = turb_segments["segment_name"].tolist()

n.add(
    "Link",
    turb_names,
    bus0=turb_segments["water_bus"].values,
    bus1=turb_segments["electric_bus"].values,
    p_nom=turb_segments["p_nom_mw"].astype(float).values,
    efficiency=np.full(len(turb_segments), TURBINE_EFFICIENCY),
    carrier=["hydro_turbine"] * len(turb_segments)
)

# Reservoir spill links
spill_names = [f"SPILL_{nid}" for nid in hydro_nodes["node_id"].values]
n.add(
    "Link",
    spill_names,
    bus0=hydro_nodes["water_bus"].values,
    bus1=["SPILL_SINK"] * len(hydro_nodes),
    p_nom=np.maximum(
        hydro_nodes["reservoir_e_nom_mwh"].astype(float).values,
        inflow_res_wide[hydro_nodes["node_id"]].max(axis=0).astype(float).values
    ),
    efficiency=np.ones(len(hydro_nodes)),
    marginal_cost=np.full(len(hydro_nodes), SPILL_COST),
    carrier=["hydro_spill"] * len(hydro_nodes)
)

# Reservoir inflow generators
inflow_res_for_hydro = (
    inflow_res_wide.reindex(columns=hydro_nodes["node_id"]).fillna(0.0)
)

inflow_res_p_nom = inflow_res_for_hydro.max(axis=0).clip(lower=MIN_P_NOM)
inflow_res_p_max_pu = (
```

```
    inflow_res_for_hydro.divide(inflow_res_p_nom, axis=1).fillna(0.0)
)

inflow_res_names = [f"INFLOW_RES_{nid}" for nid in hydro_nodes["node_id"].values]
n.add(
    "Generator",
    inflow_res_names,
    bus=hydro_nodes["water_bus"].values,
    p_nom=inflow_res_p_nom.reindex(hydro_nodes["node_id"]).astype(float).values,
    p_min_pu=inflow_res_p_max_pu[hydro_nodes["node_id"]].to_numpy(),
    p_max_pu=inflow_res_p_max_pu[hydro_nodes["node_id"]].to_numpy(),
    marginal_cost=np.zeros(len(hydro_nodes)),
    carrier=["hydro_inflow"] * len(hydro_nodes)
)
```

Description: This excerpt implements reservoir hydropower through a hydraulic layer separated from the electrical network. Each reservoir node is assigned a water bus and a Store representing energy-equivalent storage capacity. Turbine Links convert stored water into electricity, while inflow Generators inject exogenous water availability into the water bus. Spill Links release water to a spill sink when it cannot be stored or converted into useful generation. This code operationalizes the mass-balance logic described in the methodology.
Source: Author's Python/PyPSA implementation.

Appendix 3.11.5. Run-of-river hydropower representation.

Listing 3.11e. Run-of-river inflow, turbine and spill paths.

```
ror_nodes = nodes.loc[nodes["ror_capacity_MW"] > 0].copy()

if not ror_nodes.empty:
    ror_nodes["ror_water_bus"] = (
        ror_nodes["node_id"].apply(lambda x: f"WATER_ROR_{x}")
    )

    n.add(
        "Bus",
        ror_nodes["ror_water_bus"].values,
        carrier=["water_ror"] * len(ror_nodes)
    )

    inflow_ror_for_ror_nodes = (
        inflow_ror_wide.reindex(columns=ror_nodes["node_id"]).fillna(0.0)
    )

    ror_inflow_p_nom = inflow_ror_for_ror_nodes.max(axis=0).clip(lower=MIN_P_NOM)
    ror_inflow_p_max_pu = (
        inflow_ror_for_ror_nodes.divide(ror_inflow_p_nom, axis=1).fillna(0.0)
    )

    ror_inflow_names = [f"INFLOW_ROR_{nid}" for nid in ror_nodes["node_id"].values]
    n.add(
        "Generator",
        ror_inflow_names,
        bus=ror_nodes["ror_water_bus"].values,
        p_nom=ror_inflow_p_nom.reindex(ror_nodes["node_id"]).astype(float).values,
        p_min_pu=ror_inflow_p_max_pu[ror_nodes["node_id"]].to_numpy(),
        p_max_pu=ror_inflow_p_max_pu[ror_nodes["node_id"]].to_numpy(),
        marginal_cost=np.zeros(len(ror_nodes)),
```

```
        carrier=["ror_inflow"] * len(ror_nodes)
    )

    # Turbine path from run-of-river water bus to electrical bus
    ror_names = [f"ROR_{nid}" for nid in ror_nodes["node_id"].values]
    n.add(
        "Link",
        ror_names,
        bus0=ror_nodes["ror_water_bus"].values,
        bus1=ror_nodes["node_id"].values,
        p_nom=ror_nodes["ror_capacity_MW"].clip(lower=MIN_P_NOM).astype(float).values,
        efficiency=np.ones(len(ror_nodes)),
        marginal_cost=np.full(len(ror_nodes), ROR_MARGINAL_COST_FIXED),
        carrier=["ror_turbine"] * len(ror_nodes)
    )

    # Spill path for run-of-river inflow that cannot be turbinated
    ror_spill_names = [f"SPILL_ROR_{nid}" for nid in ror_nodes["node_id"].values]
    n.add(
        "Link",
        ror_spill_names,
        bus0=ror_nodes["ror_water_bus"].values,
        bus1=["SPILL_SINK"] * len(ror_nodes),
        p_nom=np.full(len(ror_nodes), 1e9),
        efficiency=np.ones(len(ror_nodes)),
        marginal_cost=np.full(len(ror_nodes), ROR_SPILL_COST),
        carrier=["ror_spill"] * len(ror_nodes)
    )
```

Description: This excerpt shows the explicit hydraulic representation of run-of-river hydropower. Unlike reservoir hydropower, run-of-river inflow is not connected to an intertemporal Store. It enters the model through an inflow Generator and must either pass through a turbine Link during the same period or be released through a spill Link. This implementation supports the distinction made in the results between reservoir spill and run-of-river spill.
Source: Author's Python/PyPSA implementation.

Appendix 3.11.6. State-dependent segmented turbine offers.

Listing 3.11f. Reservoir turbine marginal costs by fill level.

```
def get_segment_cost_vector(zone: str, fill_ratio: float) -> np.ndarray:
    zmin = float(ZONE_COST_MIN[zone])
    zmax = float(ZONE_COST_MAX[zone])
    spread = zmax - zmin

    if fill_ratio > HIGH_FILL_THRESHOLD:
        low_cost = zmin
        high_cost = 0.55 * zmax
    elif fill_ratio > LOW_FILL_THRESHOLD:
        low_cost = zmin + 0.15 * spread
        high_cost = 0.80 * zmax
    else:
        low_cost = zmin + 0.35 * spread
        high_cost = zmax

    return np.linspace(low_cost, high_cost, NUM_TURBINE_SEGMENTS, dtype=float)
```

```
def build_segmented_turbine_marginal_cost_timeseries(
    window_snaps, hydro_df, turb_segments_df, current_store_e
):
    node_meta = hydro_df[["node_id", "zone", "reservoir_e_nom_mwh"]].copy()
    node_meta["store_name"] = "RES_" + node_meta["node_id"].astype(str)
    node_meta["e_init"] = node_meta["store_name"].map(current_store_e).fillna(0.0)

    node_meta["fill_ratio"] = (
        node_meta["e_init"] /
        node_meta["reservoir_e_nom_mwh"].clip(lower=1e-9)
    ).clip(0.0, 1.0)

    segment_cost_by_name = {}

    for row in node_meta.itertuples(index=False):
        seg_costs = get_segment_cost_vector(row.zone, float(row.fill_ratio))

        for seg_label, seg_cost in zip(TURBINE_SEGMENT_LABELS, seg_costs):
            segment_cost_by_name[f"TURB_{row.node_id}_{seg_label}"] = float(seg_cost)

    ordered_costs = (
        pd.Series(segment_cost_by_name)
        .reindex(turb_segments_df["segment_name"])
        .astype(float)
        .values
    )

    cost_matrix = np.tile(ordered_costs, (len(window_snaps), 1))

    return pd.DataFrame(
        cost_matrix,
        index=pd.Index(window_snaps),
        columns=turb_segments_df["segment_name"]
    )
```

Description: This code block shows how reservoir turbine offers are made dependent on reservoir fill levels. Each reservoir turbine is divided into ten equal segments. Segment costs are lower when reservoirs are relatively full and higher when reservoirs are low. This provides a tractable piecewise-linear approximation of the increasing opportunity cost of stored water.
Source: Author’s Python/PyPSA implementation.

Appendix 3.11.7. Terminal water-value approximation.

Listing 3.11g. State- and forecast-dependent terminal water values.

```
def compute_window_endogenous_water_values(
    window_snaps, current_store_e, hydro_df, inflow_res_df, load_df
):
    out = hydro_df[[
        "node_id", "zone", "turbine_p_nom_mw", "reservoir_e_nom_mwh"
    ]].copy()

    out["store_name"] = "RES_" + out["node_id"].astype(str)

    e_nom = out.set_index("store_name")["reservoir_e_nom_mwh"].astype(float)
    e_init = current_store_e.reindex(out["store_name"]).astype(float)
```

```
out["fill0_ratio"] = (
    e_init.values / e_nom.values
).clip(0.0, 1.0)

inflow_forecast = (
    inflow_res_df.loc[window_snaps, out["node_id"]]
    .sum(axis=0)
    .reindex(out["node_id"])
    .astype(float)
    .values
)

out["forecast_inflow_mwh"] = inflow_forecast
out["forecast_inflow_ratio"] = np.minimum(inflow_forecast / e_nom.values, 1.0)

zone_turbine_tot = out.groupby("zone")["turbine_p_nom_mw"].sum().to_dict()

zone_demand = {}
for z in sorted(out["zone"].unique()):
    zone_nodes = hydro_df.loc[hydro_df["zone"] == z, "node_id"].tolist()
    zone_demand[z] = float(load_df.loc[window_snaps, zone_nodes].sum().sum())

expected_use_ratio = []

for row in out.itertuples(index=False):
    zone_total_turb = max(zone_turbine_tot.get(row.zone, 0.0), 1e-9)
    turb_share = float(row.turbine_p_nom_mw) / zone_total_turb

    node_expected_use = (
        TERMINAL_VALUE_ZONE_LOAD_WEIGHT *
        zone_demand.get(row.zone, 0.0) *
        turb_share
    )

    expected_use_ratio.append(
        min(node_expected_use / max(row.reservoir_e_nom_mwh, 1e-9), 1.0)
    )

out["expected_use_ratio"] = np.array(expected_use_ratio, dtype=float)

scarcity_score = np.clip(
    TERMINAL_VALUE_FILL_WEIGHT * (1.0 - out["fill0_ratio"])
    + TERMINAL_VALUE_LOAD_WEIGHT * out["expected_use_ratio"]
    - TERMINAL_VALUE_INFLOW_WEIGHT * out["forecast_inflow_ratio"],
    0.0,
    1.0,
)

out["scarcity_score"] = scarcity_score

out["base_terminal_water_value"] = np.clip(
    TERMINAL_VALUE_BASE_MIN
    + TERMINAL_VALUE_SCARCITY_COEF * scarcity_score,
    TERMINAL_VALUE_BASE_MIN,
    TERMINAL_VALUE_BASE_MAX,
)
```

```
    return out
```

Description: This excerpt shows how the model constructs a state- and forecast-dependent terminal water value for each reservoir node. The terminal value increases when the reservoir starts the rolling window at a lower fill level or when expected demand pressure is higher. It decreases when larger reservoir inflows are expected over the look-ahead window. This mechanism reduces the tendency of the model to overuse water at the end of each optimization window.
Source: Author’s Python/PyPSA implementation.

Appendix 3.11.8. Piecewise-linear terminal storage reward.

Listing 3.11h. Terminal storage value added to the optimization objective.

```
def extra_functionality_factory(
    res_names, hydro_nodes, apply_terminal_penalty, dynamic_wv_df=None
):
    hydro_term = hydro_nodes[["node_id", "e_final_min_mwh"]].copy()
    hydro_term["store_name"] = "RES_" + hydro_term["node_id"].astype(str)

    hydro_term = (
        hydro_term[["store_name", "e_final_min_mwh"]]
        .drop_duplicates("store_name")
        .set_index("store_name")
        .reindex(res_names)
    )

    def extra_functionality(network, snapshots_):
        if (not apply_terminal_penalty) and (not USE_ENDOGENOUS_TERMINAL_VALUE):
            return

        m = network.model
        last_snapshot = snapshots_[-1]
        store_e = m.variables["Store-e"]
        e_last = store_e.loc[last_snapshot, res_names]
        store_dim = e_last.dims[0]

        if USE_ENDOGENOUS_TERMINAL_VALUE and dynamic_wv_df is not None:
            wv_lookup = dynamic_wv_df.set_index("store_name").reindex(res_names)

            seg_ids = list(range(TERMINAL_VALUE_NUM_SEGMENTS))
            store_coord = xr.DataArray(
                res_names,
                coords={store_dim: res_names},
                dims=(store_dim,)
            )

            base_vals = wv_lookup["base_terminal_water_value"].astype(float).values
            multipliers = np.array(TERMINAL_VALUE_SEGMENT_MULTIPLIERS, dtype=float)

            slopes = np.outer(base_vals, multipliers)
```

```
        seg_caps = np.outer(
            wv_lookup["reservoir_e_nom_mwh"].astype(float).values
            / float(TERMINAL_VALUE_NUM_SEGMENTS),
            np.ones(TERMINAL_VALUE_NUM_SEGMENTS)
        )

        slope_da = xr.DataArray(
            slopes,
            coords={store_dim: res_names, "segment": seg_ids},
            dims=(store_dim, "segment")
        )

        seg_cap_da = xr.DataArray(
            seg_caps,
            coords={store_dim: res_names, "segment": seg_ids},
            dims=(store_dim, "segment")
        )

        terminal_segments = m.add_variables(
            lower=0.0,
            upper=seg_cap_da,
            coords=[store_coord, xr.DataArray(seg_ids, dims=("segment",))],
            name="terminal_water_value_segment"
        )

        m.add_constraints(
            terminal_segments.sum(dim="segment") <= e_last,
            name="terminal_water_value_fill"
        )

        m.objective += -(terminal_segments * slope_da).sum()

    if apply_terminal_penalty:
        shortfall = m.add_variables(
            lower=0.0,
            coords=[xr.DataArray(res_names, coords={store_dim: res_names}, dims=(store_dim,))],
            name="terminal_storage_shortfall"
        )

        rhs = xr.DataArray(
            hydro_term["e_final_min_mwh"].astype(float).values,
            coords={store_dim: res_names},
            dims=(store_dim,)
        )

        m.add_constraints(
            e_last + shortfall >= rhs,
            name="soft_min_final_reservoir_energy"
        )

        m.objective += TERMINAL_STORAGE_SHORTFALL_COST * shortfall.sum()

  return extra_functionality
```

Description: This excerpt shows how the terminal value of stored water is introduced into the optimization problem. End-of-window storage is assigned to several storage-value segments with decreasing marginal values.

Because the model minimizes cost, this value enters the objective as a negative cost, rewarding the model for preserving water at the end of the window. On the final rolling window, a soft minimum final storage condition is also imposed through a penalized shortfall variable.
Source: Author’s Python/PyPSA implementation.

Appendix 3.11.9. Rolling-horizon implementation and storage carry-over.

Listing 3.11i. Manual rolling-horizon solution loop.

```
all_snapshots = list(n.snapshots)
step = ROLLING_STEP_HOURS
lookahead = ROLLING_LOOKAHEAD_HOURS

rolling_results = []
water_value_records = []
rolling_nodal_prices = []
rolling_store_energy = []
rolling_line_flows = []
rolling_generator_dispatch = []

previous_final_store_e = n.stores.loc[res_names, "e_initial"].copy()

for start_idx in range(0, len(all_snapshots), step):
    end_idx = min(start_idx + lookahead, len(all_snapshots))
    window_snaps = all_snapshots[start_idx:end_idx]

    implemented_end_idx = min(start_idx + step, len(all_snapshots))
    implemented_snaps = all_snapshots[start_idx:implemented_end_idx]

    if len(window_snaps) == 0 or len(implemented_snaps) == 0:
        continue

    implemented_last_snapshot = implemented_snaps[-1]
    is_last_window = implemented_end_idx == len(all_snapshots)

    if start_idx > 0:
        n.stores.loc[res_names, "e_initial"] = (
            previous_final_store_e.reindex(res_names).astype(float).values
        )

    dynamic_wv_df = compute_window_endogenous_water_values(
        window_snaps=window_snaps,
        current_store_e=n.stores.loc[res_names, "e_initial"].copy(),
        hydro_df=hydro_nodes,
        inflow_res_df=inflow_res_for_hydro,
```

```
    load_df=load_p_set,
  )

  extra_functionality = extra_functionality_factory(
    res_names=res_names,
    hydro_nodes=hydro_nodes,
    apply_terminal_penalty=is_last_window,
    dynamic_wv_df=dynamic_wv_df,
  )

  status, condition = n.optimize(
    snapshots=window_snaps,
    extra_functionality=extra_functionality,
    **LINEAR_OPF_KWARGS
  )

  implemented_store_e = n.stores_t.e.loc[implemented_snaps, res_names].copy()
  implemented_prices = n.buses_t.marginal_price.loc[
    implemented_snaps, nodes["node_id"]
  ].copy()

  implemented_line_flows = n.lines_t.p0.loc[
    implemented_snaps, n.lines.index
  ].copy()

  previous_final_store_e = implemented_store_e.loc[
    implemented_last_snapshot
  ].copy()

  rolling_store_energy.append(implemented_store_e)
  rolling_nodal_prices.append(implemented_prices)
  rolling_line_flows.append(implemented_line_flows)

  rolling_results.append({
    "implemented_start": implemented_snaps[0],
    "implemented_end": implemented_last_snapshot,
    "status": status,
    "condition": condition,
    "final_store_energy_mwh": float(previous_final_store_e.sum())
  })
```

Description: This excerpt illustrates the rolling-horizon logic used in the model. Each optimization covers a 48-hour window, but only the first 24 hours are implemented. The final reservoir storage state of the implemented day is then used as the initial storage state for the next rolling window. This preserves chronological continuity while avoiding the computational burden of solving the full year as a single optimization problem.
Source: Author's Python/PyPSA implementation.

Appendix 3.11.10. Zonal aggregation for market clearing.

Listing 3.11j. Aggregation from nodal inputs to five bidding zones.

```
# Map each modelling node to its bidding zone
zone_map = nodes.set_index("node_id")["zone"].to_dict()

# Aggregate nodal demand to bidding zones
load_p_set_zonal = (
  load_p_set_nodal
```

```
    .rename(columns=zone_map)
    .T.groupby(level=0).sum().T
    .reindex(columns=ZONES)
    .fillna(0.0)
)

# Aggregate reservoir inflows to bidding zones
inflow_res_zone = (
    inflow_res_wide
    .rename(columns=zone_map)
    .T.groupby(level=0).sum().T
    .reindex(columns=ZONES)
    .fillna(0.0)
)

# Aggregate run-of-river inflows to bidding zones
inflow_ror_zone = (
    inflow_ror_wide
    .rename(columns=zone_map)
    .T.groupby(level=0).sum().T
    .reindex(columns=ZONES)
    .fillna(0.0)
)

# Aggregate reservoir capacity and storage to bidding zones
zone_hydro = (
    hydro_nodes.groupby("zone")[["turbine_p_nom_mw", "reservoir_e_nom_mwh"]]
    .sum()
    .reset_index()
)

zone_hydro["store_name"] = "RES_" + zone_hydro["zone"]
zone_hydro["water_bus"] = "WATER_" + zone_hydro["zone"]
zone_hydro["soc_initial_fraction"] = zone_hydro["zone"].map(INITIAL_SOC_BY_ZONE)

zone_hydro["e_initial_mwh"] = (
    zone_hydro["soc_initial_fraction"] *
    zone_hydro["reservoir_e_nom_mwh"]
)

# Build interzonal corridors from physical nodal lines
node_zone = nodes.set_index("node_id")["zone"]
lines["zone0"] = lines["bus0"].map(node_zone)
lines["zone1"] = lines["bus1"].map(node_zone)

interzonal_lines = lines.loc[lines["zone0"] != lines["zone1"]].copy()

interzonal_lines["z0"] = interzonal_lines[["zone0", "zone1"]].min(axis=1)
interzonal_lines["z1"] = interzonal_lines[["zone0", "zone1"]].max(axis=1)

zonal_corridors = (
    interzonal_lines
    .groupby(["z0", "z1"], as_index=False)["s_nom"]
    .sum()
)

zonal_corridors["line_name"] = zonal_corridors.apply(
```

```
    lambda r: f"ZL_{r['z0']}_{r['z1']}",
    axis=1
)

zonal_corridors["x"] = 1.0
```

Description: This excerpt shows how the zonal market-clearing stage is constructed. Nodal demand, reservoir inflows, run-of-river inflows, turbine capacity and reservoir storage are aggregated to the five Norwegian bidding zones. Only interzonal corridors are retained, while intra-zonal constraints are not represented at the market-clearing stage. This code implements the central simplification of the zonal market design tested in the thesis.
Source: Author’s Python/PyPSA implementation.

Appendix 3.11.11. Nodal redispatch against zonal market schedules.

Listing 3.11k. Redispatch variables and zonal generation-target constraints.

```
def nodal_redispatch_extra_factory(
    target_res_dispatch,
    target_ror_dispatch,
    target_zone_end_storage_eod
):
    res_turb_by_zone = {
        z: [
            nm
            for nid in zone_to_res_nodes.get(z, [])
            for nm in node_turb_link_names.get(nid, [])
        ]
        for z in ZONES
    }

    ror_turb_by_zone = {
        z: [f"ROR_{nid}" for nid in zone_to_ror_nodes.get(z, [])]
        for z in ZONES
    }

    def extra(network, snapshots_):
        m = network.model
        link_p = m.variables["Link-p"]
        store_e = m.variables["Store-e"]

        snap_dim = link_p.dims[0]
        link_dim = link_p.dims[1]

        snap_coord = xr.DataArray(
            list(snapshots_),
            coords={snap_dim: list(snapshots_)},
            dims=(snap_dim,)
        )

        zone_coord = xr.DataArray(
            ZONES,
            dims=("zone",),
            coords={"zone": ZONES}
        )

        up_res = m.add_variables(
            lower=0.0,
```

```
    coords=[snap_coord, zone_coord],
    name="redispatch_up_res"
)

down_res = m.add_variables(
    lower=0.0,
    coords=[snap_coord, zone_coord],
    name="redispatch_down_res"
)

up_ror = m.add_variables(
    lower=0.0,
    coords=[snap_coord, zone_coord],
    name="redispatch_up_ror"
)

down_ror = m.add_variables(
    lower=0.0,
    coords=[snap_coord, zone_coord],
    name="redispatch_down_ror"
)

for z in ZONES:
    res_links = [
        nm for nm in res_turb_by_zone.get(z, [])
        if nm in network.links.index
    ]

    if len(res_links) > 0:
        actual_res = link_p.loc[list(snapshots_), res_links].sum(dim=link_dim)
    else:
        actual_res = xr.DataArray(
            np.zeros(len(snapshots_)),
            coords={snap_dim: list(snapshots_)},
            dims=(snap_dim,)
        )

    target_res_da = xr.DataArray(
        target_res_dispatch
        .reindex(index=list(snapshots_), columns=ZONES)
        .fillna(0.0)[z]
        .to_numpy(),
        coords={snap_dim: list(snapshots_)},
        dims=(snap_dim,)
    )

    m.add_constraints(
        actual_res - up_res.loc[:, z] + down_res.loc[:, z]
        == target_res_da,
        name=f"redispatch_res_balance_{z}"
    )

    ror_links = [
        nm for nm in ror_turb_by_zone.get(z, [])
        if nm in network.links.index
    ]
```

```
        if len(ror_links) > 0:
            actual_ror = link_p.loc[list(snapshots_), ror_links].sum(dim=link_dim)
        else:
            actual_ror = xr.DataArray(
                np.zeros(len(snapshots_)),
                coords={snap_dim: list(snapshots_)},
                dims=(snap_dim,)
            )

        target_ror_da = xr.DataArray(
            target_ror_dispatch
            .reindex(index=list(snapshots_), columns=ZONES)
            .fillna(0.0)[z]
            .to_numpy(),
            coords={snap_dim: list(snapshots_)},
            dims=(snap_dim,)
        )

        m.add_constraints(
            actual_ror - up_ror.loc[:, z] + down_ror.loc[:, z]
            == target_ror_da,
            name=f"redispatch_ror_balance_{z}"
        )

    m.objective += REDISPATCH_UP_COST_RES * up_res.sum()
    m.objective += REDISPATCH_DOWN_COST_RES * down_res.sum()
    m.objective += REDISPATCH_UP_COST_ROR * up_ror.sum()
    m.objective += REDISPATCH_DOWN_COST_ROR * down_ror.sum()

  return extra
```

Description: This excerpt implements the generation-tracking component of the nodal redispatch stage. Reservoir and run-of-river generation after redispatch are compared with the zonal market schedules. Upward and downward redispatch variables measure the deviations required to restore feasibility on the physical nodal network. These deviations are penalized in the objective function, which keeps the redispatch outcome linked to the previous zonal market-clearing result.
Source: Author’s Python/PyPSA implementation.

Appendix 3.11.12. Storage-target tracking in redispatch.
Listing 3.11l. Soft end-of-day zonal storage target constraints.

```
# Inside the nodal redispatch extra-functionality block

last_snapshot = snapshots_[-1]
store_dim = store_e.dims[1]

pos_dev = m.add_variables(
  lower=0.0,
  coords=[zone_coord],
  name="redispatch_storage_pos"
)

neg_dev = m.add_variables(
  lower=0.0,
  coords=[zone_coord],
  name="redispatch_storage_neg"
```

```
)

e_last = store_e.loc[last_snapshot, :]

for z in ZONES:
    zstores = [
        nm for nm in zone_to_res_store_names.get(z, [])
        if nm in network.stores.index
    ]

    if len(zstores) > 0:
        zone_storage = e_last.loc[zstores].sum(dim=store_dim)
    else:
        zone_storage = xr.DataArray(0.0)

    target_val = float(target_zone_end_storage_eod.get(z, 0.0))

    m.add_constraints(
        zone_storage + pos_dev.loc[z] - neg_dev.loc[z] == target_val,
        name=f"redispatch_zone_storage_target_{z}"
    )

m.objective += REDISPATCH_STORAGE_DEV_COST * (
    pos_dev.sum() + neg_dev.sum()
)
```

Description: This excerpt shows how end-of-day zonal storage targets from the zonal market stage are tracked in the nodal redispatch stage. Positive and negative deviation variables allow the redispatch solution to deviate from the zonal storage target only at a cost. This prevents the redispatch stage from restoring physical feasibility by freely changing reservoir storage trajectories without recording and penalizing the deviation. This code is directly linked to the storage-target gap indicators reported in the results.
Source: Author's Python/PyPSA implementation.

Appendix 3.11.13. Load shedding as a high-penalty feasibility variable.

Listing 3.11m. Load-shedding generators.

```
n.optimize.add_load_shedding(
    buses=n.buses.index[n.buses.carrier == "AC"],
    marginal_cost=LOAD_SHEDDING_COST,
    p_nom=MAX_LOAD_SHEDDING_P_NOM,
    sign=1.0,
    suffix="_load_shedding"
)

ls_names = n.generators.index[
    n.generators.index.str.contains("_load_shedding", regex=False)
]

if len(ls_names) > 0:
    n.generators.loc[ls_names, "carrier"] = "load_shedding"
```

Description: This excerpt shows how load shedding is added as a high-penalty feasibility option. It allows the model to remain solvable when demand cannot be served under the imposed hydropower and network constraints. Because the penalty is high, load shedding is used only when the model cannot satisfy demand through available hydropower generation and transmission. In the thesis, this variable is interpreted as a model-internal indicator of residual scarcity, not as a prediction of real-world reliability problems.

Source: Author's Python/PyPSA implementation.

Note: *Load shedding is included as a high-penalty feasibility variable. It is interpreted as a model-internal residual scarcity indicator, not as a real-world blackout forecast.*

Appendix 3.11.14. Water-balance and diagnostic checks.

Listing 3.11n. Reservoir and run-of-river balance checks.

```
reservoir_balance_df = pd.DataFrame({
    "node_id": hydro_nodes["node_id"].values,
    "initial_storage_mwh": initial_storage_by_node
        .reindex(hydro_nodes["node_id"])
        .fillna(0.0)
        .values,
    "inflow_mwh": inflow_res_by_node
        .reindex(hydro_nodes["node_id"])
        .fillna(0.0)
        .values,
    "turbine_generation_mwh": turbine_gen_by_node
        .reindex(hydro_nodes["node_id"])
        .fillna(0.0)
        .values,
    "spill_mwh": spill_by_node
        .reindex(hydro_nodes["node_id"])
        .fillna(0.0)
        .values,
    "final_storage_mwh": final_storage_by_node
        .reindex(hydro_nodes["node_id"])
        .fillna(0.0)
        .values,
})

reservoir_balance_df["balance_error_mwh"] = (
    reservoir_balance_df["initial_storage_mwh"]
    + reservoir_balance_df["inflow_mwh"]
    - reservoir_balance_df["turbine_generation_mwh"]
    - reservoir_balance_df["spill_mwh"]
    - reservoir_balance_df["final_storage_mwh"]
)

print_check(
    "Reservoir balance max abs error (MWh)",
    reservoir_balance_df["balance_error_mwh"].abs().max(),
    MEDIUM_TOL,
    "exact"
)

ror_summary["ror_balance_error_mwh"] = (
    ror_summary["inflow_mwh"]
    - ror_summary["generated_mwh"]
    - ror_summary["spill_mwh"]
)

print_check(
    "ROR balance max abs error (MWh)",
    ror_summary["ror_balance_error_mwh"].abs().max(),
```

```
    MEDIUM_TOL,
    "exact"
)
```

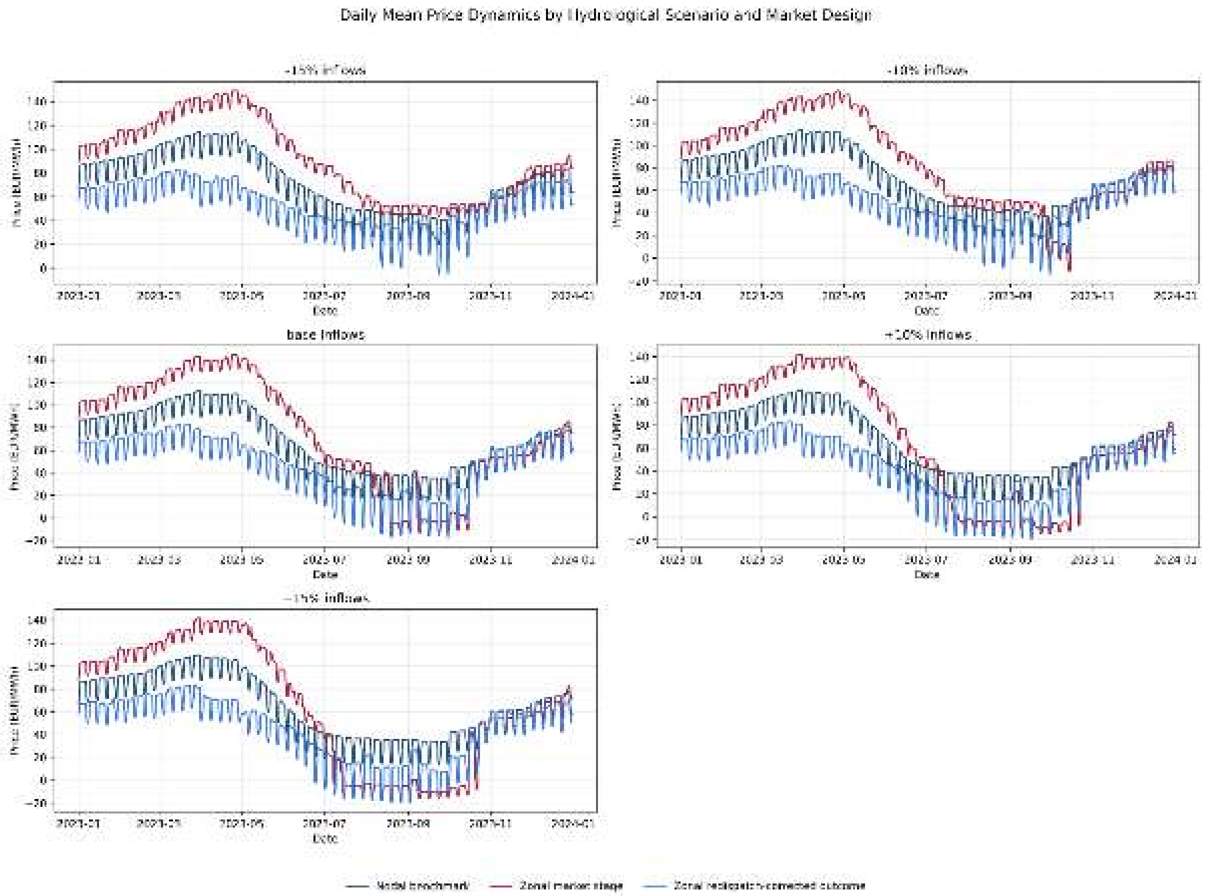
Description: This excerpt shows the internal water-balance diagnostics used to verify the consistency of the simulation outputs. For reservoir hydropower, final storage must equal initial storage plus reservoir inflow minus turbine generation and spill. For run-of-river hydropower, inflow must equal generation plus spill because no intertemporal storage is available. These checks ensure that the model respects the energy-equivalent water-balance logic before the results are interpreted.

Source: Author's Python/PyPSA implementation.

Note: *Balance errors are used as numerical diagnostics to verify internal consistency before interpreting scenario results.*

Appendix 4.1. Daily mean price dynamics by hydrological scenario and market design.

Figure 4.1a. Daily Mean Price Dynamics by Hydrological Scenario and Market Design.



Description: This figure compares daily mean electricity prices across the five hydrological inflow scenarios and the three market-design representations used in the model: the nodal benchmark, the zonal market stage, and the redispatch-corrected outcome. Prices are expressed in EUR/MWh. The figure highlights how hydrological conditions influence the level and seasonal evolution of price signals, as well as the differences that emerge between market designs. The x-axis reports the simulation date in 2023. The y-axis reports daily mean electricity prices in EUR/MWh.

Source: Author's calculations from Python/PyPSA hourly price outputs, aggregated to daily mean prices.

Appendix 4.2. Price statistics and spatial price spreads by hydrological scenario and market design.

Table 4.2a. Zone-level price statistics by hydrological scenario and market design.

Scenario	Design	Zone	Mean (EUR/MWh)	P95 (EUR/MWh)	Min. (EUR/MWh)	Max. (EUR/MWh)
-15%	Nodal	NO1	56.91	97.91	19.3	100.17
-15%	Nodal	NO2	68.82	116.63	19.3	120.87
-15%	Nodal	NO3	58.04	97.9	19.3	100.16
-15%	Nodal	NO4	63.09	108.85	19.3	112.48
-15%	Nodal	NO5	105.84	139.3	19.3	142.73
-15%	Zonal	NO1	90.82	144.72	43.34	149.43
-15%	Zonal	NO2	90.82	144.72	43.34	149.43
-15%	Zonal	NO3	91.17	144.72	43.34	149.43
-15%	Zonal	NO4	90.93	144.72	43.34	149.43
-15%	Zonal	NO5	90.82	144.72	43.34	149.43
-15%	Redispatch	NO1	34.81	59.18	-6.39	63.54
-15%	Redispatch	NO2	48.91	86.57	-6.39	94.43
-15%	Redispatch	NO3	37.83	58.84	-5.77	63.22
-15%	Redispatch	NO4	44.27	72.18	-2.77	76.56
-15%	Redispatch	NO5	92.18	119.57	-6.39	125.12
-10%	Nodal	NO1	54.82	96.94	16.57	98.67
-10%	Nodal	NO2	66.78	115.63	16.57	119.96
-10%	Nodal	NO3	55.7	96.94	16.57	98.67
-10%	Nodal	NO4	60.78	108.29	16.57	111.21
-10%	Nodal	NO5	104.45	138.49	16.57	142.06
-10%	Zonal	NO1	84.74	143.25	-11.16	148.72
-10%	Zonal	NO2	84.74	143.25	-11.16	148.72
-10%	Zonal	NO3	85.91	143.25	-10.53	148.72
-10%	Zonal	NO4	85.64	143.25	-10.53	148.72
-10%	Zonal	NO5	84.74	143.25	-11.16	148.72
-10%	Redispatch	NO1	34.14	58.17	-18.43	63.23
-10%	Redispatch	NO2	47.7	86.43	-18.43	94.13
-10%	Redispatch	NO3	36.94	57.17	-10.25	63.05
-10%	Redispatch	NO4	43.48	75.89	-7.1	79.96

Scenario	Design	Zone	Mean (EUR/MWh)	P95 (EUR/MWh)	Min. (EUR/MWh)	Max. (EUR/MWh)
-10%	Redispatch	NO5	91.24	120.48	-18.43	124.85
base	Nodal	NO1	51.76	94.57	15.42	97.47
base	Nodal	NO2	63.73	113.44	15.42	119.14
base	Nodal	NO3	52.19	94.57	15.42	97.47
base	Nodal	NO4	57.37	107.08	15.42	109.03
base	Nodal	NO5	102.39	137.29	15.42	141.46
base	Zonal	NO1	74.27	139.81	-13.55	144.88
base	Zonal	NO2	74.27	139.81	-13.55	144.88
base	Zonal	NO3	74.88	139.81	-13.55	144.88
base	Zonal	NO4	75.59	139.74	-13.55	144.88
base	Zonal	NO5	74.27	139.81	-13.55	144.88
base	Redispatch	NO1	29.7	58.02	-18.74	64.16
base	Redispatch	NO2	43.81	86.8	-18.74	94.55
base	Redispatch	NO3	32.58	54.97	-17.9	63.97
base	Redispatch	NO4	39.47	70.31	-16.86	77.06
base	Redispatch	NO5	87.99	119.81	-18.74	125.19
+10%	Nodal	NO1	49.59	92.74	14.32	95.18
+10%	Nodal	NO2	61.57	112.4	14.32	116.57
+10%	Nodal	NO3	49.72	92.74	14.32	95.14
+10%	Nodal	NO4	54.97	104.57	14.32	107.2
+10%	Nodal	NO5	100.96	136.81	14.32	140.36
+10%	Zonal	NO1	66.9	138.77	-15.53	142.08
+10%	Zonal	NO2	66.9	138.77	-15.53	142.08
+10%	Zonal	NO3	67.39	138.77	-15.53	142.08
+10%	Zonal	NO4	68.14	138.77	-15.53	142.08
+10%	Zonal	NO5	66.9	138.77	-15.53	142.08
+10%	Redispatch	NO1	26.85	58.7	-19.44	64.6
+10%	Redispatch	NO2	40.94	87.07	-19.44	92.26
+10%	Redispatch	NO3	29.27	58.39	-19.31	64.6
+10%	Redispatch	NO4	36.38	69.29	-18.74	77.09
+10%	Redispatch	NO5	86.17	120.41	-19.44	124.89
+15%	Nodal	NO1	48.7	91.61	13.35	94.39

Scenario	Design	Zone	Mean (EUR/MWh)	P95 (EUR/MWh)	Min. (EUR/MWh)	Max. (EUR/MWh)
+15%	Nodal	NO2	60.7	111.8	13.35	116.1
+15%	Nodal	NO3	48.74	91.61	13.35	94.38
+15%	Nodal	NO4	53.99	104.08	13.35	106.13
+15%	Nodal	NO5	100.36	136.35	13.35	140.05
+15%	Zonal	NO1	64.56	138.34	-15.58	141.72
+15%	Zonal	NO2	64.56	138.34	-15.58	141.72
+15%	Zonal	NO3	64.84	138.34	-15.58	141.72
+15%	Zonal	NO4	65.72	138.27	-15.58	141.72
+15%	Zonal	NO5	64.56	138.34	-15.58	141.72
+15%	Redispatch	NO1	25.63	58.69	-19.4	64.46
+15%	Redispatch	NO2	39.83	84.73	-19.4	94.21
+15%	Redispatch	NO3	27.57	57.58	-19.33	64.5
+15%	Redispatch	NO4	34.57	69.26	-19.05	77.23
+15%	Redispatch	NO5	85.44	120.17	-19.4	125.34

Description: This table reports zone-level price statistics for each hydrological inflow scenario and market design. It compares the mean, 95th percentile, minimum and maximum hourly prices across the five Norwegian bidding zones. The table is used to show how price levels and extreme price values differ between the nodal benchmark, the zonal market stage and the redispatch-corrected outcome under changing hydrological-conditions.

Source: Author’s calculations from Python/PyPSA hourly price outputs.

Table 4.2b. Spatial price spread by hydrological scenario and market design.

Scenario	Design	Avg. price (EUR/MWh)	Lowest zone	Highest zone	Spread lowest and highest (EUR/MWh)
-15%	Nodal	70.54	NO1	NO5	48.93
-15%	Zonal	90.91	NO1	NO3	0.35
-15%	Redispatch	51.6	NO1	NO5	57.37
-10%	Nodal	68.51	NO1	NO5	49.63
-10%	Zonal	85.15	NO1	NO3	1.17
-10%	Redispatch	50.7	NO1	NO5	57.1
base	Nodal	65.49	NO1	NO5	50.63
base	Zonal	74.66	NO1	NO4	1.32
base	Redispatch	46.71	NO1	NO5	58.29
+10%	Nodal	63.36	NO1	NO5	51.37
+10%	Zonal	67.25	NO1	NO4	1.24
+10%	Redispatch	43.92	NO1	NO5	59.32
+15%	Nodal	62.5	NO1	NO5	51.66
+15%	Zonal	64.85	NO1	NO4	1.16
+15%	Redispatch	42.61	NO1	NO5	59.81

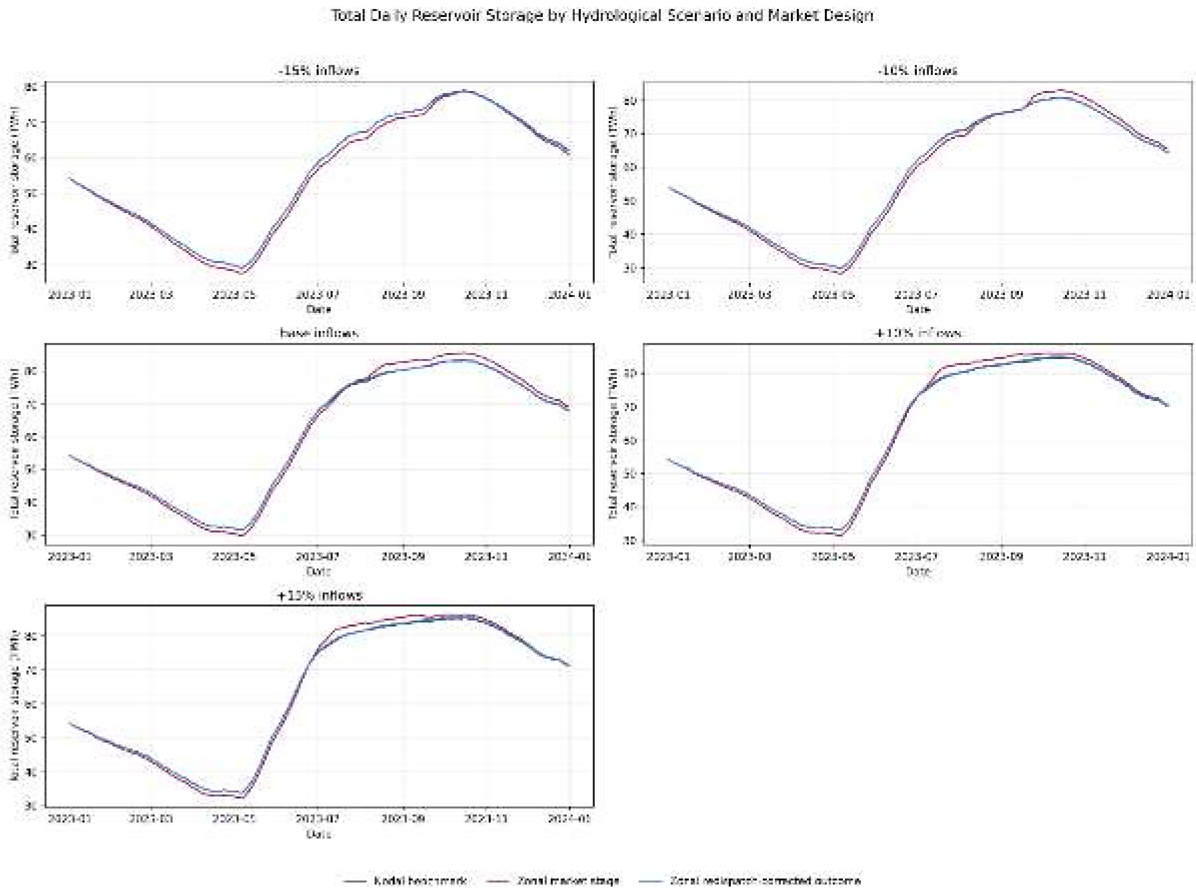
Description: This table summarizes the spatial dispersion of prices across zones for each hydrological scenario and market design. It reports the average price, the lowest-price zone, the highest-price zone and the resulting price spread. The table helps assess whether each market design produces spatially differentiated price signals or, conversely, smooths price differences across zones.

Source: Author's calculations from Python/PyPSA hourly price outputs aggregated by bidding zone.

***Note:** *The spread measures spatial price dispersion across modelled zones. It does not measure price volatility over time.*

Appendix 4.3. Total Daily Reservoir Storage by Hydrological Scenario and Market Design.

Figure 4.3a. Total Daily Reservoir Storage by Hydrological Scenario and Market Design.



Description: This figure shows the evolution of aggregate daily reservoir storage over the simulation year for each hydrological inflow scenario and market-design representation. Storage is expressed in TWh and aggregated across all modelled reservoir hydropower units. The figure illustrates how reservoir storage changes under lower, baseline and higher inflow assumptions, and whether the different market designs produce distinct storage trajectories.

Source: Author’s calculations from Python/PyPSA reservoir-storage outputs, aggregated across modelled reservoir hydropower units.

***Note:** *Storage is expressed in energy-equivalent units and should be interpreted as modelled reservoir energy storage, not as physical water volume.*

.

Appendix 4.4. Annual reservoir storage indicators by hydrological scenario and market design.

Table 4.4a. Reservoir storage summary by hydrological scenario and market design.

Scenario	Design	Final (TWh)	Min. (TWh)	Date min.	Max. (TWh)	Date max.	Drawdown (TWh)	Refill (TWh)	End-year drawdown (TWh)
-15%	Nodal	62.113	29.105	2023-05-07	78.904	2023-10-15	24.998	49.8	16.791
-15%	Zonal	60.93	27.461	2023-05-07	78.912	2023-10-15	26.639	51.45	17.982
-15%	Redispatch	61.881	29.139	2023-05-07	78.675	2023-10-15	24.964	49.537	16.794
-10%	Nodal	64.54	29.895	2023-05-07	80.841	2023-10-15	24.213	50.947	16.302
-10%	Zonal	65.48	28.259	2023-05-07	82.959	2023-10-15	25.847	54.7	17.478
-10%	Redispatch	64.427	29.93	2023-05-07	80.722	2023-10-15	24.178	50.792	16.295
base	Nodal	67.957	31.474	2023-05-07	83.319	2023-10-15	22.645	51.844	15.361
base	Zonal	69.131	29.854	2023-05-07	85.597	2023-10-15	24.263	55.743	16.466
base	Redispatch	68.16	31.511	2023-05-07	83.466	2023-10-15	22.608	51.955	15.306
+10%	Nodal	70.301	33.052	2023-05-07	84.786	2023-10-15	21.078	51.734	14.485
+10%	Zonal	70.648	31.444	2023-05-07	86.163	2023-10-15	22.683	54.719	15.516
+10%	Redispatch	70.568	33.09	2023-05-07	84.992	2023-10-15	21.039	51.901	14.423
+15%	Nodal	71.105	33.84	2023-05-07	85.159	2023-10-15	20.295	51.319	14.054
+15%	Zonal	71.125	32.239	2023-05-07	86.161	2023-10-15	21.894	53.922	15.036
+15%	Redispatch	71.38	33.879	2023-05-07	85.354	2023-10-15	20.256	51.474	13.974

Description: This table summarizes annual reservoir-storage indicators by hydrological scenario and market design. It reports final, minimum and maximum storage levels, their timing, and the main drawdown/refill metrics. All storage values are expressed in TWh.

Source: Author's calculations from daily reservoir-storage outputs generated by the Python/PyPSA simulations.

Note: *Drawdown and refill indicators are calculated from the modelled daily storage trajectory and are used to compare seasonal reservoir-management patterns across scenarios and market designs.*

Appendix 4.5. Cross-design reservoir storage differences and storage target gaps.

Table 4.5a. Cross-design differences in annual storage levels.

Scenario	Zonal - nodal final	Redispatch - nodal final	Zonal - nodal min.	Redispatch - nodal min.
-15%	-1.183	-0.232	-1.643	0.034
-10%	0.941	-0.113	-1.636	0.035
base	1.173	0.202	-1.621	0.037
+10%	0.347	0.267	-1.608	0.038
+15%	0.02	0.275	-1.601	0.039

Description: This table measures the difference in reservoir storage outcomes between market designs. It compares final and minimum annual storage levels between the zonal market stage, the nodal benchmark and the redispatch-corrected outcome. The table helps identify whether market design leads to structurally higher or lower storage levels at key points of the year. All values are expressed in TWh. Positive values indicate that the first design has a higher storage value than the comparator.

Source: Author's calculations from annual reservoir-storage indicators generated by the Python/PyPSA simulations.

Table 4.5b. Daily storage path and target-gap differences.

Scenario	Mean gap: nodal vs redispatch	Max. gap: nodal vs redispatch	Mean gap: zonal vs redispatch	Mean target gap
-15%	0.08	0.235	1.059	1.756
-10%	0.068	0.234	1.311	1.647
base	0.118	0.472	1.37	1.578
+10%	0.153	0.601	1.162	1.333
+15%	0.156	0.594	1.049	1.216

Description: This table reports daily storage-path differences between the nodal benchmark, the zonal market stage and the redispatch-corrected outcome. It distinguishes between system-level storage-path differences and storage target gaps generated by redispatch. The table is used to show whether redispatch only slightly adjusts reservoir management or whether it creates substantial deviations from the zonal market-stage storage targets. All values are expressed in TWh. Daily gaps are absolute differences in total daily storage between the compared outcomes. Target gaps are gross absolute zonal redispatch deviations, not net system storage differences.

Source: Author's calculations from daily reservoir-storage trajectories and redispatch storage-target outputs.

Appendix 4.6. Zone-level redispatch storage target deviations.

Table 4.6a. Daily redispatch storage target deviations by zone.

Scenario	Zone	Mean gap	Mean abs. gap	Max. positive	Max. negative	Max. abs.	Date max. abs.
-15%	NO1	0.099	0.109	0.346	-0.075	0.346	2023-11-24
-15%	NO2	0.795	0.795	1.804	0	1.804	2023-12-27
-15%	NO3	-0.006	0.028	0.098	-0.109	0.109	2023-08-27
-15%	NO4	-0.042	0.601	0.917	-1.053	1.053	2023-12-01
-15%	NO5	0.197	0.223	0.641	-0.128	0.641	2023-11-14
-10%	NO1	-0.001	0.097	0.318	-0.291	0.318	2023-07-20
-10%	NO2	0.566	0.628	1.376	-0.394	1.376	2023-08-12
-10%	NO3	-0.2	0.219	0.104	-0.803	0.803	2023-10-12
-10%	NO4	-0.069	0.59	0.875	-1.078	1.078	2023-12-31
-10%	NO5	0.016	0.113	0.479	-0.362	0.479	2023-08-07
+10%	NO1	-0.035	0.055	0.121	-0.288	0.288	2023-07-09
+10%	NO2	0.351	0.405	1.335	-0.477	1.335	2023-06-03
+10%	NO3	-0.143	0.15	0.05	-0.982	0.982	2023-07-25
+10%	NO4	-0.242	0.642	0.691	-1.753	1.753	2023-09-19
+10%	NO5	-0.056	0.081	0.173	-0.82	0.82	2023-07-16
+15%	NO1	-0.037	0.051	0.065	-0.337	0.337	2023-06-28
+15%	NO2	0.34	0.381	1.344	-0.522	1.344	2023-06-02
+15%	NO3	-0.127	0.134	0.041	-0.976	0.976	2023-07-14
+15%	NO4	-0.179	0.566	0.691	-1.685	1.685	2023-09-07
+15%	NO5	-0.052	0.084	0.172	-0.9	0.9	2023-07-09
base	NO1	-0.014	0.075	0.303	-0.237	0.303	2023-06-30
base	NO2	0.424	0.479	1.351	-0.46	1.351	2023-06-09
base	NO3	-0.179	0.192	0.057	-0.92	0.92	2023-08-20
base	NO4	-0.309	0.761	0.754	-1.701	1.701	2023-10-15
base	NO5	-0.048	0.072	0.172	-0.516	0.516	2023-08-10

Description: This table reports redispatch-related deviations from zonal storage targets at the bidding-zone level. For each hydrological scenario and zone, it presents the mean deviation, mean absolute deviation, maximum positive and negative deviations, and the largest absolute deviation with its date. The table is used to identify where redispatch most strongly alters the storage path implied by the zonal market stage. All gaps are expressed in TWh. The daily storage target gap is defined as realized redispatch storage minus the zonal market-stage storage target. Positive values indicate realized storage above the zonal target; negative values indicate realized storage below the zonal target.

Source: Author's calculations from redispatch storage-target tracking outputs.

Table 4.6b. Frequency of large storage target deviations by zone.

Scenario	Zone	> 0.1 TWh	> 0.25 TWh	> 0.5 TWh
-15%	NO1	151	58	0
-15%	NO2	332	275	235
-15%	NO3	2	0	0
-15%	NO4	332	277	232
-15%	NO5	207	141	50
-10%	NO1	154	32	0
-10%	NO2	305	240	196
-10%	NO3	149	129	94
-10%	NO4	331	284	243
-10%	NO5	143	60	0
+10%	NO1	49	6	0
+10%	NO2	208	167	123
+10%	NO3	107	78	40
+10%	NO4	336	299	239
+10%	NO5	85	31	13
+15%	NO1	45	10	0
+15%	NO2	212	152	116
+15%	NO3	91	62	39
+15%	NO4	318	281	217
+15%	NO5	73	38	15
base	NO1	108	12	0
base	NO2	241	203	152
base	NO3	150	113	50
base	NO4	333	300	255

Scenario	Zone	> 0.1 TWh	> 0.25 TWh	> 0.5 TWh
base	NO5	83	42	2

Description: This table counts how often redispatch storage deviations exceed selected thresholds in each bidding zone. It reports the number of days where absolute deviations are greater than 0.1 TWh, 0.25 TWh and 0.5 TWh. The table complements Table 4.6a by showing whether large deviations are isolated events or persistent features of the redispatch-corrected outcome. Values report the number of days where the absolute redispatch storage target deviation exceeds each threshold.

Source: Author’s calculations from daily redispatch storage-target deviations.

Appendix 4.7. Annual spill volumes, spill intensity and spill timing.

Table 4.7a. Annual reservoir and run-of-river spill by hydrological scenario and market design.

Scenario	Design	Reservoir spill (TWh)	RoR spill (TWh)	Total spill (TWh)	Reservoir share (%)	RoR share (%)
-15%	Nodal	2.222	2.629	4.851	45.8	54.2
-15%	Zonal	0	2.183	2.183	0	100
-15%	Redispatch	2.491	2.629	5.12	48.7	51.3
-10%	Nodal	6.17	2.947	9.117	67.7	32.3
-10%	Zonal	1.747	2.588	4.335	40.3	59.7
-10%	Redispatch	6.232	3.041	9.273	67.2	32.8
base	Nodal	15.486	3.604	19.09	81.1	18.9
base	Zonal	10.238	3.853	14.09	72.7	27.3
base	Redispatch	14.755	4.18	18.935	77.9	22.1
+10%	Nodal	25.853	4.285	30.139	85.8	14.2
+10%	Zonal	20.888	5.091	25.979	80.4	19.6
+10%	Redispatch	24.662	5.255	29.917	82.4	17.6
+15%	Nodal	31.4	4.631	36.031	87.1	12.9
+15%	Zonal	26.572	5.633	32.205	82.5	17.5
+15%	Redispatch	30.014	5.788	35.802	83.8	16.2

Description: This table reports annual spill volumes for reservoir hydro and run-of-river generation across all hydrological scenarios and market designs. It also shows the relative contribution of each technology to total spill. The table is used to assess how unused water increases under wetter inflow conditions and whether the source of spill differs between reservoir and run-of-river generation. Spill volumes are expressed in TWh. Shares are expressed as percentages of total spill. (RoR = run-of-river.)

Source: Author’s calculations from Python/PyPSA reservoir and run-of-river spill outputs.

***Note:** *Reservoir spill and run-of-river spill have different operational meanings: reservoir spill is linked to storage saturation, while run-of-river spill reflects contemporaneous absorption limits.*

Table 4.7b. Spill intensity by hydrological scenario and market design.

Scenario	Design	Reservoir spill / reservoir inflow (%)	RoR spill / RoR inflow (%)	Total spill / total inflow (%)
-15%	Nodal	2.1	35.2	4.3
-15%	Zonal	0	29.3	1.9
-15%	Redispatch	2.3	35.2	4.5
-10%	Nodal	5.5	37.3	7.6
-10%	Zonal	1.6	32.8	3.6
-10%	Redispatch	5.5	38.5	7.7
base	Nodal	12.4	41.1	14.2
base	Zonal	8.2	43.9	10.5
base	Redispatch	11.8	47.6	14.1
+10%	Nodal	18.8	44.4	20.4
+10%	Zonal	15.2	52.7	17.6
+10%	Redispatch	17.9	54.4	20.3
+15%	Nodal	21.8	45.9	23.4
+15%	Zonal	18.4	55.8	20.9
+15%	Redispatch	20.8	57.3	23.2

Description: This table expresses spill as a percentage of available inflow for reservoir hydro, run-of-river generation and the total hydro system. It measures the intensity of spill rather than its absolute volume. The table is useful for comparing how efficiently each market design converts available water into generation or storage under different hydrological conditions. Values are expressed as percentages. Spill intensity measures the share of inflow that is spilled rather than stored or converted into generation.

Source: Author’s calculations from spill outputs and scenario-specific available inflow inputs.

***Note:** *Spill intensity measures the share of available water-energy that cannot be converted into useful generation or stored within the model constraints.*

Table 4.7c. Spill timing by hydrological scenario and market design.

Scenario	Design	Peak reservoir month	Peak RoR month	Peak total month	Reservoir spill days	RoR spill days	Total spill days
-15%	Nodal	September	June	September	150	196	198
-15%	Zonal	January	June	June	None	161	161
-15%	Redispatch	September	June	September	131	196	196
-10%	Nodal	September	June	September	161	196	199
-10%	Zonal	October	June	October	18	169	169
-10%	Redispatch	September	June	September	153	196	196
base	Nodal	September	June	September	173	203	208
base	Zonal	September	June	September	64	176	176
base	Redispatch	September	June	September	163	203	203
+10%	Nodal	September	June	September	177	224	229
+10%	Zonal	September	June	September	96	178	178
+10%	Redispatch	September	June	September	168	224	224
+15%	Nodal	September	June	September	182	231	240
+15%	Zonal	September	June	September	100	183	183
+15%	Redispatch	September	June	September	169	232	232

Description: This table summarizes the timing of spill events across scenarios and market designs. It identifies the peak month for reservoir spill, run-of-river spill and total spill, and reports the number of days with nonzero spill. The table is used to show whether spill is concentrated in specific seasonal periods or becomes more persistent as inflows increase. Spill days refer to days with nonzero spill. Peak month refers to the month with the highest daily spill contribution for the relevant spill type.

Source: Author’s calculations from daily and monthly spill outputs.

Appendix 4.8. Hydrological sensitivity and cross-design differences in spill outcomes.

Table 4.8a. Hydrological sensitivity of spill relative to the base scenario.

Scenario vs base	Design	Δ reservoir spill	Δ reservoir spill (%)	Δ RoR spill	Δ RoR spill (%)	Δ total spill	Δ total spill (%)
-15%	Nodal	-13.264	-85.7	-0.975	-27.1	-14.239	-74.6
-15%	Zonal	-10.238	-100	-1.67	-43.3	-11.908	-84.5
-15%	Redispatch	-12.264	-83.1	-1.551	-37.1	-13.815	-73
-10%	Nodal	-9.315	-60.2	-0.658	-18.2	-9.973	-52.2
-10%	Zonal	-8.49	-82.9	-1.265	-32.8	-9.755	-69.2
-10%	Redispatch	-8.523	-57.8	-1.139	-27.2	-9.662	-51
+10%	Nodal	10.368	66.9	0.681	18.9	11.048	57.9
+10%	Zonal	10.651	104	1.238	32.1	11.889	84.4
+10%	Redispatch	9.906	67.1	1.076	25.7	10.982	58
+15%	Nodal	15.915	102.8	1.026	28.5	16.941	88.7
+15%	Zonal	16.334	159.6	1.78	46.2	18.114	128.6
+15%	Redispatch	15.259	103.4	1.608	38.5	16.867	89.1

Description: This table compares spill outcomes in the sensitivity scenarios with the base hydrological scenario. It reports both absolute and percentage changes in reservoir spill, run-of-river spill and total spill. The table is used to quantify how strongly spill responds to lower and higher inflow assumptions within each market-design representation. Δ indicates change relative to the base scenario within the same market design. Volume changes are expressed in TWh. Percentage changes are expressed relative to the relevant base or benchmark case.

Source: Author's calculations from annual spill outputs generated by the Python/PyPSA simulations.

***Note:** *Percentage changes should be interpreted with caution when the baseline spill volume is small, because small absolute changes can produce large relative changes.*

Table 4.8b. Cross-design comparison of reservoir spill.

Scenario	Zonal - nodal	Redispatch - nodal	Zonal - redispatch
-15%	-2.222	0.269	-2.491
-10%	-4.423	0.062	-4.485
base	-5.248	-0.731	-4.517
+10%	-4.965	-1.192	-3.773
+15%	-4.828	-1.387	-3.442

Description: This table compares annual reservoir spill across market designs for each hydrological scenario. It reports the differences between the zonal market stage, the nodal benchmark and the redispatch-corrected outcome. The table helps identify whether reservoir spill is mainly a hydrological effect or whether it is also affected by the way the market design represents network constraints and redispatch. All values are expressed in TWh. Negative values indicate lower reservoir spill in the first design relative to the comparator.

Source: Author's calculations from annual spill outputs generated by the Python/PyPSA simulations.

Table 4.8c. Cross-design comparison of run-of-river spill.

Scenario	Zonal - nodal	Redispatch - nodal	Zonal - redispatch
-15%	-0.447	0	-0.447
-10%	-0.359	0.094	-0.453
base	0.248	0.575	-0.327
+10%	0.806	0.97	-0.164
+15%	1.002	1.157	-0.155

Description: This table compares annual run-of-river spill across market designs for each hydrological scenario. It highlights how much run-of-river spill differs between the zonal market stage, the nodal benchmark and the redispatch-corrected outcome. The table is useful because run-of-river spill reflects immediate inflow utilisation constraints rather than long-term reservoir storage decisions. All values are expressed in TWh. RoR = run-of-river. Negative values indicate lower RoR spill in the first design relative to the comparator.

Source: Author's calculations from annual spill outputs generated by the Python/PyPSA simulations.

Table 4.8d. Cross-design comparison of total spill.

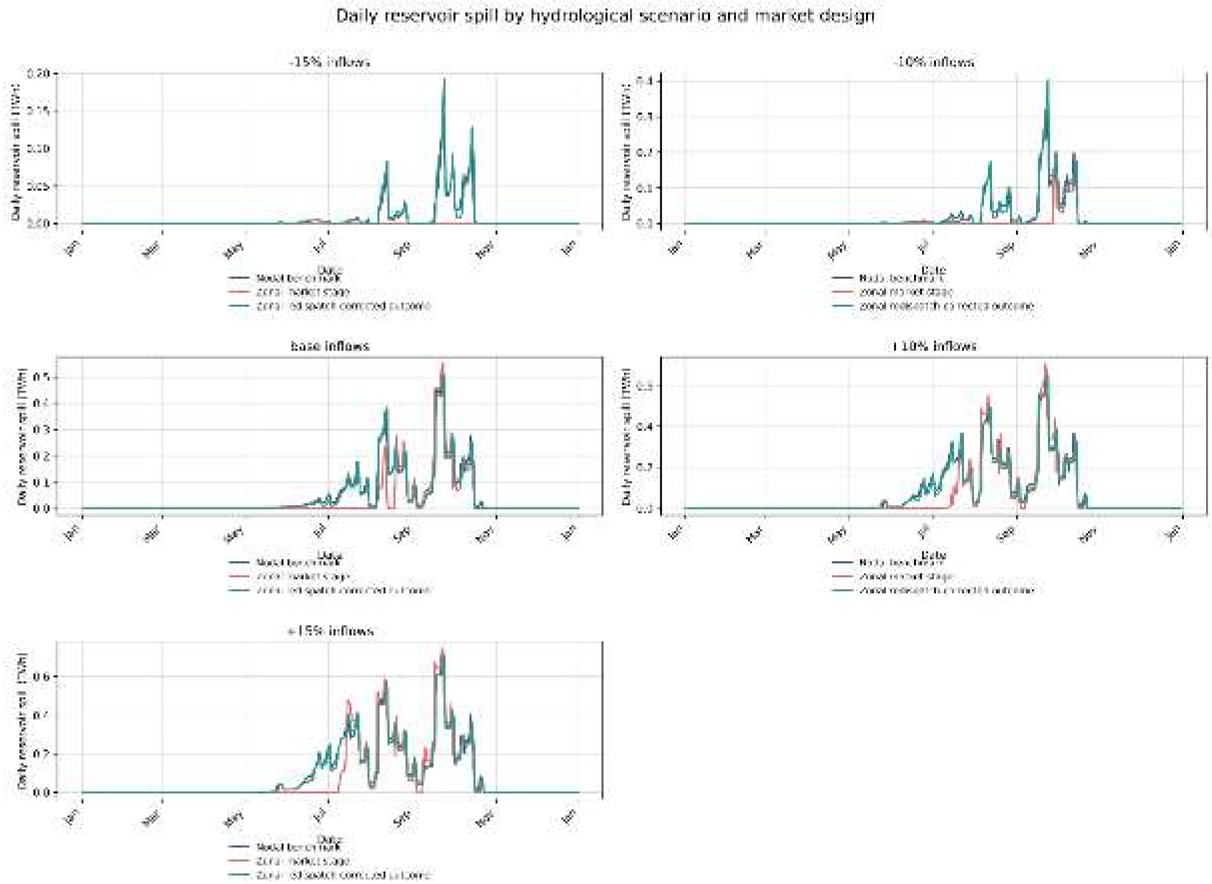
Scenario	Zonal - nodal	Redispatch - nodal	Zonal - redispatch
-15%	-2.668	0.269	-2.938
-10%	-4.782	0.156	-4.938
base	-5	-0.155	-4.844
+10%	-4.159	-0.222	-3.938
+15%	-3.826	-0.23	-3.597

Description: This table reports cross-design differences in total annual spill. It combines reservoir and run-of-river spill into a single system-level indicator and compares the zonal market stage, nodal benchmark and redispatch-corrected outcome. The table is used to assess whether total water spillage is reduced or increased by the market-design representation. All values are expressed in TWh. Negative values indicate lower total spill in the first design relative to the comparator.

Source: Author's calculations from annual spill outputs generated by the Python/PyPSA simulations.

Appendix 4.9. Daily spill profiles and storage-regime spill patterns.

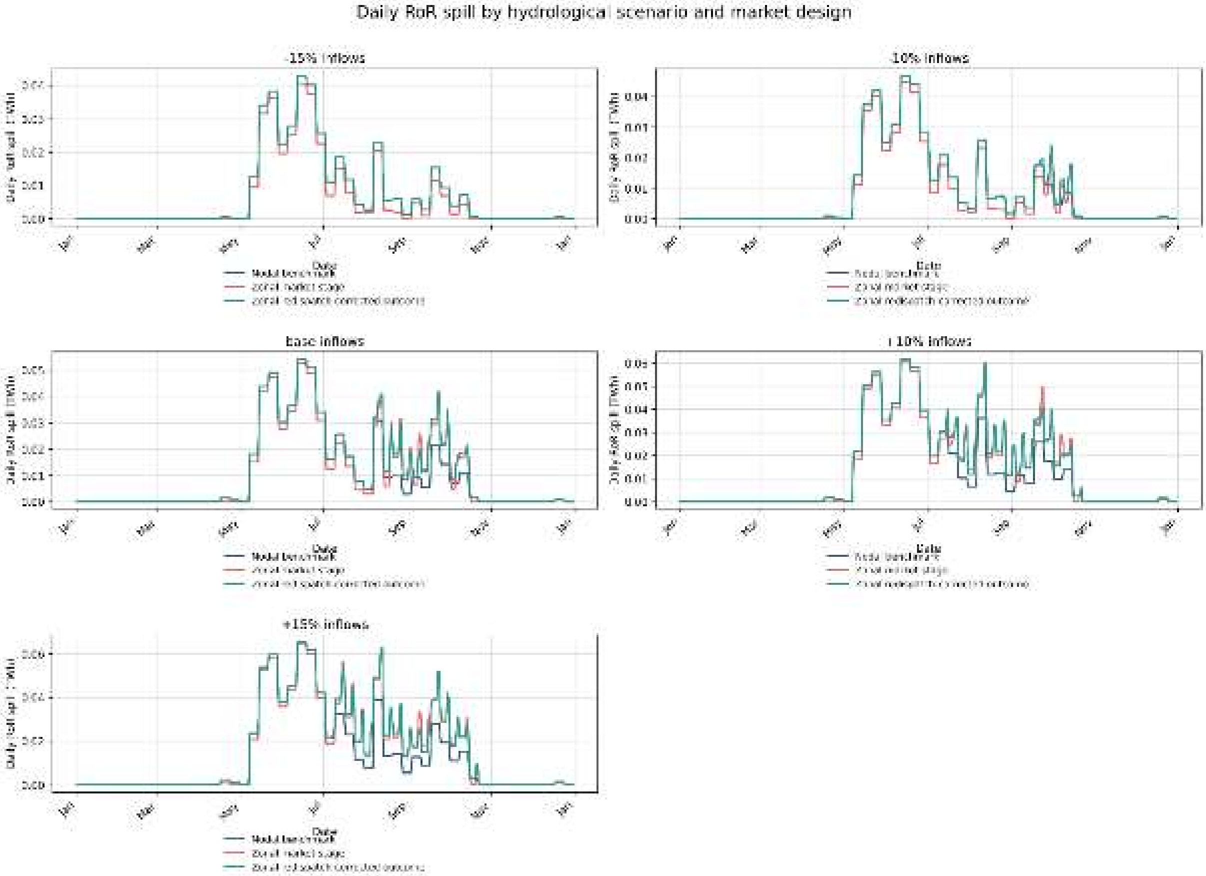
Figure 4.9a. Daily Reservoir Spill by Hydrological Scenario and Market Design.



Description: This figure reports daily reservoir spill across the five hydrological inflow scenarios and the three market-design representations: the nodal benchmark, the zonal market stage, and the zonal redispatch-corrected outcome. Reservoir spill is expressed in TWh and displayed over the full simulation year. The figure is used to show when reservoir spill occurs, how its magnitude changes with hydrological conditions, and whether the market design affects the timing and intensity of unused reservoir water. Reservoir spill refers to water allocated to reservoir hydro that cannot be stored or converted into generation within the model constraints. Daily values are aggregated from hourly simulation outputs. The five panels correspond to the -15%, -10%, base, +10% and +15% inflow scenarios.

Source: Author’s calculations from daily reservoir and run-of-river spill outputs.

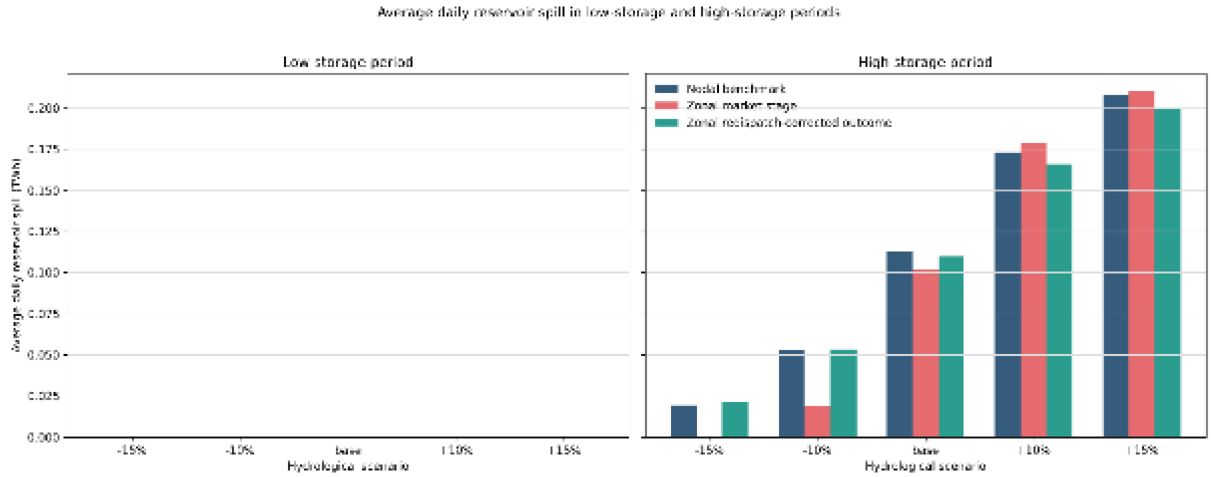
Figure 4.9b. Daily Run-of-River Spill by Hydrological Scenario and Market Design.



Description: This figure presents daily run-of-river spill across the five hydrological inflow scenarios and the three market-design representations. Run-of-river spill is expressed in TWh and shown over the full simulation year. The figure highlights the seasonal timing of run-of-river spill, especially during the spring and summer inflow period, and shows how spill patterns vary between drier and wetter hydrological assumptions. Run-of-river spill refers to contemporaneous inflow that cannot be converted into generation because of turbine or network-related constraints. Unlike reservoir spill, run-of-river spill is less directly linked to intertemporal storage decisions and more closely reflects immediate inflow availability.

Source: Author’s calculations from daily reservoir and run-of-river spill outputs.

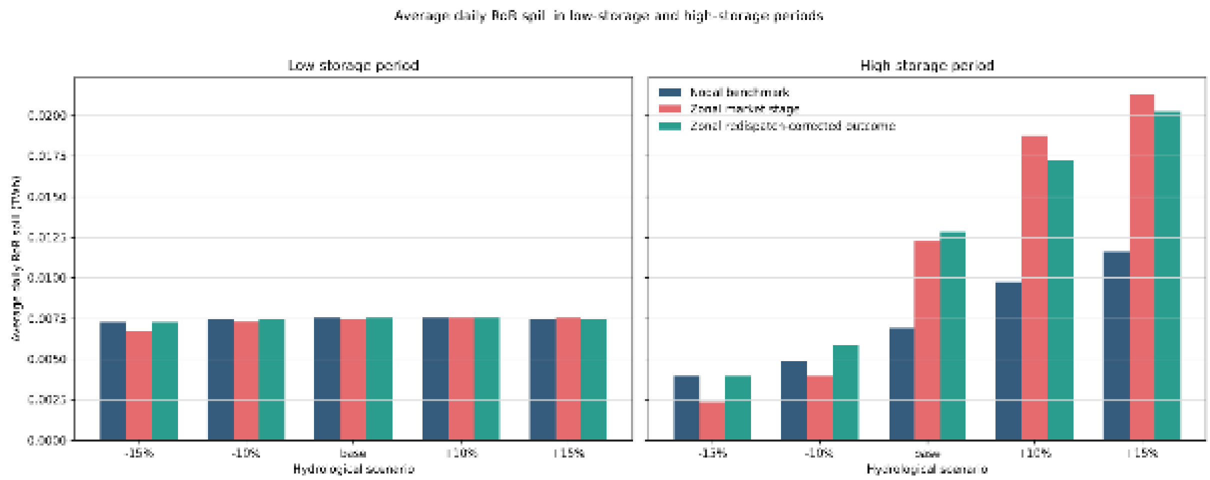
Figure 4.9c. Average Daily Reservoir Spill in Low-Storage and High-Storage Periods.



Description: This figure compares average daily reservoir spill during low-storage and high-storage periods across hydrological scenarios and market designs. Reservoir spill is expressed in TWh. The figure is used to show whether reservoir spill is associated with periods of abundant stored water and to compare how the nodal benchmark, the zonal market stage and the redispatch-corrected outcome differ when storage conditions are low or high. Low-storage and high-storage periods are defined according to the storage-period classification used in the model diagnostics. The absence, or near absence, of reservoir spill during low-storage periods indicates that reservoir spill mainly occurs when the system is already in a high-storage state. Values represent average daily spill within each storage regime, not annual totals.

Source: Author's calculations from daily spill outputs and storage-regime classification.

Figure 4.9d. Average Daily Run-of-River Spill in Low-Storage and High-Storage Periods.



Description: This figure compares average daily run-of-river spill during low-storage and high-storage periods across hydrological scenarios and market designs. Run-of-river spill is expressed in TWh. The figure is used to assess whether run-of-river spill is sensitive to aggregate reservoir-storage conditions and whether wetter scenarios increase average spill during high-storage periods. Run-of-river spill is shown separately from reservoir spill because it reflects contemporaneous inflow that cannot be stored. Average daily RoR spill remains present even during low-storage periods, but increases clearly during high-storage periods, especially under the base, +10% and +15% inflow scenarios. (RoR = run-of-river.)

Source: Author's calculations from daily spill outputs and storage-regime classification..

Appendix 4.10. Zone-level spill distribution and spill composition in selected zones.

Table 4.10a. Zone-level total spill in selected zones.

Scenario	Design	Zone	Total spill (TWh)	Zone share (%)
-15%	Nodal	NO2	1.747	36
-15%	Nodal	NO4	0.733	15.1
-15%	Nodal	NO5	0.775	16
-15%	Zonal	NO2	0.273	12.5
-15%	Zonal	NO4	0.684	31.3
-15%	Zonal	NO5	0.468	21.4
-15%	Redispatch	NO2	1.671	32.6
-15%	Redispatch	NO4	0.746	14.6
-15%	Redispatch	NO5	0.883	17.2
-10%	Nodal	NO2	3.79	41.6
-10%	Nodal	NO4	0.829	9.1
-10%	Nodal	NO5	1.704	18.7
-10%	Zonal	NO2	1.887	43.5
-10%	Zonal	NO4	0.778	18
-10%	Zonal	NO5	0.682	15.7
-10%	Redispatch	NO2	3.621	39
-10%	Redispatch	NO4	0.864	9.3
-10%	Redispatch	NO5	1.833	19.8
base	Nodal	NO2	7.998	41.9
base	Nodal	NO4	1.085	5.7
base	Nodal	NO5	4.289	22.5
base	Zonal	NO2	6.722	47.7
base	Zonal	NO4	1.058	7.5
base	Zonal	NO5	2.768	19.6
base	Redispatch	NO2	8.171	43.2
base	Redispatch	NO4	1.293	6.8
base	Redispatch	NO5	4.032	21.3
+10%	Nodal	NO2	12.319	40.9
+10%	Nodal	NO4	1.545	5.1

Scenario	Design	Zone	Total spill (TWh)	Zone share (%)
+10%	Nodal	NO5	7.097	23.5
+10%	Zonal	NO2	11.164	43
+10%	Zonal	NO4	1.805	6.9
+10%	Zonal	NO5	5.898	22.7
+10%	Redispatch	NO2	12.602	42.1
+10%	Redispatch	NO4	1.801	6
+10%	Redispatch	NO5	6.84	22.9
+15%	Nodal	NO2	14.503	40.2
+15%	Nodal	NO4	1.942	5.4
+15%	Nodal	NO5	8.572	23.8
+15%	Zonal	NO2	13.528	42
+15%	Zonal	NO4	2.304	7.2
+15%	Zonal	NO5	7.609	23.6
+15%	Redispatch	NO2	14.834	41.4
+15%	Redispatch	NO4	2.134	6
+15%	Redispatch	NO5	8.417	23.5

Description: This table reports total spill volumes in the selected bidding zones NO2, NO4 and NO5. It also shows each zone's share of total system spill within the same scenario and market design. The table is used to identify the spatial concentration of spill and to support the discussion of zones that are most relevant for the interpretation of hydro-system behaviour in Chapter 4. Only NO2, NO4 and NO5 are shown because these zones are directly discussed in Chapter 4. Total spill is expressed in TWh. Zone share is expressed as a percentage of system total spill within the same scenario-design combination.

Source: Author's calculations from zone-level spill outputs generated by the Python/PyPSA simulations.

Table 4.10b. Zone-level spill composition in selected zones.

Scenario	Design	Zone	Reservoir spill (TWh)	RoR spill (TWh)	Reservoir share (%)	RoR share (%)
-15%	Nodal	NO2	1.41	0.337	80.7	19.3
-15%	Nodal	NO4	0.037	0.697	5	95
-15%	Nodal	NO5	0.243	0.532	31.3	68.7
-15%	Zonal	NO2	0	0.273	0	100
-15%	Zonal	NO4	0	0.684	0	100
-15%	Zonal	NO5	0	0.468	0	100
-15%	Redispatch	NO2	1.334	0.337	79.8	20.2

Scenario	Design	Zone	Reservoir spill (TWh)	RoR spill (TWh)	Reservoir share (%)	RoR share (%)
-15%	Redispatch	NO4	0.05	0.697	6.6	93.4
-15%	Redispatch	NO5	0.351	0.532	39.7	60.3
-10%	Nodal	NO2	3.415	0.375	90.1	9.9
-10%	Nodal	NO4	0.056	0.773	6.8	93.2
-10%	Nodal	NO5	1.105	0.598	64.9	35.1
-10%	Zonal	NO2	1.568	0.319	83.1	16.9
-10%	Zonal	NO4	0	0.778	0	100
-10%	Zonal	NO5	0.126	0.555	18.5	81.5
-10%	Redispatch	NO2	3.238	0.383	89.4	10.6
-10%	Redispatch	NO4	0.069	0.795	7.9	92.1
-10%	Redispatch	NO5	1.209	0.624	66	34
base	Nodal	NO2	7.545	0.453	94.3	5.7
base	Nodal	NO4	0.156	0.929	14.4	85.6
base	Nodal	NO5	3.553	0.736	82.8	17.2
base	Zonal	NO2	6.272	0.451	93.3	6.7
base	Zonal	NO4	0	1.058	0	100
base	Zonal	NO5	1.951	0.817	70.5	29.5
base	Redispatch	NO2	7.668	0.503	93.8	6.2
base	Redispatch	NO4	0.248	1.045	19.2	80.8
base	Redispatch	NO5	3.156	0.876	78.3	21.7
+10%	Nodal	NO2	11.785	0.534	95.7	4.3
+10%	Nodal	NO4	0.455	1.09	29.5	70.5
+10%	Nodal	NO5	6.22	0.877	87.6	12.4
+10%	Zonal	NO2	10.582	0.582	94.8	5.2
+10%	Zonal	NO4	0.493	1.312	27.3	72.7
+10%	Zonal	NO5	4.838	1.06	82	18
+10%	Redispatch	NO2	11.984	0.618	95.1	4.9
+10%	Redispatch	NO4	0.512	1.289	28.4	71.6
+10%	Redispatch	NO5	5.739	1.101	83.9	16.1
+15%	Nodal	NO2	13.927	0.576	96	4
+15%	Nodal	NO4	0.772	1.17	39.7	60.3
+15%	Nodal	NO5	7.623	0.949	88.9	11.1
+15%	Zonal	NO2	12.89	0.638	95.3	4.7

Scenario	Design	Zone	Reservoir spill (TWh)	RoR spill (TWh)	Reservoir share (%)	RoR share (%)
+15%	Zonal	NO4	0.878	1.426	38.1	61.9
+15%	Zonal	NO5	6.439	1.17	84.6	15.4
+15%	Redispatch	NO2	14.157	0.676	95.4	4.6
+15%	Redispatch	NO4	0.725	1.408	34	66
+15%	Redispatch	NO5	7.215	1.202	85.7	14.3

Description: This table decomposes zone-level spill into reservoir spill and run-of-river spill for the selected zones NO2, NO4 and NO5. It reports both absolute spill volumes and the relative shares of each technology. The table helps clarify whether spill in each zone is mainly driven by reservoir storage limitations or by run-of-river inflow that cannot be converted into generation. Only NO2, NO4 and NO5 are shown because these zones are directly discussed in Chapter 4. Spill volumes are expressed in TWh. Shares are expressed as percentages of total zone-level spill.

Source: Author's calculations from zone-level reservoir and run-of-river spill outputs.

Appendix 4.11. Annual and peak load-shedding indicators.

Table 4.11a. Annual load shedding by hydrological scenario and market design.

Scenario	Design	Total LS (TWh)	Demand (TWh)	LS / demand (%)	LS hours	LS days	Main LS zone	Main zone share
-15%	Nodal	3.852	105.088	3.7	4275	322	NO5	87.8
-15%	Zonal	0	105.088	0	0	0	NO1	NA
-15%	Redispatch	3.893	105.088	3.7	4284	322	NO5	86.9
-10%	Nodal	3.841	105.088	3.7	4275	322	NO5	88
-10%	Zonal	0	105.088	0	0	0	NO1	NA
-10%	Redispatch	3.889	105.088	3.7	4284	322	NO5	86.9
base	Nodal	3.826	105.088	3.6	4275	322	NO5	88.3
base	Zonal	0	105.088	0	0	0	NO1	NA
base	Redispatch	3.878	105.088	3.7	4279	322	NO5	87.1
+10%	Nodal	3.813	105.088	3.6	4273	322	NO5	88.5
+10%	Zonal	0	105.088	0	0	0	NO1	NA
+10%	Redispatch	3.862	105.088	3.7	4274	322	NO5	87.4
+15%	Nodal	3.806	105.088	3.6	4269	322	NO5	88.6
+15%	Zonal	0	105.088	0	0	0	NO1	NA
+15%	Redispatch	3.855	105.088	3.7	4271	322	NO5	87.5

Description: This table reports annual load shedding outcomes for each hydrological scenario and market design. It includes total unserved energy, total demand, load shedding as a share of demand, the number of load-shedding hours and days, and the main zone affected by load shedding. The table is used to assess the extent and spatial concentration of scarcity events in the model. LS = load shedding. Total LS and demand are expressed in TWh. LS / demand and main zone share are expressed as percentages. Load shedding is interpreted as unserved energy within the stylized optimization model, not as a real-world reliability forecast.

Source: Author's calculations from Python/PyPSA load-shedding outputs and hourly demand inputs.

Table 4.11b. Peak load-shedding intensity by hydrological scenario and market design.

Scenario	Design	Max. hourly LS (MWh)	Max. daily LS (GWh)	Mean LS during LS hours (MWh)
-15%	Nodal	2582.99	25.858	900.94
-15%	Zonal	0	0	NA
-15%	Redispatch	2609.98	26.131	908.83
-10%	Nodal	2577.95	25.817	898.55
-10%	Zonal	0	0	NA
-10%	Redispatch	2607.76	26.083	907.89
base	Nodal	2575.1	25.75	895.05
base	Zonal	0	0	NA
base	Redispatch	2593.9	26.081	906.19
+10%	Nodal	2558.61	25.626	892.24
+10%	Zonal	0	0	NA
+10%	Redispatch	2569.76	25.758	903.61
+15%	Nodal	2528.31	25.323	891.58
+15%	Zonal	0	0	NA
+15%	Redispatch	2568.4	25.735	902.69

Description: This table reports indicators of load-shedding intensity during scarcity events. It presents the maximum hourly load shedding, maximum daily load shedding and the mean load shedding during hours when load shedding occurs. The table complements annual load-shedding totals by showing whether scarcity is driven by a few severe hours or by repeated moderate shortages. LS = load shedding. Mean LS during LS hours is calculated only over hours in which load shedding is positive.

Source: Author's calculations from hourly and daily load-shedding outputs.

Appendix 4.12. Cross-design comparison of load shedding.

Table 4.12a. Cross-design comparison of load shedding.

Scenario	Δ LS: Zonal - nodal	Δ LS: Redispatch - nodal	Δ LS: Zonal - redispatch	Δ hours: Zonal - nodal	Δ hours: Redispatch - nodal	Δ hours: Zonal - redispatch
-15%	-3.852	0.042	-3.893	-4275	9	-4284
-10%	-3.841	0.048	-3.889	-4275	9	-4284
base	-3.826	0.051	-3.878	-4275	4	-4279
+10%	-3.813	0.049	-3.862	-4273	1	-4274
+15%	-3.806	0.049	-3.855	-4269	2	-4271

Description: This table compares load-shedding outcomes across market designs. It reports differences in total load shedding and in the number of load-shedding hours between the zonal market stage, the nodal benchmark and the redispatch-corrected outcome. The table is used to show how much the representation of market design changes the amount and persistence of unserved demand in the model. Δ indicates cross-design difference. Load-shedding differences are expressed in TWh; hour differences are expressed in hours. Negative values indicate lower load shedding or fewer shedding hours in the first design relative to the comparator.

Source: Author's calculations from annual load-shedding indicators by market design and scenario.

Appendix 4.13. Zone-level load shedding in selected scarcity zones.

Table 4.13a. Zone-level load shedding in selected scarcity zones.

Scenario	Design	Zone	Annual LS (TWh)	Demand	LS / demand	Share of system LS (%)	LS hours	LS days	Max. daily LS (GWh)
-15%	Nodal	NO2	0.337	24.615	1.4	8.7	1350	150	5.036
-15%	Nodal	NO4	0.134	15.055	0.9	3.5	1228	150	2.194
-15%	Nodal	NO5	3.381	14.141	23.9	87.8	4275	322	18.968
-15%	Redispatch	NO2	0.335	24.615	1.4	8.6	1350	150	4.95
-15%	Redispatch	NO4	0.176	15.055	1.2	4.5	1310	150	2.22
-15%	Redispatch	NO5	3.383	14.141	23.9	86.9	4284	322	18.968
-10%	Nodal	NO2	0.335	24.615	1.4	8.7	1350	150	4.996
-10%	Nodal	NO4	0.127	15.055	0.8	3.3	1209	150	2.188
-10%	Nodal	NO5	3.38	14.141	23.9	88	4275	322	18.967
-10%	Redispatch	NO2	0.333	24.615	1.4	8.6	1350	150	4.966
-10%	Redispatch	NO4	0.175	15.055	1.2	4.5	1302	150	2.215
-10%	Redispatch	NO5	3.382	14.141	23.9	86.9	4284	322	18.967
base	Nodal	NO2	0.332	24.615	1.3	8.7	1350	150	4.987
base	Nodal	NO4	0.118	15.055	0.8	3.1	1184	150	2.175
base	Nodal	NO5	3.377	14.141	23.9	88.3	4275	322	18.967

Scenario	Design	Zone	Annual LS (TWh)	Demand	LS / demand	Share of system LS (%)	LS hours	LS days	Max. daily LS (GWh)
base	Redispatch	NO2	0.331	24.615	1.3	8.5	1350	150	4.965
base	Redispatch	NO4	0.168	15.055	1.1	4.3	1288	150	2.205
base	Redispatch	NO5	3.379	14.141	23.9	87.1	4279	322	18.967
+10%	Nodal	NO2	0.329	24.615	1.3	8.6	1350	150	4.986
+10%	Nodal	NO4	0.109	15.055	0.7	2.9	1162	150	2.161
+10%	Nodal	NO5	3.374	14.141	23.9	88.5	4273	322	18.966
+10%	Redispatch	NO2	0.329	24.615	1.3	8.5	1350	150	4.964
+10%	Redispatch	NO4	0.157	15.055	1	4.1	1268	150	2.195
+10%	Redispatch	NO5	3.377	14.141	23.9	87.4	4274	322	18.966
+15%	Nodal	NO2	0.328	24.615	1.3	8.6	1350	150	4.963
+15%	Nodal	NO4	0.105	15.055	0.7	2.8	1150	150	2.119
+15%	Nodal	NO5	3.373	14.141	23.9	88.6	4269	322	18.965
+15%	Redispatch	NO2	0.328	24.615	1.3	8.5	1350	150	4.963
+15%	Redispatch	NO4	0.152	15.055	1	4	1263	150	2.19
+15%	Redispatch	NO5	3.375	14.141	23.9	87.5	4271	322	18.965

Description: This table reports zone-level load shedding for NO2, NO4 and NO5, which are the zones most directly discussed in Chapter 4. Annual load shedding is expressed in TWh, while maximum daily load shedding is expressed in GWh. The zonal market stage is omitted because it reports zero load shedding in all scenarios.

Source: Author's calculations from nodal load-shedding outputs aggregated to selected bidding zones.

Appendix 4.14. Nodal congestion and load-shedding overlap.

Table 4.14a. Nodal congestion and load-shedding overlap by hydrological scenario.

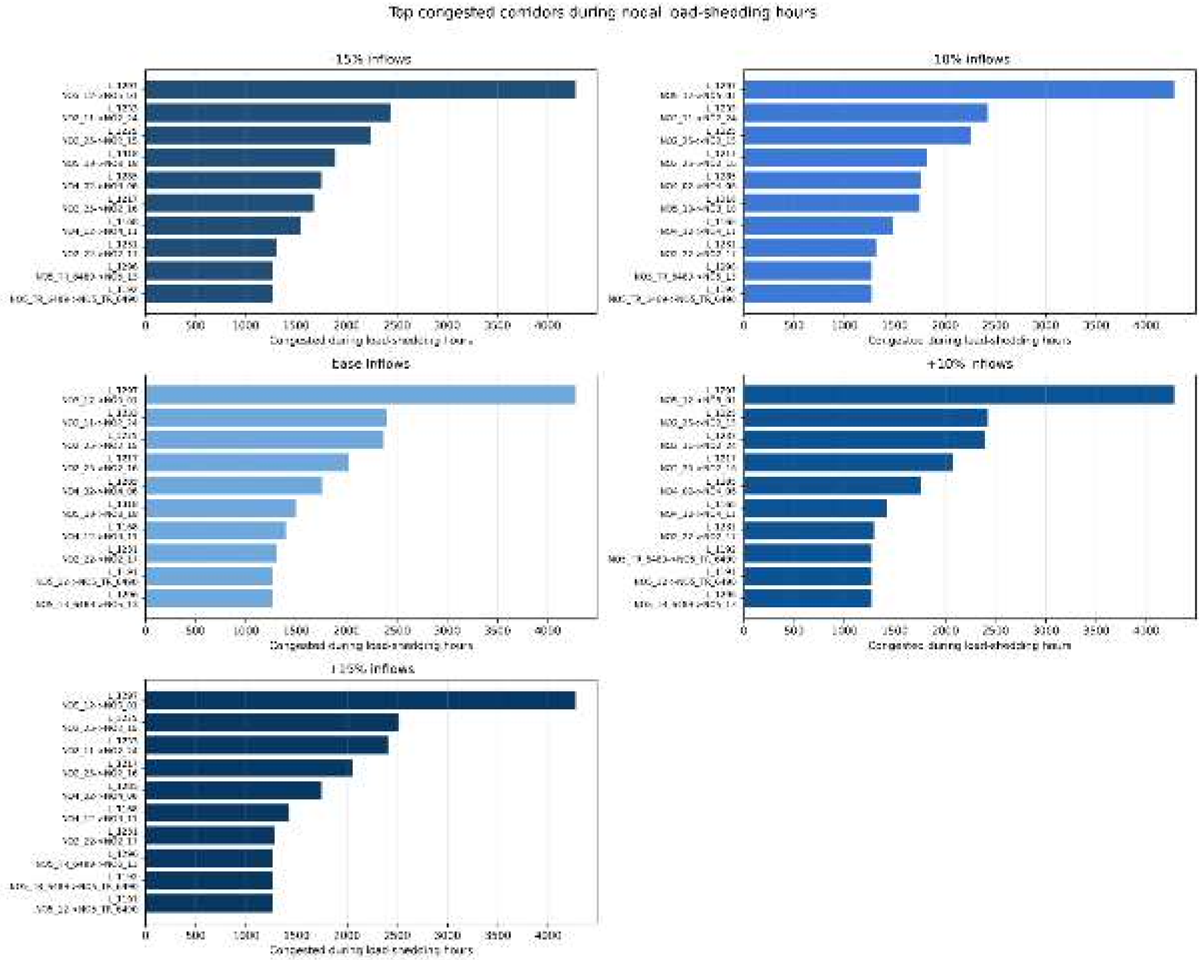
Scenario	LS hours	LS hours with congestion	Share with congestion	LS during congested hours	Share of LS during congestion	Main LS zone
-15%	4275	4275	100	3.852	100	NO5
-10%	4275	4275	100	3.841	100	NO5
base	4275	4275	100	3.826	100	NO5
+10%	4273	4273	100	3.813	100	NO5
+15%	4269	4269	100	3.806	100	NO5

Description: This table examines the relationship between load shedding and network congestion in the nodal benchmark. It reports the number of load-shedding hours, the number and share of these hours that coincide with congestion, and the share of total load shedding occurring during congested hours. The table supports the interpretation that scarcity events in the nodal model are closely associated with binding transmission constraints. LS refers to load shedding. Congestion is defined as an absolute line-loading ratio greater than or equal to 0.999, and LS during congested hours is expressed in TWh. This analysis is limited to the nodal benchmark. Redispatch congestion and load-shedding overlap are treated separately in Appendix 4.16 using reconstructed redispatch line-loading indicators.

Source: Author’s calculations from nodal benchmark load-shedding outputs, hourly line-flow outputs and matched line-capacity data.

Appendix 4.15. Congested transmission corridors during nodal load-shedding hours.

Figure 4.15a. Top Congested Transmission Corridors during Nodal Load-Shedding Hours.

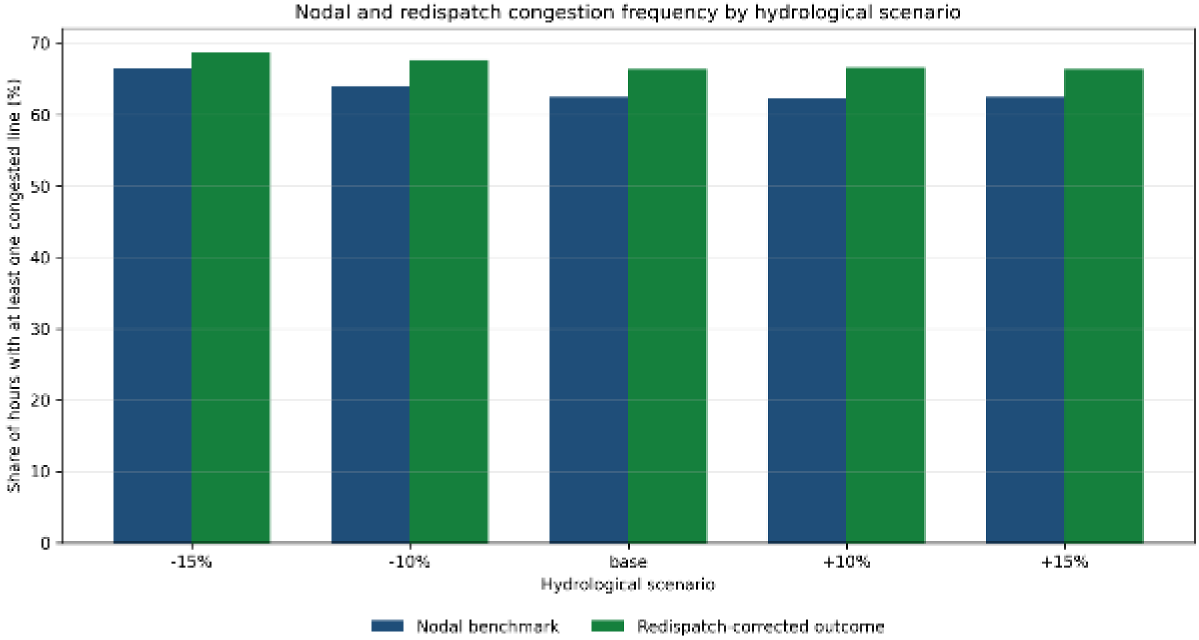


Description: This figure identifies the transmission corridors most frequently congested during hours in which load shedding occurs in the nodal benchmark. The x-axis reports the number of load-shedding hours during which each corridor is congested. The y-axis reports the model corridor labels, including the line identifier and connected nodes. The figure is shown separately for each hydrological inflow scenario in order to assess whether the same network bottlenecks remain binding under drier and wetter hydrological conditions. Congestion is counted for hours with nodal load shedding. A corridor is considered congested when its loading reaches the congestion threshold used in the model diagnostics.

Source: Author's calculations from nodal benchmark hourly line-loading outputs and load-shedding indicators.

Appendix 4.16. Redispatch congestion patterns and load-shedding overlap.

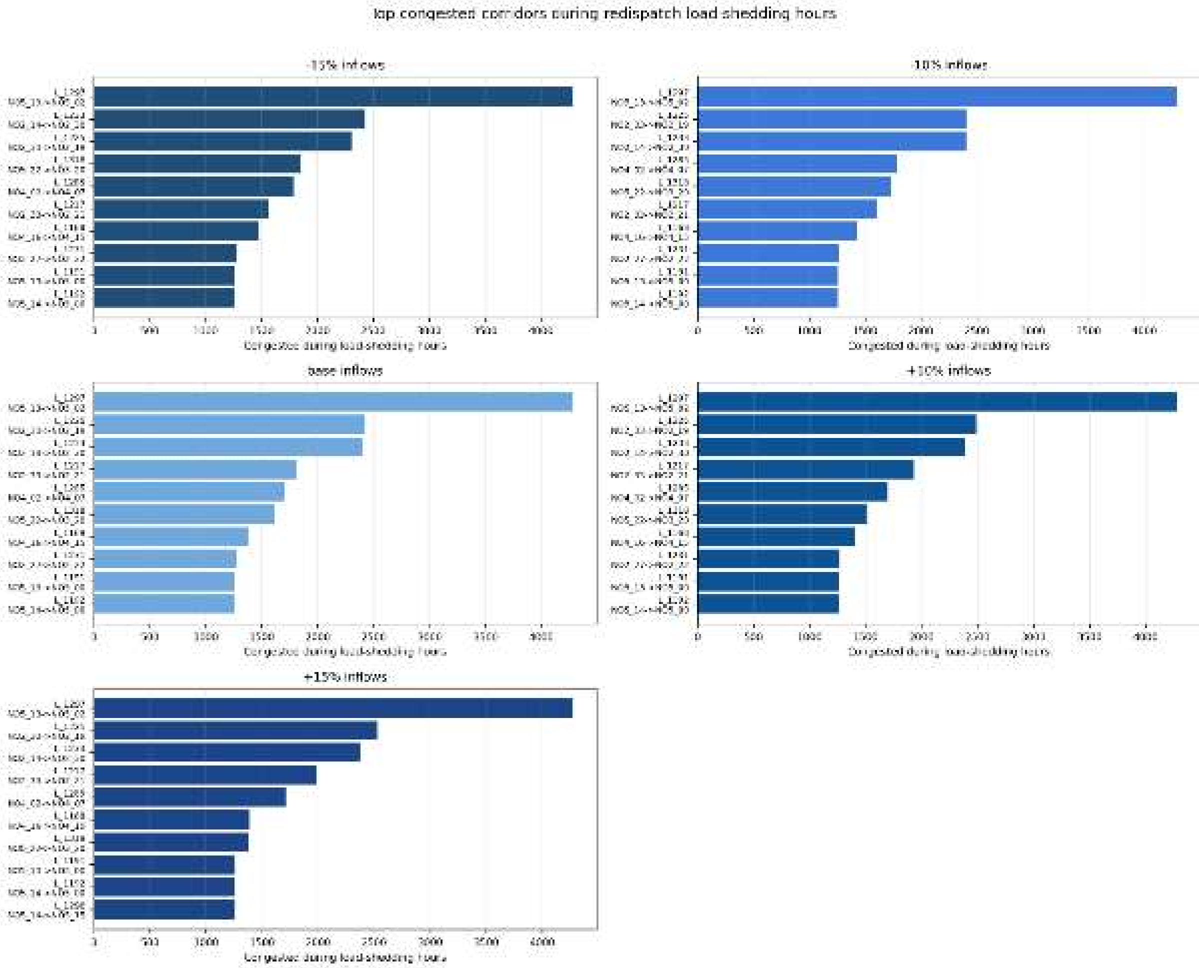
Figure 4.16a. Congestion frequency by market design and hydrological scenario.



Description: This figure compares the share of hours with at least one congested line in the nodal benchmark and in the redispatch-corrected outcome across the five hydrological scenarios. It shows that congestion remains persistent across all inflow conditions and that the redispatch-corrected outcome presents a slightly higher congestion frequency than the nodal benchmark. The figure therefore supports the interpretation that restoring physical feasibility through redispatch does not remove network constraints as a relevant feature of the stylized system.

Source: Author's calculations from nodal benchmark and redispatch-corrected hourly line-flow outputs and matched line-capacity data.

Figure 4.16b. Congested corridors during redispatch load-shedding hours.



Description: This figure presents the model corridors that are most frequently congested during hours with positive redispatch load shedding, for each hydrological scenario. It shows that congestion during scarcity periods is not randomly distributed across the stylized network, but tends to recur on a limited set of corridors. The figure therefore supports the interpretation that residual scarcity has a spatial-deliverability component: some demand remains unmet even though water-energy may be available elsewhere in the system.

Source: Author's calculations from redispatch-corrected hourly line-loading outputs and redispatch load-shedding indicator