NTNU – Trondheim
Norwegian University of
Science and Technology

A stochastic optimization approach for pricing hydropower regulation capacity in electricity markets

Bjørnar Fjelldal
Sjur Messel Nafstad

Master of Energy and Environmental Engineering
Submission date: June 2014
Supervisor: Olav B Fosso, ELKRAFT
Co-supervisor: Gro Klæboe, Powel AS

Norwegian University of Science and Technology
Department of Electric Power Engineering

i. Summary

With the introduction of an increasing number of regulation markets, hydropower producers face multiple opportunities regarding how to utilize their water optimally. Such regulation markets can increase profits drastically compared to a single day-ahead spot market, thus the use and comprehension of multi-market optimization tools are valuable for all hydropower producers with storage.

This report contains the development of a stochastic multi-stage optimization model where the effects of committing upon different levels of regulation obligations in electricity markets are investigated seen from a hydropower producer's perspective. A mathematical formulation was written, and a corresponding optimization code implemented. An analysis was further conducted through a case study where a stochastic spot price was constructed and optimized upon by an implemented watercourse with authentic properties.

Comparison between different levels of regulation obligations, the effect concerning the introduction of new information during the stochastic optimization period and the benefits of multiple executions (Monte Carlo-simulations) are among the essential output data analyzed.

The results indicate such comparisons of regulation obligations to be useful when deciding whether or not to provide regulation capacity, and if so, at which level. Also, the value of good information should not be underestimated when participating in markets with uncertainty regarding the future.

The market designs are affecting the choice and design of the optimization strategy used – in markets with more complex properties, analyzing multiple regulation obligations may prove hard to present and time consuming to process. Thus, using a forecasted price of regulation markets is often preferred, but unfortunately such forecasts are hard to obtain. Nevertheless, decision-making tools regarding optimal levels of regulation obligations can be utilized by considering factors such as the needed regulation price to break even from a potential loss in the spot market, as well as the marginal cost of increasing a regulation obligation.

ii. Sammendrag

Med et økende antall reguleringsmarkeder å ta hensyn til, har vannkraftprodusenter mange muligheter i forhold til hvordan man skal disponere vannet sitt optimalt. Slike reguleringsmarkeder kan øke fortjenesten kraftig i forhold til et enkelt spotmarked, så bruken og forståelsen av multimarkedsoptimeringsverktøy er verdifull for alle vannkraftprodusenter med muligheten til å lagre energi.

Denne rapporten inneholder utviklingen av en stokastisk flerstegs optimeringsmodell hvor effekten ved å forholde seg til forskjellige nivå av reguleringsforpliktelser i kraftmarkeder er undersøkt, sett fra en vannkraftprodusents ståsted. En matematisk formulering ble utformet, og en korresponderende optimeringskode ble implementert. Videre ble en analyse utført gjennom et casestudie hvor en stokastisk spotpris ble konstruert og optimert mot med et vassdrag som innehar autentiske egenskaper.

Sammenligninger mellom forskjellige nivå av reguleringsforpliktelser, effekten vedrørende introduksjonen av ny informasjon under den stokastiske optimeringsperioden og fordelene av flere modellkjøringer (Monte Carlo-simuleringer) er blant de essensielle utdataene som ble analysert.

Resultatene indikerer at slike sammenligninger av reguleringsforpliktelser er nyttige når valget om å levere eller ikke levere reguleringskapasitet skal tas, og i så fall, på hvilket nivå. I tillegg bør ikke verdien av god informasjon bli undervurdert når man deltar i markeder med en usikkerhet angående fremtiden.

Utformingen av markedet påvirker hvilken optimeringsstrategi som bør bli brukt – i markeder med mer komplekse egenskaper kan analyseringen av multiple reguleringsforpliktelser være vanskelig å presentere fornuftig, samt være tidskrevende å prosessere. Derfor vil en prognose på prisen i et reguleringsmarked være å foretrekke, men dessverre er gode slike prognoser vanskelige å fremskaffe. Likevel, beslutningsverktøy som tar hensyn til optimale nivå av reguleringsforpliktelser kan bli benyttet ved å vurdere faktorer som den nødvendige reguleringsprisen for å gå i balanse fra et potensielt tap i spotmarked, i tillegg til marginalkostnaden ved å øke en reguleringsforpliktelse.

iii. Preface

This report is written as a master thesis during the Master of Science degree for the *Energy and Environmental Engineering* program at the *Norwegian University of Science and Technology* (NTNU) under the *Department of Electric Power Engineering*.

It must also be noted that in addition to this work, an article [1] based on last fall's specialization project [2] was written during this semester. It was approved, presented and published at the *11th International Conference on the European Energy Market* (EEM14) that took place in Cracow 28th-30th of May 2014, and is available as appendix I.

We would like to thank our supervisor Olav Bjarte Fosso (Department of Electric Power Engineering, NTNU) for the guidance provided to us during our final semester at NTNU. Also, we would like to thank Powel AS for the great cooperation and the kindness of providing an additional working space at their headquarters and the possibility of making use of their competence. Lastly, we thank Gro Klæboe (Powel AS) for her continuous support throughout this master thesis process.

Trondheim, June 2014

Bjørnar Fjelldal Sjur Messel Nafstad

iv. Nomenclature

Table 1: Indices

e	Scenario e
h	Hour h
i	Index i
o	Regulation obligation o
r	Reservoir r
s	Station s
t	Step t
u	Unit u

Table 2: Sets

E	Number of scenarios
$\tilde{E}$	Number of scenarios that can occur given the node in the scenario tree that the model is currently standing on
H	Number of hours
I	Number of indices
O	Number of regulation obligations
R	Number of reservoirs
S	Number of stations
T	Number of steps
U	Number of units at station s

Table 3: Variables

$b^{t,\mathrm{o},r,h}$	Bypass from reservoir r in hour h for regulation obligation o optimized at step t	m^3/s
$f^{t,\mathrm{o},r,h}$	Flood from reservoir r in hour h for regulation obligation o optimized at step t	m^3/s
$q^{t,\mathrm{o},s,u,h}$	Discharge through unit u at station s in hour h for regulation obligation o optimized at step t	m^3/s
$x^{t,\mathrm{o},s,u,h}$	Production on unit u at station s in hour h for regulation obligation o optimized at step t	MW
$x_{tot}^{t,\mathrm{o},h}$	Total production in hour h for regulation obligation o optimized at step t	MW
$\alpha^{t,\mathrm{o},r}$	Energy equivalent of final reservoir level at reservoir r for regulation obligation o optimized at step t	MWh
$\beta^{t,\mathrm{o},s,u,h}$	Start indicator for regulation obligation o optimized at step t, 1 if unit u at station s starts in hour h, 0 otherwise	{0,1}
$\gamma^{t,\mathrm{o},r,h}$	Reservoir level at reservoir r in hour h for regulation obligation o optimized at step t	m^3
$\zeta^{t,\mathrm{o},s,u,h}$	Run indicator for regulation obligation o optimized at step t, 1 if unit u at station s runs in hour h, 0 otherwise	{0,1}
$\theta^{t,\mathrm{o},r}$	Final reservoir level at reservoir r for regulation obligation o optimized at step t	m^3

Table 4: Parameters

$A_{\max}^{r}$	Maximum reservoir regulation per hour at reservoir r	m^3
$B_{\max}^{r}$	Maximum bypass from reservoir r	m^3/s
$B_{\min}^{r}$	Minimum bypass from reservoir r	m^3/s
$C^{s,u}$	Startup cost for unit u at station s	NOK
$G_{\max}^{s,u}$	Maximum ramping per hour on unit u at station s	MW
g	Gravity constant	m/s^2
$H^{s,\mathrm{u}}$	Head for unit u at station s	m
$H_{tot}^{s,u}$	Total system head for unit u at station s	m
$L_{\max}^{r}$	Maximum reservoir volume in reservoir r	m^3
$L_{\min}^{r}$	Minimum reservoir volume in reservoir r	m^3
$N^{r,h}$	Inflow to reservoir r in hour h	m^3/s
$P_{spot}^{e,h}$	Spot price for scenario e in hour h	NOK/MWh
$Q_{\max}^{s,u}$	Maximum discharge through unit u at station s	m^3/s
$Q_{\min}^{s,u}$	Minimum discharge through unit u at station s	m^3/s
W^{r}	Water value for reservoir r	NOK/MWh
$X_{\max}^{s,u}$	Maximum production on unit u at station s	MW
$X_{\min}^{s,u}$	Minimum production on unit u at station s	MW
$Y_{sym}^{o,h}$	Regulation obligation o in hour h	MW
$\eta_{\max}^{s,u}$	Turbine efficiency at best-point on unit u at station s	%
Π^{e}	Probability of scenario e	%
ρ	Water density	kg/m^3
Γ^{i}	Slope of cut i in the piecewise linearized P-Q-curve	-
Ω^{i}	Intercept of cut i in the piecewise linearized P-Q-curve	-

v. Table of contents

1 Introduction

Hydropower producers have an increased amount of opportunities and ways to utilize their water in order to gain as much profit as possible. With the introduction of additional ancillary services - like newly designed regulation markets - the process of spending the energy stored in reservoirs optimally prove to become more difficult than ever before.

During the recent years the proportion of renewable energy sources, such as solar and wind power, has increased in the European region [3]. These energy sources are non-controllable and unpredictable yielding an increased value of the energy sources suited for regulation. Hence, these newly designed regulation markets are especially profitable for participants with the ability to change the production on a short notice, such as hydropower producers with reservoir capacities.

This thesis develops a stochastic multi-stage optimization model for pricing such services. No specific market is investigated - the goal is to develop and investigate a model that puts a price on the reduced flexibility a hydropower producer faces in the day-ahead spot market when taking on a regulation obligation.

Many regulation markets have longer time horizons than the day-ahead horizon in the spot market. This challenges the decision of going into such markets, since the hydropower producer reduces the flexibility in the day-head spot market for an equally long horizon when taking on an obligation in a regulation market.

The model developed in this thesis investigates such a case; it optimizes the production to the day-ahead spot market given different levels of regulation obligations for a period of 9 days while utilizing a stochastic spot price. The physical scheduling of the watercourse under such conditions is analyzed, and the costs related to committing to the different regulation obligations under investigation are then calculated. This facilitates the possibility to calculate the break-even price[1] and the marginal cost [2] a hydropower producer faces when committing to regulation obligations.

[1] See chapter 3.7.3 for a definition of the break-even price

[2] See chapter 3.7.4 for a definition of the marginal regulation cost

2 Theory

For more extensive theory describing the subjects mentioned in chapter 2.1, 2.2 and 2.3, see appendix B, which is a copy of chapter 2 from [2].

2.1 The state of electric power systems

The state of the electrical power system can be assessed through investigation of the frequency quality, the voltage quality and the time of operation with reduced reliability (time outside the N-1 criteria). There are different measures that can be taken to influence these factors [4]. This report will focus on the frequency quality and the actions that can be taken to improve this.

In an electrical power system it has to be a perfect balance between production and consumption at all times. Any deviation from this requirement would lead to a frequency deviation from the nominal frequency in the system. If there is a production surplus compared with the consumption the frequency will increase, and if the consumption exceeds the production the frequency will decrease [5].

2.2 Increased frequency deviations

In the recent years the reliability of the grid in the Nordic region has decreased due to an increased proportion of time with frequency deviations.

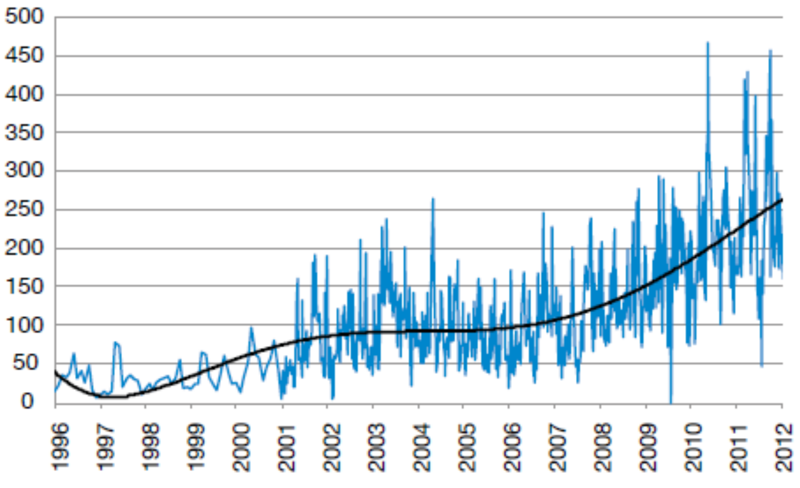


Figure 1: Development in frequency quality given in minutes outside 49.90-50.10 HZ per month [4], also used in [2]

There are several reasons behind this development [4]:

- Deviations around hour shifts will occur since the consumption changes continuously while the production in the spot market is specified for each hour and only changes around hour shifts.
- More HVDC-connections to the continent are installed, and this changes the power flow.
- Some participants are excluded from contributing with regulation capacity due to congestions in the grid. This leads to less available regulation capacity. With an increased load of the grid the recent years this problem increases.
- The grid is operated without the large hydropower units suited for regulation in an increased proportion of the time due to more and more HVDC-connections to the continent, more solar power and more wind power.
- 60-seconds oscillations occurs, probably caused by different responses on corresponding units in the Nordic region.

2.3 Regulation markets

To avoid variations in frequency caused by differences between production and consumption the TSO (Transmission System Operator) controls several regulation markets. Since it is almost impossible to control consumption on short notice, the regulation must lie on the production side. The regulation markets can be divided into different levels [6]:

- Primary regulation
- Secondary regulation
- Tertiary regulation

The actually name of the markets may vary from region to region but all regulation markets can be placed in one of these categories. The markets can be ranked after activation time and energy volume. Primary regulation has the shortest activation time and lowest volume, while tertiary regulation has the longest activation time and largest volume.

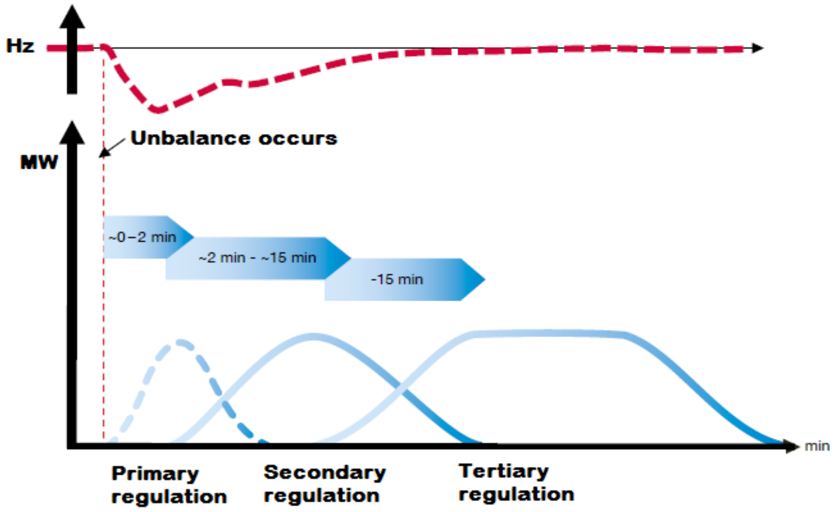


Figure 2: Regulation markets (adapted from [4]), also used in [2]

One of the main purposes of having several levels of regulation is to have the opportunity to free the regulation capacity with shortest activation time after it is activated to make the system ready for another change in consumption. Hence, all the markets are equally important for the frequency stability in the system.

The most important market for most power producers is the day-ahead spot market. However, producers who have the possibility to regulate their production on a short notice can increase the profit by participating in regulation markets in addition to the main day-ahead spot market [1]. This opportunity is especially well suited for hydropower producers with reservoir capacity. Run-of-rivers hydropower producers are not too well suited, especially not for delivering regulation with large energy volumes (i.e. tertiary regulation), since this requires the opportunity to store energy - in this case water - for later use. Generally there are two factors that must be met to make a power producer suited for participating in the regulation markets and deliver regulation capacity:

- The opportunity the change the production quickly.
- The opportunity to store energy (water) for later use.

2.4 Market structures

Different power markets have different structures regarding time horizons, time blocks and other properties describing the market. This influences how a hydropower producer can utilize these markets and how they should be included in the optimization process.

2.4.1 Time horizons in different markets

The day-ahead spot market, where the delivery of power for the following day is traded between buyers and sellers in a power exchange market, is the main market for most hydropower producers. As indicated this is a day-ahead market meaning that both buyers and sellers specify their bids for the following day based on the information available today. However, the regulation markets often have either a long or short time horizon meaning that hydropower producers must specify the bids in these markets based on different available information than in the day-ahead spot market. The time horizons span from weeks in markets with long horizons to hours in intraday markets where the bids are given to the TSO just before the delivery hour [6].

2.4.2 Time blocks in different markets

Some regulation markets also have time blocks meaning that the bid can be changeable within certain segments during the day, and for some markets the bid into one time block one day must be equal the bid into the same time block the next day for the entire time horizon of the market [7].

Another factor that complicates this process is the differences regarding the length of the time blocks defining each market. In the day-ahead spot market the time blocks last for one hour meaning that the biddable volume must be constant within each hour. In the regulation market the time blocks are often longer meaning that the bid must be constant for a longer period of time. Most regulation markets have time blocks lasting from one hour up to one week. For instance, the Swiss primary regulation market has a one-week continuous time block, while the Norwegian secondary regulation market has a structure consisting of three daily time blocks. These blocks may have different volumes, however the same block must be equal between each day of the one-week period [7], [8].

2.4.3 Symmetrical and unsymmetrical regulation markets

Symmetrical and unsymmetrical properties also complicate the issue of bidding into regulation markets. In a symmetrical regulation market the same biddable volume must be available as regulation capacity both upward and downward for the entire time block. In an unsymmetrical market it is possible to specify different upward and downward regulation capacity bids for each time block. Symmetrical properties can for instance be found in the Swiss primary regulation market, while the Norwegian secondary regulation market is an example of a regulation market with unsymmetrical properties [7], [8].

2.4.4 Challenges regarding different market structures

The main challenge with these issues is that the bids into the different markets must be specified based on different information about the future. Also, the bid into one market influences the bids into other markets where bids can be decided with more available information. If a hydropower producer bids into a regulation market with a one-week time horizon this restricts the flexibility in the day-ahead spot market for the upcoming week.

Due to the uncertainty regarding the future, especially when it comes to the price in the different markets as well as the inflow, a combination of all these properties discussed above complicates the process of how to distribute and bid the limited capacity into the different markets.

2.5 Stochastic vs. deterministic optimization of hydropower scheduling

When a hydropower producer optimizes the production for the upcoming period there are basically two different approaches that can be utilized; a deterministic or a stochastic approach. If a deterministic optimization approach is applied, all relevant input parameters must be specified. A stochastic approach on the other hand takes the uncertainty about the future into account and attempts to find a robust solution given all the

possible outcomes of the uncertain input parameters. Since it almost always will be some uncertainty regarding the future - for hydropower producers this especially concerns future price and inflow - a stochastic approach will in most cases be desirable from a theoretical point of view.
However, deterministic models are often used since these are easier to utilize in the real life. Stochastic models need good input data with realistic probability distributions of the uncertain parameters to be useful in real life, and they are often more difficult to solve. Results from such models may also be harder to structure and utilize in practice. The combination of these factors can often be challenging and deterministic models are therefore often used instead [9].

2.5.1 The scenario tree

Scenario trees are often used to describe all possible outcomes and the corresponding uncertainties in stochastic models.
Generally, a scenario tree can be built up by basic elements such as nodes, branches, full paths, and step.

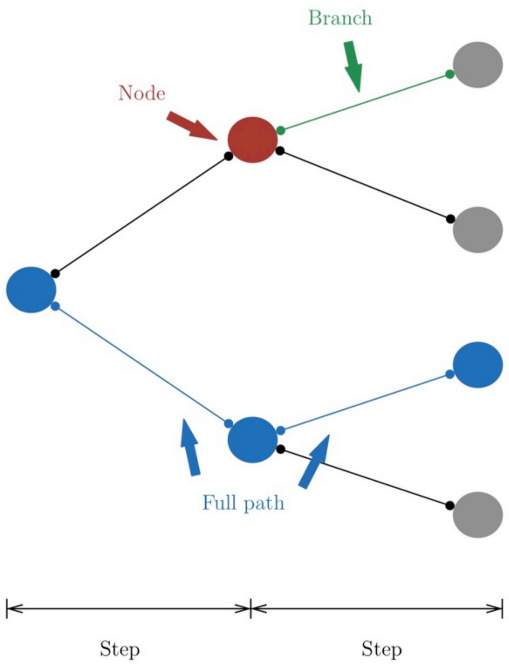


Figure 3: Description of a scenario tree and its main properties

A node is a point in the scenario tree, which contains information about the uncertain parameter. Branches connect the nodes. The number of branches going out from a node describes how many possible outcomes the uncertain parameter can get given the value of the parameter specified in the given node. All branches are described by a probability stating the likelihood of the value lying at the end node of the given branch being the next realized value of the uncertain parameter.

Steps, also known as stages, can be interpreted as the fundamental time unit describing the scenario tree. Generally, new information will be revealed between each step in the scenario tree. A full path is a unique way through the scenario tree going over all the steps defining the tree. In a stochastic model the total number of possible unique scenarios will be equal the total number of unique full paths in the scenario tree. Thus, a full path is equal to an individual scenario [10].

When a scenario tree is used as input to a linear stochastic optimization model describing the uncertain input parameters, the model should do the optimization based on the current available information in the scenario tree.

Some stochastic models receive new information when the model moves to the next step in the scenario tree, revealing previous uncertain parameters so that the optimal result may change. A challenge with such models is how to choose which branch to follow next when the model has optimized one step based on the current available information and should move to the next step. One way to solve this is through a random function in combination with the probability describing each output branch from the given node, also referred to as a probability function.
Due to this randomness the optimal result from this kind of stochastic optimization model will change to some degree between each execution of the model.

2.5.2 The Monte Carlo method

A more advanced approach to analyze the results from a stochastic model is by utilizing the Monte Carlo method. This method is a useful approach to establish a distribution of results concerning a function or a model that contain probabilistic, stochastic factors. Hence, the method is well suited for analyzing stochastic optimization of hydropower scheduling [11].

Generally the method implies pulling a random number from an unknown probability distribution a large number of times. This sample can them be used to find a statistical estimate of the random variable.
In stochastic optimization the main purpose of a Monte Carlo-simulation is to create statistical estimates of the stochastic variables that can be used to calculate confidence intervals [12].

This can be done by executing the model a large number of times, saving the output data for each execution and analyzing the overall results.

2.5.2.1 Key figures from a Monte Carlo-simulation

One of the main challenges when considering a stochastic model that iterates through a scenario tree, utilizing a probability function deciding the further path through the tree, is that the results from the model will change to some degree between each execution of the model.

The results from the model consist of a large number of decision variables. If a stochastic model is utilized these can be interpreted as stochastic variables due to the probability function within the stochastic model. The simplest way to solve this issue is to execute the model a large number of times and calculate the mean and the variance of the sampled data. These are some of the key figures that are useful to analyze from a Monte Carlo-simulation. To be able to calculate these, a set of sampled data is needed, and is gained through this simulation strategy.

In statistics a sample is a subset of a population [13]. In the case of a stochastic hydropower optimization the population should be interpreted as the solution space for the results from the stochastic model and the sample should be interpreted as the results from a large number of executions of the model, i.e. a Monte Carlo-simulation. This is applicable both for the decision variables from the model and the resulting economic parameters.

From a theoretical point of view it is assumed that the population - the solution space from the stochastic model - has an expected value and a variance. If the sample is of a sufficient size, it can be assume that the expected value of the random variable equals the mean of the sample and that the variance of the random variable equals the sample variance [13].

Another key figure that is interesting when analyzing the results from a Monte Carlo-simulation is the mode. This is the value in a sampled data that appears most often, i.e. in stochastic optimization this is the solution with the highest probability.

2.5.2.2 Sampling distributions

If it is assumed that a result from the stochastic model is a stochastic variable with a given probability distribution, it is possible to estimate this distribution. This is called a sampling distribution [13].
If the model is executed a large number of times, i.e. by utilizing a Monte Carlo-simulation, it will be possible to find the probability distribution of the results. From a hydropower producer's point of view it is useful to know how sensitive the optimal solution is to changes in the outcome of the stochastic input parameters. This can be described through this distribution.

In a real life case like stochastic hydropower optimization it is unlikely that the results from a Monte Carlo simulation will fit a theoretical probability distribution perfectly. Nevertheless, in some cases it can be useful and interesting to approximate the results to a mathematically defined probability distribution since this will facilitate further analysis of the results.

2.5.2.3 Confidence intervals

If the probability distributions of the solutions is known it is possible to establish a confidence interval. This interval is calculated from a set of sampled data taken from a population and gives an estimate range of values that will include a stochastic variable drawn from this population with a probability equal to the confidence level. The confidence level is a distinctive number describing the confidence interval. This value gives the probability of drawing a random variable from the probability distribution within the estimated range of value describing the confidence interval [14].

3 Case study

This study investigates how a hydropower producer in charge of a watercourse could optimally bid into a spot market with uncertainty, when also attending regulation markets considering different amounts of reserved regulation capacity obligations.

Foreseeing the future prices in spot markets is a difficult task, dependent on many variables. Thus, the price relies very much on uncertainty. A way to describe this uncertainty is by introducing a stochastic price. The outcome of such a price is based on probabilities, and uncertainty of the possible outcomes increases further into the optimization period.

When introducing yet another variable in the form of an additional biddable regulation market, defining a regulation obligation that must be fulfilled, the problem becomes even more complex and a optimization model is therefore developed with the goal of providing decision support for hydropower producers in these situations.

In this study, a stochastic spot price time series was reduced into a scenario tree of spot prices with corresponding probabilities. A stochastic multi-stage optimization model was then developed to take such scenario tree prices into account when considering optimal production patterns in a day-ahead spot market. This model was then implemented in the programming language Mosel, and then solved by IVE-Xpress[3].

The model takes the fulfillment of obligations for a symmetric regulation market into account, making sure the production plan has sufficient capacity at all times throughout the period to provide the obligated regulation. The symmetric regulation market has a continuous time-block lasting throughout the entire period – yielding the regulation obligation to be constant at all times – while the spot market accepts different bids for each hour of the period.

[3] http://www.fico.com/en/products/fico-xpress-optimization-suite/

3.1 The price reduction process and the scenario tree creation

3000 possible spot market prices were created using the approach described in [15]. They describe 9 days of the Norwegian spot price starting at week 35 of 2010, and were reduced into a scenario tree with characteristics mentioned in Table 5, using the SCENRED 2-tool[4] [16].

The original amount of spot price time series was reduced into a selection of 512 spot price time series – also defined as full paths in the scenario tree - each with a corresponding probability describing the likelihood of the price scenario to occur.
It was chosen to double the amount of branches for each incremental step of the scenario tree, starting with two branches in the first step, ending with 512 possible branches in the last step. This is illustrated in Figure 4.

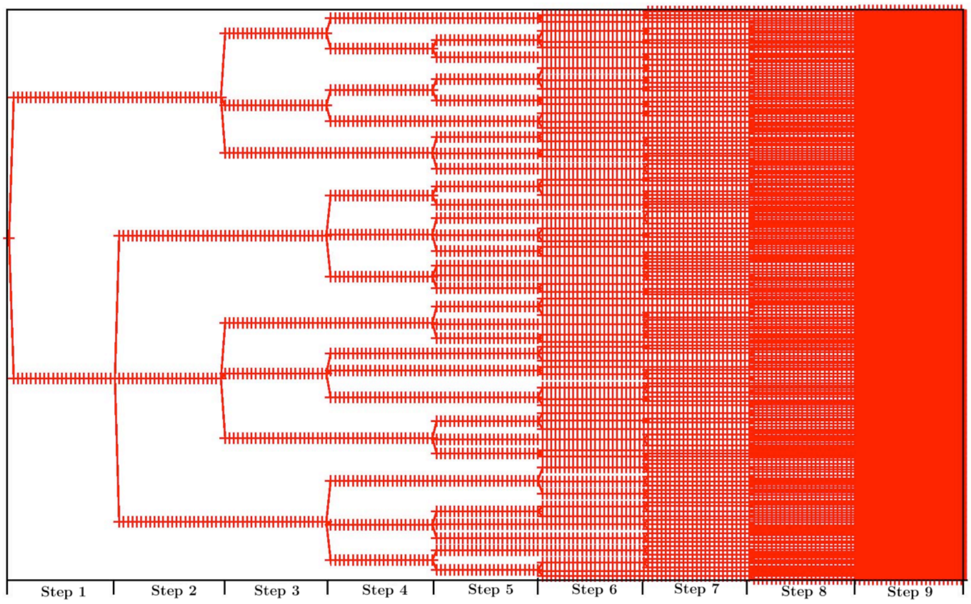


Figure 4: The reduced scenario tree with corresponding step lengths

Table 5: The amount of possible branches in the reduced scenario tree for each step

Step	1	2	3	4	5	6	7	8	9
Branches	2	4	8	16	32	64	128	256	512

Of course, the number of branches that is possible for each step is given the actual outcome of the previous step. The model excludes the

[4] http://www.gams.com/dd/docs/solvers/scenred2.pdf

branches that get unavailable when new information arrives, stating which way the price actually went. Each step consists of 24 hours, making a step equal to a full day.

It is noteworthy that all the discussed properties of the scenario tree are alterable within the model itself, so it could easily have another way of dividing branches and describing step lengths. However, for a short-term stochastic multi-stage optimization model, these characteristics prove plausible and easily understandable.

For an extensive explanation of the process of reducing spot price time series from SCENRED and making the new scenario spot price series with corresponding probabilities compatible with the optimization model, valuable information and an example is found in appendix 11. This appendix is important for the further comprehension of the developed model, and should be read.

3.2 The possible price outcomes

Figure 5 illustrates the development of the stochastic price, given the scenario reduction mentioned in chapter 0. The shaded area shows the maximum and minimum possible outcome of the stochastic spot price during each hour. The scope of opportunities increases with each step, as more branches are realized along with the uncertainty of the price itself, which increases further away from the present time.

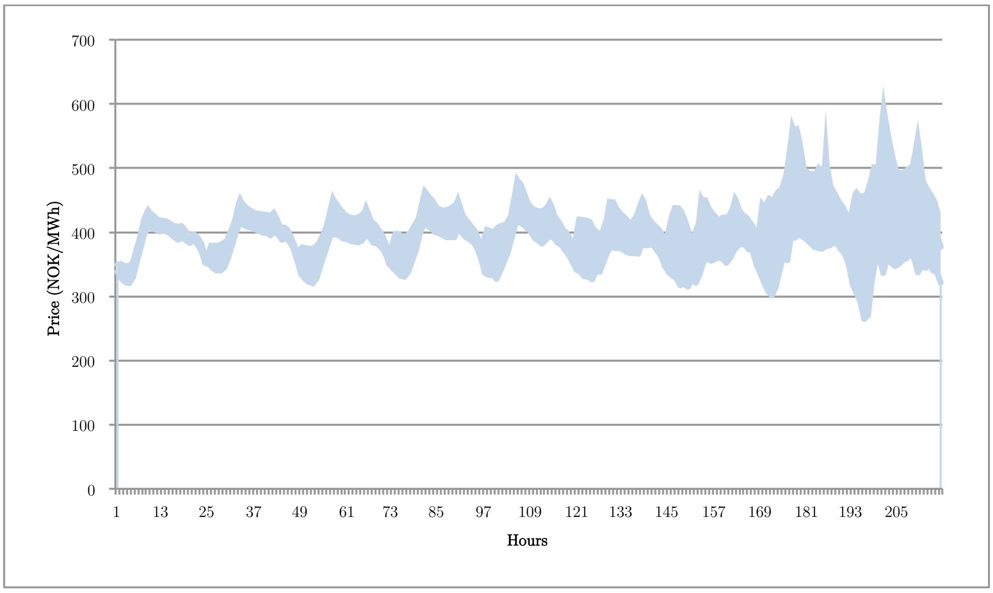


Figure 5: The possible price outcomes for the reduced stochastic scenario spot price

3.3 The watercourse

The same watercourse as in the preliminary specialization project [2], made possible because of the cooperation with Powel AS, is still used as the basis for all technical aspects related to the case study.

It is an anonymous watercourse with authentic characteristics, and consists of two reservoirs with two corresponding stations. The upper reservoir, Reservoir 1, is over twelve times as large as the lower reservoir, Reservoir 2, and the installed discharge capacity on Station 1 below Reservoir 1 is over twice as large as the installed discharge capacity on Station 2 below Reservoir 2. This unbalance between reservoirs and stations yields some difficult situations for certain outcomes, but on the other hand it can give more interesting results compared to a watercourse where the topology do not vary as much.

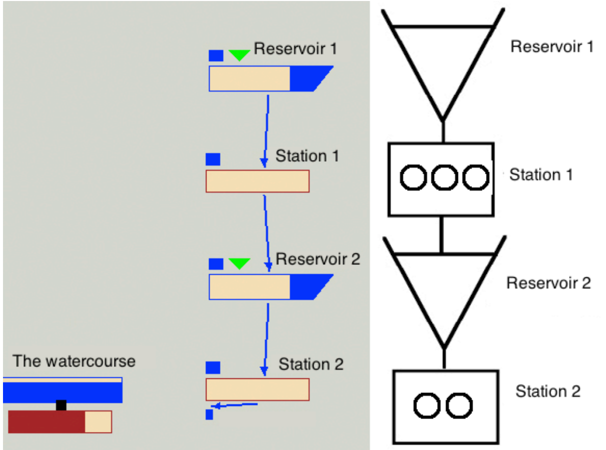


Figure 6: An overview of the watercourse, left: from Powel Sim, right: basic sketch

Table 6: Reservoir properties

	Reservoir 1	Reservoir 2
Volume [m^3]	33625000	2700000
HRV/LRV [masl]	1335/1250	865.2/852

Table 7: Station properties

	Station 1	Station 2
Number of units [no.]	3	2
Unit capacity [MW]	110	44,1
Station capacity [MW]	330	88,2
Head [m]	435-470	360
Turbine type	Francis	Francis
Maximum discharge [m^3/s]	26	12

Due to some simplifications in the model, the same relative efficiency curve is used for all units at all stations.
This is a plausible approximation since all Francis turbines have the same characteristic efficiency curve, and each unit's maximum discharge determines where on the efficiency curve it lies. A constant head factor is used at both stations.

3.4 Other general input data

Other general input data to the model that is important for understanding and investigating the results is presented below. To achieve a consistent analysis these were held constant throughout the analysis.

- **Water value:** The water value is assumed known from a long term and seasonal model. In practice it was set to the average of all spot price series in the presented scenario tree.

- **Minimum unit production:** The minimum production was set to 50 % of the maximum production on each unit.

- **Initial reservoir volume:** The initial reservoir volumes of both reservoirs were set to 50 % of maximum reservoir volume.

- **Maximum bypass:** This was set to a large value making the restriction none binding.

- **Maximum reservoir regulation per hour:** This was set to a large value making the restriction none binding.

- **Maximum ramping:** This was set to a large value making the restriction none binding.

- **Startup costs:** The startup cost of each unit was set to 5000 NOK.

- **Inflow:** Historical inflow data was used for this specific watercourse.

The values of the startup cost and the inflow are real values describing the watercourse gained access to through the cooperation with Powel AS.

3.5 Regulation obligations

Forecasted prices for regulation markets are difficult to predict. [17] indicates that pricing ancillary services such as balancing markets are challenging. In the preliminary specialization project [2], historic price data was used as input to the model regarding the regulation markets. However, this is not a satisfactory approximation when optimizing against future market prices. Therefore, using regulation capacity obligations as input is preferred. When using such an approach, it is possible to find the expected cost – due to less flexibility in the spot market – related to different regulation obligations.

It must be noted that this study only estimates the cost of reserving the regulation capacity. Thus it is assumed that the regulation markets investigated upon only provides income from reserving capacity, and not for the energy possibly used if the regulation obligation is activated. Some markets do have this additional way of providing income, i.e. the Norwegian secondary regulation market [7], and if that is the case it should be taken into consideration when evaluating the results.

Different combinations of regulation obligations were carried out when testing the watercourse's ability to deliver obligated regulation capacity. A trade-off between reasonable values of obligations regarding the biddable volume in different regulation markets compared to the physical limitations of the watercourse, gave reasonable values for further investigation [7], [8].

Regarding this specific watercourse, there was an incentive to challenge the behavior of the watercourse as much as possible within reasonable limits. Units on Station 1 can provide a maximum of 27.5 MW of symmetric regulation capacity, while units on Station 2 can provide 11 MW, hence the watercourse can theoretically deliver a total symmetric regulation capacity of 104.5 MW. This value however, is probably unrealistic. A range of obligations taking this into account was being investigated, making sure the highest regulation obligation succeeded the maximum capacity of both units on Station 2. Table 8 shows the combinations of regulation obligations mainly investigated upon.

Table 8: The combination of regulation obligations mainly investigated

Obligation [MW]	0 (Spot only)	6	12	18	24

In addition to this, all integer values between 1 MW and 25 MW were investigated in order to calculate marginal costs. Only monetary results were taken into account in these incremental model executions.

3.6 Approaches when creating output

The goal behind this study was to observe the optimal production pattern with and without regulation obligations present, considering an uncertain stochastic spot price. The developed model also creates detailed results in terms of the behavior of the entire watercourse, i.e. unit production, reservoir levels, total profits, the needed break-even price in the regulation market to cover the loss in optimal production only considering the spot market, and the marginal cost of increasing the regulation obligation. Many of these results are investigated thoroughly, however some of the most detailed results are only used as guidance for the overall analysis.

3.6.1 Comparison between regulation obligations

One of the main decisions a hydropower producer needs to take is the amount of regulation obligations to provide into a regulation market. A comparison between different levels of regulation obligations is therefore vital in order to study the different effects such obligations have on the watercourse and on the expected monetary values.

3.6.2 Stepwise decision variables

Given the right output data, decision variables from each individual step can be examined, providing the chance to observe how the model corrects itself after each step when new information is revealed. This information is useful for understanding how the model works, and to observe how the behavior upon the uncertain market is conducted. These results also indicate the value of proper information about the future.

3.6.3 Monte Carlo-simulations

In order to obtain a good distribution of the possible outcomes of the stochastic model, a large amount of optimizations have to be made. These results are needed when calculating the expected value and the confidence interval of the needed regulation price in regulation markets to make a regulation obligation profitable – i.e. the break-even price - when reserving the obligated capacity. Such multiple optimizations were made with the different combinations of regulation obligations mentioned above.

3.7 Wanted output for further analysis

Below is a more detailed list of the output wanted in order to conduct a more in depth analysis of the model executions considering stochastic variables. When processing such vast amounts of data, it is necessary to structure it properly in order to achieve useful results.

3.7.1 Optimal production patterns and reservoir behavior

One of the main and most important results for a hydropower producer is the optimal production pattern. The reservoir behavior is also essential. In addition, physical variables such as discharge, bypass and flood are retrieved to support the analysis.

All these factors will be calculated for all given levels of regulation obligation.

3.7.2 Change in spot market profit

When a hydropower producer commits to deliver a certain regulation obligation for a given period, the income from the day-ahead spot market will decrease since this restricts the producer's flexibility in this market.

To decide if it is profitable to participate in a regulation market or not it is interesting to know how much the income from the spot market is reduced if a given regulation obligation is committed upon.

3.7.3 Break-even regulation price

When the change in spot market profit for different levels of regulation obligation is known, it is possible to calculate the break-even price for these levels of obligations. This should be interpreted as the necessary price to make a given regulation obligation profitable for the hydropower producer. It is calculated by comparing the objective value where no regulation obligation is included with the objective value considering the given regulation obligation

$$P_{break-even} = \frac{Z(Y_0) - Z(Y_{sym})}{Y_{sym}\,H} \tag{1}$$

where $P_{break-even}$ is the needed price in the regulation market to break even considering the loss of spot market profit, $Z(Y_0)$ is the objective value of the model only considering a spot market, $Z(Y_{sym})$ is the objective value of the model also considering the current regulation obligation Y_{sym} , and H is the number of hours in the optimization period.

If the hydropower producer has some kind of forecast for the prices in the regulation market this provides a good decision tool for deciding if a given regulation obligation is profitable to commit upon or not.

3.7.4 Marginal regulation cost

If the difference between the various levels of regulation obligation is set to one basic unit (1 MW) it is possible to calculate the marginal cost of delivering regulation obligation as a function of the regulation obligation. This is an approximation of the derivative of the cost function and should be interpreted as the cost of increasing the regulation obligation with one unit.

From basic microeconomic theory it is known that it is profitable for the producer to increase the delivered volume as long as the price is above the marginal cost [18]. Hence, this factor is suited for the hydropower producer when deciding the optimal level of regulation obligation that should be bid into the regulation market if some kind of forecast for the given regulation market is available.

4 Model description

This chapter discusses and describes the developed model for bidding into a day-ahead spot market considering regulation markets. In addition to this, appendix D and E includes both a compact mathematical model description as well as the implemented code used to execute the model and obtain results.

4.1 Implementation techniques and pre-assumptions

The technical aspects of the model presented in chapter 3.3 - regarding the design of the watercourse - are based on the model developed during the specialization project [2]. The efficiency curves of the turbines are implemented using a P-Q-curve (power-discharge-curve), doing a piecewise linearization due to the optimization solving algorithm's limitations of having a linearized problem.
Some simplifications regarding the model have been made as well;

- Constant head factor of reservoirs during production
- The same relative efficiency curve for all units
- No change in use of water when regulation is activated
- Cascade-connected reservoirs
- Best-point strategy when calculating the end reservoir volume
- Water values are assumed known and independent
- All markets are assumed perfect

For a more extensive description of the implementation techniques and pre-assumptions, see appendix C, which is a copy of chapter 4.1 and 4.2 of [2].

4.2 The approach to a stochastic multi-stage optimization formulation

When considering stochastic multi-stage optimization models, the actual realized result from multiple optimizations must be taken into account and realized step by step, as new information is stepwise revealed.

This model iterates through multiple steps by doing an individual optimization for each current step length. A step length spans from the start of the current step until the end of the entire optimization period.

For the first step length, the entire optimization period is considered, but only the optimal decision variables from the first step are specified for the final result. For the subsequent step, the model does not take the previous step into account when optimizing, but sees only a period lasting from the start of the second step until the end of the last step. The decision variables are then specified only for the first step that the model see - that is from the start until the end of the second step. The final optimization sees only the step length of the last step, spanning from the start of the last step until the end of it.

The decision variables from the last step length are then specified, and all the determined decision variables are then combined forming the complete actual realized decision variables.

If each step consists of 24 hours, the first optimization being done when standing in the first step will optimize from hour 1 of day 1 until hour 24 of the last day within the period. However, only the decision variables from hours 1-24 would be specified. When standing in the last step, hour 1 until hour 24 of the last day is optimized upon, and the corresponding decision variables are specified. Variables not dependent of hours but only steps (e.g. the end reservoir volumes) are specified only in the last step - they are purely guidance variables for the previous steps needed to optimize correctly.

To be able to compare solutions with a range of different regulation obligations, it is important that the same stochastic outcome is considered for each obligation if the comparison should have any consistency. For each individual step, the model therefore iterates over all regulation obligations, making the total amount of individual optimizations equal to the number of steps times the number of regulation obligations considered.

4.3 The mathematical formulation of the stochastic day-ahead market model considering regulation markets

$$\max \frac{\sum_{e\in\tilde{E}}\sum_{h=24t-23}^{H}\Pi^{e}P_{spot}^{e,h}x_{tot}^{t,\mathrm{o},h}}{\sum_{e\in\tilde{E}}\Pi^{e}}+\sum_{r\in R}W^{r}\alpha^{t,\mathrm{o},r}-\sum_{s\in S}\sum_{u\in U}\sum_{h=24t-23}^{H}C^{s,u}\beta^{t,\mathrm{o},s,u,h} \qquad \forall t\in T,\forall o\in O \qquad (2)$$

The objective function (2) maximizes profit from all price forecast scenarios that can occur given the node in the scenario tree that the model is currently standing on.

In addition, the value of the remaining water in the reservoir is taken into consideration, along with the costs of starting up units that are not currently active.

$$\gamma^{t,\mathrm{o},r,24t}=\gamma^{t-1,\mathrm{o},r,24t} \qquad \forall t\in T\neq t=1,\forall o\in O,\forall r\in R \qquad (3)$$

$$f^{\,t,\mathrm{o},r,24t}=f^{\,t-1,\mathrm{o},r,24t} \qquad \forall t\in T\neq t=1,\forall o\in O,\forall r\in R \qquad (4)$$

$$b^{t,\mathrm{o},r,24t}=b^{t-1,\mathrm{o},r,24t} \qquad \forall t\in T\neq t=1,\forall o\in O,\forall r\in R \qquad (5)$$

$$q^{\,t,\mathrm{o},s,u,24t}=q^{\,t-1,\mathrm{o},s,u,24t} \qquad \forall t\in T\neq t=1,\forall o\in O,\forall s\in S,\forall u\in U \qquad (6)$$

$$x^{t,\mathrm{o},s,u,24t}=x^{t-1,\mathrm{o},s,u,24t} \qquad \forall t\in T\neq t=1,\forall o\in O,\forall s\in S,\forall u\in U \qquad (7)$$

$$\beta^{t,\mathrm{o},s,u,24t}=\beta^{t-1,\mathrm{o},s,u,24t} \qquad \forall t\in T\neq t=1,\forall o\in O,\forall s\in S,\forall u\in U \qquad (8)$$

$$\zeta^{t,\mathrm{o},s,u,24t}=\zeta^{t-1,\mathrm{o},s,u,24t} \qquad \forall t\in T\neq t=1,\forall o\in O,\forall s\in S,\forall u\in U \qquad (9)$$

Equation (3)-(9) couples the solutions from previous steps, setting the decision variables of the actual reservoir volume, flood, bypass, discharge, unit production, running indicator and start indicator from the last hour of the previous step equal to the decision variables in the hour before the first hour of the new step optimized upon. This is done in order to make the constraints work flawlessly between each step optimization and to obtain a continuous solution.

$$x_{tot}^{t,\mathrm{o},h}+Y_{sym}^{o,h}-\sum_{s\in S}\sum_{u\in U}X_{\max}^{s,u}\zeta^{t,\mathrm{o},s,u,h}\leq 0 \qquad \forall t\in T,\forall o\in O,h=24t-23..H \qquad (10)$$

$$x_{tot}^{t,\mathrm{o},h}-Y_{sym}^{o,h}-\sum_{s\in S}\sum_{u\in U}X_{\min}^{s,u}\zeta^{t,\mathrm{o},s,u,h}\geq 0 \qquad \forall t\in T,\forall o\in O,h=24t-23..H \qquad (11)$$

In order to obtain the given obligations to the symmetrical regulation market, equation (10) and (11) makes sure that there is sufficient regulation capacity both upward and downward to meet the committed obligation at any time.

$$\gamma^{t,o,r,h} + N^{r,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} - \gamma^{t,o,r,(h+1)} = 0 \tag{12}$$

$$t = 1, \forall o \in O, r = 1, s = 1, \forall u \in U, h = 24t - 23..H - 1$$

$$\gamma^{t,o,r,h} + N^{r,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} - \gamma^{t,o,r,(h+1)} = 0 \tag{13}$$

$$\forall t \in T \neq t = 1, \forall o \in O, r = 1, s = 1, \forall u \in U, h = 24(t-1)..H - 1$$

$$\gamma^{t,o,r,h} + N^{r,h} + \sum_{s \in S} \sum_{u \in U} q^{t,o,(s-1),u,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} + b^{t,o,(r-1),h} - \gamma^{t,o,r,(h+1)} = 0 \tag{14}$$

$$t = 1, \forall o \in O, \forall r \in R \neq r = 1, \forall s \in S \neq s = 1, \forall u \in U, h = 24t - 23..H - 1$$

$$\gamma^{t,o,r,h} + N^{r,h} + \sum_{s \in S} \sum_{u \in U} q^{t,o,(s-1),u,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} + b^{t,o,(r-1),h} - \gamma^{t,o,r,(h+1)} = 0 \tag{15}$$

$$\forall t \in T \neq t = 1, \forall o \in O, \forall r \in R \neq r = 1, \forall s \in S \neq s = 1, \forall u \in U, h = 24(t-1)..H - 1$$

$$\gamma^{t,o,r,h} + N^{r,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} - \theta^{t,o,r} = 0 \tag{16}$$

$$\forall t \in T, \forall o \in O, r = 1, s = 1, \forall u \in U, h = H$$

$$\gamma^{t,o,r,h} + N^{r,h} + \sum_{s \in S} \sum_{u \in U} q^{t,o,(s-1),u,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} + b^{t,o,(r-1),h} - \theta^{t,o,r} = 0 \tag{17}$$

$$\forall t \in T, \forall o \in O, \forall r \in R \neq r = 1, \forall s \in S \neq s = 1, \forall u \in U, h = H$$

Equation (12)-(17) describes the water balance of each reservoir in terms of initial water volume, inflow, discharge, flooding and bypass to other lower reservoirs. These equations assume a cascade-connected watercourse. The last hour of the previous step is included for the second step and its successors, making sure the optimal decision variables from the previous step is taken into account correctly, ensuring the same amount of water is in the reservoir when changing steps.

$$\alpha^{t,o,r} = \frac{\theta^{t,o,r} H_{tot}^{s,u} \rho g \eta_{\max}^{s,u}}{3.6 \cdot 10^{9}} \qquad \forall t \in T, \forall o \in O, \forall r \in R, \forall s \in S, \forall u \in U \tag{18}$$

The conversion from reservoir volume (m^3) to energy (MWh) is done by equation (18). Since it is unknown how the plant will operate with its future amounts of water, a best point strategy is used. This strategy says that the plant most likely will run at optimal efficiency levels, ensuring the corresponding amount of power being produced from the water it uses.

$$x^{t,o,s,u,h} \leq \left(\Gamma^{i} \frac{q^{t,o,s,u,h}}{Q_{\max}^{s,u}} + \Omega^{i} \right) Q_{\max}^{s,u} H^{s,u} \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t - 23..H, \forall i \in I \tag{19}$$

When calculating the power output, data from an efficiency curve has been converted into a P-Q-curve. A piecewise linearization of that curve, the linear cuts described by the matrices in equation (19), determines the power output a unit will get from the amount of discharged water.

$$B_{\min}^{r} \leq b^{t,o,r,h} \leq B_{\max}^{r} \qquad \forall t \in T, \forall o \in O, \forall r \in R, h = 24t-23..H \tag{20}$$

$$L_{\min}^{r} \leq \gamma^{t,o,r,h} \leq L_{\max}^{r} \qquad \forall t \in T, \forall o \in O, \forall r \in R, h = 24t-23..H \tag{21}$$

$$L_{\min}^{r} \leq \theta^{t,o,r,} \leq L_{\max}^{r} \qquad \forall t \in T, \forall o \in O, \forall r \in R \tag{22}$$

$$Q_{\min}^{s,u} \leq q^{t,o,s,u,h} \leq Q_{\max}^{s,u} \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \tag{23}$$

Equation (20)-(23) simply describes the minimum and maximum allowed levels of the respective bypass from reservoirs, reservoir volumes, end reservoir volumes and discharge from units,

$$\left| x^{t,o,s,u,h} - x^{t,o,s,u,(h-1)} \right| - G_{\max}^{s,u} \leq 0 \qquad t = 1, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-22..H \tag{24}$$

$$\left| x^{t,o,s,u,h} - x^{t,o,s,u,(h-1)} \right| - G_{\max}^{s,u} \leq 0 \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \tag{25}$$

$$\left| \gamma^{t,o,r,h} - \gamma^{t,o,r,(h+1)} \right| - A_{\max}^{r} \leq 0 \qquad t = 1, \forall o \in O, \forall r \in R, h = 24t-23..H-1 \tag{26}$$

$$\left| \gamma^{t,o,r,h} - \gamma^{t,o,r,(h+1)} \right| - A_{\max}^{r} \leq 0 \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall r \in R, h = 24(t-1)..H-1 \tag{27}$$

whereas (24)-(27) limits the maximum change in reservoir volume (regulation) and production (ramping). These constraints also take the optimal decision variables from the previous step's last hour into account making the coupling between steps flawless.

$$x^{t,o,s,u,h} - X_{\max}^{t,o,s,u} \zeta^{t,o,s,u,h} \leq 0 \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \tag{28}$$

$$x^{t,o,s,u,h} - X_{\min}^{s,u} \zeta^{t,o,s,u,h} \geq 0 \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \tag{29}$$

(28) and (29) limits the maximum and minimum production on each unit - as well as initializing a binary running indicator.

$$\beta^{t,o,s,u,h} - \left(\zeta^{t,o,s,u,h} - \zeta^{t,o,s,u,(h-1)} \right) \geq 0 \qquad t = 1, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-22..H \tag{30}$$

$$\beta^{t,o,s,u,h} - \left(\zeta^{t,o,s,u,h} - \zeta^{t,o,s,u,(h-1)} \right) \geq 0 \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \tag{31}$$

$$\beta^{t,o,s,u,h} - \left(\zeta^{t,o,s,u,(h+1)} - \zeta^{t,o,s,u,h} \right) \geq 0 \qquad t = 1, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23 \tag{32}$$

This binary indicator is again used in equation (30)-(32) to tell whether or not a unit is starting up at hour h. Due to the binary properties of the running indicator, the startup indicator remains continuous, as it will either be 0 or 1 anyway because of its presence in the objective function.

Avoiding a binary property for this variable reduces the calculation time of the model. If the unit is running in the last hour of the previous step, this is taken into account avoiding possible startup costs that should not occur.

$$\sum_{s\in S}\sum_{u\in U} x^{t,o,s,u,h} - x_{tot}^{t,o,h} = 0 \qquad \forall t\in T, \forall o\in O, h = 24t-23..H \tag{33}$$

For simplicity, the spot productions from all units are summed up in equation (33), enhancing the readability of the objective function.

$$\zeta^{t,o,s,u,h} \in \{0,1\} \tag{34}$$
$$\forall t\in T, \forall o\in O, \forall s\in S, \forall u\in U, \forall h\in H$$

$$x^{t,o,s,u,h}, x_{tot}^{t,o,h}, \alpha^{t,r}, \beta^{t,o,s,u,h}, \gamma^{t,o,r,h}, q^{t,o,s,u,h}, f^{t,o,r,h}, b^{t,o,r,h} \geq 0 \tag{35}$$

$$\forall t\in T, \forall o\in O, \forall r\in R, \forall s\in S, \forall u\in U, \forall h\in H$$

Equation (34) and (35) states the binary and non-negativity conditions of the variables.

4.4 A more thorough explanation of the model's behavior

This chapter gives an explanation of how the mathematical formulation was implemented into the programming language used. A short overview of the process is presented in the subchapters below. The complete implemented Mosel code with explanatory comments can be found in appendix E.

4.4.1 Defined terms used by the model

Scenarios: In this model a scenario is defined as one unique full path through the entire scenario tree describing the spot price. Hence, the total number of scenarios will be equal to the number of end-nodes in the scenario tree.

Steps: A step is in this model defined as a point where new information is revealed, thus giving a chance of changing decision variables for the remainder of the period. This model optimizes against a day-ahead spot market, and a step will therefore consist of 24 hours.

Obligation index: A counter used to iterate over the different levels of regulation obligations. Each obligation index represents one given level of regulation obligation. The regulation obligation associated with obligation index 1 should always be zero. This means it is a pure spot market optimization not taking any regulation market into account, needed for utilizing equation (1).

Model index: A counter that is utilized only when a Monte Carlo-simulation is carried out. The purpose is to create a loop forcing the model to run many times in order to carry out as many simulations as desired in the Monte Carlo-simulation.

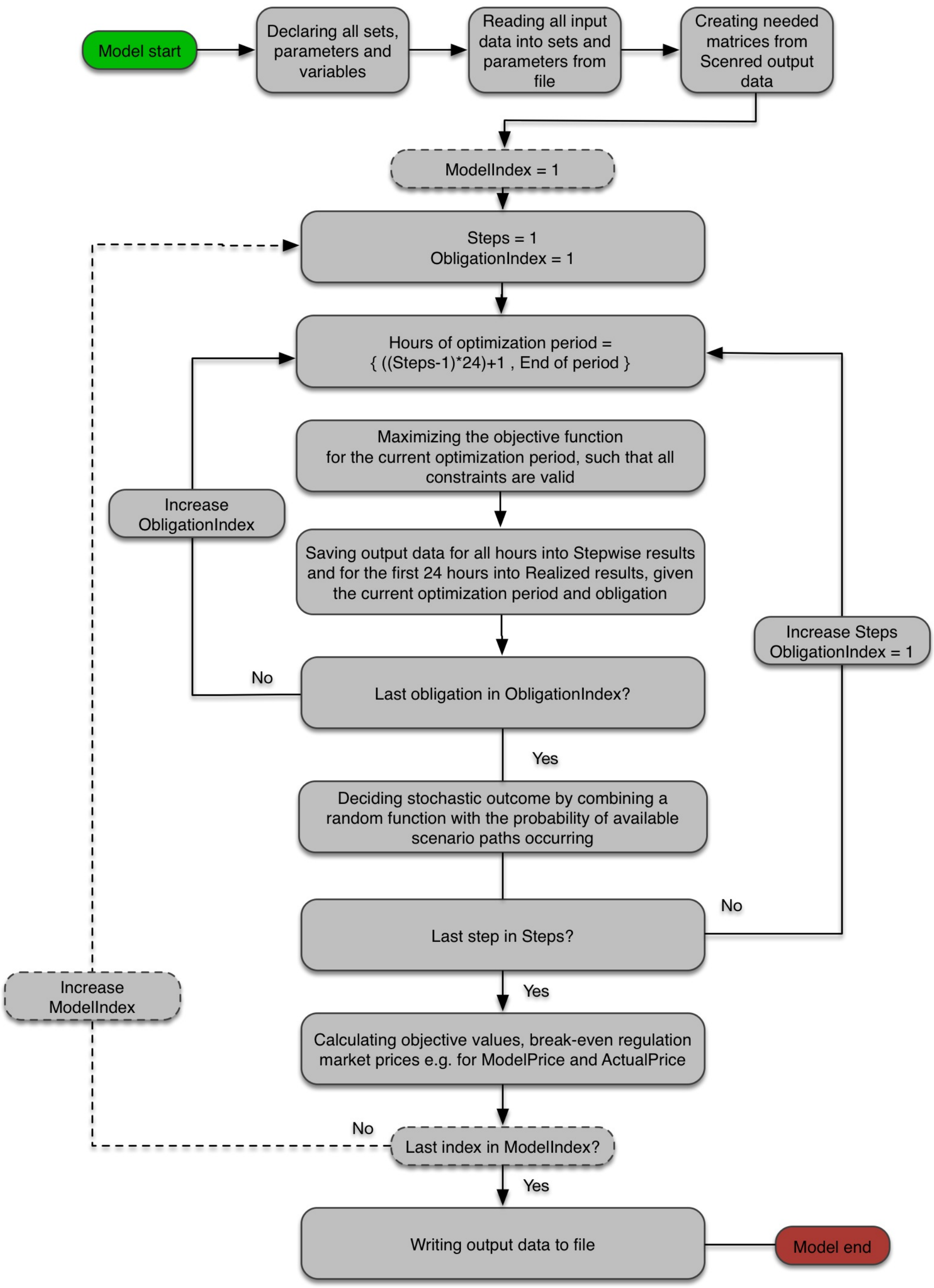
Model price: The price that the model sees for each step, and which it optimizes upon. It is a probability-weighted price where all scenario paths that are still possible to obtain are included.

Actual price: The simulated price that occurred for each step. After the model has optimized taking the model price into consideration, a probability function based on the probability distribution of the remaining scenario paths are utilized, deciding which way the price goes. A new model price is then calculated for the next step, based on the scenario paths that now are available given the outcome of the actual price for the previous step.

Realized results: The optimal decision variables for the first 24 hours of the current step length is stored into parameters used for consecutive steps and for the final result. These results represent the actual scheduled behavior of the watercourse.

Stepwise results: Results from each stepwise optimization. Only the first 24 hours is actually utilized for the final result, however the rest of the variables are useful for analytical purposes.

4.4.2 A flow chart of the execution of the model



Model start
Declaring all sets, parameters and variables
Reading all input data into sets and parameters from file
Creating needed matrices from Scenred output data
ModelIndex = 1
Steps = 1
ObligationIndex = 1
Hours of optimization period =
{ ((Steps-1)*24)+1 , End of period }
Maximizing the objective function for the current optimization period, such that all constraints are valid
Increase ObligationIndex
Saving output data for all hours into Stepwise results and for the first 24 hours into Realized results, given the current optimization period and obligation
Increase Steps
ObligationIndex = 1
No
Last obligation in ObligationIndex?
Yes
Deciding stochastic outcome by combining a random function with the probability of available scenario paths occurring
No
Last step in Steps?
Increase ModelIndex
Yes
Calculating objective values, break-even regulation market prices e.g. for ModelPrice and ActualPrice
No
Last index in ModelIndex?
Yes
Writing output data to file
Model end

Figure 7: An illustrative flow chart of how the model works

4.4.3 The pseudo code of the model

```
Declare all sets, variables, parameters and restrictions
Read input data from file
Create needed matrixes from SCENRED output data (appendix 0)
for each $m \in M$
    for each $t \in T$
        for each $o \in O$
            maximize objective function (equation (2))
            subject to all constraints (equation (3)-(35))
            if $t>1$
                for $h=24(t-1)$
                    Fix all variables to the solution from previous step
                    in the first hour of the current optimization period
                end do
            end if
            for each $h=[24(t-1),24t]$
                save all variables to Realized<variablename>
            end do
            for each $h \in H$
                save all variables to Stepwise<variablename>
            end do
            for each $h=[24(t-1),24t]$
                save  ModelPrice
            end do
            for each $h=[24(t-1),24t]$
                save ActualPrice
            end do
        end do
        Run the probability function to find out which branch to
        follow in the scenario tree to the next step.
    end do
    Calculate Objective values and break-even price based on both ModelPrice and ActualPrice
end do
Write output data to file
end model
```

4.4.4 An in depth explanation of the model's workflow process

This chapter gives an in depth explanation of the logic used when developing the stochastic model and describes how the model works in a thoroughly manner. It can be skipped, however it may prove to be useful reading in order to get a wider understanding of the later presented results.

Initial declaration of parameters and input data

First all sets, parameters, variables and restrictions that are going to be used in the model are declared, the input data is read from Excel files and matrices describing the scenario tree is created. The output from SCENRED cannot be utilized directly by the stochastic model. Hence, a subscript to manipulate these data into a format that can be utilized by the stochastic model is necessary. A further explanation of this method can be found in appendix 0.

The loop hierarchy of the model

The model index-loop is first established. This loop is only utilized when Monte Carlo simulations are carried out. Since the results from the model will vary to some degree due to the probability function within the stochastic model, the entire calculation process is carried out multiple times giving more reliable results.

The step-loop is then established within the model index-loop. Since the optimal solution will change when new information about the spot price is revealed when the model moves to the next step, a new optimization is carried out for each step.

Within each step an individual optimization should be carried out for all levels of regulation obligations. Hence, an obligation index-loop iterating over all levels of regulation obligations is needed. The given regulation obligation influences some of the physical restrictions, since it must be specified that the given regulation obligation must be available as rotating reserve within the system. To be able to compare the different regulation obligations the obligation index-loop must be inside the step-loop. The same outcome of the stochastic properties will then be used on all regulation obligations.

The objective function and basic physical constraints

The basic part of the linear model - an objective function and the necessary linear constraints needed to describe the physics of the watercourse and the other constraints affecting the solution space - is then specified within the model index-loop, the step-loop and the obligation index loop.

Coupling of variables between steps

When the model moves to the next step and new information is revealed a new optimization is carried out, but this time only from the current step to the end of the entire period. However, since it is assumed that the spot market under consideration is a day-ahead market the production for the next 24 hours must always be specified based on the current information about the spot price. Hence, there must be a coupling between the optimization in each step to be able to establish a continuous optimal solution for the entire period. This coupling is specified for all obligation indices. These restrictions specify that the decision variables in the hour prior to the first hour of the current step must be equal to the last hour of the previous step, considering the same regulation obligation.

Saving realized results

As mentioned earlier, the solution will change as new information about the spot price is revealed. However, it is always necessary to decide the optimal production for the next 24 hours based on the current available information. Therefore, to find the optimal solution for the entire period the solution for the next 24 hours must be saved from the optimization in each step. This is called the realized solution.

Saving stepwise results

Since there is a disclosure of new information regarding the spot price when the model moves to the next step, the optimal solution will change to some degree. Hence, it is possible to see how much the solution changes if the optimal solution for the entire optimization period is saved for each stepwise optimization. These solutions are named stepwise results. However, since the model only optimizes from the current step to the end of the entire period, these stepwise results will always be zeros for the hours already specified. Hence, it is only possible to compare the solutions in the hours that are not yet specified.

Model price and actual price

There are two price series that are interesting to save and compare from each execution of the model; the model price and the actual price. The model price is the price series that the model optimizes against in each step. It is the probability weighted average of all the price series that still can occur given the current node the model stands on in the scenario tree. This price series will change as new information is revealed when the model moves to the next step, because it will rule out approximately half of the scenarios that were possible in the last step.

The actual price is the price that occurred given the results from the probability function that determines the models movement through the scenario tree.

Recalculating the actual objective value

Since the model only optimizes over the time period from the current step to the end of the entire period, and the solution only will be realized for the first 24 hours of each step length, the objective value from the optimization in each step cannot be utilized directly. The values have to be recalculated based on the solutions that were previously saved in the model.

There are two approaches to calculate the objective value; either the model price or the actual price could be utilized. Both methods are used in the model, but the result should be interpreted in different ways. The objective value calculated based on the model price should be interpreted as the value of the objective function the hydropower producers should expect during the given optimization period.

The actual price however is not a calculated average but the spot price scenario that actually turned out to occur given the results from the probability function. The objective value calculated from the actual price should instead be interpreted as the realized objective value the hydropower producer actually observed standing at the end of the period looking back in time.

The probability function deciding the path through the scenario tree

In each step, the model is standing in a branching point in the scenario tree. There are multiple branches with different probabilities the model could follow during the next time step. To choose which branch to follow the model utilizes a probability function. This function utilized a randomly generated number between 0 and 1 and the probabilities associated with each branch to choose which branch to follow in the scenario tree when moving to the next step. The probability function should be outside the obligation index-loop but within the step-loop, to achieve some consistency in the analysis.

Calculating the break-even price

One of the reasons for calculating objective values is to be able to find the break-even price for each level of regulation obligation. This is the price that is needed to make it profitable for the producer to commit to the given regulation obligation. It is calculated by utilizing equation (1).

Since there are two ways to calculate the objective values there will also be two different break-even prices for each regulation obligation. As with the objective values these should be interpreted in different ways. Utilizing the model price, the break-even price should be interpreted as the price the producer should expect is needed in order to make it profitable committing to a regulation obligation. On the other hand, utilizing the actual price, the break-even price should be interpreted as the break-even price that actually turned out to be necessary to make it profitable to commit to the given regulation obligation standing at the end of the period investigating the past.

The model also makes it possible to see how the break-even prices develops when the model moves to the next step and new information is revealed. This can be done by investigating the stepwise results. However, if the given regulation market has a longer time block than the spot market this would not be interesting in the real world since the decision must be taken based on the information that was available in the past, not the information available during the period the regulation obligation applies for.

Calculating the marginal regulation cost

In addition to the break-even price the model calculates the marginal cost for the different levels of regulation obligations. This is the cost related to increasing the regulation obligation with one unit (1 MW). It is calculated as the difference in objective value between to incremental values of the regulation obligation dived by the number of hours in the optimization period.

Just as for the break-even price, either the actual price or the model price could be utilized to calculate the objective values used to calculate the marginal costs. If the model price is used the marginal cost should be interpreted as the expected marginal cost standing within the optimization period. On the other hand, if the actual price is used it should be interpreted as the marginal cost the producers actually faced standing at the end of the optimization period investigating the past.

Also, similarly as with the break-even price it is possible to see how the marginal cost changes as new information about the future spot price is revealed as the model moves through the scenario tree, by investigating the stepwise results.

Utilization of the model

If the model should be utilized in practice the hydropower producer should emphasize using the break-even price and marginal cost calculated based on the model price, when considering if a regulation obligation should be accepted or not. This is because it is less dependent of the outcome from the probability function, and hence is a better decision support than the results calculated based on the actual price. The most reliable solution can be found by carrying out a Monte Carlo-simulation and use these results to find the mean break-even price and marginal cost for each regulation obligation based on the model price.

5 Results and analysis

All results presented below are unaltered output data from executions done on the stochastic multi-stage optimization model described in chapter 4. The model was implemented using the optimization language Mosel, and executed in Xpress-IVE.

The following subchapters contain results and corresponding analysis of a stochastic routine. Given the uncertainty and variations a model with such properties yields, different situations occur for each execution of the model. However, the results show how the model behaves considering the stochastic outcome of the individual model executions presented.

The actual price that occurs with the introduction of new information for each step is compared against the model price. For each regulation obligation that is compared within the same stochastic outcome of the spot price, the optimal production patterns, reservoir behaviors, profit, break-even price and marginal cost are analyzed.

Analyzing how the model optimizes within each step is also interesting. Thus, the same subjects mentioned in the paragraph above are also investigated stepwise by examining how the outcome of the stochastic uncertainty affects the decision variables and corresponding results.

Lastly, to establish a more certain behavior of the model's result, Monte Carlo-simulations with a vast amount of executions yielding different stochastic outcomes were performed. The expected values of the break-even price and marginal cost were calculated. In addition to this, confidence intervals of the break-even price were created.

5.1 Comparison of prices

As mentioned in chapter 4.4.4, the model calculates an average price based on the possible scenarios that are available during each step called the model price.
However, after the completion of each stepwise optimization, a random event based on probability occurs, yielding one of the possible outcomes to have actually happened.

Figure 8 shows the difference between the simulated actual price and the probability weighted model price the model optimizes against. Dependent on which event to occur, the actual price will either lie below, above or almost on the model price. Of course, the model price is - like the actual price - also dependent on the stochastic outcomes due to it's probability weighted average of the still achievable spot prices given the current position in the scenario tree. But unlike the actual price, the model price has a more average based behavior regarding the steps not yet optimized.

In this case, the actual price turned out to be above the model price in steps 1, 5 and 7, respectively, and below in steps 2 and 8. For the remaining steps, the model price and the actual price were approximately equal.

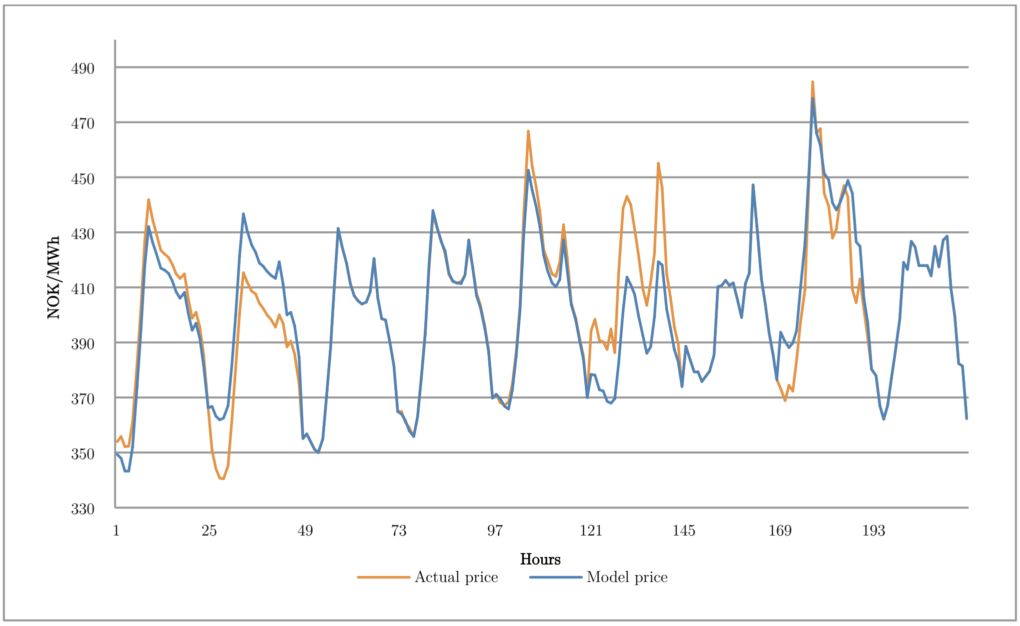


Figure 8: Comparison between the actual realized price and the probability weighted model price optimized upon

5.1.1 The model price vs. the actual price

For analytical purposes, the model price is most essential, being the price profile that the model optimizes against. The results based on this price should be the ones hydropower producers rely on when utilizing the model to decide the bidding strategy for the upcoming period.

On the other hand, the actual price is a good pinpoint to what could happen, giving a simulated scenario-based result that varies between optimizations. In a retrospective deterministic perspective, the actual price would yield the optimal results.

5.2 Comparison of regulation obligations

While maintaining the same stochastic outcome of the price scenarios, the model also takes different regulation obligations into consideration, giving a good overview of how the physical properties of the watercourse, as well as the economic output varies with an increasing amount of capacity reserved for the regulation markets.

5.2.1 Optimal production patterns

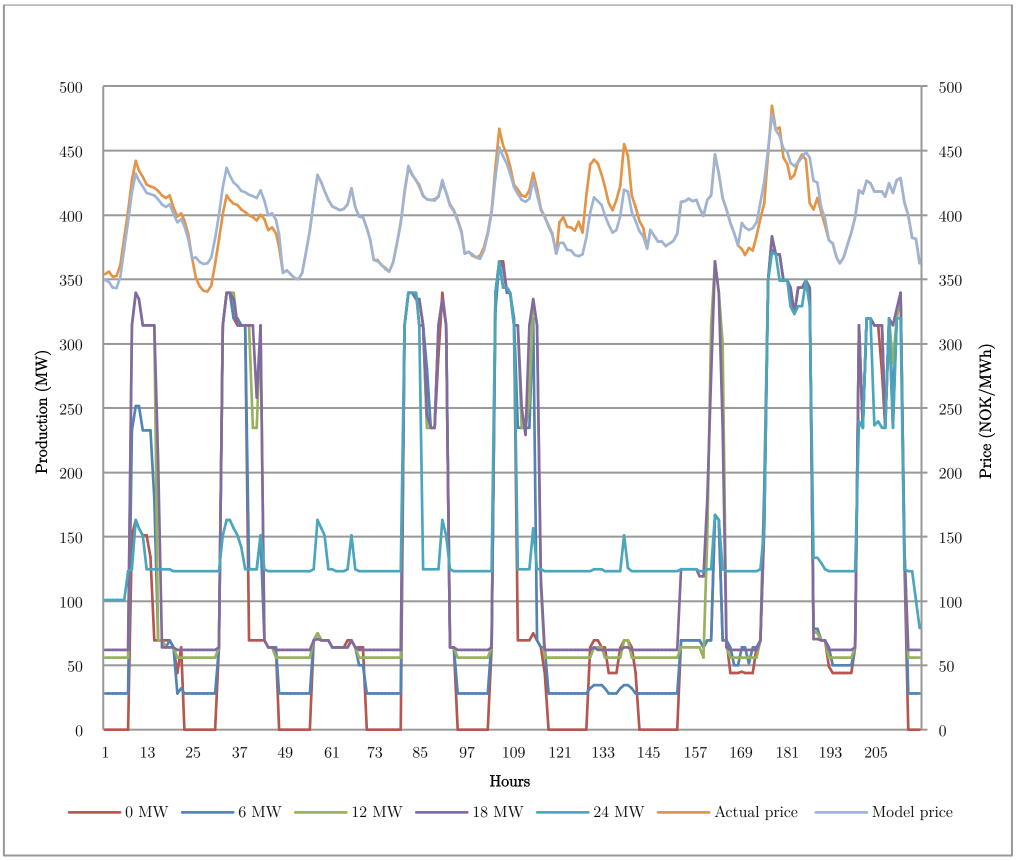


Figure 9: Optimal production pattern considering different regulation obligations along with both the model price and the actual price

The graphs displayed in Figure 9 describe the optimal, realized scheduled production over all steps, considering different obligations.
In this specific optimization, the model execution not considering regulation obligations is producing in high price curves, but has no production at all in time segments where the price is significantly lower than the water value. The combination of the difference between the spot price and the water value and the cost of restarting units defines when to produce and not produce.

The price is never sufficiently high enough to give the units an incentive to produce above their best point on the efficiency curve. The maximal capacity of the watercourse is 418.2 MW, however the peak production never succeeds 360 MW when all units are activated.

When introducing symmetric regulation obligations that must be fulfilled during the entire optimization period, no production stops are permitted. In low price segments, the trend is that the units combined produce just enough to be able to deliver the committed regulation obligation. Because the reduced flexibility in upward production does not intervene with the best point of the generators - the maximum production in the highest price peaks is just as high for all regulation obligations compared to the model execution where only a spot market is considered. However, since some units must run at all times, the model executions considering regulation obligations are taking advantage of the lower price peaks as well, trying to minimize the loss in the spot market as much as possible being that they must produce anyway.

5.2.2 Reservoir behaviors

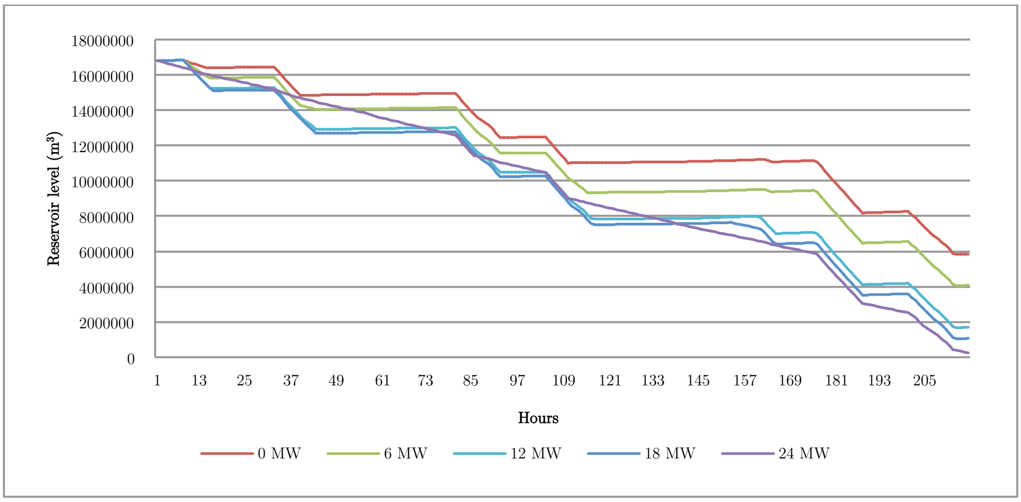


Figure 10: Levels of Reservoir 1 considering different regulation obligations

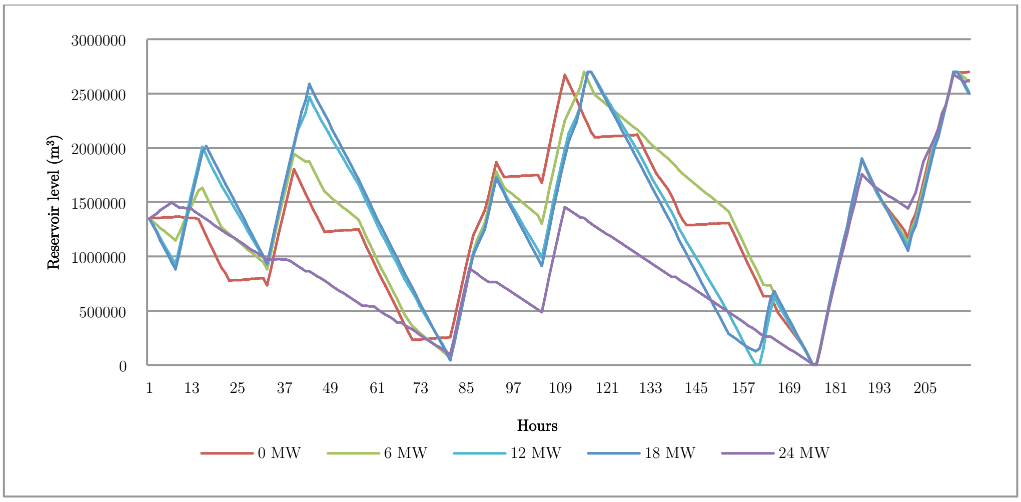


Figure 11: Levels of Reservoir 2 considering different regulation obligations

Considering the reservoir levels for each regulation obligation, the twelve times larger Reservoir 1 has a clear trend of using more water with each increasing level of regulation obligation. This is logical, because a higher obligation means a higher minimum production at all times, yielding a higher consumption of water.

Reservoir 2, with the physical properties of being quite small, is heavier regulated with the introduction of regulation obligations. The maximum regulation between hours and maximum unit ramping restrictions are not binding, causing a rapidly changing reservoir level of Reservoir 2. It is clearly used as a regulation reservoir. However the limitations regarding its size are still a bottleneck in the watercourse. This sensitivity analysis is further discussed in chapter 6.2.

5.2.3 The change in spot market profit

Optimizing only considering a spot market gives the best possible objective value in the model, reflecting the profit a hydropower producer sees in the day-ahead spot market.

When introducing regulation obligations, the profit from the day-ahead spot market decreases in accordance with increasing obligations. Adding additional constraints that restrict the solution space of the model – i.e. reducing the flexibility to benefit from price differences - will in most cases result in a lower value of the objective function, at best sustain its value [19].

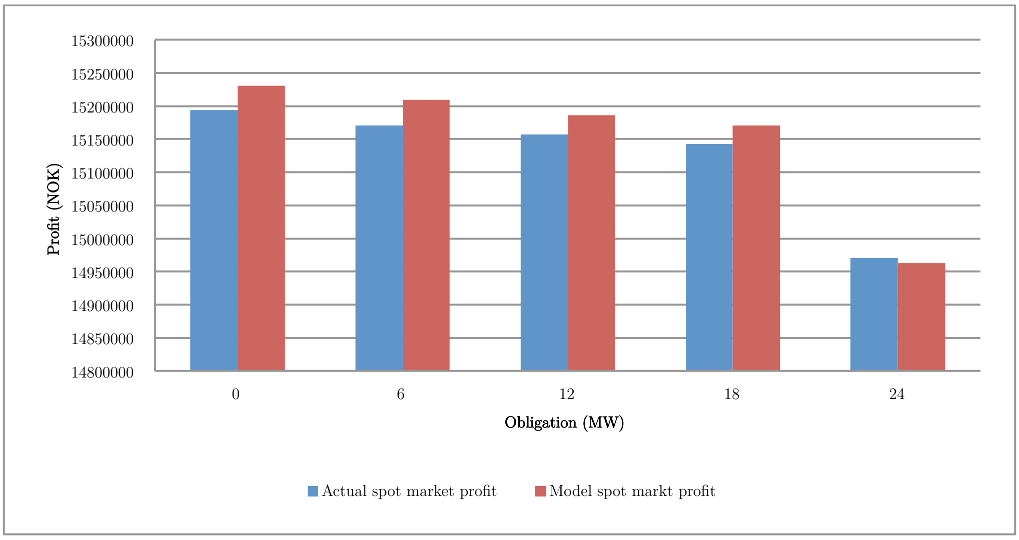


Figure 12: Total profits considering different regulation obligations

As shown in Figure 12, the profit from the spot market is gradually decreasing with each regulation obligation level. There is a vast increase of costs when an additional unit must start in order to fulfill a regulation obligation. This is due to the fact that the unit must produce at unfavorable times, often at a minimum production capacity plus the capacity needed for the obligation. For 24 MW of regulation obligation this is the case. Station 2 is preferred for delivering regulation capacity because of the risk of flooding Reservoir 2 when not producing. However, at 24 MW Station 2 is not able to cover the obligation alone, the two units has a maximal symmetric regulation contribution of 11 MW each, yielding at least 2 MW to come from the Station 1. This causes a lot of water to be discharged from the upper station during low price segments, thus making the change in profit between 18 MW and 24 MW extra noticeable. This is further discussed in chapter 6.4.

5.2.4 The break-even price in the regulation market

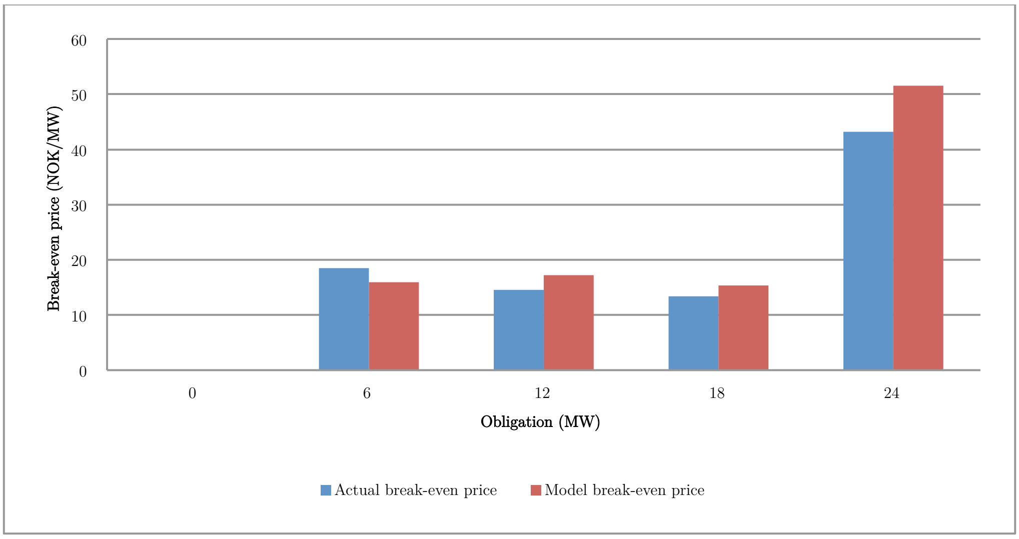


Figure 13: The break-even price in the regulation market considering different regulation obligations

Perhaps the most essential part of information a hydropower producer needs when deciding whether or not to participate in a regulation market, is an estimation of what the price in such a regulation market needs to be in order to make an increased profit compared to only bidding into the spot market. Figure 13 illustrates this break-even price needed in order to at least break even with the optimization only considering a day-ahead spot market.

The effect of reserving capacities above 2x11 MW – as mentioned the watercourse prefers to deliver regulation from Station 2's two units – is a drastic change in profit from 18 MW to 24 MW because Station 1 must start to deliver regulation as well, causing a much higher break-even price. This price will decrease again with an increased amount of regulation obligation, until another unit must be initiated in order to fulfill the obligation committed upon. Chapter 6.4 further discusses this.

5.2.5 Marginal costs when entering regulation markets

The marginal regulation cost is calculated as the decrease in the objective function when increasing the regulation obligation with one unit, in this case 1 MW, divided by the number of hours considered in the optimization period.

The decrease in the objective function is equivalent to the loss of profit in the spot market when reserving one more unit of regulation capacity for other markets.

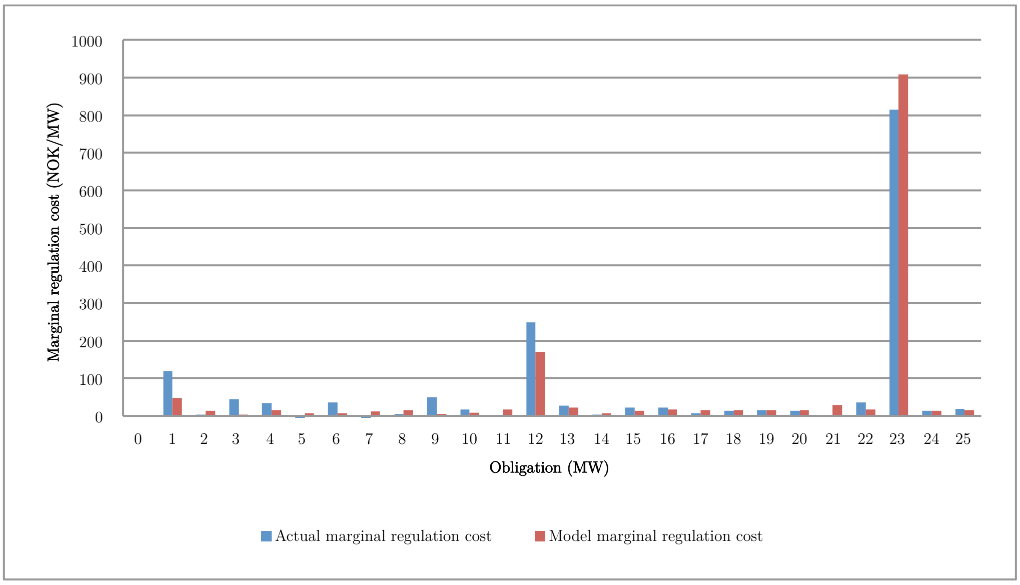


Figure 14: Marginal costs considering all regulation obligations between 1 MW and 25 MW

Figure 14 illustrates this marginal regulation cost. The observed trend is that at the exact obligation when a new unit must be utilized to deliver regulation capacity, the cost is noticeably increased. This is because of the minimum production constraint on the unit causing at least 50% production at unfavorable periods. However, the marginal price for the following regulation obligation is often significantly lower, as the new loss in the spot market when already deciding to utilize a new unit is not necessarily as high. As mentioned, see chapter 6.4 for a further discussion regarding this observation.

The use of such a graph is to see how much the regulation obligation can be increased, given that you already have decided upon delivering some regulation. It can be used in combination with the break-even price graph, which is further discussed in chapter 6.5.

5.3 Stepwise analysis

The model optimizes the production from the current step the model stands on to the end of the entire period. When the model moves to the next step some of the previously possible full paths through the scenario tree are excluded. Hence, more information about the future is revealed and the model gets a new scope of opportunities regarding the spot price it optimizes against.

Of course, in a day-ahead spot market the production in the next step must always be decided and specified based on the current information about the future. For each step, the optimal results for the next steps are specified as the optimal strategy. However, it is possible to change the production for the rest of the period as new information gets revealed at the beginning of the next step.

The model saves the results from each time step as it iterates through the entire period. Hence, it is possible to see how the optimal results change as new information about the future is revealed. This also gives an indication of how dependent the results are of the price compared to other factors such as the topology and physical restrictions in the watercourse. If only small changes in the optimal production occurs as new information about the future spot price is revealed, this indicates that the topology of the watercourse or other physical restrictions are influencing the results more than the actual market price. [1] indicated that this often is the case.

5.3.1 Stepwise changes in the model price

The price profile that the model optimizes upon changes with each step, as new information is revealed. In the first step, all 512 price scenarios are probability weighted, but in the successive steps, many scenarios are no longer achievable, thus not taken into account when probability weighing the remaining possible scenarios.

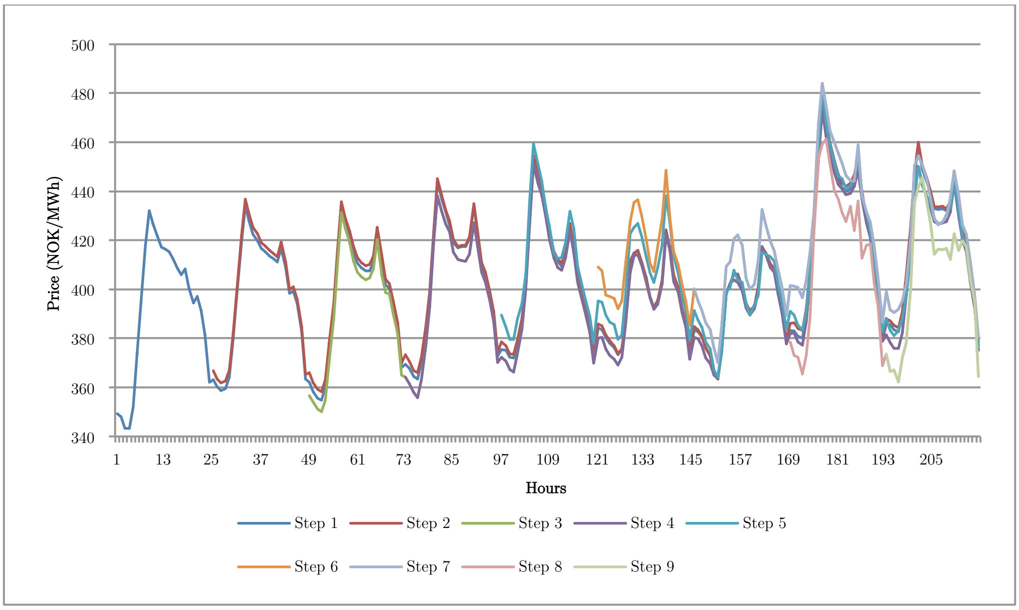


Figure 15: The model price for each step during a stochastic optimization

Figure 15 illustrates how the model price is affected by the different price profiles taken into consideration for each stepwise iteration. For instance, when considering the possible scenario prices in step 2, some scenarios are no longer available dependent on where in the scenario tree the random event tells the model to stand. This yields a higher probability weighted price for the rest of the period than when considering the possible scenario prices in step 1. This effect is observed at every step change, and the most prominent occurrence is seen when considering step 6. The magnitude of these effects will change between each execution of the model due to the discussed probability function within the stochastic model.

5.3.2 Stepwise optimal production patterns

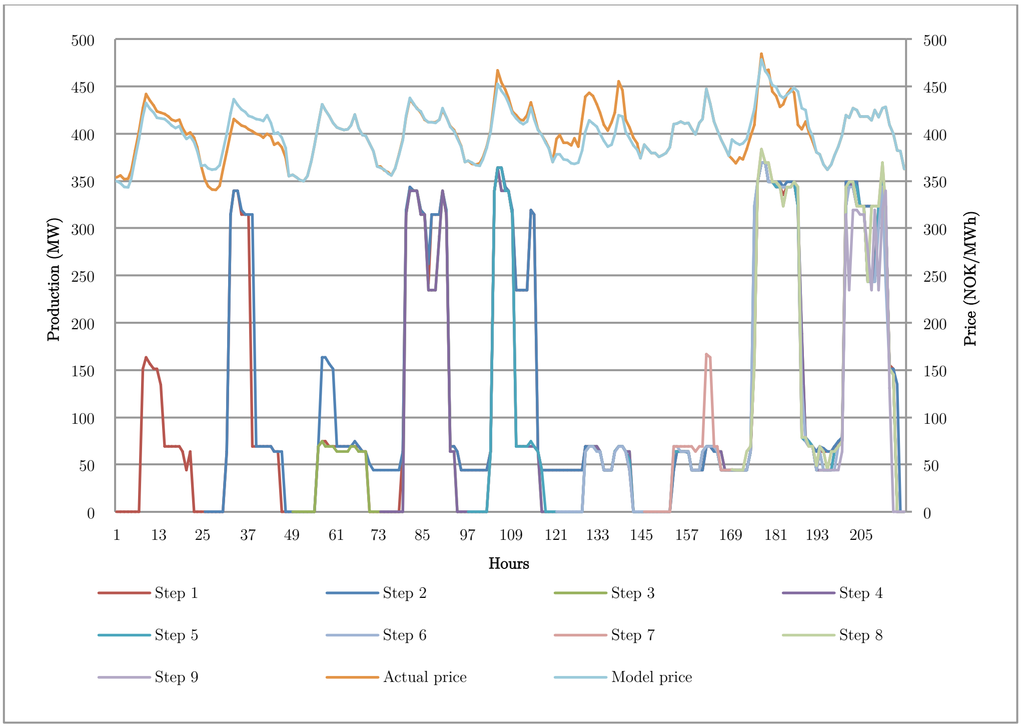


Figure 16: Optimal production pattern with no regulation obligation from each time step along with both the model price and the actual price

Figure 16 illustrates the optimal production strategy for a case with no regulation obligations when the producer only participates in the day-ahead spot market. It indicates that as new information about the future is revealed in each step, the optimal production pattern for the rest of the period changes.

For instance, the optimal production pattern when standing in step 3 is adjusted downward compared to the optimal production pattern in step 2. This is explained by looking at the price profiles. The actual price to occur in step 2 turned out much lower than the probability weighted model price used to optimize step 2, thus the model corrects itself during the optimization in step 3.

When looking back in time after the current optimization period the producers will see that that optimal decision in the start of the period often was either too optimistic or too pessimistic given what actually happened with the random variable. How far away from the strategy that would be optimal in a deterministic back seeing perspective the actual scheduled production patterns is, are often very dependent of how wide the scope of opportunities for the stochastic input variables is, in this case the spot price. Since the model utilizes the model price of all the full paths in the scenario tree that can occur in the given step, the optimal solution from a stochastic model often lies towards the center of the space of possible outcomes. Hence, the optimal solution from a deterministic back seeing perspective will be further away from the current optimal solution from the stochastic multi-stage model, if the realized value of the stochastic spot price is at the edge of the scope of opportunities.

The model optimizes the bidding into a day-ahead market several days into the future, which can seem a bit unnecessary since it is a day-ahead market. This is due to the fact that regulation markets often have longer time blocks and time horizons. If the producer participates in such a market and commits to deliverer a certain amount of available regulation capacity for the whole period, this will restrict the window of opportunities for the whole week also in the day-ahead spot market. Hence, the production pattern for the day-ahead spot market is to some degree already decided upon. The solution space in the day-ahead spot market optimization gets limited when the producer is committed to deliver regulation capacity.

5.3.3 Stepwise optimal reservoir behavior

It is also interesting to see how the reservoir levels given by the optimal production strategies changes as new information about the future spot price is revealed. This is illustrated in Figure 17 and Figure 18.

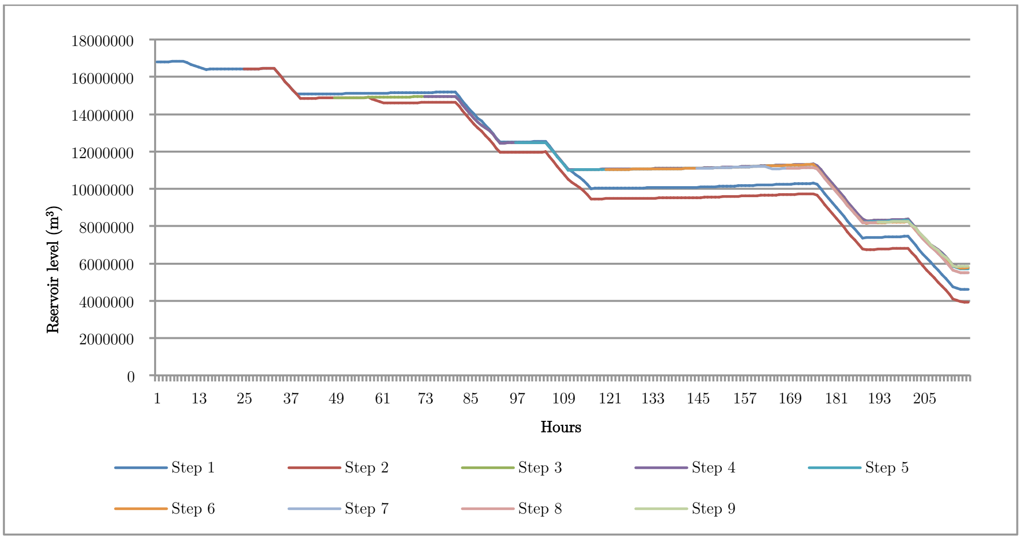


Figure 17: Reservoir levels at Reservoir 1 given optimal production with no regulation obligation for each time step

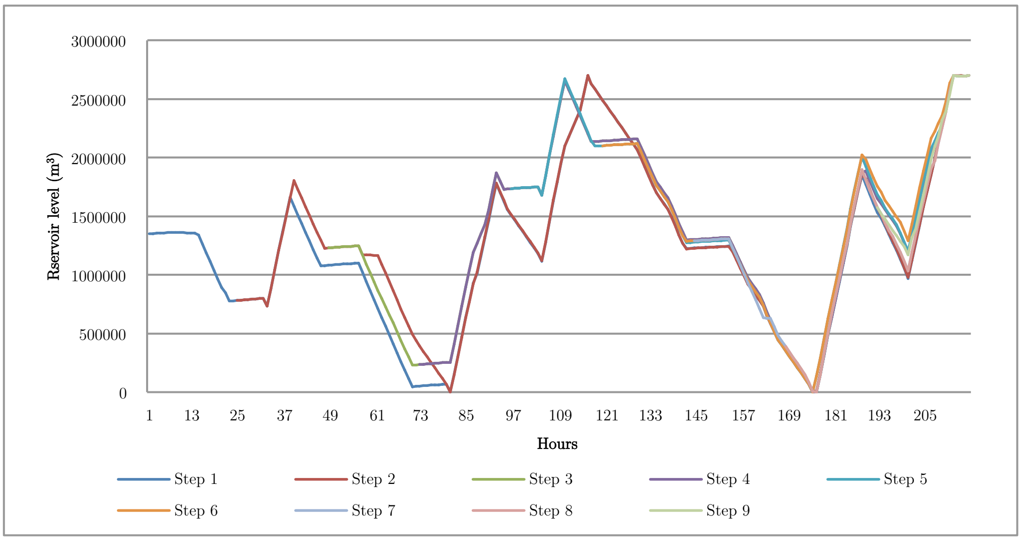


Figure 18: Reservoir levels at Reservoir 2 given optimal production with no regulation obligation for each time step

Since Reservoir 1 is much larger than Reservoir 2 relative to the maximum discharge of the stations the relative change in the optimal production strategy from step to step are much larger at Reservoir 2 than at Reservoir 1. As discussed in chapter 5.3.2, the model was more optimistic regarding the future price when optimizing step 2 compared to step 3. This can again be observed through the end reservoir level of Reservoir 1, where more water is used in the optimization in step 2 than when optimizing step 3.

5.3.4 Stepwise changes in spot market profit

The expected value of the objective function also changes as new information is revealed. Of course, the actual value of the objective function will always decrease during this process since the model always maximizes the income from the current step to the end of the period. Hence, when the optimization period gets shorter the value of the objective function will decrease.

Nevertheless, it is possible to see how the objective function would change as new information is revealed, if the income from the spot market and the startup costs from the past steps are taken into account. If the realized income from the spot market as well as the incurred startup costs for all past steps are added to the objective function it is possible to compare the values of the objective function from the different steps. This procedure should be done for all steps except the first. The value of the remaining water in end reservoirs from the past time steps should not be considered since this will be taken into account by the optimized objective function in the current step. This method makes it possible to see how the disclosure of new information influences the expected value of the objective function over the entire period.

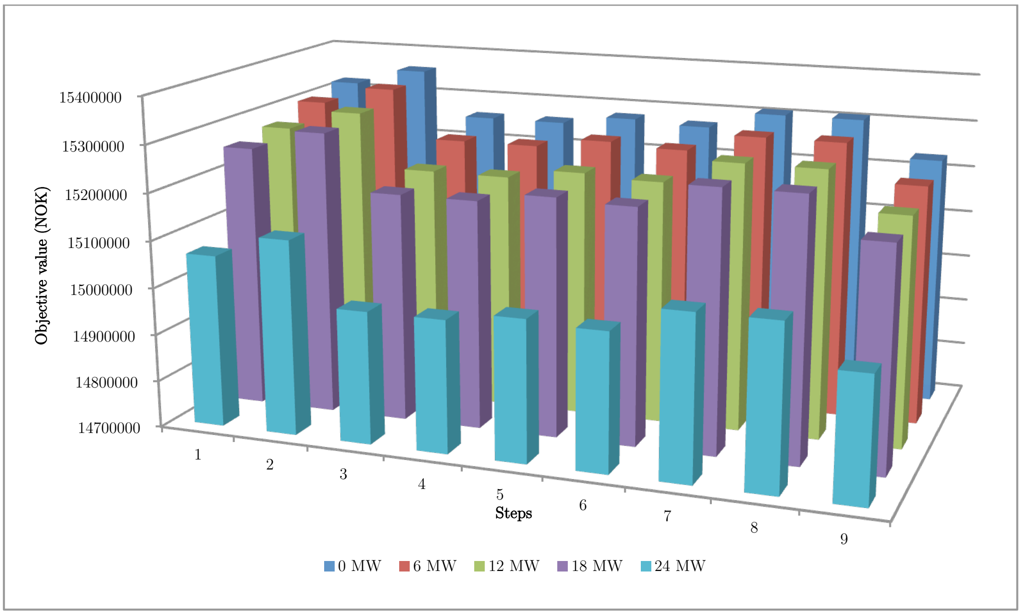


Figure 19: Stepwise objective values for different regulation obligations when the model price is used to calculate the value from the past

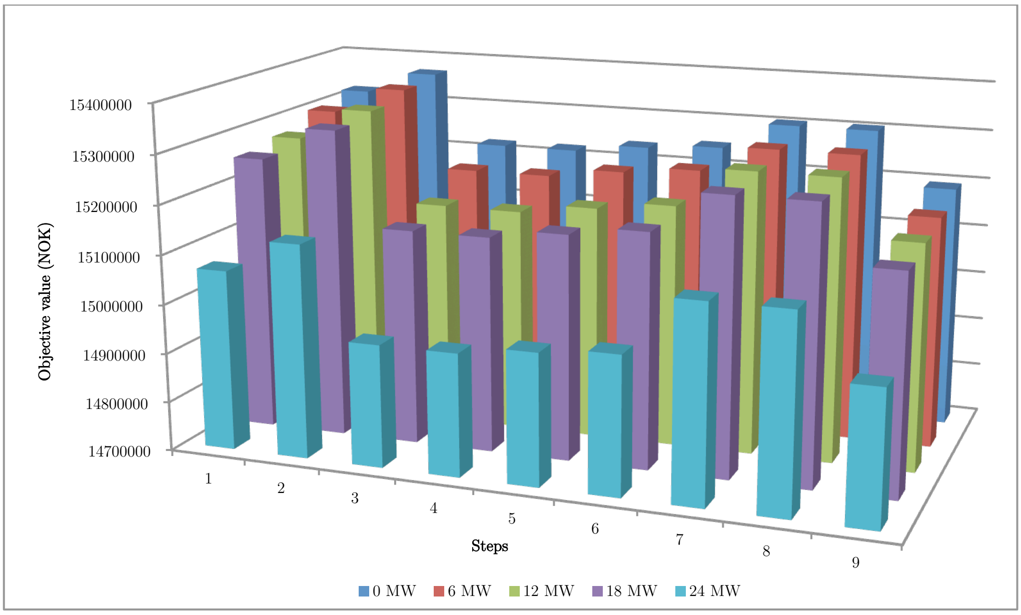


Figure 20: Stepwise objective values for different regulation obligations when the actual price is used to calculate the value from the past

Figure 20 illustrates the expected objective value for each step when the actual price, the price that actually occurred given the outcome from the probability function, is used to calculate the income from the past steps. In Figure 19 the expected objective value for each step when the model price, the probability weighted spot price that the models sees and optimizes against, is used to calculate the income from the past steps.

For all regulation obligations, the above figures show a decrease in the objective function after step 2, due to the new information available after this step. In step 7 and 8, a new optimism regarding the spot price is again observed. However, the last step indicates a more pessimistic outcome in the actual price.

5.3.5 Stepwise break-even price

From the objective values it is possible to investigate how the disclosure of new information influences the expected break-even price related to committing to a given regulation obligation. This is illustrated in Figure 21 and Figure 22.

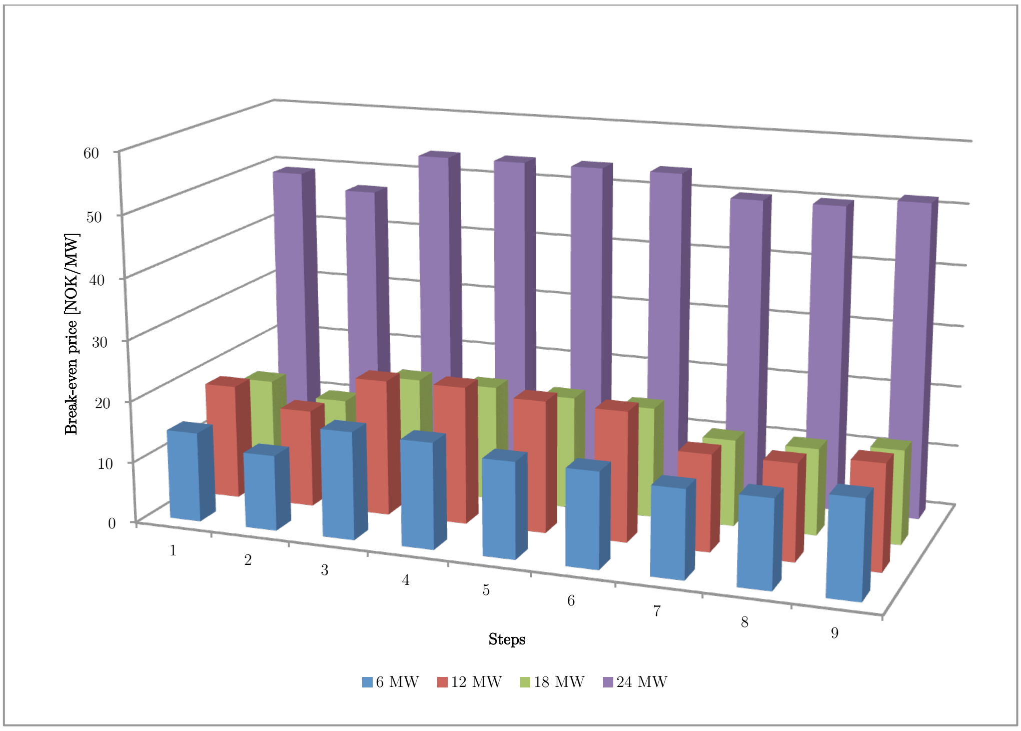


Figure 21: Stepwise break-even prices for different regulation obligations when the model price is used to calculate the income from past steps

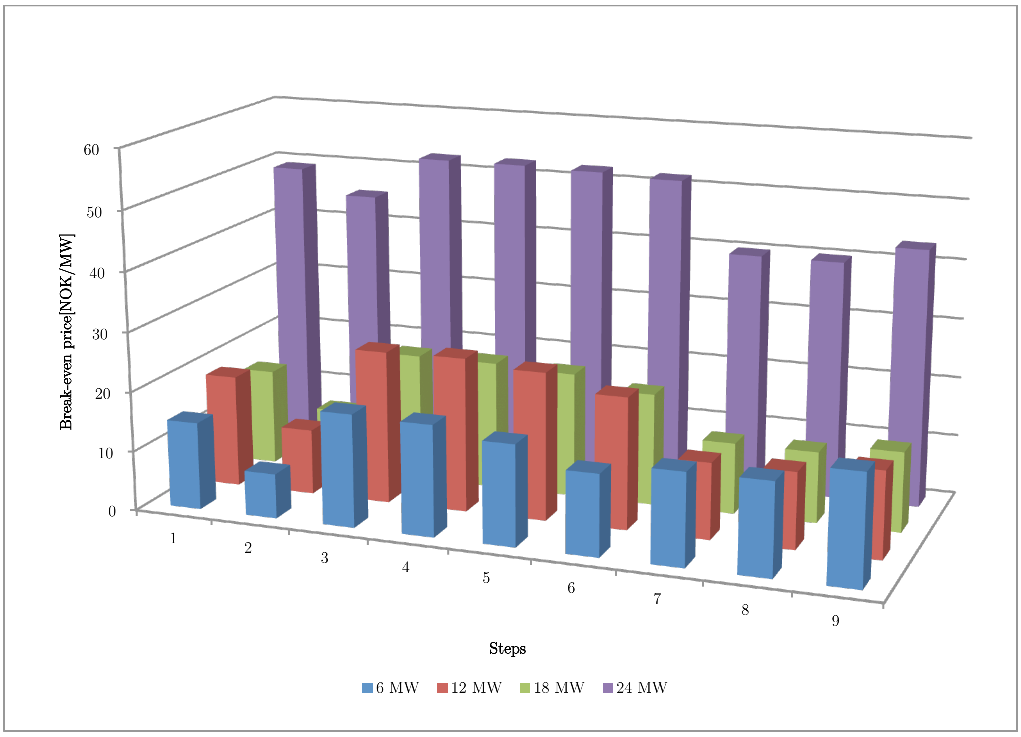


Figure 22: Stepwise break-even prices for different regulation obligations when the actual price is used to calculate the income from past steps

The trend regarding the development between step 2 and step 3 are also present in these figures, for the break-even price it is reflected in an increased value between the steps. This effect is generally more present when utilizing the actual price compared to when utilizing the model price, because the solutions calculated based on the model price will lie more towards the center of the possible solution space for the break-even price.

The break-even price calculated with the model price should be interpreted as the expected break-even price when standing within the time horizon of the given regulation obligation, while the break-even price calculated with the actual price should be interpreted as the break-even seen by the producer in a back seeing deterministic perspective standing at the end of the period.

It should be noticed that the expected break-even price related to a regulation obligation varies significantly when new information is revealed in the model. This applies both when the actual price and when the model price is used to calculate the income from the past steps, and indicates that also with a stochastic multi-stage model there is great uncertainty related to the break-even price of committing to a regulation obligation. The break-even price related to a certain level of regulation obligation is highly dependent on which of the possible scenarios outlined in the scenario tree that actually occurred from the outcome of the probability function within the stochastic model.

Regulation markets often have longer time blocks than the spot market, i.e. an entire week. Hence, a hydropower producer would use the model to optimize over the entire period and use the calculated break-even price related to committing upon a regulation obligation, when considering if the regulation obligation would be profitable for the next week. In this case it would not be possible to recalculate after the disclosure of new information has happened, the decision of committing to a regulation obligation or not would already have been made. Accordingly, stepwise results are only interesting for analytic purposes, not for an actual scheduling of the production.

5.3.6 Stepwise marginal cost

It is also possible to investigate how the marginal cost for the different levels of regulation obligation changes as the model moves to the next step and new information about the future spot price is revealed. Similarly as for the break-even price, this can either be done by using the model price or the actual price to calculate the income from the past steps when calculating the objective values.

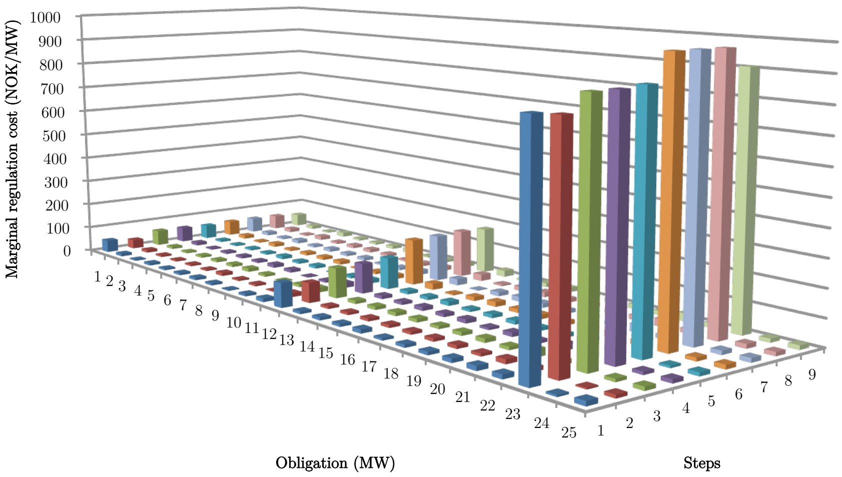


Figure 23: Marginal costs for all regulation obligations when the model price is used to calculate the income from past steps

As for the break-even price the marginal cost calculated with the model price should be interpreted as the expected marginal cost standing within the time horizon of the given regulation obligation while the marginal cost calculated with the actual price should be interpreted as the marginal cost seen by the producer in a back seeing deterministic perspective standing at the end of the period.

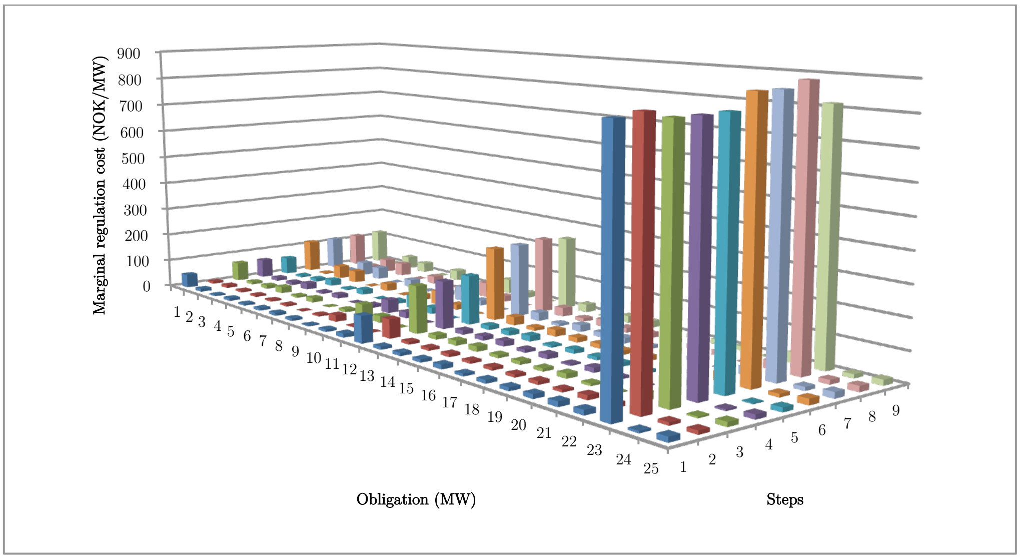


Figure 24: Marginal costs for all regulation obligations when the actual price is used to calculate the income from past steps

Figure 23 and Figure 24 illustrates the stepwise marginal costs when utilizing the model price and the actual price respectively.

They describe the same correlation regarding the development of the stepwise price, as mentioned in chapter 5.3.1. Due to the optimistic price forecast of step 6, the marginal cost correspondingly increases at this step.

It should be noticed that the plots are dominated by the very large marginal cost occurring when it is necessary to activate another unit for the entire period to conduct the given regulation obligation. Hence, it is difficult to see how the marginal cost changes for the other regulation obligation from step to steps since these are of much smaller values.

5.4 Monte Carlo analysis

Since the results vary to some degree between each full execution of the stochastic multi-stage model - due to the probability function within the model deciding the path through the scenario tree - Monte Carlo-simulations were carried out. The model was executed a large number of times and the mean values of the key economic figures as well as the distributions of the results were analyzed. Because of limited computation time it was also investigated if a chosen number of executions was sufficient enough to establish stable results, and was done by comparing the results from a Monte Carlo-simulation with 100 executions against a Monte Carlo-simulation with 1000 executions.

Two different Monte Carlo-simulations is presented and used in this chapter:

- A simulation with 100 model executions considering all regulation obligations between 1 MW and 25 MW. This simulation is used in 5.4.1, 5.4.2, 5.4.3 and 5.4.6
- A simulation with 1000 model executions considering regulation obligations of 6 MW, 12 MW, 18 MW and 24 MW. This simulation is used in 5.4.4 and 5.4.6.

Chapter 5.4.6 motivates that the empiric results obtained in both 100 and 1000 model executions does not differ significantly enough to influence the analysis.

When carrying out a Monte Carlo-simulation it is inefficient and unnecessary to analyze all results from each execution of the model. It is challenging and time-consuming to interpret all technical results describing the physics of the watercourse, e.g. production patterns, reservoir levels and run indicators. Hence, the focus should lie on the key figures. In this case this means objective values, break-even prices and marginal costs for the different levels of regulation obligations. Since the latter two figures are calculated from the objective values, and the objective value do not have any practical meaning in such a scenario based setting, the focus should lie on the break-even prices and marginal costs.

5.4.1 Spot prices in a Monte Carlo-simulation

When a Monte Carlo-simulation is carried out, the actual price occurring within the model will change between each execution. These price series are illustrated in Figure 25.

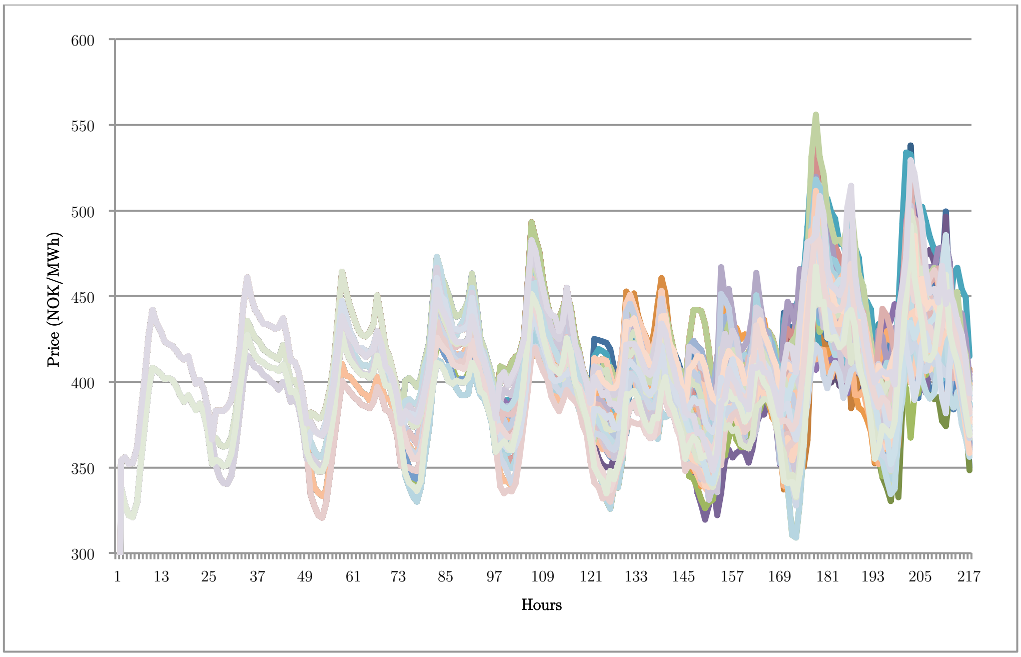


Figure 25: Actual price series from a Monte Carlo-simulation with 100 executions of the model

It illustrates that the number of possible outcomes for the spot price is increasing for each step, and is due to the structure of the scenario tree used in the model, described in Table 5. With a large number of executions, such a figure starts to look like the total scope of opportunities for the stochastic spot price, shown in Figure 5.

5.4.2 Break–even price for different regulation obligations

The development of the break-even price when the regulation obligation increases is illustrated in Figure 26 and Figure 27, and can be explained based on the physical properties of the watercourse.

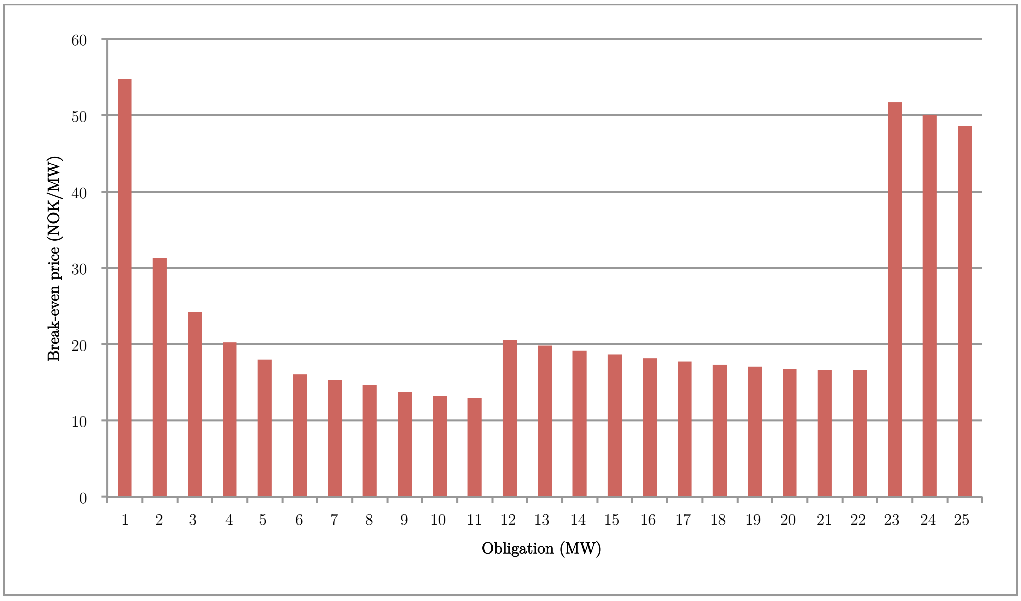


Figure 26: Mean break-even price for different levels of regulation obligations when the model price is used to calculate the objective values

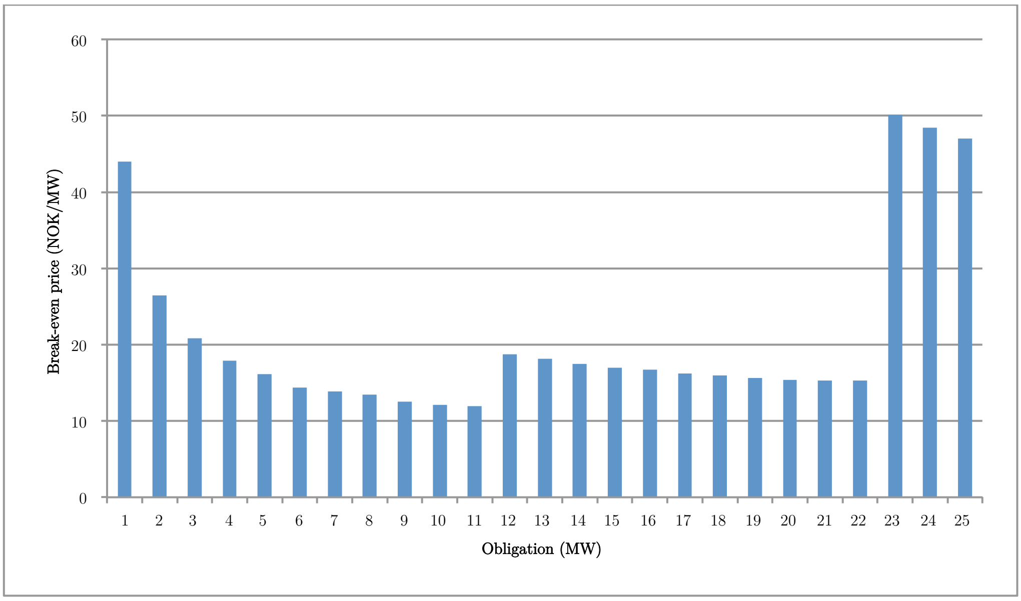


Figure 27: Mean break-even price for different levels of regulation obligations when the actual price is used to calculate the objective values

When it is necessary to let another unit run continuously throughout the entire period in order to meet the given regulation obligation, the break-even price will increase. In this watercourse and for the regulation obligations investigated, this occurs for regulation obligation levels of 1 MW, 12 MW and 23 MW. After such an increase the break-even price will decrease gradually until it is necessary to let yet another unit run for the entire period in order to meet the regulation obligation.

5.4.3 Marginal cost for different regulation obligations

While the break-even price reflects the cost of committing to a given regulation obligation the marginal cost reflects the cost of increasing the regulation obligation with one unit (1 MW). Just as with the break-even price the development of the marginal cost with increasing levels of regulation obligations can be explained by the physics of the watercourse.

As Figure 28 and Figure 29 illustrates, the marginal cost is significantly higher for certain levels of regulation obligations. These are the same levels as described in 5.4.2 where it is necessary to activate another unit for the entire period in order to meet the given regulation obligation. For the other levels of regulation obligations the marginal cost is relatively low.

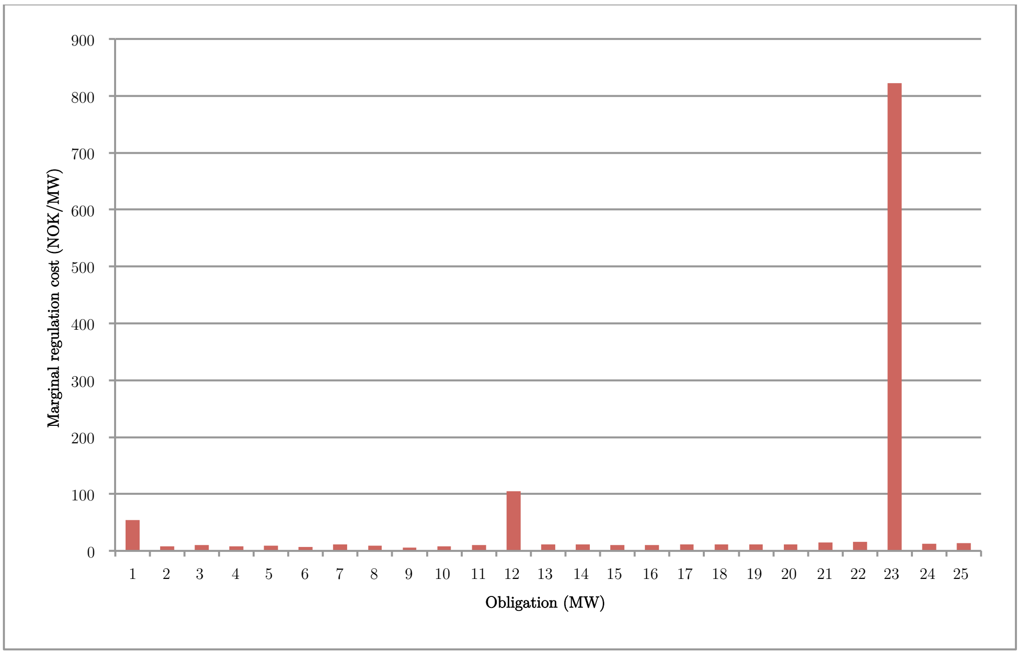


Figure 28: Mean marginal regulation cost for different levels of regulation obligations when the model price is used to calculate the objective values

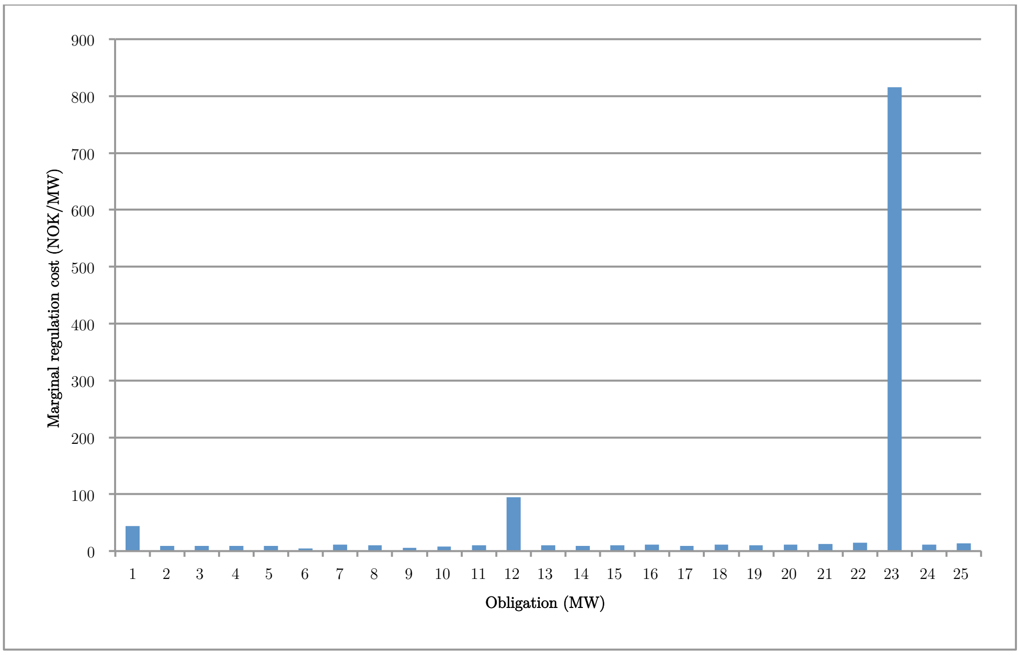


Figure 29: Mean marginal regulation cost for different levels of regulation obligations when the actual price is used to calculate the objective values

This indicates that it often will be profitable to commit upon a regulation obligation just below the level where it is necessary to activate another unit. Often, this will also be a local minima on the break-even price curve, see Figure 26 and Figure 27. A further discussion of how to use these results in order to optimize the bidding strategy in a regulation market can be found in chapter 6.5.

5.4.4 Distribution of the results

It was also investigated if it was possible to establish a general mathematical probability distribution of the results from a Monte Carlo-simulation of the stochastic multi-stage optimization model. This would provide a more accurate description of the expected results than what is obtained by simply calculating the mean results from the Monte Carlo-simulation.

For this analysis a Monte Carlo-simulation with 1000 executions of the model, but only with regulation obligations of 6 MW, 12 MW, 18 MW and 24 MW, was used. The reason for this simplification is computation time, a Monte Carlo simulation with 1000 simulations considering all relevant regulation obligations would need a disproportionately long computation time.

Histograms illustrating the distribution of the break-even price for the different levels of regulation obligations are presented in Figure 30 - Figure 33.

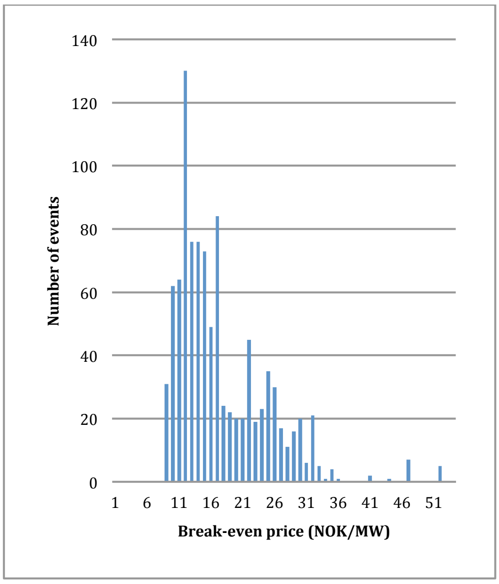


Figure 30: Real distribution of break-even prices with 6 MW regulation obligation

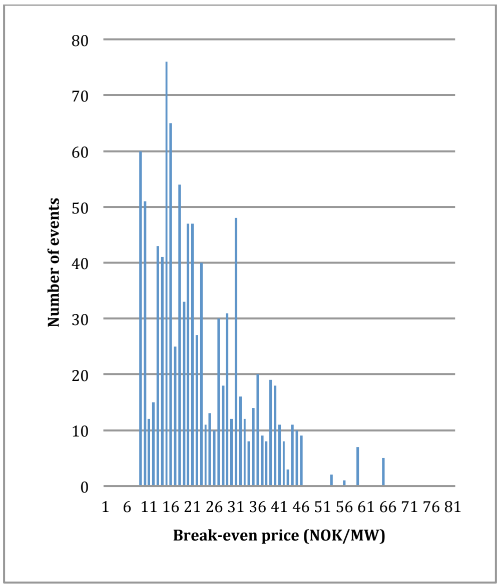


Figure 31: Real distribution of break-even prices with 12 MW regulation obligation

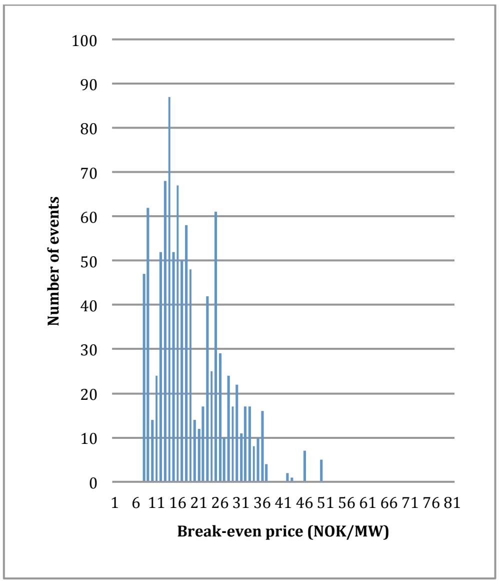


Figure 32: Real distribution of break-even prices with 18 MW regulation obligation

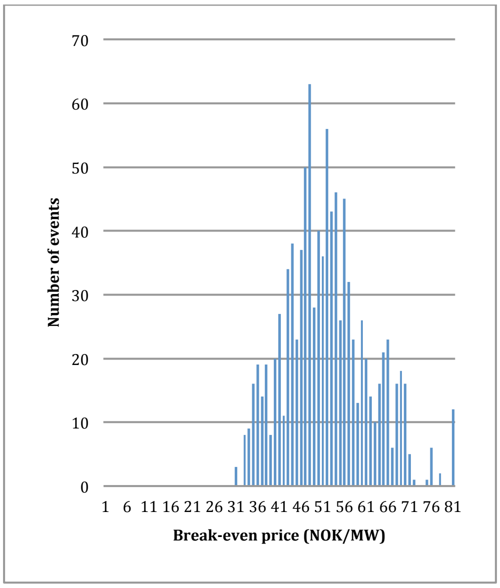


Figure 33: Real distribution of break-even price with 24 MW regulation obligation

The tool *Minitab 17 Statistical Software*[5] was used to investigate if any general mathematical probability distribution could describe these results in a satisfying way. Many common probability distributions were analyzed, see appendix F.

A *Goodness of Fit Test* was conducted for each distribution considering each level of regulation obligation. This is a form of statistical hypothesis testing, where a null-hypothesis is either accepted or rejected based on a chosen significance level [20]. In these tests, a significance level of 5% was used [21].

However, all null-hypothesizes were rejected meaning that none of the tested distributions could describe the distribution of the break-even prices for the different levels of regulation obligations. The results from this testing can be found in appendices F.1 - F.4.

In addition to the fact that it proves difficult to describe the distributions with a general mathematical formulation, Figure 30 - Figure 33 illustrates that the general shape of the distribution changes between different levels of regulation obligations. This indicates that the physical properties of the watercourse - most likely the regulation levels where it is necessary to activate another unit to meet the given regulation obligation, further discussed in chapter 6.4 - influences the shape of the distribution. Hence, a regulation obligation close to one of these levels will have a different distribution than a regulation obligation lying approximately midway in between two such levels.

5.4.5 Confidence intervals

The main motive for finding a probability distribution was to facilitate the creation of confidence intervals describing the expected break-even prices. However, since no general mathematical probability distribution was found, an alternative solution was to create empirical confidence intervals. If a confidence level of 95 % is desired, this is simply done by removing 2.5 % of the tail in both ends in the distributions presented in Figure 30 - Figure 33.

[5] http://www.minitab.com/en-us/products/minitab/

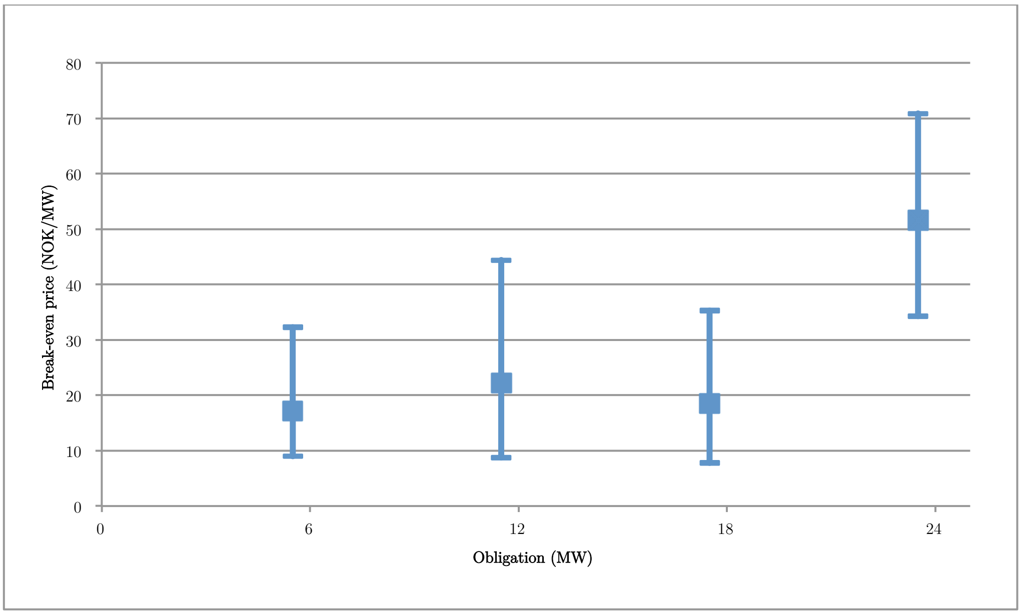


Figure 34: Empirical confidence intervals for the break-even price for different levels of regulation obligations

Figure 34 illustrates both the expected value and the range of the 95% empirical confidence interval for the break-even price after considering 1000 stochastic model executions. It is observed that the square representing the expected value lies closer to the lower tail of the confidence interval for all investigated regulation obligations. This indicates that it is more likely to achieve a larger break-even price than the expected value compared to a lower price. Thus, the distributions are asymmetrical.

5.4.6 Sufficient data sets

In order to have trustworthy data in a Monte Carlo-simulation, the number of individual simulations must be satisfactory. The actual number depends on the input data and the structure of the model itself.

A simulation containing 100 iterations was compared to a simulation with ten times the number of iterations, featuring 1000 individual model executions.

The average value of the break-even price was investigated for four different levels of regulation obligation, and the results reported almost the same values.

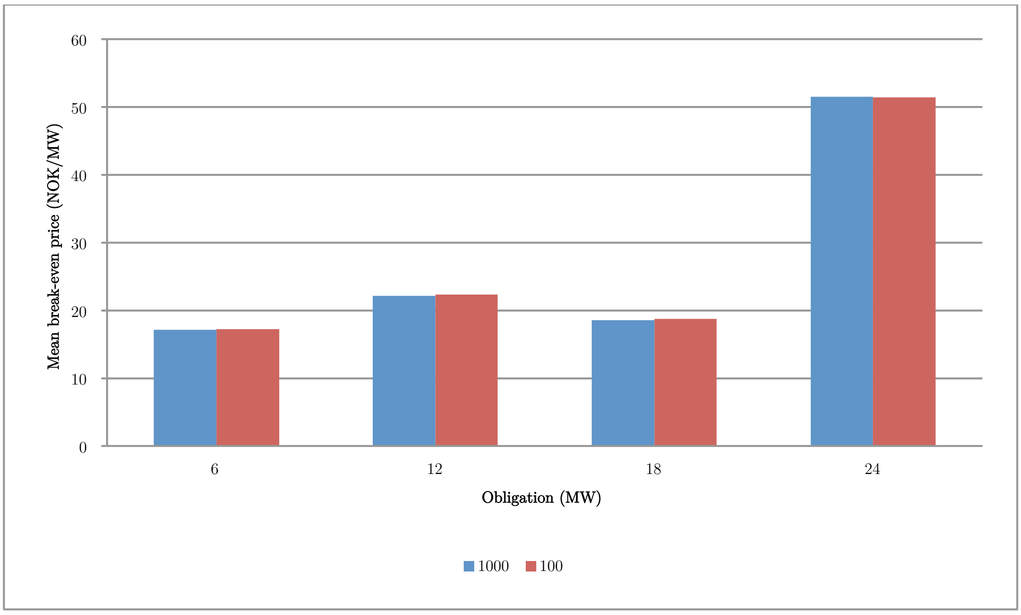


Figure 35: Mean of break-even prices considering Monte Carlo-simulations of 100 and 1000 optimizations

Table 9: Percentage change in mean values of the break-even prices considering 100 and 1000 simulations

Obligation [MW]	6	12	18	24
Percentage change in mean value [%]	0,46	0,69	0,98	0,99

The fact that the percentage change in the mean break-even prices never exceeds 1% is adequate information yielding that 100 optimizations are a sufficient number when utilizing a Monte Carlo-simulation with incremental regulation obligations.

6 Discussion

6.1 The significance of the regulation market's structure and its properties

The properties and guidelines of different regulation markets are varying and often changed. The markets can either be symmetric - which yields the same bidding volume for both upward and downward regulation - or asymmetric - the possibility of bidding different volumes of upward and downward regulation. Such markets may also have a block-based structure - dividing the regulation period into different blocks of hours that must have the same regulation capacity, but the blocks itself can consist of different bidding volumes [7]. When considering how to model optimal bidding into such markets, the best approach may be dependent on the properties of the regulation market itself.

In this study, a symmetric regulation market consisting of only one time block over the entire time horizon was investigated. This structure made it ideal to iterate over different regulation obligations, with the goal of calculating the break-even price in the regulation market as well as the marginal regulation cost for corresponding regulation obligations.

However, if the market was to have either an asymmetric property or consist of block-bids, using a multiple regulation obligation optimization strategy would cause challenges regarding both an increased amount of execution time as well as how to present and analyze the results in an efficient, comparable way.

If the model were to consider $\boldsymbol{O}$ obligation regulations, using a symmetric, block-free market as in this study, the different combinations of obligations needing comparison is equal to $\boldsymbol{O}$. When adding an option to bid separate volumes in upward and downward regulation given an asymmetric property, this would yield $\boldsymbol{O}^2$ combinations. With the inclusion of block bids, the possible obligation combinations increase into $\boldsymbol{O}^B$ possible outcomes, where B is the number of blocks in the biddable period. And if combining both an asymmetric property with block-bids, the possible combinations of regulation obligations would be equal to $\boldsymbol{O}^{2B}$.

Even with only one obligation, $O = 2$ (the model always need an optimal solution only taking a spot market into account for comparison), the increase in obligation combinations is doubled when considering asymmetric properties or introducing two blocks, and quadrupled when combining those properties. With a multiple regulation obligations, these numbers increase rapidly.

Table 10: The number of obligation combinations needed to be taken into consideration by the model with different regulation market properties

Number of obligations:	2	3	4	5
Regulation market properties:				
Symmetric market (O)	2	3	4	5
Asymmetric market (O^2)	4	9	16	25
Symmetric & 2 blocks (O^2)	4	9	16	25
Symmetric & 3 blocks (O^3)	8	27	64	125
Asymmetric & 2 blocks (O^4)	16	81	256	625
Asymmetric & 3 blocks (O^6)	64	729	4096	15625

When considering other market structures than symmetric, block-free bids, the desirable tool for describing a regulation market would be to optimize against a price profile, similar as to what is done against the day-ahead spot market. This approach was investigated in [2] for a deterministic model, both with asymmetric properties and with block-bids. The difference between this approach and the approach conducted in this study, is that instead of finding the optimal bidding regulation volume into the market given the price, the break-even price needed given the bidding regulation volume is found.

However, to do this it is necessary to have a good price forecast for the regulation market or balancing market. This proves difficult to obtain [17].

6.2 The sensitivity of reservoir volumes

[1] used the same watercourse as in this analysis for all analytic purposes. Then as now, a short time optimization model taking both a spot market and a regulation market into account was utilized. However, the difference between the previous work and this report is that the regulation market volume was set as a variable and not as an obligation. One of the main conclusions from [1] was the significance of the physical properties regarding the watercourse in the results. Physical restrictions proved to be more important input factors than the price forecast for the regulation market under investigation.

Therefore, a sensitivity analysis was carried out. The physical property that proved to be most influencing in [1] was the bottleneck at Station 2 along with the quite small Reservoir 2. Station 2 has low total maximum discharge compared with the upper Station 1 and Reservoir 2 is small compared to the maximum discharge of Station 1 above. This influences the results heavily since participating in a regulation market with Station 1 very often will lead to flooding at Reservoir 2.

To avoid the influence of these physical restrictions in this stochastic analysis, the total reservoir volume of both reservoirs were increased by a factor of ten. A Monte Carlo-simulation was then carried out to see how the results - mainly the break-even price of regulation markets related to different regulation obligations and the marginal cost of attending such markets - varied between this case and the case with the actual reservoir volumes.

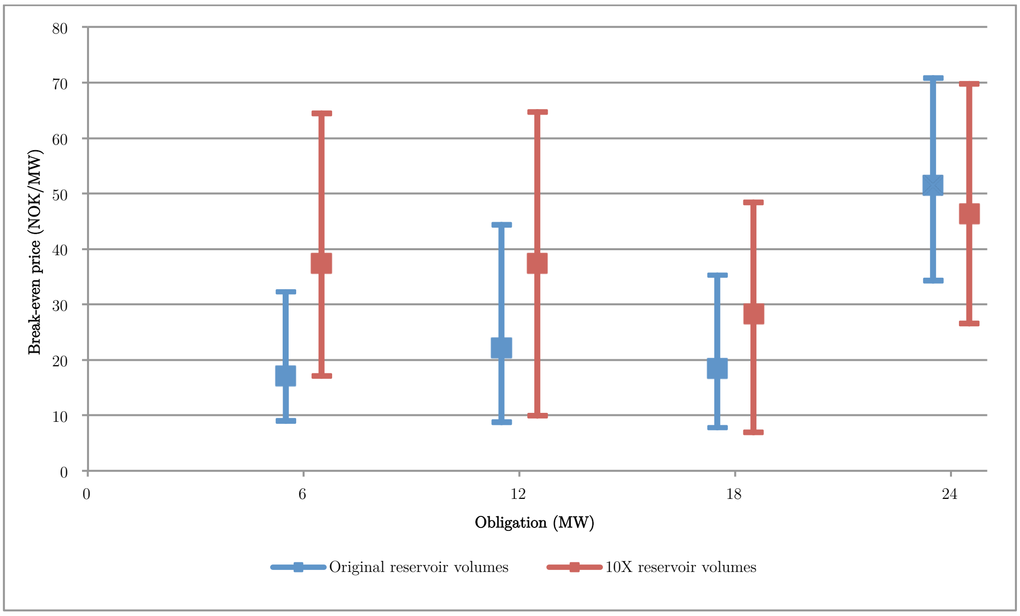


Figure 36: Empirical 95% confidence intervals for the break-even prices of regulation obligations considering original reservoir volumes and ten times the original reservoir volumes

Figure 36 illustrates empirical 95% confidence intervals of the needed break-even prices in regulation markets after a Monte Carlo-simulation of 1000 model executions both considering the original reservoir volumes and a relaxed version with ten times the original volumes. The square represents the expected value of the break-even price, and the line illustrates the empirical confidence interval itself.

It easily observed that the confidence intervals concerning relaxed reservoirs are less asymmetric than when original reservoir levels are considered. Relaxing the reservoirs eliminates the bottlenecks previously described, which strongly influenced the distributions of the results observed in chapter 5.4.4.

There is also another interesting trend regarding the comparison between the confidence intervals. The relaxed Monte Carlo-simulation has a higher expected break-even price and a larger total confidence interval than the Monte Carlo-simulation with original reservoir volumes, considering the first three regulation obligations. But when reaching the highest amount of regulation obligations, 24 MW, the expected value of the break-even price for the expanded reservoirs has a lower expected value than the original reservoir Monte Carlo-simulation.

This can be explained by the same bottleneck as discussed in chapter 3.3. When the smallest Reservoir 2 has its original size, Station 2 must produce whenever Station 1 produces to avoid flooding. Thus, the solution only taking a spot market into account includes production and therefore the possibility of regulation capacity at Station 2 at almost all hours even before considering regulation obligations.

This effect is illustrated in Figure 37, which compares the spot production for two stochastic scenarios with original reservoir levels and ten times the size.

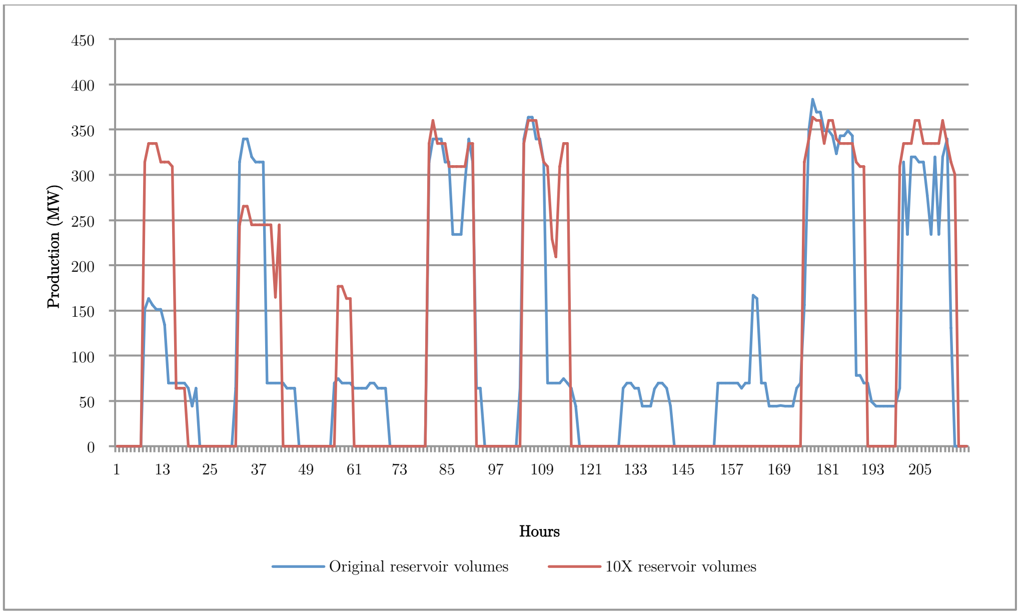


Figure 37: Optimal production patterns when only considering a spot market with original reservoir levels and ten times the original reservoir levels

As a consequence of this, the difference in the objective value between the optimization only taking a spot market into account and the optimizations considering the smallest regulation obligations are larger when relaxing the reservoirs – illustrated through the break-even prices in Figure 36 - but are less indifferent considering original reservoir levels. However, when the regulation obligation is sufficient enough, causing a unit on Station 1 to deliver regulation capacity, this difference is smaller when analyzing results from the relaxed reservoirs.

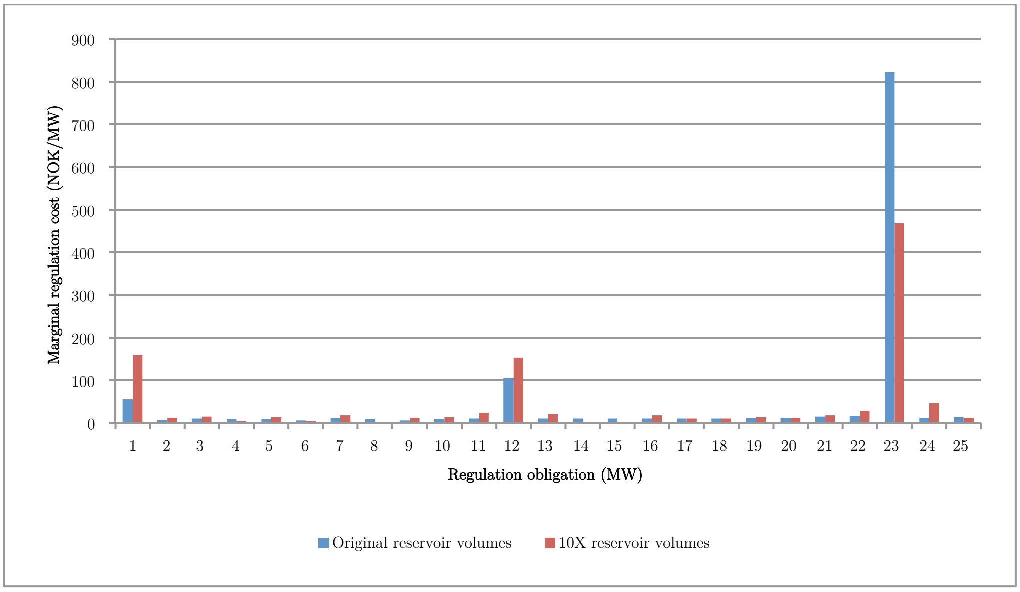


Figure 38: Marginal regulation costs considering original reservoir levels and ten times the original reservoir levels

When comparing marginal costs, the observed trend shows a slightly higher cost of reserving regulation capacity when considering a relaxed reservoir for almost all obligations. This is rather easily explained by the fact that when considering the original reservoir levels, units on Station 2 must contribute regulation capacity regardless of the incremental increase in obligations to avoid flooding, thus the cost of contributing with another unit of regulation is lower compared to reservoir levels where flooding is not a concern.

Nevertheless, for the 23 MW level of obligation, the marginal cost is significantly higher for the original reservoir level. This is the obligation level where Station 1 must start contributing regulation capacity at all times.

The reason behind this sudden change of which marginal cost is the highest, are yet again subject to physical limitations of the original reservoir level. When considering ten times larger reservoirs and a regulation obligation where Station 1 must deliver regulation at all times, the optimization model tries to minimize the loss in the spot market by producing close to maximum capacity in certain peaks of the spot price profile. This however, is not possible without causing a flood considering the original reservoir levels, thus the marginal cost for these reservoir levels are higher than when relaxing the reservoirs. This effect also explains the decline in the break-even price for a ten times larger reservoir with a regulation obligation of 24 MW, compared to the original reservoir levels where the previous break-even prices were lower.

6.3 Negative marginal cost in certain stochastic outcomes

The results from the stochastic multi-stage model changes to some degree between each execution due to the probability function within the model that decides the path through the scenario tree. In a small proportion of the model executions the results indicate a negative marginal cost for a few of the regulation obligation levels investigated.

This should generally be impossible from a mathematical optimization point of view. To get a negative marginal cost the value of the objective function must increase when the regulation obligation increases. Increasing the regulation obligation means restricting the solution space for the optimization model, which never should increase the value of the objective function [19]. However, this phenomenon can be explained when a stochastic multi-stage optimization process is conducted. This observation only occurs in certain sequential outcomes of the probability function within the model.

When the model stands in one step the production is optimized with the same stochastic outcome of the spot price for all regulation obligations. Then, the value of the objective function will always decrease with increasing regulation obligation. However, if the probability function creates an outcome of the spot price in the next step that is higher than the model price in the given step, this sometimes creates a situation where higher regulation obligations give a higher realized objective value investigating the entire period. Then it would have been optimal to produce more in the spot market in the previous step - given the realized path chosen by the probability function - and for higher regulation obligations the model is forced to do so.

Hence, the realized objective value calculated over the entire period can in a few cases increase in line the with regulation obligations. This happens if the actual price is distinctively higher than the model price in the past steps during a sufficiently large share of the period.

As mentioned earlier the marginal cost can be calculated in two ways; by using an objective value based on either the actual price or the model price. Since the model price always will lie towards the center of the opportunity scope of the uncertain spot price - it is a probability weighted average of the possible outcomes - this may produce a negative marginal cost in fewer cases than when the actual price is used to calculate the marginal cost.

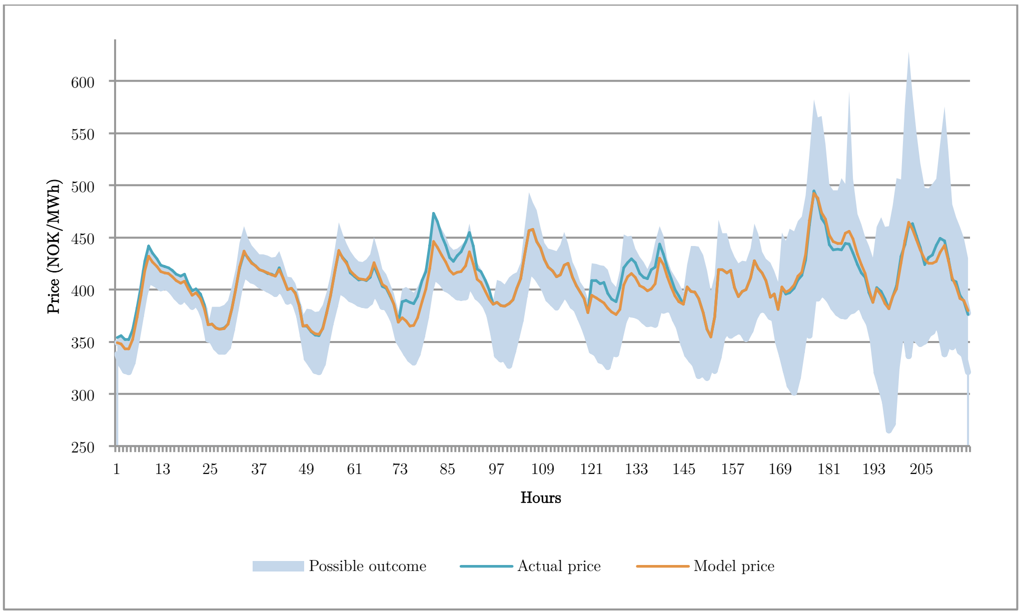


Figure 39: Price outcome in a simulation that produced some negative marginal costs

Figure 39 illustrates a case where the actual price became higher than the model price in a sufficient large proportion of the steps for this phenomenon to happen. In this simulation this occurred in step 4 and step 6. For this stochastic outcome considering 25 different levels of regulation obligations, a negative marginal cost occurred for seven regulation levels when the actual price was used in the calculation. Correspondingly, only one regulation obligation produced a negative marginal cost when the model price was used in the calculation. See appendix G for a list of these marginal costs. It should be pointed out that this was an extreme case, in most executions of the model the marginal cost were positive for all regulation obligations.

The stepwise results would never produce negative marginal cost, in this case the normal mathematical logic applies meaning that the objective value always will decrease with increasing regulation obligations giving positive marginal costs. It is only the total objective value calculated after the last step based on the optimal production specified in each step that can yield negative marginal costs.

This phenomenon can of course be seen upon as a weakness of the stochastic multi-stage model, since it should be impossible to increase the value of the objective function when the solution space is decreased. However, as explained the reason is related to the uncertainty regarding the future spot price. Since this also is the case in practice, it illustrates the difficult optimization task the hydropower producers phase on a daily basis considering uncertain stochastic input variables and how they affect the production and the related costs.

6.4 The increased break-even price and marginal cost when starting new units

Figure 26 - Figure 29 in chapter 5.4.2 and 5.4.3 illustrates the expected break-even price and marginal cost for all levels of regulation obligations ranging from 1 MW to 25 MW calculated through a Monte Carlo-simulation.

Both these graphs have some characteristic breakpoints where the cost increases dramatically at the same levels of regulation obligations. For this watercourse and for the regulation obligations investigated in this study, these breakpoints are occurring at 1 MW, 12 MW and 23 MW.

The marginal costs in Figure 28 and Figure 29 are generally low, except at these specific levels of regulation obligations. When it comes to the break-even prices in Figure 26 and Figure 27, they have a distinctive increase at these levels of regulation obligations before they decrease slowly until the next mentioned regulation level where it increases distinctively yet again.

The explanation behind this behavior can be found by investigating how the model chooses to deliver the different levels of regulation obligations, together with the physical properties of the watercourse. The watercourse consists of three units at Station 1, which each can deliver a maximum symmetrical regulation capacity of 27.5 MW, and two units at Station 2, which each can deliver a maximum symmetrical regulation capacity of 11 MW. Due to reservoir properties - Reservoir 2 is small compared to the discharge from Station 1 - it is always most profitable to deliver regulation capacity from Station 2 before a unit at Station 1 is continuously activated. A continuous running of one of the units at Station 1 very often leads to a risk of flooding Reservoir 2 and restricts the flexibility of the units at Station 2 as well. A continuous discharge from Station 1 would make it necessary to utilize both units at Station 2 at high production levels the entire time.

Hence, for regulation obligations up to 11 MW the optimal solution is to let one unit at Station 2 run continuously, and for regulation obligations between 11 MW and 22 MW the optimal solution is to let both units at Station 2 run continuously. After this the optimal solution is less unambiguous, the optimal way to meet the given regulation changes depending on reservoir levels and physical factors. However, the break-even price is always higher for regulation obligation levels above 22 MW, since it will be necessary to use one of the units at Station 1 to some degree in order to meet the regulation obligation anyway.

Thus, the mentioned regulation levels in Figure 26 - Figure 29 matches the levels where it is necessary to continuously active another unit in order to meet the given regulation obligation. When the regulation obligation increases to a level where this is necessary, this yields a drastic increase in the costs related to this regulation obligation due to the reduced flexibility it gives in the day-ahead spot market.

This discontinuous behavior of the cost should be considered when choosing the bidding volume into a regulation market. It will often be most profitable to choose a regulation obligation right below one of these breakpoints at the break-even price and marginal cost curves. A more thorough discussion of how to use the break-even prices and the marginal costs to optimize the regulation bidding volume can be found in chapter 6.5.

6.5 Combining the break-even price and marginal regulation cost as a decision making tool

If the forecasted price of a regulation market is unknown, a model with regulation obligations as parameters and the break-even price and marginal cost of providing such regulation as results can be a suitable solution. However, if the forecasted price of a regulation market is known, this price can be used as an input parameter and an optimal level of regulation obligation can be found, as shown in [2].

The former approach is suboptimal opposed to the latter, but forecasts of regulation market and balancing market prices are hard to achieve [17], thus many hydropower producers are optimizing production with a predetermined regulation obligation. Nevertheless, an optimal level of regulation obligation can be found without an advanced regulation price forecast input, given a few conditions. By combining the break-even price of incremental regulations obligations along with the corresponding marginal costs of providing such a regulation, it is possible to rule out regulation obligations that are less valuable than others.

The conditions are that a lower and an upper limit of regulation obligations must be decided upon. This is rarely a problem, as either the market or the physical properties of the watercourse restricts the solution space of regulation capacity to some degree. Also, an expected value of the regulation market price must be agreed upon. This value is not used as an input parameter in the optimization, but is needed in order to indicate whether or not the break-even price and the marginal cost is lower than the expected market price.

In the following example graphs that accompany the explanation of this method, the break-even price and marginal regulation cost are dependent on the model price. This is because of it being the price the producer should expect when optimizing. A lower and upper limit of 0 MW and 25 MW correspondingly is used, with an expected regulation price of 20 NOK/MW.

6.5.1 Analyzing the break-even price graph

The first graph that needs consideration is the break-even price graph.

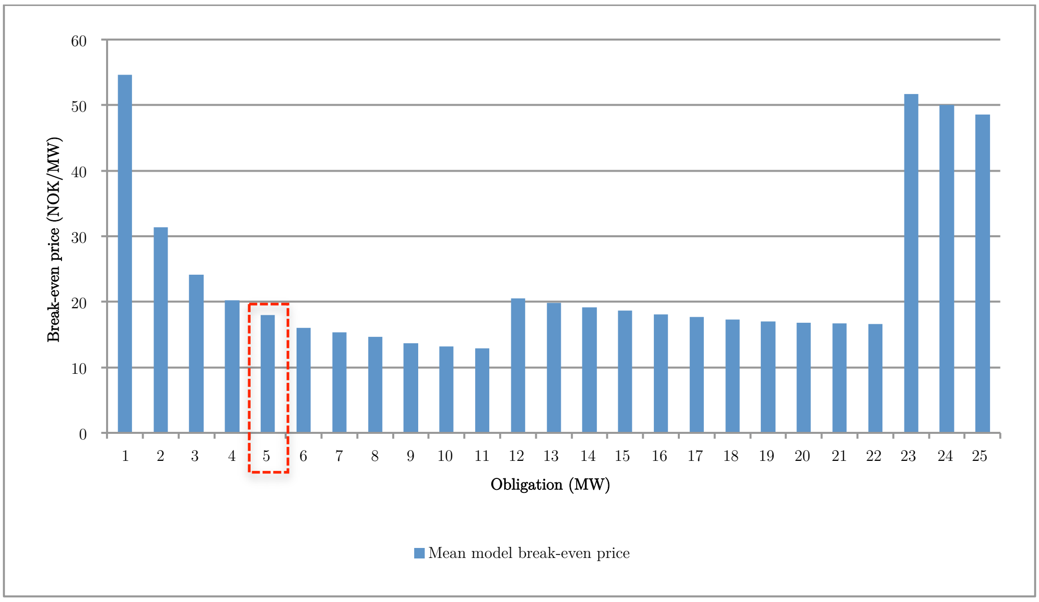


Figure 40: Example of the first regulation obligation with a break-even price below an expected regulation price of 20 NOK/MW

The first regulation obligation within the limits decided upon that has a break-even price below the expected regulation price is considered the current optimal regulation obligation. Then, the graph illustrating the marginal costs must be investigated.

6.5.2 Analyzing the marginal regulation cost graph

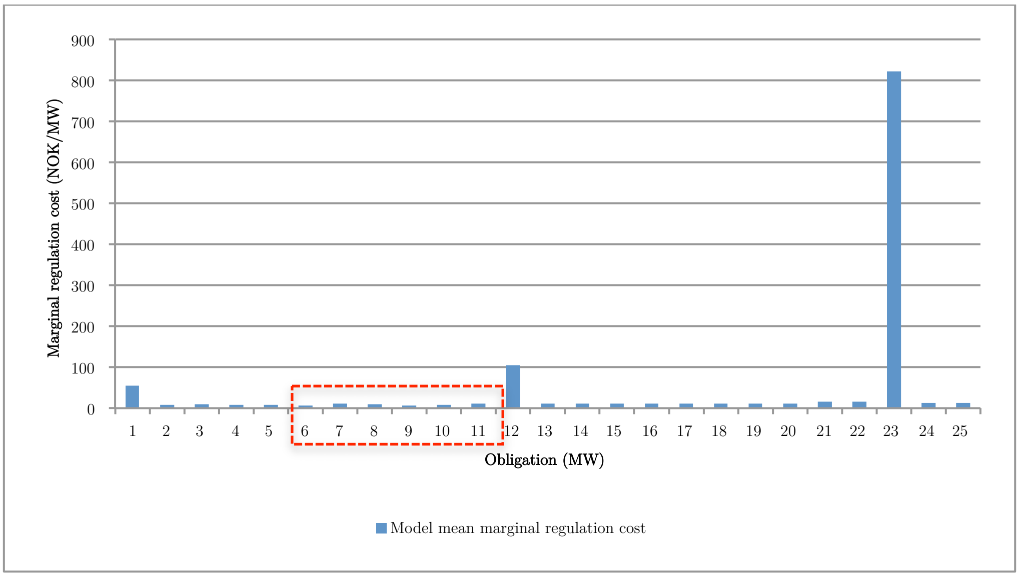


Figure 41: Example of all successive regulation obligations with a marginal cost below 20 NOK/MW

If, and for as long as the next regulation obligation's marginal cost is lower than the expected regulation price, this regulation obligation is the current optimal level. When the marginal cost exceeds the expected regulation price, the break-even price graph must again be examined. The last regulation obligation with a marginal cost below the expected price is now the current optimal regulation obligation.

If there are no future regulation obligations with a break-even price (including the current regulation obligation examined in the marginal cost graph) below the expected regulation price, the current optimal level is the final optimal level. However, if there exists regulation obligations with break-even prices below the expected regulation price, a more extensive test should be carried out.

6.5.3 Considering the surplus formula

The first successive regulation obligation with a break-even price below the expected regulation price is then examined against the following surplus formula;

$$P_{sym}\left(Y_{sym}-\dot{Y}_{sym}\right) > \sum_{i=\dot{Y}_{sym}+1}^{Y_{sym}} C_{MC}(i) \tag{36}$$

where P_{sym} is the expected regulation price, Y_{sym} is the regulation obligation currently examined, $\dot{Y}_{sym}$ is the current optimal regulation obligation and C_{MC} is the corresponding marginal cost of the regulation obligation. The formula states that the income from the regulation market when increasing regulation obligations must be higher than the incurred cost of losing capacity in the spot market due to this increase.

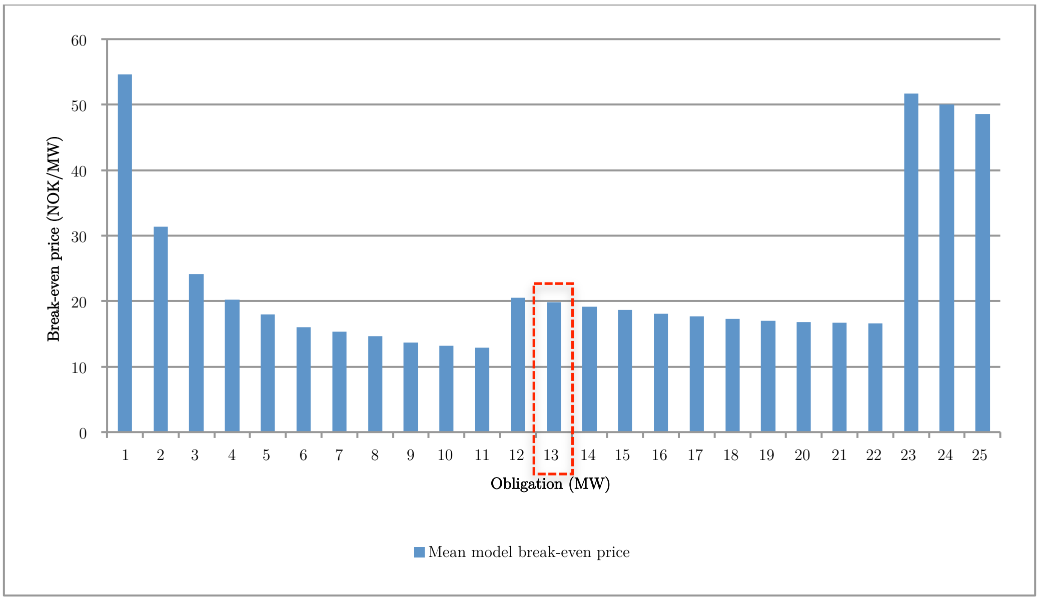


Figure 42: Example of the next successive regulation obligation with a break-even price below an expected regulation price of 20 NOK/MW. However it does not satisfy the surplus equation

As long as this equation proves false, the current obligation must be increased. If there exists a regulation obligation level where this equation proves true, this is the new current optimal regulation obligation.

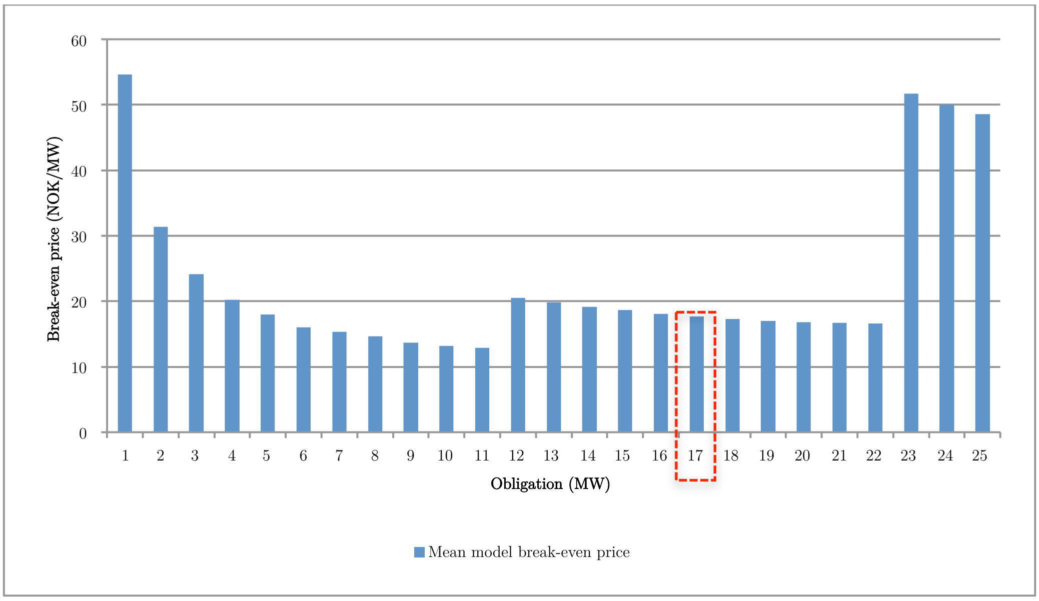


Figure 43: Example of the first successive regulation obligation with a break-even price below 20 NOK/MW that satisfies the surplus equation

If this is the case, it is then sufficient to yet again compare the next regulation obligation's marginal cost with the expected regulation price, and increase the current optimal regulation obligation as long as the marginal cost is below the expected regulation price.

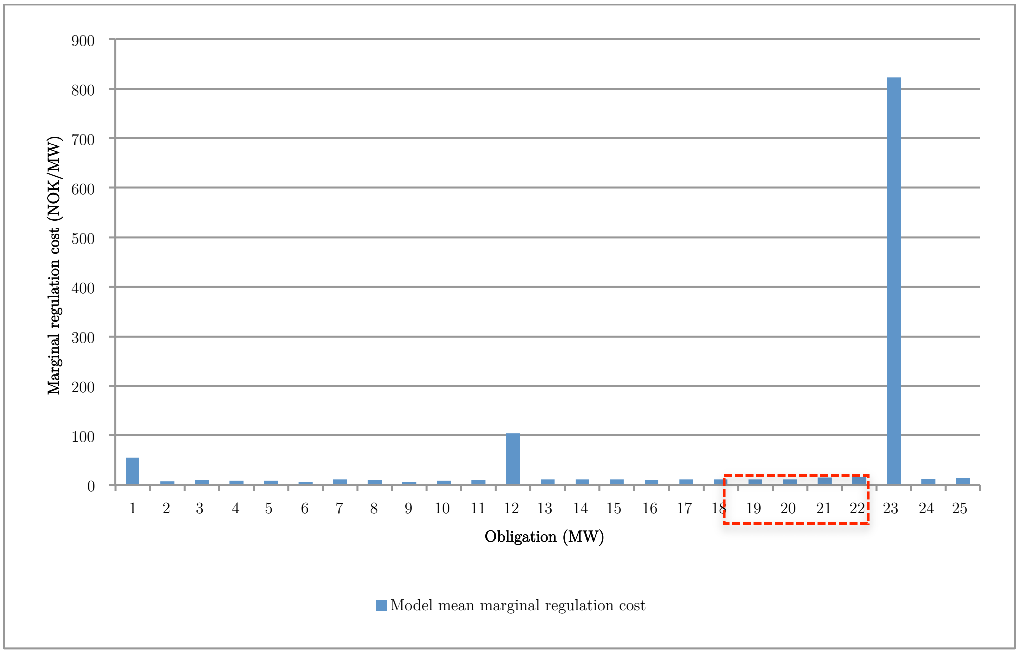


Figure 44: Example of all successive regulation obligations with a marginal cost below 20 NOK/MW

This routine is of course only valid within the lower and upper limits decided upon beforehand.

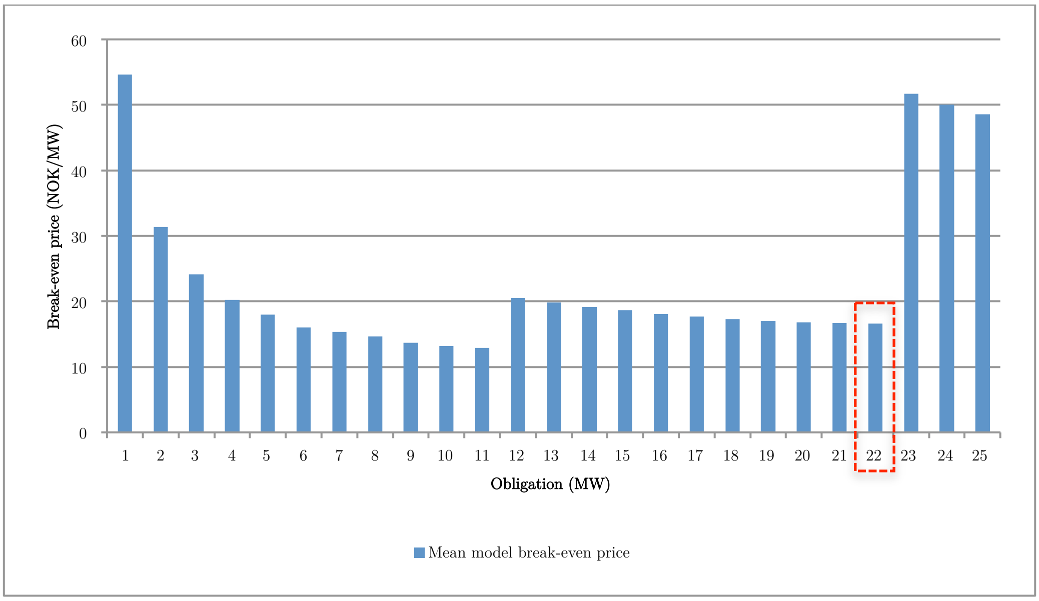


Figure 45: Example of the optimal level of regulation obligation considering an expected regulation price of 20 NOK/MW

This method purposes a way to rule out inefficient levels of regulation obligations, if the possibility of having flexible decisions regarding the regulation obligation exists. The optimal solution will often lie at the regulation obligation present before a huge rise in the marginal cost, often caused by a new unit delivering regulation, as discussed in chapter 6.4.

However, the strength of this method is that it compares which units and stations are most profitable of delivering regulation, thus finding the optimal regulation obligation by using the surplus formula mentioned to add up all accruing costs and expected revenues within the predetermined limits.

7 Conclusion

A stochastic multi-stage optimization model proves to be useful for foreseeing the optimal behavior of a watercourse given different regulation obligations, and for estimating the cost a hydropower producer faces when committing to a given regulation obligation.
Since estimating the price in regulation markets and balancing markets remains difficult [17], an alternative solution is to estimate the cost related to different bidding volumes into a regulation market - i.e. a range of regulation obligations - instead of optimizing the bidding volume based on a price forecast for the given regulation market.

The market design is affecting the choice and design of the optimization strategy

Generally it would be desirable with a model using a forecasted price in both the day-ahead spot market and the regulation market as input parameters and optimize the biddable volume into both markets based on this.
Nevertheless, since price forecasts for regulation markets as mentioned are difficult to obtain, an alternative strategy can be to calculate the cost related to committing to different regulation obligations. However, if the regulation market consists of several time blocks, is unsymmetrical or has a combination of these properties, this strategy proves difficult to conduct as discussed in chapter 6.1. In such a case a model taking the forecasted price in the regulation market as input would be a preferable.

Considering the physical behavior of the watercourse under different regulation obligations proves useful

A model optimizing the production in a day-ahead spot market assuming that an obligation in a regulation market could be committed upon, gives hydropower producers the ability to see how different physical properties such as hourly unit production, reservoir levels, discharge, bypass, flood etc. are influenced by the different regulation obligations. This is valuable information when considering bidding into a regulation market and committing to a regulation obligation for a longer time horizon than the day-ahead, as it gives a compact overview of how the watercourse behaves for a range of different regulation obligations.

Stepwise results from stochastic outcomes indicates the value of good information

As new information about the future is revealed when the model iterates to the next step - corresponding to the next day in this short-term model - the optimal solution often changes to some degree. This reflects the issue hydropower producers' face in practice when bidding into a regulation market with a longer time horizon than the day-ahead spot market. The decision of going into such a regulation market must be taken with less information about the future than what is available the day-ahead, and this influences the production schedule for the upcoming period. This illustrates how valuable accurate forecasts of the market price are.

Monte Carlo-simulations yield more trustworthy results

Some stochastic models, including the model developed in this thesis, produces different results between each execution of the model. Hence, the results can be regarded as stochastic variables with corresponding probability distributions. In such cases a Monte Carlo-simulation strategy produces more trustworthy results. However, calculation times may prove to be an issue.

A bidding strategy can be developed based on calculated costs and prices related to a range of regulation obligations

Calculating the cost related to committing upon regulation obligations facilitates the calculation of break-even prices and the marginal costs related to different regulation obligations. The break-even price reflects the needed price in the given regulation market to make the given regulation obligation profitable, while the marginal cost reflects the cost related to increasing the regulation obligation with one unit (1 MW).

None of these figures alone provide precise information when it comes to finding optimal bid into a regulation market. They only indicate if a regulation obligation is profitable or not, and if it is desirable to increase the regulation obligation. Thus, it is necessary to obtain an expected value for the given regulation market to compare these values against. However, the most profitable bidding volume will still be unknown. Nevertheless, if these figures are combined in an optimization strategy as discussed in chapter 6.5 it is possible to find the most profitable regulation obligation, i.e. the volume the hydropower producer should bid into the regulation market under investigation.

The design and physical properties of the watercourse are the most influencing factor of the results

As indicated in [1] and [2], the physical properties of the watercourse often influences the results more than the market prices. This is also present in this stochastic study, where the same watercourse is considered.

The break-even prices and marginal costs increase significantly when it is necessary to continuously activate a new unit in order to meet the given regulation obligations, especially when Station 1 is initiated. The discussed physical bottleneck of the system proves to be decisive for the optimal results obtained.

8 Further work

One of the main developments in this master thesis compared to the work carried out in the specialization project [2] was the introduction of stochastic properties. So far, the spot price has been considered stochastic, however other input parameters have the same uncertain behavior and could be modeled accordingly.

For instance the uncertainty of inflow can be modeled stochastic, along with price forecasts from all considered regulation markets. The latter parameters are rather hard to forecast correctly, but would be a more elegant solution to a multi-market optimization problem because of the relevance to practical decisions made by the hydropower producers.

Regarding multi-market optimization, a more advanced model could possibly consider multiple regulation markets in addition to the spot market, finding the optimal distribution of production and regulation capacity between for instance a spot market, a primary regulation market, a secondary regulation market and a tertiary regulation market. However, since these markets are dependent and sometimes interconnected with each other, it would prove a challenging task to complete.

It would also be interesting to analyze how the interaction between multiple participants in these markets are influencing for instance forecasting of regulation prices, given that all participants tries to maximize profit from the spot market and the considered regulation markets.

Lastly, it would prove useful to integrate many of the discussed parts of this report into a more streamlined decision tool for finding optimal levels of regulation obligations by implementing algorithms that analyses results and delivers helpful decision support.

9 Figures and tables

10 References

[1] Bjørnar Fjelldal, Sjur Messel Nafstad, and Gro Klæboe, "Optimal day-ahead electricity market bidding considering different ancillary services," *11th International Conference on the European Energy Market*, 2014.

[2] Bjørnar Fjelldal and Sjur Messel Nafstad, "Optimal pre-spot bidding considering different regulation markets," Department of Electric Power Engineering, Norwegian University of Science and Technology, 2013.

[3] The European Wind Energy Association, "Wind in power 2011 European statistics," The European Wind Energy Association, 2011.

[4] Statnett SF, *Systemdrifts- og marketsutviklingsplan 2012.*, 2012.

[5] Janusz W. Bialek, James R. Bumby Jan Machowski, *Power System Dynamics - Stability and Control*, 2nd ed.: Wiley, vol. 2012, pp. 335-382.

[6] Ivar Wangensteen, *Power System Economics - the Nordic Electricity Market*, 2nd ed.: Tapir academic press, 2011, pp. 275-309.

[7] Statnett. (2012) statnett.no. [Online]. http://www.statnett.no/Global/Dokumenter/Kraftsystemet/Marketsinformasjon/Frekvensstyrte%20og%20sekund%C3%A6re/LFC%20Vilk%C3%A5r%20for%20marketRevTilImplementering.pdf

[8] Swissgrid. (2014) swissgrid.ch. [Online]. http://www.swissgrid.ch/dam/swissgrid/experts/ancillary_services/Dokumente/D140101_AS-Products_V7R0_en.pdf

[9] Gerard Doorman, *Hydro Power Scheduling.*: Department of Electrical Power engineering, NTNU, 2012, pp. 11-20.

[10] Turid Follestad, "Documentation of the C program Scentreegen for scenario tree generation and reduction ," SINTEF Energi AS, 2012.

[11] John R. Birge and Franois Louveaux, *Introduction to Stochastic Programming.*: Springer, 1997, pp. 331-352.

[12] Eric C. Anderson. (2014, May) Monte Carlo Methods and Importance Sampling. [Online]. http://ib.berkeley.edu/labs/slatkin/eriq/classes/guest_lect/mc_lecture_notes.pdf

[13] Ronald E. Waypole, Raymond H. Myers, Sharon L. Myers, and Keying Ye, *Probability & Statistics for engineers and scientists.*: Pearson, 2012, pp. 225-264.

[14] Ronald E. Waypole, Raymond H. Myers, Sharon L. Myers, and Keying Ye,

Probability & Statistics for engineers and scientists.: Pearson, 2012, pp. 265-318.

[15] Gro Klæboe and Olav Bjarte Fosso, "Forecasting hourly electricity prices for bidding optimization," Department of Electric Power Engineering, Norwegian University of Science and Technology, 2013.

[16] Turid Follestad, "A description of the scenario tree reduction and construction algorithms implemented in Scentreegen ," Sintef Energi, 2012.

[17] Gro Klæboe, Anders Lund Eriksrud, and Stein-Erik Fleten, "Benchmarking time series based forecasting models for electricity balancing market prices," *Energy Systems*, December 2013.

[18] Ivar Wangensteen, *Power System Economics - the Nordic Electricity Market.*: Tapir academic press, 2011, pp. 15-32.

[19] Jan Lundgren, Mikael Rönnqvist, and Peter Värbrand, *Optimization.*: Studentlitteratur, 2010, pp. 77-106.

[20] Ronald E. Waypole, Raymond H. Myers, Sharon L. Myers, and Keying Ye, *Probability & Statistics for engineers and scientists.*: Pearson, 2012, pp. 319-388.

[21] Minitab. (2014, May) Identifying the Distribution of Your Data. [Online]. http://www.minitab.com/en-us/Support/Tutorials/Identifying-the-Distribution-of-Your-Data/

[22] Olav Bjarte Fosso, TET4515 ELK-14 Transmission System Operating Planning, 2013, University course at NTNU.

[23] Gerard Doorman, *Hydro Power Scheduling.*: Department of Electrical Power engineering, NTNU, 2012, pp. 34-55.

[24] NordPoolSpot. The day-ahead market - Elspot. [Online]. http://www.nordpoolspot.com/How-does-it-work/Day-ahead-market-Elspot-/

[25] NordPoolSpot. No. 02/2013 - 2012 Sees new record volume for Nord Pool Spot. [Online]. http://www.nordpoolspot.com/Message-center-container/Exchange-list/2013/01/No-22013-2012-Sees-new-record-volume-for-Nord-Pool-Spot/

[26] NordPoolSpot. Intraday market. [Online]. http://www.nordpoolspot.com/How-does-it-work/Intraday-market-Elbas/

[27] Nico van der Wijst, *Finance - A Quantitative Introduction*, 1st ed.: Cambridge, 2013.

[28] S.-O. Fridolfsson and T. Tanger˚as, "Market power in the Nordic electricity whole- sale market: A survey of the empirical evidence ," Research Institute of Industrial Economics (IFN), 2009.

11 Appendices

A A thorough explanation of the entire stochastic mechanism of the developed model

A.1 The SCENRED output data

The output from SCENRED is fully understandable, but needs to have a different structure in order to be utilized by the developed model. Whereas SCENRED depends on nodes (individual hours) in a scenario tree, the developed model depends on steps (24 hour blocks). Thus the output data from SCENRED needs a conversion in order to work in this model.

A.1.1 Definition of some terms

A few terms further explained in chapter 4.4.1, will also be used during this explanation;

Scenario: A full, unique path throughout the scenario tree, consisting of as many nodes as hours in the optimization period. Some nodes can (and will) be used by several scenarios, as the number of scenario tree branches increases throughout the scenario tree.

Node: A specific hour in a specific scenario, consisting of a node number, predecessor node, probability and price.

Step: A branching point in the scenario tree, where new information typically is revealed. In this case a step is equal to a day – 24 hours.

A.1.1 Description of output data from SCENRED

Two output files are used in this conversion process:

A.1.1 Tree.dat output

First, the *tree.dat* file consists of tree parameters:

- The predecessor node of the current node
- The probability in the current node
- The price in the current node

Table I: Example of output data in the tree.dat file from SCENRED

tree.dat:

(NODE)	PRED	PROB	PRICE
1	1	1.0000000000	348.7000
2	1	0.2926666667	338.1464
3	1	0.7073333333	354.0349
4	2	0.2926666667	328.3813
5	3	0.7073333333	355.9664
⋮	⋮	⋮	⋮
336	330	0.2110000000	381.6977

The node column is not explicitly written, but increases with each new line in the output file. Node 1 is a dummy node needed to have a branch at the first time step, and therefore has a probability equal to 1.

It is also noteworthy to understand that the node indices increase vertically, then horizontally, as illustrated in Figure I.

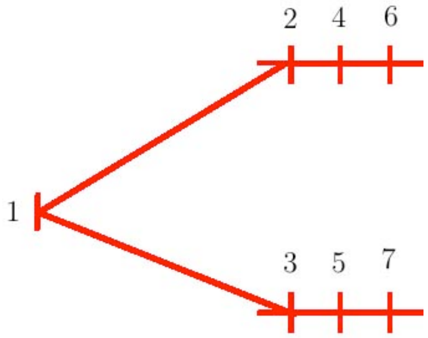


Figure I: How SCENRED indexes nodes in tree.dat relative to a scenario tree

A.1.2 Row.dat output

Secondly, the *row.dat* file consists of two parameters:

- The probability for a full, unique scenario path
- The price hour-by-hour for each scenario

Table II: Example of output data in the row.dat file from SCENRED

row.dat:	Explanation of each row
8.83333333333334e-002	Probability of Scenario 1
3.4870000000e+002	Price in hour 0 of Scenario 1 (dummy)
3.3814636210e+002	Price in hour 1 of Scenario 1
3.2838127347e+002	Price in hour 2 of Scenario 1
⋮	Price in hour 3 -> second last hour of Scenario 1
3.7722932700e+002	Price in last hour of Scenario 1
	Free space indicating a new scenario
⋮	*Reoccurring pattern for scenario 2 -> second last scenario*
2.10999999999998e-001	Probability of last scenario
3.4870000000e+002	Price in hour 0 of last scenario (dummy)
3.5403492181e+002	Price in hour 1 of last scenario
3.5596642491e+002	Price in hour 2 of last scenario
⋮	Price in hour 3 -> second last hour of last scenario
3.8169766065e+002	Price in last hour of last scenario

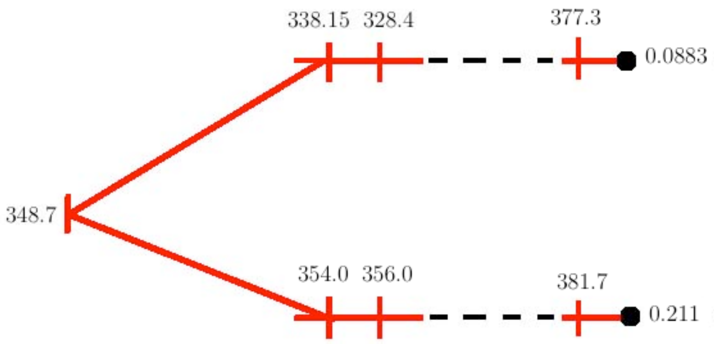


Figure II: How SCENRED indexes scenario prices and probabilities in row.dat relative to a scenario tree

The scenario number is not explicitly written for each scenario price column, but the file begins with scenario 1 and as a free space-row and a new probability appears, a new scenario is present.

A.2 Structuring the SCENRED output data and creating new matrices

Now, all the discussed parameters (except from the price relative to each node) must be read into matrices usable by the model. The node predecessors and node probabilities is obtained from *tree.dat*, whereas the scenario path prices and probabilities regarding each full scenario path is obtained from *row.dat*.
They are read into four matrices in the model, named

- ***NodePredecessor(n):*** Predecessor node for each node in the scenario tree.

- ***NodeProbability(n):*** Probability for each node in the scenario tree.

- ***ScenarioSpotPrice(e,h):*** The hourly price for each scenario in the scenario tree.

- ***EndScenarioProbability(e):*** The probability for each scenario in the scenario tree

where n is all nodes, e is all scenarios and h is all hours found in the scenario tree.

In addition to this, the model needs to know how many branches the scenario divides itself into during each step. This is read into a matrix named

- ***StepNumberOfBranch(t):*** The number of branches for each step in the scenario tree. *(This matrix must comply with the input parameters from SCENRED)*

where t is all steps found in the scenario tree.

A.2.1 Creating new matrices from SCENRED output data

As new information about the stochastic spot price occurs whenever a new step is initiated, the goal is to transform the recently initialized matrices mentioned above into matrices that work on a step-level instead, containing only the value for the first hour of the current step instead of all values for all hours/nodes. Therefore, the information stored in *NodePredecessor(n)* and *NodeProbability(n)* needs to be transformed into two steps-scenarios-matrices

- StepProbabilityDynamic(e,t): The probability of a scenario to occur at a given step

- StepPredecessorDynamic(e,t): The predecessor node of the node corresponding to the probability in *StepProbabilityDynamic(e,t)*

where e is all scenarios and t is all steps found in the scenario tree.

By doing this, it is possible to obtain information about which probability occurs at each step for all scenarios, and which predecessor node the corresponding node is connected to.
The dynamic property is related to the fact that they only contain as many probabilities and predecessor nodes for each unique path in the scenario tree – so if the tree branches into 2, 4 and 8 branches in steps 1, 2 and 3, these matrices will only contain two values in the first column, four values in the second column and eight values in the third column.
In addition to this, a matrix linking the predecessor nodes to the scenario price series must be created. Because the matrix *ScenarioSpotPrice(e,t)* contains price series for all scenarios and thus is not dynamic, the *StepPredecessorDynamic(e,t)* matrix must be converted into a full matrix, filling all elements in the last matrix needed by the model, named;
StepPredecessorFull(e,t): The predecessor node of the node corresponding to the price series in *ScenarioSpotPrice(e,t)*

A.3 An illustrative example of how the model converts the output data into usable matrices

The best way to explain the algorithm used to model the stochastic workflow of the model is by illustrating it via a small-scale example. Consider a three-step scenario tree with 2 branches in step 1, 4 branches in step 2 and 8 branches in step 3, as shown in Figure III.

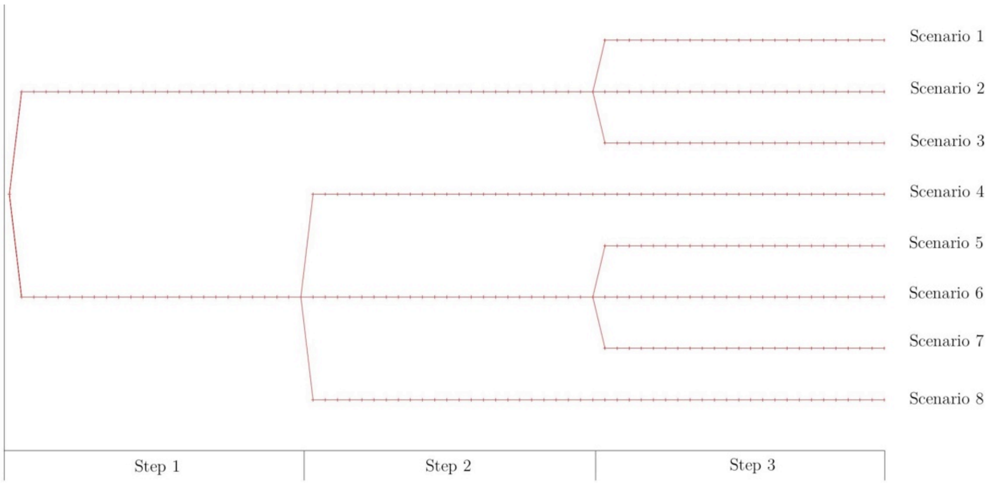


Figure III: A scenario tree with 2 branches in step 1, 4 branches in step 2 and 8 branches in step 3, yielding a total of 8 scenarios

The output from SCENRED is more or less read directly into the five following matrices;

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ 3 \\ \vdots \\ 330 \end{pmatrix}$$

Matrix I: NodePredecessor(n)

$$\begin{pmatrix} 1.000000000 \\ 0.2926666667 \\ 0.7073333333 \\ 0.2926666667 \\ 0.7073333333 \\ \vdots \\ 0.2110000000 \end{pmatrix}$$

Matrix II: NodeProbability(n)

$$\begin{pmatrix} 8.83333333333334\text{e-}002 \\ 1.15333333333334\text{e-}001 \\ 8.90000000000001\text{e-}002 \\ 2.21333333333331\text{e-}001 \\ 7.83333333333333\text{e-}002 \\ 1.10333333333334\text{e-}001 \\ 8.63333333333334\text{e-}002 \\ 2.10999999999998\text{e-}001 \end{pmatrix}$$

Matrix III: EndScenarioProbability(n)

$$\begin{pmatrix} 3.3814636210\text{e+}002 & \cdots & 3.5321259813\text{e+}002 & \cdots & 3.6996504957\text{e+}002 & \cdots & 3.7722932700\text{e+}002 \\ 3.3814636210\text{e+}002 & \cdots & 3.5321259813\text{e+}002 & \cdots & 3.4776862724\text{e+}002 & \cdots & 3.7556428774\text{e+}002 \\ 3.3814636210\text{e+}002 & \cdots & 3.5321259813\text{e+}002 & \cdots & 3.2230479252\text{e+}002 & \cdots & 3.7181173959\text{e+}002 \\ 3.5403492181\text{e+}002 & \cdots & 3.4430806643\text{e+}002 & \cdots & 3.5122913388\text{e+}002 & \cdots & 3.7428140736\text{e+}002 \\ 3.5403492181\text{e+}002 & \cdots & 3.6334452893\text{e+}002 & \cdots & 3.3520182830\text{e+}002 & \cdots & 3.7687502228\text{e+}002 \\ 3.5403492181\text{e+}002 & \cdots & 3.6334452893\text{e+}002 & \cdots & 3.5681045540\text{e+}002 & \cdots & 3.7646913274\text{e+}002 \\ 3.5403492181\text{e+}002 & \cdots & 3.6334452893\text{e+}002 & \cdots & 3.7862378559\text{e+}002 & \cdots & 3.8063768916\text{e+}002 \\ 3.5403492181\text{e+}002 & \cdots & 3.8334410033\text{e+}002 & \cdots & 3.6960673717\text{e+}002 & \cdots & 3.8169766065\text{e+}002 \end{pmatrix}$$

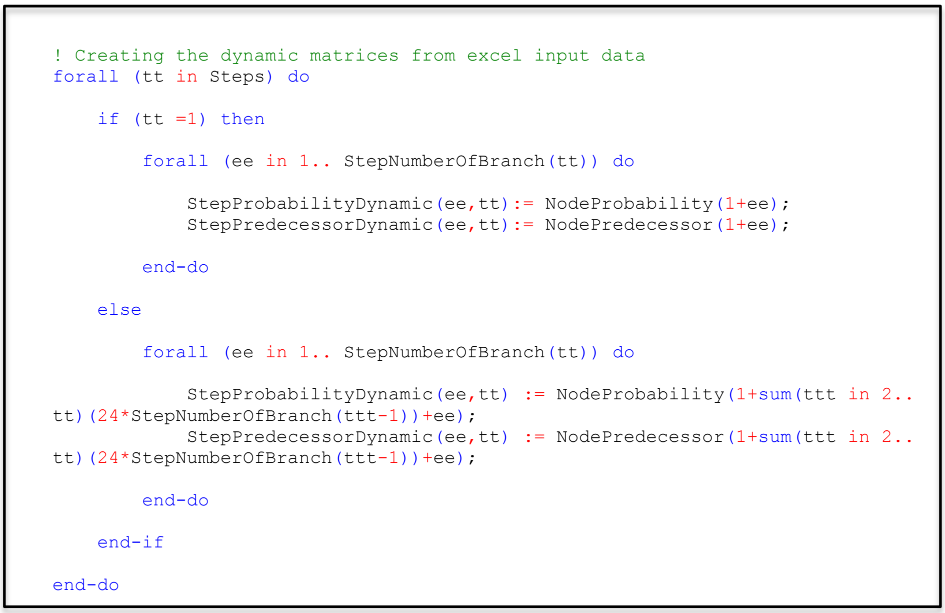
Matrix IV: ScenarioSpotPrice(e,t)

$$\begin{pmatrix} 2 & 4 & 8 \end{pmatrix}$$

Matrix V: StepNumberOfBranches(t)

Now, the matrices dependent on nodes must be converted into matrices dependent on steps.

This is done by the following code, also available as a part of appendix E;



```
! Creating the dynamic matrices from excel input data
forall (tt in Steps) do

    if (tt =1) then

        forall (ee in 1.. StepNumberOfBranch(tt)) do

            StepProbabilityDynamic(ee,tt):= NodeProbability(1+ee);
            StepPredecessorDynamic(ee,tt):= NodePredecessor(1+ee);

        end-do

    else

        forall (ee in 1.. StepNumberOfBranch(tt)) do

            StepProbabilityDynamic(ee,tt)  := NodeProbability(1+sum(ttt in 2..
tt)(24*StepNumberOfBranch(ttt-1))+ee);
            StepPredecessorDynamic(ee,tt)  := NodePredecessor(1+sum(ttt in 2..
tt)(24*StepNumberOfBranch(ttt-1))+ee);

        end-do

    end-if

end-do
```

This results in two new matrices:

$$\begin{pmatrix} 0.29 & 0.29 & 0.09 \\ 0.71 & 0.22 & 0.11 \\ - & 0.28 & 0.09 \\ - & 0.21 & 0.22 \\ - & - & 0.08 \\ - & - & 0.11 \\ - & - & 0.09 \\ - & - & 0.21 \end{pmatrix}$$

Matrix VI: StepProbabilityDynamic(e,t)

$$\begin{pmatrix} 1 & 48 & 142 \\ 1 & 49 & 142 \\ - & 49 & 142 \\ - & 49 & 143 \\ - & - & 144 \\ - & - & 144 \\ - & - & 144 \\ - & - & 145 \end{pmatrix}$$

Matrix VII: StepPredecessorDynamic(e,t)

With this information, it is possible to plot the scenario tree and adding the probabilities and predecessor nodes, giving a greater overview of the stochastic problem;

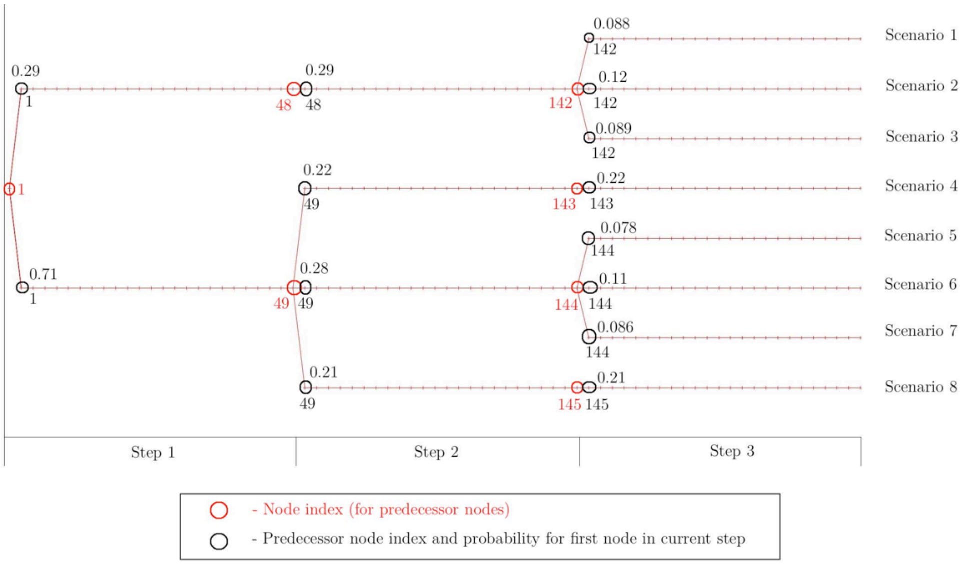


Figure IV: The scenario tree with probabilities and predecessor nodes

As mentioned, the *StepPredecessorDynamic(e,t)* must also be converted into an explicit version, making it able to correspond with the scenario prices found in *ScenarioSpotPrice(e,h)*. This is done by the following code, also available as a part of appendix E;

```
!Creating the explicit matrix from the dynamic matrix

    !Creating last column of the explicit matrix
    forall (ee in Scenarios) do

        StepPredecessorFull(ee,getsize(Steps)):=StepPredecessorDynamic(ee,getsize(Steps));

    end-do

    ! Creating the remaining columns of the explicit matrix
    forall (tt in 1.. getsize(Steps)-1) do

        EqualNumbersCounter := 0;
        IteratorCounter:= 1;
        Counter:= 1;

        forall (ee in 2.. getsize(Scenarios)) do

            if (StepPredecessorFull(ee-1,1+getsize(Steps)-tt) =
StepPredecessorFull(ee,1+getsize(Steps)-tt)) then

                EqualNumbersCounter:= EqualNumbersCounter + 1;

            else
                forall(pp in IteratorCounter.. IteratorCounter+EqualNumbersCounter) do

                        StepPredecessorFull(pp,(getsize(Steps)-tt)) :=
StepPredecessorDynamic(Counter,getsize(Steps)-tt);
                end-do

                Counter:= Counter + 1;
                IteratorCounter := IteratorCounter + EqualNumbersCounter +1;
                EqualNumbersCounter :=0;

            end-if

        end-do

    end-do

        forall (tt in 1.. getsize(Steps)-1) do

            StepPredecessorFull(getsize(Scenarios),getsize(Steps)-tt):=
            StepPredecessorDynamic(StepNumberOfBranch(getsize(Steps)-tt),getsize(Steps)-
tt);

        end-do
```

which creates the last matrix needed in order to have the stochastic algorithm in the model work:

$$\begin{pmatrix} 1 & 48 & 142 \\ 1 & 48 & 142 \\ 1 & 48 & 142 \\ 1 & 49 & 143 \\ 1 & 49 & 144 \\ 1 & 49 & 144 \\ 1 & 49 & 144 \\ 1 & 49 & 145 \end{pmatrix}$$

Matrix VIII: StepPredecessorFull(e,t)

Now, with all matrices created, the model is able to iterate through the scenario tree.

A.3.1 Step 1

Before starting the optimization, the model always stands in node 1, ready to start step 1.

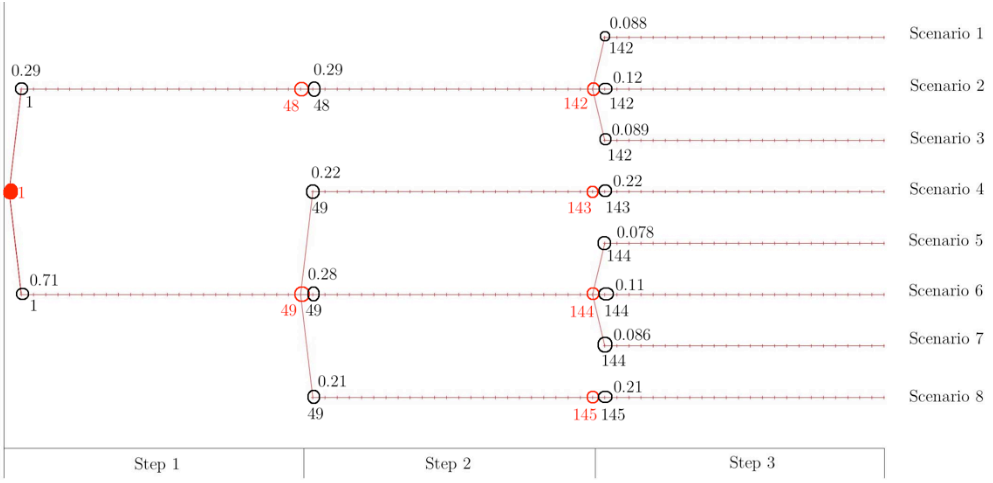
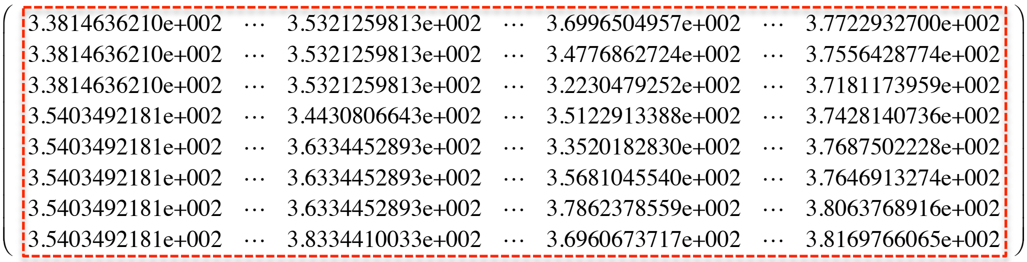


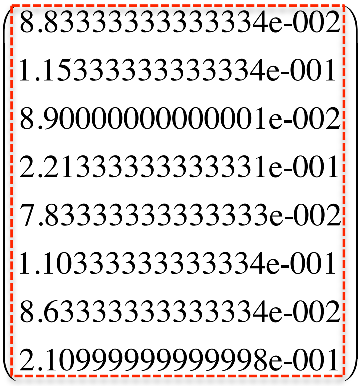
Figure V: Where the model stands in the scenario tree before step 1

Not knowing the stochastic price outcome of step 1, the model takes a probability weighted average of all scenarios - also called the model price - into account when optimizing all hours from the start of step 1 until the end of step 3.



$$\begin{pmatrix} 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.6996504957e{+}002 & \cdots & 3.7722932700e{+}002 \\ 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.4776862724e{+}002 & \cdots & 3.7556428774e{+}002 \\ 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.2230479252e{+}002 & \cdots & 3.7181173959e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.4430806643e{+}002 & \cdots & 3.5122913388e{+}002 & \cdots & 3.7428140736e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.3520182830e{+}002 & \cdots & 3.7687502228e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.5681045540e{+}002 & \cdots & 3.7646913274e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.7862378559e{+}002 & \cdots & 3.8063768916e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.8334410033e{+}002 & \cdots & 3.6960673717e{+}002 & \cdots & 3.8169766065e{+}002 \end{pmatrix}$$

Matrix IX: Scenario spot price series taken into account by the model before step 1



$$\begin{pmatrix} 8.83333333333334e{-}002 \\ 1.15333333333334e{-}001 \\ 8.90000000000001e{-}002 \\ 2.21333333333331e{-}001 \\ 7.83333333333333e{-}002 \\ 1.10333333333334e{-}001 \\ 8.63333333333334e{-}002 \\ 2.10999999999998e{-}001 \end{pmatrix}$$

Matrix X: Corresponding end-scenario probabilities taken into account by the model before step 1

All scenario prices with corresponding probabilities are therefore taken into account at the start of step 1.

A.3.2 Step 2

After the first step is optimized, new information regarding the actual outcome of step 1 is revealed, before optimizing at the start of step 2. This is done by a random event, drawing a number between 0 and 1. Referring to Matrix VI, if the number is below or equal to $\mathbf{0.29}$, Scenario 1-3 will be happening, and all scenarios with a corresponding probability having a predecessor node equal to $\mathbf{48}$ can occur in step 2. If else, Scenario 4-8 will be happening, and all scenarios with a corresponding probability having a predecessor node equal to $\mathbf{49}$ can occur in step 2. The random event produces the number $\mathbf{0.57}$, and the latter alternative happens:

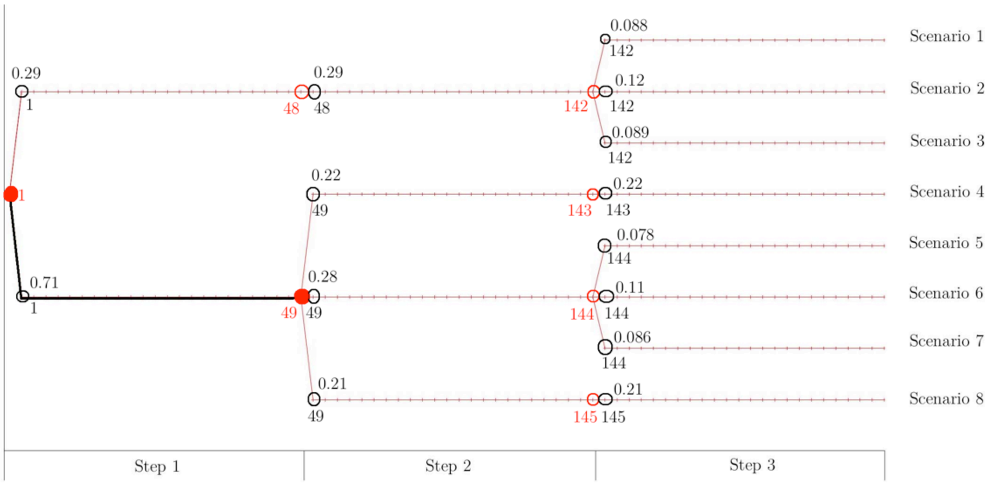


Figure VI: Where the model stands in the scenario tree before step 2

Now, having the same structure in *StepProbabilityDynamic(e,t)* and *StepPredecessorDynamic(e,t)*, the algorithm says that since **0.71** lies on the **second** row in the first column (step 1) in *StepProbabilityDynamic(e,t)*, every scenario with a predecessor equal to the **second** numeric quantity in the second column (step 2) in *StepPredecessorDynamic(e,t)* should be probability weighted when optimizing step 2.

$$\begin{pmatrix} 0.29 & 0.29 & 0.09 \\ \boxed{0.71} & 0.22 & 0.11 \\ - & 0.28 & 0.09 \\ - & 0.21 & 0.22 \\ - & - & 0.08 \\ - & - & 0.11 \\ - & - & 0.09 \\ - & - & 0.21 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 48 & 142 \\ 1 & \boxed{49} & 142 \\ - & 49 & 142 \\ - & 49 & 143 \\ - & - & 144 \\ - & - & 144 \\ - & - & 144 \\ - & - & 145 \end{pmatrix}$$

Matrix XI & XII: The connection between the realized probability in step 1 and the predecessor node for the achievable probabilities in step 2

In this case, this yields all scenarios with the predecessor node **49** from step 2. Thus, all scenarios with the same scenario numbers as the rows in *StepPredecessorFull(e,t)* is optimized upon with corresponding probability weights for the remaining hours.

$$\begin{pmatrix} 1 & 48 & 142 \\ 1 & 48 & 142 \\ 1 & 48 & 142 \\ 1 & 49 & 143 \\ 1 & 49 & 144 \\ 1 & 49 & 144 \\ 1 & 49 & 144 \\ 1 & 49 & 145 \end{pmatrix}$$

$$\Downarrow$$

$$\begin{pmatrix} 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.6996504957e{+}002 & \cdots & 3.7722932700e{+}002 \\ 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.4776862724e{+}002 & \cdots & 3.7556428774e{+}002 \\ 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.2230479252e{+}002 & \cdots & 3.7181173959e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.4430806643e{+}002 & \cdots & 3.5122913388e{+}002 & \cdots & 3.7428140736e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.3520182830e{+}002 & \cdots & 3.7687502228e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.5681045540e{+}002 & \cdots & 3.7646913274e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.7862378559e{+}002 & \cdots & 3.8063768916e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.8334410033e{+}002 & \cdots & 3.6960673717e{+}002 & \cdots & 3.8169766065e{+}002 \end{pmatrix}$$

$$\Downarrow$$

$$\begin{pmatrix} 8.83333333333334e{-}002 \\ 1.15333333333334e{-}001 \\ 8.90000000000001e{-}002 \\ 2.21333333333331e{-}001 \\ 7.83333333333333e{-}002 \\ 1.10333333333334e{-}001 \\ 8.63333333333334e{-}002 \\ 2.10999999999998e{-}001 \end{pmatrix}$$

Matrix XIII, XIV & XV: The connection between the scenarios still achievable in step 2 due to the predecessor node, the corresponding scenario spot prices and the corresponding end scenario probabilities

In this case, scenarios 3-8 are still achievable, and will be probability weighted when optimizing step 2.

A.3.3 Step 3

After step 2 is optimized, new information is again revealed in anticipation of step 3. The scenarios still achievable when calculating what actually occurred in step 2, are the probabilities situated on the same row indexes in the **second** column of *StepProbabilityDynamic(e,t)* as the predecessor that occurred during step 2 situated on the **second** column of *StepPredecessorDynamic(e,t).*

$$\begin{pmatrix} 1 & 48 & 142 \\ 1 & 49 & 142 \\ - & 49 & 142 \\ - & 49 & 143 \\ - & - & 144 \\ - & - & 144 \\ - & - & 144 \\ - & - & 145 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.29 & 0.29 & 0.09 \\ 0.71 & 0.22 & 0.11 \\ - & 0.28 & 0.09 \\ - & 0.21 & 0.22 \\ - & - & 0.08 \\ - & - & 0.11 \\ - & - & 0.09 \\ - & - & 0.21 \end{pmatrix}$$

Matrix XVI & XVII: The connection between the predecessor node that occurred in step 2 and the scenario probabilities still achievable in step 3

The random event is yet again initiated, drawing a new number between 0 and 1.

0.45 occurs, but this number needs to be adjusted due to the fact that some scenarios from step 2 no longer are achievable, as shown in Matrix XVI.

The random number is adjusted by multiplying with the previous realized probability (**0.71**), because the sums of the probabilities of the branches in step 2 from the previous probability in step 1 always equal each other.

This gives an adjusted value of **0.3195**, yielding the second probability to actually occur (**0.28**), since **0.3195** is larger than 0.22, but smaller than 0.22+0.28.

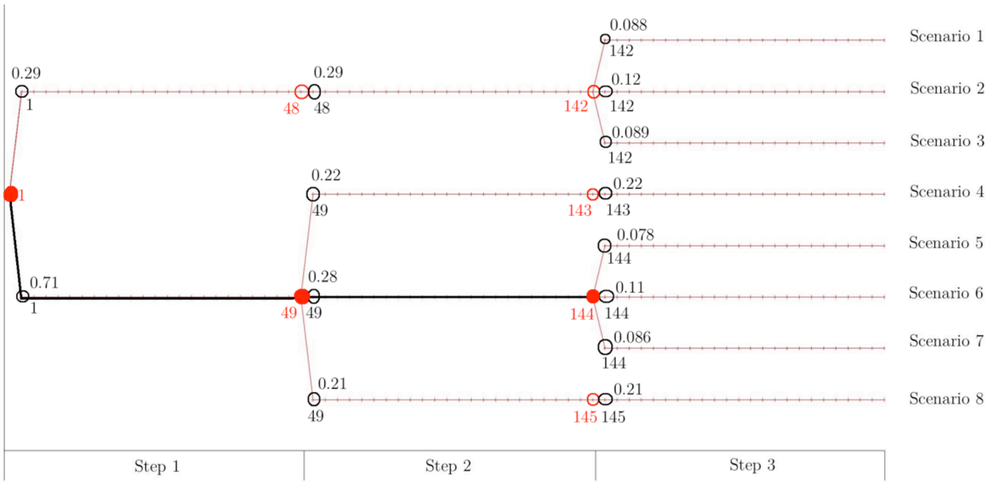


Figure VII: Where the model stands in the scenario tree before step 2

Since **0.28** lies on the **third** row in the **second** column (step 2) in *StepProbabilityDynamic(e,t)*, every scenario with a predecessor equal to the **third** numeric quantity in the **third** column (step 3) in *StepPredecessorDynamic(e,t)* should be probability weighted when optimizing step 3.

$$\begin{pmatrix} 0.29 & 0.29 & 0.09 \\ 0.71 & 0.22 & 0.11 \\ - & \boxed{0.28} & 0.09 \\ - & 0.21 & 0.22 \\ - & - & 0.08 \\ - & - & 0.11 \\ - & - & 0.09 \\ - & - & 0.21 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 48 & 142 \\ 1 & 49 & 142 \\ - & 49 & 142 \\ - & 49 & 143 \\ - & - & \boxed{144} \\ - & - & 144 \\ - & - & 144 \\ - & - & 145 \end{pmatrix}$$

Matrix XVIII & XIX: The connection between the realized probability in step 2 and the predecessor node for the achievable probabilities in step 3

Every scenario price with a corresponding predecessor node of **144** is now being taken into account.

$$\begin{pmatrix} 1 & 48 & 142 \\ 1 & 48 & 142 \\ 1 & 48 & 142 \\ 1 & 49 & 143 \\ 1 & 49 & 144 \\ 1 & 49 & 144 \\ 1 & 49 & 144 \\ 1 & 49 & 145 \end{pmatrix}$$

$$\Downarrow$$

$$\begin{pmatrix} 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.6996504957e{+}002 & \cdots & 3.7722932700e{+}002 \\ 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.4776862724e{+}002 & \cdots & 3.7556428774e{+}002 \\ 3.3814636210e{+}002 & \cdots & 3.5321259813e{+}002 & \cdots & 3.2230479252e{+}002 & \cdots & 3.7181173959e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.4430806643e{+}002 & \cdots & 3.5122913388e{+}002 & \cdots & 3.7428140736e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.3520182830e{+}002 & \cdots & 3.7687502228e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.5681045540e{+}002 & \cdots & 3.7646913274e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.6334452893e{+}002 & \cdots & 3.7862378559e{+}002 & \cdots & 3.8063768916e{+}002 \\ 3.5403492181e{+}002 & \cdots & 3.8334410033e{+}002 & \cdots & 3.6960673717e{+}002 & \cdots & 3.8169766065e{+}002 \end{pmatrix}$$

$$\Downarrow$$

$$\begin{pmatrix} 8.83333333333334e{-}002 \\ 1.15333333333334e{-}001 \\ 8.90000000000001e{-}002 \\ 2.21333333333331e{-}001 \\ 7.83333333333333e{-}002 \\ 1.10333333333334e{-}001 \\ 8.63333333333334e{-}002 \\ 2.10999999999998e{-}001 \end{pmatrix}$$

Matrix XX, XXI & XXII: The connection between the scenarios still achievable in step 3 due to the predecessor node, the corresponding scenario spot prices and the corresponding end scenario probabilities

In this case, scenarios 5-7 are still achievable, and will be probability weighted when optimizing step 3.

Since this was the last step of the scenario tree, no further calculations needs to be done. However, in order to complete the actual realized price profile including the last period as well, a final random event is initiated. This information will not have any influence on the optimization results as the model is done optimizing the last step, but it is useful to observe what actually occurred during this last step as well.

The scenarios still achievable when calculating what actually occurred in step 3, are the probabilities situated on the same row indexes in the **third** column of *StepProbabilityDynamic(e,t)* as the predecessor that occurred during step 3 situated on the **third** column of *StepPredecessorDynamic(e,t)*.

$$\begin{pmatrix} 1 & 48 & 142 \\ 1 & 49 & 142 \\ - & 49 & 142 \\ - & 49 & 143 \\ - & - & \boxed{144} \\ - & - & \boxed{144} \\ - & - & \boxed{144} \\ - & - & 145 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.29 & 0.29 & 0.09 \\ 0.71 & 0.22 & 0.11 \\ - & 0.28 & 0.09 \\ - & 0.21 & 0.22 \\ - & - & \boxed{0.08} \\ - & - & \boxed{0.11} \\ - & - & \boxed{0.09} \\ - & - & 0.21 \end{pmatrix}$$

Matrix XXIII & XXIV: The connection between the predecessor node that occurred in step 3 and the final scenario probabilities achievable after step 3

The random event produces the number **0.77**, and gets adjusted by being multiplied with **0.28**, so it equals **0.2156**. The actual probability to occur in step 3 is therefore **0.09**, because **0.2156** is greater than 0.08, greater than 0.08+0.11 but smaller than 0.08+0.11+0.09.

$$\begin{pmatrix} 0.29 & 0.29 & 0.09 \\ 0.71 & 0.22 & 0.11 \\ - & 0.28 & 0.09 \\ - & 0.21 & 0.22 \\ - & - & 0.08 \\ - & - & 0.11 \\ - & - & \boxed{0.09} \\ - & - & 0.21 \end{pmatrix}$$

Matrix XXV: The actual, realized scenario probability realized after step 3

As illustrated in Figure VIII, Scenario 7 was the scenario that in the end actually occurred.

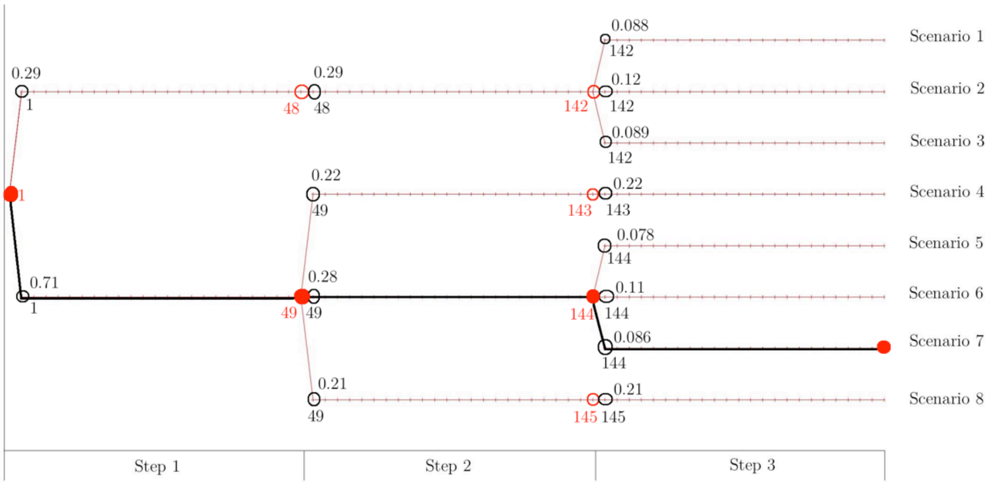


Figure VIII: The finished, actual path of the scenario tree after step 3

B Additional theory (chapter 2 from [2])

There are three main properties that measure the state of the electrical power system [4]:

- The frequency quality
    - Frequency containment
    - Frequency restoration
    - Reserve replacement
- The time of operation with reduced reliability (time outside the N-1 criteria)
- The voltage quality

B.1 Frequency deviation

This project focuses on optimization between a spot market and a corresponding regulation market. The purpose of regulation markets is to increase the frequency quality. Hence, given the three main properties that describe the state of an electrical power system it is the first property that is focused on in this project.

An electrical power system cannot store electrical energy and there has to be a perfect balance between production and consumption at all times. If there is an unbalance between production and consumption in the system this will cause a frequency deviation from the nominal frequency in the system. If the consumption is greater than the production the frequency will decrease. On the other hand, if the production is greater than the consumption the frequency will increase [5].

This physical property combined with the fact that consumers turn on and off their loads at all times, results in a need to regulate the system in order to obtain a stable operation and frequency. This responsibility lies on the Transmission System Operator (TSO) [4].

In theory the TSO could regulate either the production, the consumption or both of them in order to obtain balance in the system. However, due to physical and practical factors, the absolutely most common thing today is to regulate the producers in order to obtain balance. This may change in a future Smart Grid system where consumers also can deliver regulating services and hence can be used by the TSO to obtain balance in the system [22].

However, not all producers are suited to regulate their power production on short notice, and hence not able to deliver regulation services to the system. This, in general, applies to producers of thermal power, solar power, wind power and nuclear power. Thermal and nuclear power plants do not meet the requirement of being able to change the production quickly. When it comes to solar and wind power, these sources are non-predictable and hence not suited to contribute with regulation. Hydropower, especially hydro plants with reservoirs, on the other hand is highly suited for delivering such services.
Therefore, it is possible for hydropower producers to bid into both the spot market and the regulation markets. By doing this they get an extra income because of their possibility to be flexible [23].

B.2 The current state of the grid frequency

Over the recent years, the frequency quality in the Nordic region has decreased. Figure 1 illustrates that the proportion of time with a frequency outside 49.90-50.10 Hz has drastically increased the last 15 years.

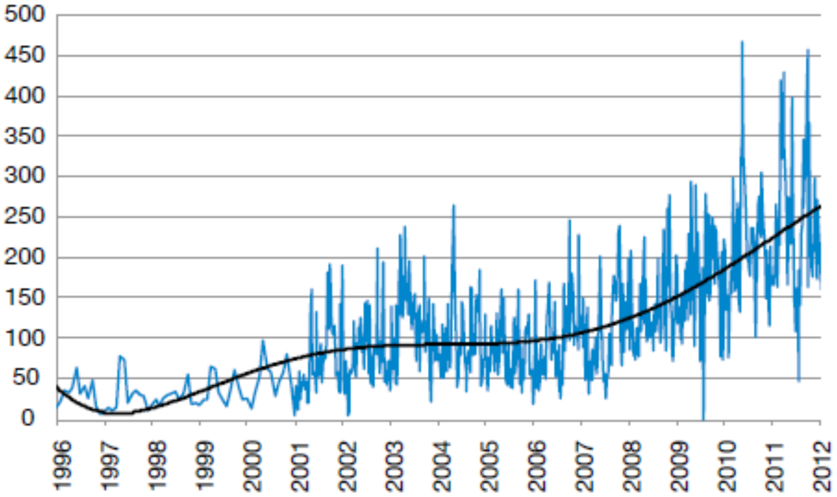


Figure IX: Development in frequency quality given in minutes outside 49.90-50.10 HZ per month [4]

This reduces the reliability of the system, because when the frequency is outside of the 49.90-50.10 Hz interval some of the regulation capacity is already in used. Hence, it would not be able to contribute with regulation in order to keep the system in equilibrium should a production failure occur.

There are several reasons why the frequency quality has decreased the recent years. A large proportion of the unbalance happens during hour shifts, see Figure X. This is because the producers have unit commitments that change every hour, while the consumption changes continually throughout the day. This causes unbalance between production and consumption during the hour shifts. This phenomenon happens most often during the parts of the day with large change of consumption in the system, especially early in the morning and in the evening. More HVDC connections and more changes in the power flow direction throughout the day also increase the unbalance [4].

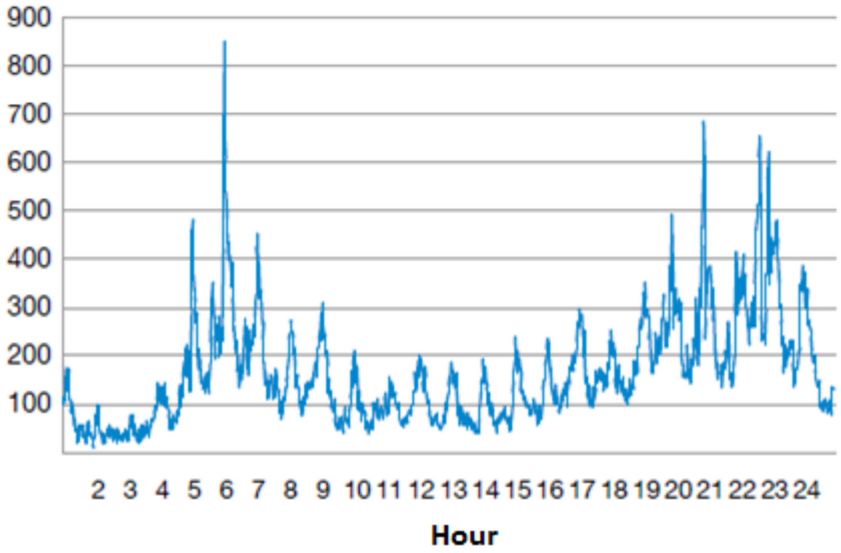


Figure X: Number of events with frequency outside 49.90-50.10 HZ distributed over a day (adapted from [4])

Another reason to this negative development in frequency deviation is that the total load of the grid has increased during the recent years. This leads to more congestion in the grid, which again excludes producers in some part of the grid from contributing with regulation capacity. Hence, the total regulation capacity available to the TSO decreases. Regulation capacity can be defined as available and flexible production capacity that can be utilized to prevent unbalance between production and consumption on a short notice [4].

A third reason for this increase is that the power system in an increasingly proportion of the time is operated without the large hydropower units with large reservoirs. This is due to the fact that more cables between the Nordic region and the continent yields more import and expert. In these situations the load is covered by import and production that cannot be regulated that easily. The amount of regulation capacity in the Nordic system is strongly connected to these large hydropower units. Hence, when these units are turned off the available regulation capacity in the system is strongly reduced [4].

The last thing that contributes to this increase in frequency deviation in the Nordic region is something called 60-seconds-oscillations. These are slow oscillations in the frequency that probably are caused by different settings and responses on corresponding units in the Nordic region. These oscillations have increased over the recent years [4].

B.3 Different power markets

B.3.1 The spot market

The spot market is the main market for a hydropower producer. This is the market where the hydropower producer decide how much he wants to produce and at which price. It is a day-ahead market meaning the contracts agreed upon defines prices and volumes for the following day. In the Nordic region, this market is called Elspot and is organized by Nord Pool Spot [6], [23].

B.3.2 The internal trading market

The internal trading market is an intra-day-market where participants can buy and sell themselves into balance until one hour before delivery. In this case balance means the balance between the producers actual production and their obligation to the market manager from the day-ahead spot market. The majority of the power is traded in the spot market, but if something unexpected happens participants can use this market to buy themselves into balance before the hour of delivery. In the Nordic region, this market is called Elbas and is organized by Nord Pool Spot. This market has become more important the last years due to the increasing share of non-controllable renewable energy in the system.

However, the volume traded in this market is still very low (3.2 TWh in 2012) [6], [24], [25].

B.4 Regulation markets

To avoid unbalance between production and consumption the TSO controls several regulation capacity markets with different characteristics.
There are three levels of regulation:
- Primary regulation capacity
- Secondary regulation capacity
- Tertiary regulation capacity

These regulation levels have different characteristics in terms of how they are activated and what level of power they can deliver.

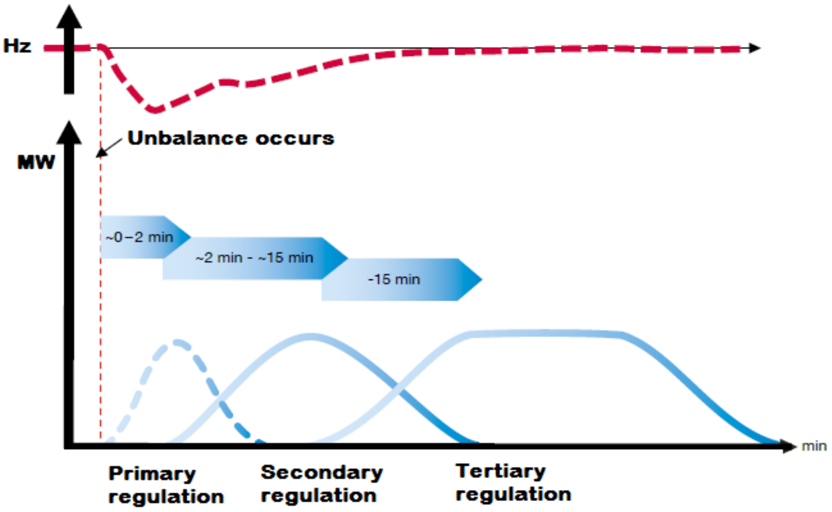


Figure XI: Regulation markets (adapted from [4])

B.4.1 The primary regulation market

The first thing that will happen if something causes the consumption to exceed the production, is that the rotating energy of the system will be converted to electrical power. This will cause the frequency in the system to decrease. On the other hand, if the production exceeds the consumption, the rotating energy in the system and hence the frequency, will increase [5].

This leads to the activation of the primary regulation in the system. Primary regulation is defined as regulation of turbine governors keeping the set-point of the generator constant. In this case the governor systems means the system controlling the power delivered to the prime mover shaft of the unit. In the case of hydropower this will be the blade position regulating the intake of water into the turbine [5].

The primary regulation is controlled by an automatic regulating system, and is activated within seconds after a frequency deviation [4]. For a generator to be able to contribute with primary regulation it must be rotating when the frequency deviation occurs. The amount of primary regulation power a generator contributes with in a case of frequency deviation is given by the generator droop setting [5].

When a power producer bids capacity into the primary market, he specifies how much power he will contribute with in a case of a given frequency deviation. Hence, the producer specifies what kind of droop setting he will use on his generators as well. It is also important to make sure that the generator is capable of increasing its power output. If the generator already produces at its maximum or its minimum it cannot respectively increase or decrease its output as a response to a frequency deviation.

B.4.1.1 Droop settings

How much the power output from a unit changes with a change in frequency is given by the droop settings of the unit. The formal definition of the droop coefficient **d** is given in Figure XII.

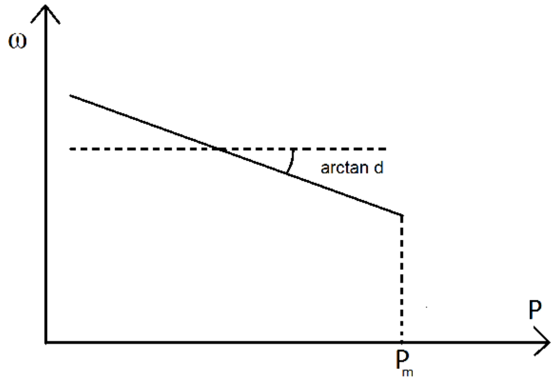


Figure XII: Definition of droop coefficient [5]

A more physical interpretation of the droop coefficient is the required percentage change in frequency to move the valves controlling the water inlet to the turbine from a fully open to a fully closed position. Mathematically this can be described by equation (I).

$$d = \frac{\frac{\Delta\omega}{\omega_0}}{\frac{\Delta P}{P_m}} \tag{I}$$

This formula assumes a linear relationship between the valve positions and the mechanical power delivered to the generator. $\Delta\omega$ is representing the change in frequency, ω_0 is the nominal frequency, ΔP is the change in output power with the given change in frequency, and P_m is the maximum possible production on the unit.

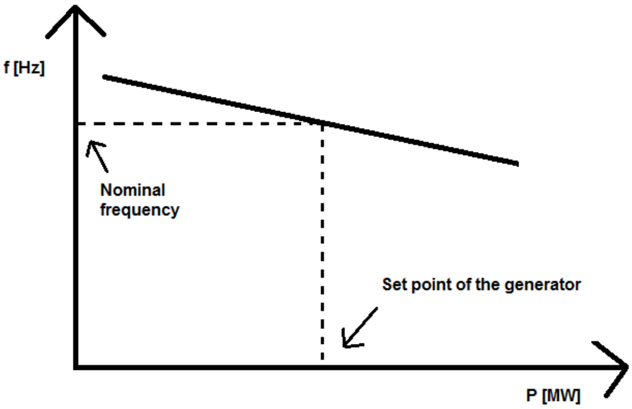


Figure XIII: Droop setting (adapted from [5])

When bidding into both the spot market and the primary regulation market the producers specifies the curve in Figure XIII for each unit during each hour. This figure describes the set point of the unit and how rapidly the production should change in the case of a frequency deviation. It is a linear function where the slope is given by the droop setting and the intersection is given by the set point of the generator.

B.4.2 The secondary regulation market

The secondary regulation has many similarities compared with the primary regulation. However, while the primary regulation is activated automatically by the droop setting on each unit, the secondary regulation is activated automatically by the TSO. For a unit to be able to contribute with secondary reserves it must be connected directly to the TSOs secondary regulation controlling system. This is an automated system that detects frequency deviations too large or too long lasting for the primary regulation to handle. The system then sends a signal to the units contributing with secondary regulation telling them to increase or decrease the production by changing the setpoint of the generators [4].

The most important difference between the primary and secondary regulation is the activation time. While the primary regulation responds in seconds to a change in frequency, the response time for the secondary regulation are 120-240 seconds after the signal from the TSO is received. In addition to bringing the frequency back to 50 Hz an important purpose of the secondary capacity is to free the primary regulation. When the primary regulation is activated the system has reduced capability to respond immediately to further frequency deviations. Hence, it is important to free the primary regulation capacity as quickly as possible after the activation. This makes the system capable of responding to another change in frequency [4].

B.4.3 The tertiary regulation market

The last regulation market is the tertiary market, also known as the balancing market. Unlike the primary and secondary regulation, this regulation is manually activated and has a maximum response time of 15 minutes. The purpose is to handle differences between production and consumption in the system over time, and to free the primary and secondary regulation. Producers can bid into this market by telling the market manager how much compensation they require to change their production [4].

C Implementation techniques and pre-assumptions (chapter 4.1 and 4.2 from [2])

C.1 Implementation of efficiency curves

One of the key factors of a watercourse model is the relation between the discharge of water through the turbine and the power output from the corresponding generator. The input data is typically an efficiency curve, describing different sets of turbine efficiencies with respect to water discharge.

The efficiency curve is often given as an efficiency-discharge-curve, where the turbines efficiency is plotted against the discharged water.
Transforming the efficiency-discharge-curve into at power-discharge-curve, and then doing a piecewise linearization of the new curve (illustrated in Figure XIV), gives the advantage of easily reading off the given power output at the intersection between the lowest linear curve and the chosen water discharge.
The solution space is limited by the cuts and the x-axis due to the non-negativity properties of the variables. The solution will lie on the border of the solution space because of the optimization process, and will often be found as a corner solution at the intersection of two cuts.

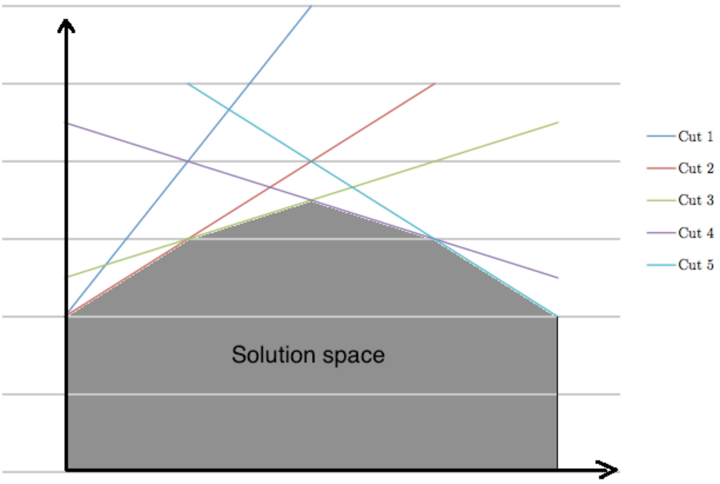


Figure XIV: A piecewise linearization of a concave function

The piecewise linearization is a consequence of the limitations given by the solver of the model - it uses mathematical algorithms that require all equations to be linear in order to solve the problem linearly.

It is also worth noting that for the piecewise linearization to give proper results, the original function needs to be concave. If not, each linear cut needs to have a restricted region of operation.

C.2 Pre-assumptions and simplifications

When building a model from scratch, some pre-assumptions were made in order to produce results within a given period of time. In this case, the following simplifications was done:

C.2.1 Constant head factor during production

The head of the water with respect to the turbine is proportional to the produced power in the unit, as equation (II) clearly states.

$$P = \rho g H Q \eta \tag{II}$$

This means a lower power output will be achieved from the same amount of water when the reservoir is half full rather than at HRV. The head of the water and the discharge is also interconnected, making the problem non-linear. Thus, this feature was neglected in this case.

C.2.2 Same relative efficiency curve for all units

Each production unit has it's own unique efficiency curve, stating how much of the potential energy of the water the turbine is able to convert into electrical energy. But, given that all units on the same station are using the same type of turbine (i.e. Francis), the shape of the efficiency curve is relatively similar between the units. By using the same shape, but correspondingly different levels of maximum discharge, the input data to the model is heavily simplified, and the code itself gets more compact. This yields no difference between units of the same size, they are practically identical and will be used interchangeably, but the analysis of the watercourse itself is still realistic enough.

C.2.3 No change in use of water when regulation is activated

When upward regulation is activated, there is a small increase of water used for a short period of time, and vice versa for downward regulation. This effect is neglected in both models taking a regulation market into account. Because of differences in activated volume between the primary and the secondary regulation market, the effect of this is most prominent for the secondary regulation market. However, the water spared during downward regulation counteracts some of the increase in water used during upward regulation. This assumption implies a close-to-none cost of participating in the regulation markets. The only exception is when this leads to a production that lies far from the best point on the efficiency curve.

C.2.4 Cascade-connected reservoirs

The watercourse in this case study consists of cascade-connected reservoirs, and the model was built to conform to this property.
Hence, the reservoirs need to be connected to each other in series, rather than in parallel. This is due to the water balance constraints, which says that discharge/bypass from Reservoir r is transferred to Reservoir (r+1), and from Reservoir (r+1) to Reservoir (r+2) accordingly. One could of course have multiple reservoirs discharging into the same lower elevated reservoir, but as of now they need to be summed up in to one larger reservoir in the input data.

C.2.5 Best-point strategy when calculating the end reservoir value

When calculating the total value of the remaining water in the reservoir at the end of the optimization period, one does not know how the water is going to be used in the future. It is possible to either produce at low discharge to fulfill a regulation bid, produce optimally to get the most out of the water, or produce at maximum discharge to avoid spilling. The most plausible assumption is that the unit produces at its best-point; therefore this efficiency value was being used when calculating the value of the end reservoir.

C.2.6 Water values assumed known and independent

The water values are not calculated, but assumed known to the producer, i.e. from a seasonal/long term scheduling model. Water values are not obtainable in this case study. The model uses independent water values, meaning that the value of the water is constant during the optimization period, and not depended on the production during the period. The water values for a reservoir will be constant and optimal when production is utilized optimally [23].

C.2.7 All markets are assumed perfect

The assumption that all participants in the different power markets are price takers was made. That means among other things [26] that no participant regardless of production capacity or bidding volume can influence the market price, because there are too many participants both demanding and supplying power. This however, is not completely true when it comes to hydropower producers as some of them have a huge market share compared to the market itself. But then again, the forecasted price would not be valid, and the model would depend on which producer using it.

Nevertheless, this situation is not that present in the Nordic countries [27].

D The compact mathematical model formulations

A note about the deterministic model presented in appendix D.2;

The first model developed taking obligation regulations into consideration was a deterministic extension of the model developed in the specialization project, developed prior to the stochastic model.

Instead of finding an optimal regulation capacity given a regulation price, this extended model finds a needed break-even regulation price given different regulation obligations. This model does an optimization iteration over O obligation regulations only having equation (XXXIV)-(XXXVI) inside a obligation loop. All other equations ((XXXVII)-(LV)) must be valid for all optimization iterations. The results of each iteration was saved into output files for further use. It was merely a proof of concept that comparing O regulation obligations proved useful for further development of the model, taking stochastic properties into account.

Even though this model is not used to produce any results in this report, it was a necessary step towards the final stochastic model, and should be presented as well.

D.1 The stochastic multi-stage optimization model

$$\max \sum_{e \in \tilde{E}^t} \sum_{h=24t-23}^{H} \frac{\Pi^e P_{spot}^{e,h} x_{tot}^{t,o,h}}{\sum_{e \in \tilde{E}^t} \Pi^e} + \sum_{r \in R} W^r \alpha^{t,o,r} - \sum_{s \in S} \sum_{u \in U} \sum_{h=24t-23}^{H} C^{s,u} \beta^{t,o,s,u,h} \qquad \forall t \in T, \forall o \in O \qquad \text{(III)}$$

s.t.

$$\gamma^{t,o,r,24t} = \gamma^{t-1,o,r,24t} \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall r \in R \qquad \text{(IV)}$$

$$f^{t,o,r,24t} = f^{t-1,o,r,24t} \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall r \in R \qquad \text{(V)}$$

$$b^{t,o,r,24t} = b^{t-1,o,r,24t} \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall r \in R \qquad \text{(VI)}$$

$$q^{t,o,s,u,24t} = q^{t-1,o,s,u,24t} \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall s \in S, \forall u \in U \qquad \text{(VII)}$$

$$x^{t,o,s,u,24t} = x^{t-1,o,s,u,24t} \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall s \in S, \forall u \in U \qquad \text{(VIII)}$$

$$\beta^{t,o,s,u,24t} = \beta^{t-1,o,s,u,24t} \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall s \in S, \forall u \in U \qquad \text{(IX)}$$

$$\zeta^{t,o,s,u,24t} = \zeta^{t,o,s,u,24t} \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall s \in S, \forall u \in U \qquad \text{(X)}$$

$$x_{tot}^{t,o,h} + Y_{sym}^{o,h} - \sum_{s \in S} \sum_{u \in U} X_{\max}^{s,u} \zeta^{t,o,s,u,h} \leq 0 \qquad \forall t \in T, \forall o \in O, h = 24t - 23..H \qquad \text{(XI)}$$

$$x_{tot}^{t,o,h} - Y_{sym}^{o,h} - \sum_{s \in S} \sum_{u \in U} X_{\min}^{s,u} \zeta^{t,o,s,u,h} \qquad \forall t \in T, \forall o \in O, h = 24t - 23..H \qquad \text{(XII)}$$

$$\gamma^{t,o,r,h} + N^{r,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} - \gamma^{t,o,r,(h+1)} = 0 \qquad t = 1, \forall o \in O, r = 1, s = 1, \forall u \in U, h = 24t - 23..H - 1 \qquad \text{(XIII)}$$

$$\gamma^{t,o,r,h} + N^{r,h} + \sum_{s \in S} \sum_{u \in U} q^{t,o,(s-1),u,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} + b^{t,o,(r-1),h} - \gamma^{t,o,r,(h+1)} = 0 \qquad t = 1, \forall o \in O, \forall r \in R \neq r = 1, \forall s \in S \neq s = 1, \forall u \in U, h = 24t - 23..H - 1 \qquad \text{(XIV)}$$

$$\gamma^{t,o,r,h} + N^{r,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} - \gamma^{t,o,r,(h+1)} = 0 \qquad \forall t \in T \neq t = 1, \forall o \in O, r = 1, s = 1, \forall u \in U, h = 24(t-1)..H - 1 \qquad \text{(XV)}$$

$$\gamma^{t,o,r,h} + N^{r,h} + \sum_{s \in S} \sum_{u \in U} q^{t,o,(s-1),u,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} + b^{t,o,(r-1),h} - \gamma^{t,o,r,(h+1)} = 0 \qquad \forall t \in T \neq t = 1, \forall o \in O, \forall r \in R \neq r = 1, \forall s \in S \neq s = 1, \forall u \in U, h = 24(t-1)..H - 1 \qquad \text{(XVI)}$$

$$\gamma^{t,o,r,h} + N^{r,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} - \theta^{t,o,r} = 0 \qquad \forall t \in T, \forall o \in O, r = 1, s = 1, \forall u \in U, h = H \qquad \text{(XVII)}$$

$$\gamma^{t,o,r,h} + N^{r,h} + \sum_{s \in S} \sum_{u \in U} q^{t,o,(s-1),u,h} - \sum_{s \in S} \sum_{u \in U} q^{t,o,s,u,h} - f^{t,o,r,h} - b^{t,o,r,h} + b^{t,o,(r-1),h} - \theta^{t,o,r} = 0 \qquad \forall t \in T, \forall o \in O, \forall r \in R \neq r = 1, \forall s \in S \neq s = 1, \forall u \in U, h = H \qquad \text{(XVIII)}$$

$$\alpha^{t,o,r} = \frac{\theta^{t,o,r} H_{tot}^{s,u} \rho g \eta_{\max}^{s,u}}{3.6 \cdot 10^{9}} \qquad \forall t \in T, \forall o \in O, \forall r \in R, \forall s \in S, \forall u \in U \qquad \text{(XIX)}$$

$$x^{t,o,s,u,h} \leq \left(\Gamma^{i} \frac{q^{t,o,s,u,h}}{Q_{\max}^{s,u}} + \Omega^{i} \right) Q_{\max}^{s,u} H^{s,u} \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H, \forall i \in I \qquad \text{(XX)}$$

$$B_{\min}^{r} \leq b^{t,o,r,h} \leq B_{\max}^{r} \qquad \forall t \in T, \forall o \in O, \forall r \in R, h = 24t-23..H \qquad \text{(XXI)}$$

$$L_{\min}^{r} \leq \gamma^{t,o,r,h} \leq L_{\max}^{r} \qquad \forall t \in T, \forall o \in O, \forall r \in R, h = 24t-23..H \qquad \text{(XXII)}$$

$$L_{\min}^{r} \leq \theta^{t,o,r,} \leq L_{\max}^{r} \qquad \forall t \in T, \forall o \in O, \forall r \in R \qquad \text{(XXIII)}$$

$$Q_{\min}^{s,u} \leq q^{t,o,s,u,h} \leq Q_{\max}^{s,u} \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \qquad \text{(XXIV)}$$

$$\left| x^{t,o,s,u,h} - x^{t,o,s,u,(h-1)} \right| \leq G_{\max}^{s,u} \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H-1 \qquad \text{(XXV)}$$

$$\left| \gamma^{t,o,r,h} - \gamma^{t,o,r,(h+1)} \right| \leq A_{\max}^{r} \qquad \forall t \in T, \forall o \in O, \forall r \in R, h = 24t-23..H-1 \qquad \text{(XXVI)}$$

$$x^{t,o,s,u,h} - X_{\max}^{t,o,s,u} \zeta^{t,o,s,u,h} \leq 0 \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \qquad \text{(XXVII)}$$

$$x^{t,o,s,u,h} - X_{\min}^{s,u} \zeta^{t,o,s,u,h} \geq 0 \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \qquad \text{(XXVIII)}$$

$$\beta^{t,o,s,u,h} - \left(\zeta^{t,o,s,u,h} - \zeta^{t,o,s,u,(h-1)} \right) \geq 0 \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-22..H \qquad \text{(XXIX)}$$

$$\beta^{t,o,s,u,h} - \left(\zeta^{t,o,s,u,(h+1)} - \zeta^{t,o,s,u,h} \right) \geq 0 \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23 \qquad \text{(XXX)}$$

$$\sum_{s \in S} \sum_{u \in U} x^{t,o,s,u,h} - x_{tot}^{t,o,h} = 0 \qquad \forall t \in T, \forall o \in O, h = 24t-23..H \qquad \text{(XXXI)}$$

$$\zeta^{t,o,s,u,h} \in \{0,1\} \qquad \forall t \in T, \forall o \in O, \forall s \in S, \forall u \in U, h = 24t-23..H \qquad \text{(XXXII)}$$

$$x^{t,o,s,u,h}, x_{tot}^{t,o,h}, \alpha^{t,r}, \beta^{t,o,s,u,h}, \gamma^{t,o,r,h}, q^{t,o,s,u,h}, f^{t,o,r,h}, b^{t,o,r,h} \geq 0 \qquad \forall t \in T, \forall o \in O, \forall r \in R, \forall s \in S, \forall u \in U, h = 24t-23..H \qquad \text{(XXXIII)}$$

D.2 The deterministic model

$$\max \sum_{h\in H} P_{spot}^{h} x_{tot}^{h} + \sum_{r\in R} W^{r}\alpha^{r} - \sum_{s\in S}\sum_{u\in U}\sum_{h\in H} C^{s,u}\beta^{s,u,h} \qquad \forall o\in O \qquad \text{(XXXIV)}$$

s.t.

$$x_{tot}^{t,h} + Y_{sym}^{o,h} - \sum_{s\in S}\sum_{u\in U} X_{\max}^{s,u}\zeta^{t,s,u,h} \le 0 \qquad \forall o\in O, \forall s\in S, \forall u\in U, \forall h\in H \qquad \text{(XXXV)}$$

$$x_{tot}^{t,h} - Y_{sym}^{o,h} - \sum_{s\in S}\sum_{u\in U} X_{\min}^{s,u}\zeta^{t,s,u,h} \ge 0 \qquad \forall o\in O, \forall s\in S, \forall u\in U, \forall h\in H \qquad \text{(XXXVI)}$$

$$\gamma^{r,h} + N^{r,h} - \sum_{s\in S}\sum_{u\in U} q^{s,u,h} - f^{r,h} - b^{r,h} - \gamma^{r,(h+1)} = 0 \qquad r=1,\, s=1, \forall u\in U, \forall h\in H \ne h=H \qquad \text{(XXXVII)}$$

$$\gamma^{r,h} + N^{r,h} + \sum_{s\in S}\sum_{u\in U} q^{(s-1),u,h} - \sum_{s\in S}\sum_{u\in U} q^{s,u,h} - f^{r,h} - b^{r,h} + b^{(r-1),h} - \gamma^{r,(h+1)} = 0 \qquad \forall r\in R \ne r=1, \forall s\in S \ne s=1, \forall u\in U, \forall h\in H \ne h=H \qquad \text{(XXXVIII)}$$

$$\gamma^{r,h} + N^{r,h} - \sum_{s\in S}\sum_{u\in U} q^{s,u,h} - f^{r,h} - b^{r,h} - \theta^{r} = 0 \qquad r=1,\, s=1, \forall u\in U, h=H \qquad \text{(XXXIX)}$$

$$\gamma^{r,h} + N^{r,h} + \sum_{s\in S}\sum_{u\in U} q^{(s-1),u,h} - \sum_{s\in S}\sum_{u\in U} q^{s,u,h} - f^{r,h} - b^{r,h} + b^{(r-1),h} - \theta^{r} = 0 \qquad \forall r\in R \ne r=1, \forall s\in S \ne s=1, \forall u\in U, h=H \qquad \text{(XL)}$$

$$\alpha^{r} = \frac{\theta^{r} H_{tot}^{s,u}\rho g \eta_{\max}^{s,u}}{3.6\cdot 10^{9}} \qquad \forall r\in R, \forall s\in S, \forall u\in U \qquad \text{(XLI)}$$

$$x^{s,u,h} \le \left(\Gamma^{i}\frac{q^{s,u,h}}{Q_{\max}^{s,u}} + \Omega^{i}\right) Q_{\max}^{s,u} H^{s,u} \qquad \forall s\in S, \forall u\in U, \forall h\in H, \forall i\in I \qquad \text{(XLII)}$$

$$B_{\min}^{r} \le b^{r,h} \le B_{\max}^{r} \qquad \forall r\in R, \forall h\in H \qquad \text{(XLIII)}$$

$$L_{\min}^{r} \le \gamma^{r,h} \le L_{\max}^{r} \qquad \forall r\in R, \forall h\in H \qquad \text{(XLIV)}$$

$$L_{\min}^{r} \le \theta^{r,} \le L_{\max}^{r} \qquad \forall r\in R \qquad \text{(XLV)}$$

$$Q_{\min}^{s,u} \le q^{s,u,h} \le Q_{\max}^{s,u} \qquad \forall s\in S, \forall u\in U, \forall h\in H \qquad \text{(XLVI)}$$

$$\left|x^{s,u,h} - x^{s,u,(h-1)}\right| \le G_{\max}^{s,u} \qquad \forall s\in S, \forall u\in U, \forall h\in H \ne h=H \qquad \text{(XLVII)}$$

$$\left|\gamma^{r,h} - \gamma^{r,(h+1)}\right| \le A_{\max}^{r} \qquad \forall r\in R, \forall h\in H \ne h=H \qquad \text{(XLVIII)}$$

$$x^{s,u,h} - X_{\max}^{s,u}\zeta^{s,u,h} \le 0 \qquad \forall s\in S, \forall u\in U, \forall h\in H \qquad \text{(XLIX)}$$

$$x^{s,u,h} - X_{\min}^{s,u}\zeta^{s,u,h} \ge 0 \qquad \forall s\in S, \forall u\in U, \forall h\in H \qquad \text{(L)}$$

$$\beta^{s,u,h} - \left(\zeta^{s,u,h} - \zeta^{s,u,(h-1)}\right) \ge 0 \qquad \forall s\in S, \forall u\in U, \forall h\in H \ne h=1 \qquad \text{(LI)}$$

$$\beta^{s,u,h} - \left(\zeta^{s,u,(h+1)} - \zeta^{s,u,h}\right) \ge 0 \qquad \forall s\in S, \forall u\in U, h=1 \qquad \text{(LII)}$$

$$\sum_{s\in S}\sum_{u\in U} x^{s,u,h} - x_{tot}^{h} = 0 \qquad \forall h\in H \qquad \text{(LIII)}$$

$$\zeta^{s,u,h} \in \{0,1\} \qquad \forall s\in S, \forall u\in U, \forall h\in H \qquad \text{(LIV)}$$

$$x^{s,u,h}, x_{tot}^{h}, \alpha^{r}, \beta^{s,u,h}, \gamma^{r,h}, q^{s,u,h}, f^{r,h}, b^{r,h} \ge 0 \qquad \forall r\in R, \forall s\in S, \forall u\in U, \forall h\in H \qquad \text{(LV)}$$

E The implemented model in Mosel

E.1 The stochastic multi-stage optimization model

```
model stochastic !Model name
options explterm !All statements must be terminated by a semicolon
options noimplicit !No implicit declarations
uses "mmxprs"; !Gain access to the Xpress-Optimizer library
uses "mmodbc"; !Gain access to SQL for Excel input
uses "mmsystem"; !Gain access to datetime

!Declaration of tic/toc
declarations
    ModelStartTime: datetime;
    ModelEndTime: datetime;
    OptimizationStartTime: datetime;
    OptimizationEndTime: datetime;
end-declarations

ModelStartTime:= datetime(SYS_NOW);

!Declaration of sets
declarations
    Hours:          set of integer;
    Units:          set of integer;
    Stations:       set of integer;
    Reservoirs:     set of integer;
    Index:          set of integer;
    ObligationIndex:set of integer;
    Scenarios:      set of integer;
    Steps:          set of integer;
    Nodes:          set of integer;
end-declarations

!Reading input data of sets from excel
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\input_216.xlsx');
    SQLexecute("select HOURS from HH", Hours);
    SQLexecute("select UNITS from UU", Units);
    SQLexecute("select STATIONS from SS", Stations);
    SQLexecute("select RESERVOIRS from RR", Reservoirs);
    SQLexecute("select INDEX from II", Index);
    SQLexecute("select OBLIGATIONINDEX from OO", ObligationIndex);
SQLdisconnect;

SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\stoch_input_216.xlsx');
    SQLexecute("select SCENARIOS from EE", Scenarios);
    SQLexecute("select STEPS from TT", Steps);
    SQLexecute("select NODES from NN", Nodes);
SQLdisconnect;

!Declaration of general parameters
declarations
    !Spotprice
    SpotPrice:                  array(Hours)                of real;
    !Symmetrical regulation obligation
    SymRegObligation:           array(ObligationIndex)      of real;
    !Water value
    WaterValue:                 array(Reservoirs)           of real;
    !Intial reservoir volume
    InitialWaterVolume:         array(Reservoirs)           of real;
    !Inflow
    InflowRes:                  array(Reservoirs,Hours)     of real;
    !Max production on each unit
    MaxProduction:              array(Stations,Units)       of real;
    !Min production on each unit
    MinProduction:              array(Stations,Units)       of real;
    !Max bypass m3/s
    MaxBypass:                  array(Reservoirs)           of real;
    !Min bypass m3/s
    MinBypass:                  array(Reservoirs)           of real;
    !Max reservoar volume m3
    MaxReservoirVolume:         array(Reservoirs)           of real;
    !Min reservoar volume m3
    MinReservoirVolume:         array(Reservoirs)           of real;
    !Max regulation of reservoar per hour m3/h
    MaxRegulatingPerHour:       array(Reservoirs)           of real;
    !Start-up cost at unit u at station s
    StartUpCost:                array(Stations,Units)       of real;
    !Max ramping at unit u at station s
    MaxRamping:                 array(Stations,Units)       of real;
    !Slopes of the piecewise linear PQ-curve
    Slope:                      array(Index)                of real;
    !Interceptions of the piecewise linear PQ-curve
    Intercept:                  array(Index)                of real;
    !Head for each unit
    HeadUnit:                   array(Stations,Units)       of real;
    !System head for each unit
    TotHeadUnit:                array(Stations,Units)       of real;
    !Max discharge through unit u at station s
    MaxDiscUnit:                array(Stations,Units)       of real;
    !Min discharge through unit u at station s
    MinDiscUnit:                array(Stations,Units)       of real;
    !Best point efficiency for each unit
    EffBPUnit:                  array(Stations,Units)       of real;

    !Declaration of parameters used by the stochastic part of the model

    !The spot prices for all possible Scenarios
    ScenarioSpotPrice:          array(Scenarios,Hours)  of real;
    !The distribution of probabilities for each Scenario to occur
    EndScenarioProbability:     array(Scenarios)        of real;
    !The predecessor node given each node
    NodePredecessor:            array(Nodes)            of integer;
    !The probability for each node in the scenario tree
    NodeProbability:            array(Nodes)            of real;
    !The number of branches in each step in the scenario tree
    StepNumberOfBranch:         array(Steps)            of integer;
    !The matrices created by the model in order to use Scenred data
    StepProbabilityDynamic:             array(Scenarios,Steps)  of real;
    StepPredecessorDynamic:     array(Scenarios,Steps)  of integer;
    StepPredecessorFull:        array(Scenarios,Steps)  of real;
    !Counters needed in order to create the above matrices
    EqualNumbersCounter:        integer;
    IteratorCounter:            integer;
    Counter:                    integer;
    CoordCounter:               integer;
    Indicator:                  integer;
    !Storing the random function deciding the stochastic outcomes
    Dice:                       array(Steps) of real;
    DiceAdjusted:               array(Steps) of real;
    !Declaration of parameters used to store stochastic results
    RealizedPredecessor:        array(Steps) of real;
    RealizedPredDynCoord:       array(Steps) of integer;
    RealizedProbDynCoord:       array(Steps) of integer;
    RealizedProbability:        array(Steps) of real;

    !Declaration of parameters used to store variables of stepwise runs
    StepwisePTotSpot:               array(ObligationIndex,Steps,Hours)                  of real;
    StepwiseQInRes:                 array(ObligationIndex,Steps,Reservoirs,Hours)       of real;
    StepwiseRunIndUnit:             array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwisePUnit:                  array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwiseQBypassRes:             array(ObligationIndex,Steps,Reservoirs,Hours)       of real;
    StepwiseQFloodRes:              array(ObligationIndex,Steps,Reservoirs,Hours)       of real;
    StepwiseQProdUnit:              array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwiseStartIndUnit:           array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwisePEndRes:                array(ObligationIndex,Steps,Reservoirs)             of real;
    StepwiseObjectiveValue:         array(ObligationIndex,Steps)                        of real;
    StepwiseQEndRes:                array(ObligationIndex,Steps,Reservoirs)             of real;
    StepwiseBreakEvenRegPrice:      array(ObligationIndex,Steps)                        of real;
    StepwiseTurbEff:                array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwiseModelTotObjVal:         array(ObligationIndex,Steps)                        of real;
    StepwiseModelBreakEvenRegPrice: array(ObligationIndex,Steps)                        of real;
    StepwiseActualTotObjVal:        array(ObligationIndex,Steps)                        of real;
    StepwiseActualBreakEvenRegPrice:array(ObligationIndex,Steps)                        of real;
    StepwiseModelMarginalRegCost:   array(ObligationIndex,Steps)                        of real;
    StepwiseActualMarginalRegCost:  array(ObligationIndex,Steps)                        of real;
    StepwiseModelPrice:             array(Steps,Hours)                                  of real;


    !Declaration of parameters used to store variables of realized runs
    ActualPrice:                array(Hours) of real;
    ActualObjectiveValue:       array(ObligationIndex)                          of real;
    ActualBreakEvenRegPrice:    array(ObligationIndex)                          of real;
    ActualMarginalRegCost:      array(ObligationIndex)                          of real;
    RealizedPTotSpot:           array(ObligationIndex,Hours)                    of real;
    RealizedQInRes:             array(ObligationIndex,Reservoirs,Hours)         of real;
    RealizedRunIndUnit:         array(ObligationIndex,Stations,Units,Hours)     of real;
    RealizedPUnit:              array(ObligationIndex,Stations,Units,Hours)     of real;
    RealizedQBypassRes:         array(ObligationIndex,Reservoirs,Hours)         of real;
    RealizedQFloodRes:          array(ObligationIndex,Reservoirs,Hours)         of real;
    RealizedQProdUnit:          array(ObligationIndex,Stations,Units,Hours)     of real;
    RealizedStartIndUnit:       array(ObligationIndex,Stations,Units,Hours)     of real;
    RealizedPEndRes:            array(ObligationIndex,Reservoirs)               of real;
    RealizedQEndRes:            array(ObligationIndex,Reservoirs)               of real;
    RealizedTurbEff:            array(ObligationIndex,Stations,Units,Hours)     of real;
    RealizedRunningUnits:       array(ObligationIndex,Stations,Hours)           of real;

    !Declaration of parameters used to store the price and marginal cost for regulation
    !(and correspondig objective value) that the model sees when optimizing
    ModelPrice:                 array(Hours)            of real;
    ModelObjectiveValue:        array(ObligationIndex)  of real;
    ModelBreakEvenRegPrice:     array(ObligationIndex)  of real;
    ModelMarginalRegCost:       array(ObligationIndex)  of real;

end-declarations

!Reading input data of parameters from excel
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\input_216.xlsx');
    SQLexecute("select SPOTPRICE from HH", SpotPrice);
    SQLexecute("select SYMREGOBLIGATION from OO", SymRegObligation);
    SQLexecute("select WATERVALUE from RR", WaterValue);
    SQLexecute("select INITIALWATERVOLUME from RR", InitialWaterVolume);
    SQLexecute("select INFLOWRES from RRHH", InflowRes);
    SQLexecute("select MAXPRODUCTION from SSUU", MaxProduction);
    SQLexecute("select MINPRODUCTION from SSUU", MinProduction);
    SQLexecute("select MAXBYPASS from RR", MaxBypass);
    SQLexecute("select MINBYPASS from RR", MinBypass);
    SQLexecute("select MAXRESERVOIRVOLUME from RR", MaxReservoirVolume);
    SQLexecute("select MINRESERVOIRVOLUME from RR", MinReservoirVolume);
    SQLexecute("select MAXREGULATINGPERHOUR from RR", MaxRegulatingPerHour);
    SQLexecute("select STARTUPCOST from SSUU", StartUpCost);
    SQLexecute("select MAXRAMPING from SSUU", MaxRamping);
    SQLexecute("select HEADUNIT from SSUU", HeadUnit);
    SQLexecute("select MAXDISCUNIT from SSUU", MaxDiscUnit);
    SQLexecute("select MINDISCUNIT from SSUU", MinDiscUnit);
    SQLexecute("select TOTHEADUNIT from SSUU", TotHeadUnit);
    SQLexecute("select EFFBPUNIT from SSUU", EffBPUnit);
    SQLexecute("select SLOPE from II", Slope);
    SQLexecute("select INTERCEPT from II", Intercept);
    SQLdisconnect;

    SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\stoch_input_216.xlsx');
    SQLexecute("select SCENARIOSPOTPRICE from EEHH", ScenarioSpotPrice);
    SQLexecute("select ENDSCENARIOPROBABILITY from EE", EndScenarioProbability);
    SQLexecute("select NODEPREDECESSOR from NN", NodePredecessor);
    SQLexecute("select NODEPROBABILITY from NN", NodeProbability);
    SQLexecute("select STEPNUMBEROFBRANCH from TT", StepNumberOfBranch);
SQLdisconnect;

writeln("All input data successfully read from Excel file");

!Declaration of variables
declarations

    !Total spot volume in hour h
    ptotspot:       array(ObligationIndex,Steps,Hours)                of mpvar;
    !Reservoir volume at the end of the period
    qendres:        array(ObligationIndex,Steps,Reservoirs)           of mpvar;
    !Energy in the reservoir at the end of the period
    pendres:        array(ObligationIndex,Steps,Reservoirs)           of mpvar;
    !Discharge through unit u at station s in hour h
    qprodunit:      array(ObligationIndex,Steps,Stations,Units,Hours) of mpvar;
    !Flood from reservoir r in hour h
    qfloodres:      array(ObligationIndex,Steps,Reservoirs,Hours)     of mpvar;
    !Bypass from reservoir r in hour h
    qbypassres:     array(ObligationIndex,Steps,Reservoirs,Hours)     of mpvar;
    !reservoar volume in reservoir r in hour h
    qinres:         array(ObligationIndex,Steps,Reservoirs,Hours)     of mpvar;
    !Spot production from unit u at station s in hour h
    punit:          array(ObligationIndex,Steps,Stations,Units,Hours) of mpvar;
    !Start indicator, 1 if unit u starts in hour h
    startindunit:   array(ObligationIndex,Steps,Stations,Units,Hours) of mpvar;
    !Run indicator, 1 if unit u runs in hour h
    runindunit:     array(ObligationIndex,Steps,Stations,Units,Hours) of mpvar;


! Declaration of the objective function
maxz:                   linctr;

!Declaration of constranits

! Sets initial reservoir volume in qinres for all reservoirs
restInitWaterBalance:       array(Reservoirs)               of linctr;
!Other water balances
restWaterBalance1:          array(Reservoirs,Hours)         of linctr;
restWaterBalance2:          array(Reservoirs,Hours)         of linctr;
restWaterBalance3:          array(Reservoirs,Hours)         of linctr;
restWaterBalance4:          array(Reservoirs,Hours)         of linctr;
restWaterBalance5:          array(Reservoirs,Hours)         of linctr;
restWaterBalance6:          array(Reservoirs,Hours)         of linctr;
restWaterBalance7:          array(Reservoirs,Hours)         of linctr;
restWaterBalance8:          array(Reservoirs,Hours)         of linctr;
!Total spot production from unit to system level
restTotSpotProd:            array(Hours)                    of linctr;
!Max bypass
restMaxBypassRes:           array(Reservoirs,Hours)         of linctr;
!Min bypass
restMinBypassRes:           array(Reservoirs,Hours)         of linctr;
!Max reservoar volume
restMaxVolumeRes:           array(Reservoirs,Hours)         of linctr;
!Min reservoar volume
restMinVolumeRes:           array(Reservoirs,Hours)         of linctr;
!Max reservoar regulation  equation 1
restMaxRegRes1:             array(Reservoirs,Hours)         of linctr;
!Max reservoar regulation equation 2
restMaxRegRes2:             array(Reservoirs,Hours)         of linctr;
!Start-up cost
restStartupCostUnit:        array(Stations,Units,Hours)     of linctr;
!Max ramping equation 1
restMaxRampUnit1:           array(Stations,Units)           of linctr;
!Max ramping equation 2
restMaxRampUnit2:           array(Stations,Units)           of linctr;
!Max production on unit u at station s in hour h
restMaxProdUnit:            array(Stations,Units,Hours)     of linctr;
!Min production on unit u at station s in hour h
```

```
restMinProdUnit:            array(Stations,Units,Hours)     of linctr;
!Production indicator for each unit in each hour
restProdIndUnit:            array(Stations,Units,Hours)     of linctr;
!Max reservoir volume at the end of the period
restMaxVolumeEndRes:        array(Reservoirs)               of linctr;
!Min reservoir volume at the end of the period
restMinVolumeEndRes:        array(Reservoirs)               of linctr;
!Calculates the equivalent energy of the water in the reservoar at the end of the
!period in MWh
restEndResEnergy:           array(Reservoirs)               of linctr;
!Max discharge thruogh unit u at station s in hour h
restMaxDiscUnit:            array(Stations,Units,Hours)     of linctr;
!Min discharge thruogh unit u at station s in hour h
restMinDiscUnit:            array(Stations,Units,Hours)     of linctr;
!PQ-curve
restPQCurve1:               array(Stations, Units, Hours)   of linctr;
!PQ-curve
restPQCurve2:               array(Stations, Units, Hours)   of linctr;
!PQ-curve
restPQCurve3:               array(Stations, Units, Hours)   of linctr;
!PQ-curve
restPQCurve4:               array(Stations, Units, Hours)   of linctr;
!PQ-curve
restPQCurve5:               array(Stations, Units, Hours)   of linctr;
!PQ-curve
restPQCurve6:               array(Stations, Units, Hours)   of linctr;
!PQ-curve
restPQCurve7:               array(Stations, Units, Hours)   of linctr;
!Reserving upward regulation capacity
restMaxProdObligation:      array(Hours)                    of linctr;
!Reserving downward regulation capacity
restMinProdObligation:      array(Hours)                    of linctr;
!Connecting optimal decision variables from step to step
restStepChangeQInRes:       array(Reservoirs)               of linctr;
restStepChangeQProdUnit:    array (Stations, Units)         of linctr;
restStepChangeQFloodRes:    array(Reservoirs)               of linctr;
restStepChangeQBypassRes:   array(Reservoirs)               of linctr;
restStepChangePUnit:        array (Stations, Units)         of linctr;
restStepChangeRunIndUnit:   array (Stations, Units)         of linctr;
restStepChangeStartIndUnit: array (Stations, Units)         of linctr;

end-declarations

! Creates binary variable
forall (oo in ObligationIndex, tt in Steps,ss in Stations,uu in Units,hh in Hours) do
    runindunit(oo,tt,ss,uu,hh) is_binary;
end-do

forall (oo in ObligationIndex, tt in Steps,ss in Stations,uu in Units,hh in Hours) do
    startindunit(oo,tt,ss,uu,hh) is_binary;
end-do

writeln("All restrictions and variables successfully created");
writeln("Creating stochastic matrices needed by the optimization model");

! Creation of needed matrices for the stochastic calculations

! Creating the dynamic matrices from excel input data
forall (tt in Steps) do

    if (tt =1) then

        forall (ee in 1.. StepNumberOfBranch(tt)) do

            StepProbabilityDynamic(ee,tt):= NodeProbability(1+ee);
            StepPredecessorDynamic(ee,tt):= NodePredecessor(1+ee);

        end-do

    else

        forall (ee in 1.. StepNumberOfBranch(tt)) do

            StepProbabilityDynamic(ee,tt) := NodeProbability(1+sum(ttt in 2.. tt)(24*StepNumberOfBranch(ttt-
1))+ee);
            StepPredecessorDynamic(ee,tt) := NodePredecessor(1+sum(ttt in 2.. tt)(24*StepNumberOfBranch(ttt-
1))+ee);

        end-do

    end-if

end-do

    !Creating the explicit matrix from the dynamic matrix

    !Creating last column of the explicit matrix
    forall (ee in Scenarios) do

        StepPredecessorFull(ee,getsize(Steps)):=StepPredecessorDynamic(ee,getsize(Steps));

    end-do


    ! Creating the remaining columns of the explicit matrix
    forall (tt in 1.. getsize(Steps)-1) do

        EqualNumbersCounter := 0;
        IteratorCounter:= 1;
        Counter:= 1;

        forall (ee in 2.. getsize(Scenarios)) do

            if (StepPredecessorFull(ee-1,1+getsize(Steps)-tt) = StepPredecessorFull(ee,1+getsize(Steps)-tt))
then

                EqualNumbersCounter:= EqualNumbersCounter + 1;

            else

                forall(pp in IteratorCounter.. IteratorCounter+EqualNumbersCounter) do

                        StepPredecessorFull(pp,(getsize(Steps)-tt)) :=
StepPredecessorDynamic(Counter,getsize(Steps)-tt);
                end-do

                Counter:= Counter + 1;
                IteratorCounter := IteratorCounter + EqualNumbersCounter +1;
                EqualNumbersCounter :=0;

            end-if

        end-do

    end-do

        forall (tt in 1.. getsize(Steps)-1) do

            StepPredecessorFull(getsize(Scenarios),getsize(Steps)-tt):=
            StepPredecessorDynamic(StepNumberOfBranch(getsize(Steps)-tt),getsize(Steps)-tt);

        end-do

    writeln("Model run start");

    ! Model algoritm begins

forall (tt in Steps) do ! Iterating over all steps

        if (tt =1) then ! For the first step only

        !Establishing initial values

            RealizedProbability(tt):= 1;
            RealizedPredecessor(tt):=1;
            RealizedPredDynCoord(tt):=1;

            !Optimizing the entire period with all scenarios weighted according to corresponding probability



            !Saving the price optimized upon for the first 24 hours
            forall (hh in 1..24) do
                ModelPrice(hh) :=sum(ee in Scenarios)(ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee));
            end-do

        else    ! for all other steps

                ! Emulating what actually happend from the previous step

            Dice(tt-1):= random;    ! Emulating which way the market will go
            DiceAdjusted(tt-1):= Dice(tt-1)*RealizedProbability(tt-1);  ! Adjusting it to the same size as
corresponding


            ! Storing the coordinate to the probability that was realized in the previous step
            Indicator:=0;

            forall(ee in Scenarios | StepPredecessorDynamic(ee,tt-1) = RealizedPredecessor(tt-1)) do

                if (DiceAdjusted(tt-1) < sum(pp in RealizedPredDynCoord(tt-
1)..ee)(StepProbabilityDynamic(pp,tt-1))) then

                    if (Indicator = 0) then

                    RealizedProbDynCoord(tt-1):=ee;
                    RealizedProbability(tt):= StepProbabilityDynamic(ee,tt-1);
                    Indicator:=1;

                    end-if

                end-if

            end-do

            ! Storing the predecessor for the probabilities that shall be weighted

            Counter:=1;
            CoordCounter:=1;
            while (Counter < RealizedProbDynCoord(tt-1)) do

                if (StepPredecessorDynamic(CoordCounter,tt) <>  StepPredecessorDynamic(CoordCounter+1,tt))
then

                    Counter:= Counter +1;

                end-if

                CoordCounter:= CoordCounter +1;

            end-do

            RealizedPredDynCoord(tt):= CoordCounter;
            RealizedPredecessor(tt):= StepPredecessorDynamic(CoordCounter,tt);

            ! Storing the actual realized spot price (by picking the first price series with the
corresponding predecessor)

            Counter:=1;

            while (StepPredecessorFull(Counter,tt) <> RealizedPredecessor(tt)) do

                Counter:= Counter + 1;

            end-do

            forall (hh in (24*tt-47) .. 24*(tt-1)) do

                ActualPrice(hh):= ScenarioSpotPrice(Counter,hh);

            end-do

            ! Maximizing over all periods remaining considering valid probabilities and price series

            !Storing the price optimized upon for each step
            forall (hh in 24*(tt-1)..24*tt) do
                ModelPrice(hh) :=sum(ee in Scenarios | StepPredecessorFull(ee,tt) = RealizedPredecessor(tt))
                ((ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee))
                /(sum(eee in Scenarios | StepPredecessorFull(eee,tt) =
RealizedPredecessor(tt))(EndScenarioProbability(eee))));
            end-do

        end-if

        forall(oo in ObligationIndex)do !Iterating over all obligations

        if (tt = 1) then
            !Setting initial volume of reservoirs
            forall (rr in Reservoirs) do
                restInitWaterBalance(rr) := qinres(oo,tt,rr,1) = InitialWaterVolume(rr);
            end-do

            !Water balance upper reservoir for first step
            forall (rr in 1..1, hh in (24*tt-23)..(24*getsize(Steps)-1)) do
                restWaterBalance1(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)
                -sum(ss in 1..1, uu in Units)(qprodunit(oo,tt,ss,uu,hh)*3600)-qfloodres(oo,tt,rr,hh)-
qbypassres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) =0;
            end-do

            !Water balance other reservoirs for first step
            forall (rr in 2.. getsize(Reservoirs), hh in 24*tt-23 .. (24*getsize(Steps)-1)) do
                restWaterBalance2(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)+ sum(uu in
Units)(qprodunit(oo,tt,(rr-1),uu,hh)*3600)
                -sum(uu in Units)(qprodunit(oo,tt,rr,uu,hh)*3600)-
qfloodres(oo,tt,rr,hh)+qbypassres(oo,tt,(rr-1),hh)-qbypassres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) =0;
            end-do

            !Max ramping equation 1
            forall (ss in Stations, uu in Units, hh in 24*tt-22 .. 24*getsize(Steps)) do
                restMaxRampUnit1(ss,uu) := punit(oo,tt,ss,uu,hh) - punit(oo,tt,ss,uu,(hh-1)) <=
MaxRamping(ss,uu);
            end-do

            !Max ramping equation 2
            forall (ss in Stations, uu in Units, hh in 24*tt-22 .. 24*getsize(Steps)) do
                restMaxRampUnit2(ss,uu) := punit(oo,tt,ss,uu,(hh-1)) - punit(oo,tt,ss,uu,hh) >= -
MaxRamping(ss,uu);
            end-do

            !Max reservoir regulation equation 1
            forall (rr in Reservoirs, hh in 24*tt-23 .. (24*getsize(Steps)-1)) do
                restMaxRegRes1(rr,hh) := qinres(oo,tt,rr,(hh+1))-qinres(oo,tt,rr,hh) <=
MaxRegulatingPerHour(rr);
            end-do

            !Max reservoir regulation equation 2
            forall (rr in Reservoirs, hh in 24*tt-23 .. (24*getsize(Steps)-1)) do
                restMaxRegRes2(rr,hh) := qinres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) <=
MaxRegulatingPerHour(rr);
            end-do

                !Start-up costs
            forall (ss in Stations, uu in Units,hh in 24*tt-22 .. 24*getsize(Steps)) do
```

```
                restStartupCostUnit(ss,uu,hh) := startindunit(oo,tt,ss,uu,hh) - (runindunit(oo,tt,ss,uu,hh) -
runindunit(oo,tt,ss,uu,(hh-1))) >= 0 ;
            end-do

            !Start-up costs
            forall (ss in Stations, uu in Units, hh = 24*tt-23) do
                restStartupCostUnit(ss,uu,hh) := startindunit(oo,tt,ss,uu,hh) -
(runindunit(oo,tt,ss,uu,(hh+1)) - runindunit(oo,tt,ss,uu,hh)) >= 0 ;
            end-do

            !Objective value for first step
            maxz:= sum(ee in Scenarios, hh in (24*tt-23) .. 24*getsize(Steps))
            (ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee)*ptotspot(oo,tt,hh))
            + sum(rr in Reservoirs)(WaterValue(rr)*pendres(oo,tt,rr))
            - sum(ss in Stations, uu in Units, hh in (24*tt-23) .. 24*getsize(Steps))
            (StartUpCost(ss,uu)*startindunit(oo,tt,ss,uu,hh));

            else

            ! Setting all variables in current step for hour 24 equal to hour 24 of previous step
            forall (rr in Reservoirs, ss in Stations, uu in Units) do
                restStepChangeQInRes(rr) := qinres(oo,tt,rr,(24*(tt-1))) = StepwiseQInRes(oo,tt-1,rr,(24*(tt-
1)));
                restStepChangeQProdUnit(ss,uu):= qprodunit(oo,tt,ss,uu,(24*(tt-1))) =
StepwiseQProdUnit(oo,tt-1,ss,uu,(24*(tt-1)));
                restStepChangeQFloodRes(rr) := qfloodres(oo,tt,rr,(24*(tt-1))) = StepwiseQFloodRes(oo,tt-
1,rr,(24*(tt-1)));
                restStepChangeQBypassRes(rr) := qbypassres(oo,tt,rr,(24*(tt-1))) = StepwiseQBypassRes(oo,tt-
1,rr,(24*(tt-1)));
                restStepChangePUnit(ss,uu):= punit(oo,tt,ss,uu,(24*(tt-1))) = StepwisePUnit(oo,tt-
1,ss,uu,(24*(tt-1)));
                restStepChangeRunIndUnit(ss,uu) := runindunit(oo,tt,ss,uu,(24*(tt-1))) =
StepwiseRunIndUnit(oo,tt-1,ss,uu,(24*(tt-1)));
                restStepChangeStartIndUnit(ss,uu) := startindunit(oo,tt,ss,uu,(24*(tt-1))) =
StepwiseStartIndUnit(oo,tt-1,ss,uu,(24*(tt-1)));
            end-do

            !Water balance upper reservoir for other steps
            forall (rr in 1..1, hh in (24*(tt-1))..(24*getsize(Steps)-1)) do
                restWaterBalance3(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)
                -sum(ss in 1..1, uu in Units)(qprodunit(oo,tt,ss,uu,hh)*3600)-qfloodres(oo,tt,rr,hh)-
qbypassres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) =0;
            end-do

            !Water balance other reservoirs for other steps
            forall (rr in 2.. getsize(Reservoirs), hh in (24*(tt-1)) .. (24*getsize(Steps)-1)) do
                restWaterBalance4(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)+ sum(uu in
Units)(qprodunit(oo,tt,(rr-1),uu,hh)*3600)
                -sum(uu in Units)(qprodunit(oo,tt,rr,uu,hh)*3600)-
qfloodres(oo,tt,rr,hh)+qbypassres(oo,tt,(rr-1),hh)-qbypassres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) =0;
            end-do

            !Max ramping equation 1
            forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
                restMaxRampUnit1(ss,uu) := punit(oo,tt,ss,uu,hh) - punit(oo,tt,ss,uu,(hh-1)) <=
MaxRamping(ss,uu);
            end-do

            !Max ramping equation 2
            forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
                restMaxRampUnit2(ss,uu) := punit(oo,tt,ss,uu,(hh-1)) - punit(oo,tt,ss,uu,hh) >= -
MaxRamping(ss,uu);
            end-do

            !Max reservoir regulation equation 1
            forall (rr in Reservoirs, hh in 24*(tt-1) .. (24*getsize(Steps)-1)) do
                restMaxRegRes1(rr,hh) := qinres(oo,tt,rr,(hh+1))-qinres(oo,tt,rr,hh) <=
MaxRegulatingPerHour(rr);
            end-do

            !Max reservoir regulation equation 2
            forall (rr in Reservoirs, hh in 24*(tt-1) .. (24*getsize(Steps)-1)) do
                restMaxRegRes2(rr,hh) := qinres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) <=
MaxRegulatingPerHour(rr);
            end-do

            !Start-up costs
            forall (ss in Stations, uu in Units,hh in 24*tt-23 .. 24*getsize(Steps)) do
                restStartupCostUnit(ss,uu,hh) := startindunit(oo,tt,ss,uu,hh) - (runindunit(oo,tt,ss,uu,hh) -
runindunit(oo,tt,ss,uu,(hh-1))) >= 0 ;
            end-do

            ! The maximization function for all steps except tt=1
            maxz:=
            sum(ee in Scenarios | StepPredecessorFull(ee,tt) = RealizedPredecessor(tt), hh in (24*tt-23) ..
24*getsize(Steps))
            ((ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee))
            /(sum(eee in Scenarios | StepPredecessorFull(eee,tt) =
RealizedPredecessor(tt))(EndScenarioProbability(eee)))
            *ptotspot(oo,tt,hh))
            + sum(rr in Reservoirs)(WaterValue(rr)*pendres(oo,tt,rr))
            - sum(ss in Stations, uu in Units, hh in (24*tt-23) .. 24*getsize(Steps))
            (StartUpCost(ss,uu)*startindunit(oo,tt,ss,uu,hh));

        end-if
        !Water balance last hour upper reservoir
        forall (rr = 1, hh = 24*getsize(Steps)) do
            restWaterBalance5(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)
            -sum(ss in 1..1, uu in Units)(qprodunit(oo,tt,ss,uu,hh)*3600)-qfloodres(oo,tt,rr,hh)-
qbypassres(oo,tt,rr,hh)-qendres(oo,tt,rr) =0;
        end-do

        !Water balance last hour other reservoirs
        forall (rr in 2.. getsize(Reservoirs), hh = 24*getsize(Steps)) do
            restWaterBalance6(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)+ sum(uu in
Units)(qprodunit(oo,tt,(rr-1),uu,hh)*3600)
            -sum(uu in Units)(qprodunit(oo,tt,rr,uu,hh)*3600)-qfloodres(oo,tt,rr,hh)+qbypassres(oo,tt,(rr-
1),hh)-qbypassres(oo,tt,rr,hh)-qendres(oo,tt,rr) =0;
        end-do

        !Calculates the equivalent energy of the water in the reservoar at the end of the
        !period in MWh
        forall(rr in Reservoirs) do
            restEndResEnergy(rr) := pendres(oo,tt,rr) =
qendres(oo,tt,rr)*TotHeadUnit(rr,1)*1000*9.81*EffBPUnit(rr,1)/(3600*1000000);
        end-do

        !Cuts of PQ-curves
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps) |MaxDiscUnit(ss,uu) > 0) do
            restPQCurve1(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(1)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(1))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve2(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(2)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(2))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve3(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(3)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(3))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve4(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(4)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(4))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve5(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(5)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(5))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve6(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(6)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(6))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
        end-do

        !Max bypass
        forall (rr in Reservoirs,  hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxBypassRes(rr,hh) := qbypassres(oo,tt,rr,hh)<= MaxBypass(rr);
        end-do

        !Min bypass
        forall (rr in Reservoirs, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMinBypassRes(rr,hh) := qbypassres(oo,tt,rr,hh)>= MinBypass(rr);
        end-do

        !Max resevoir volume
        forall (rr in Reservoirs, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxVolumeRes(rr,hh) := qinres(oo,tt,rr,hh) <= MaxReservoirVolume(rr);
        end-do

        !Min reservoir volume
        forall (rr in Reservoirs, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMinVolumeRes(rr,hh) := qinres(oo,tt,rr,hh) >= MinReservoirVolume(rr);
        end-do

        !Max resevoir volume at the end of the period
        forall (rr in Reservoirs) do
            restMaxVolumeEndRes(rr) := qendres(oo,tt,rr) <= MaxReservoirVolume(rr);
        end-do

        !Min resevoir volume at the end of the period
        forall (rr in Reservoirs) do
            restMinVolumeEndRes(rr) := qendres(oo,tt,rr) >= MinReservoirVolume(rr);
        end-do

        !Max Discharge
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxDiscUnit(ss,uu,hh) := qprodunit(oo,tt,ss,uu,hh) <= MaxDiscUnit(ss,uu);
        end-do

        !Min Discharge
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMinDiscUnit(ss,uu,hh) := qprodunit(oo,tt,ss,uu,hh) >= MinDiscUnit(ss,uu);
        end-do

        !Total production in the spot market from unit level to system level
        forall (hh in 24*tt-23 .. 24*getsize(Steps)) do
            restTotSpotProd(hh) := sum(ss in Stations, uu in Units)(punit(oo,tt,ss,uu,hh)) -
ptotspot(oo,tt,hh) = 0 ;
        end-do


        !Max production on each unit
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxProdUnit(ss,uu,hh) := punit(oo,tt,ss,uu,hh)-
MaxProduction(ss,uu)*runindunit(oo,tt,ss,uu,hh) <= 0;
        end-do

        !Min production on each unit
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
```

```
            restMinProdUnit(ss,uu,hh) := punit(oo,tt,ss,uu,hh)-
MinProduction(ss,uu)*runindunit(oo,tt,ss,uu,hh) >= 0;
        end-do

        !Reserving upward regulation capacity
        forall(hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxProdObligation(hh) := ptotspot(oo,tt,hh)+SymRegObligation(oo)-sum(ss in Stations, uu in
Units)MaxProduction(ss,uu)*runindunit(oo,tt,ss,uu,hh) <= 0;
        end-do

        !Reserving downward regulation capacity
        forall(hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMinProdObligation(hh) := ptotspot(oo,tt,hh)-SymRegObligation(oo)-sum(ss in Stations, uu in
Units)MinProduction(ss,uu)*runindunit(oo,tt,ss,uu,hh) >= 0;
        end-do

        !Running optimizations for all steps and obligations
        writeln("Optimizing Step: ",tt,", ObligationIndex: ",oo);
        OptimizationStartTime:= datetime(SYS_NOW);
        maximize(maxz);

        StepwiseObjectiveValue(oo,tt):= getobjval;
        OptimizationEndTime:= datetime(SYS_NOW);

        !Check for optimiality / feasibility
        if  (getprobstat = XPRS_INF) then
            writeln("Infeasible solution");
            writeln("Step: ",tt,", ObligationIndex: ",oo);
        end-if
        if (not getprobstat = XPRS_OPT) then
            writeln("Not solved to optimality");
            writeln("Step: ",tt,", ObligationIndex: ",oo);
        else
            writeln("Step: ",tt,", ObligationIndex: ",oo," is optimized successfully");
            writeln("Elapsed time of optimization: ",OptimizationEndTime-OptimizationStartTime," seconds");

        end-if

        !Saving the realized  solutions from each step to show the actual behavior throughout the step
        forall(rr in Reservoirs, ss in Stations, uu in Units, hh in 24*tt-23 .. 24*tt) do

            RealizedPTotSpot(oo,hh):= getsol(ptotspot(oo,tt,hh));
            RealizedQInRes(oo,rr,hh):= getsol(qinres(oo,tt,rr,hh));
            RealizedRunIndUnit(oo,ss,uu,hh):= getsol(runindunit(oo,tt,ss,uu,hh));
            RealizedPUnit(oo,ss,uu,hh):= getsol(punit(oo,tt,ss,uu,hh));
            RealizedQBypassRes(oo,rr,hh):= getsol(qbypassres(oo,tt,rr,hh));
            RealizedQFloodRes(oo,rr,hh):= getsol(qfloodres(oo,tt,rr,hh));
            RealizedQProdUnit(oo,ss,uu,hh):= getsol(qprodunit(oo,tt,ss,uu,hh));
            RealizedStartIndUnit(oo,ss,uu,hh):= getsol(startindunit(oo,tt,ss,uu,hh));
            RealizedPEndRes(oo,rr):= getsol(pendres(oo,tt,rr));
            RealizedQEndRes(oo,rr):= getsol(qendres(oo,tt,rr));
            if (getsol(qprodunit(oo,tt,ss,uu,hh)) >0) then
                RealizedTurbEff(oo,ss,uu,hh):=
(1000000*getsol(punit(oo,tt,ss,uu,hh))/(getsol(qprodunit(oo,tt,ss,uu,hh))*9.81*1000*HeadUnit(ss,uu)));
            end-if
        end-do

        !Saving the Stepwise solutions from each step to show the Stepwise behavior throughout the step
        forall (rr in Reservoirs, ss in Stations, uu in Units, hh in Hours) do
            StepwisePTotSpot(oo,tt,hh):= getsol(ptotspot(oo,tt,hh));
            StepwiseQInRes(oo,tt,rr,hh):= getsol(qinres(oo,tt,rr,hh));
            StepwiseRunIndUnit(oo,tt,ss,uu,hh):= getsol(runindunit(oo,tt,ss,uu,hh));
            StepwisePUnit(oo,tt,ss,uu,hh):= getsol(punit(oo,tt,ss,uu,hh));
            StepwiseQBypassRes(oo,tt,rr,hh):= getsol(qbypassres(oo,tt,rr,hh));
            StepwiseQFloodRes(oo,tt,rr,hh):= getsol(qfloodres(oo,tt,rr,hh));
            StepwiseQProdUnit(oo,tt,ss,uu,hh):= getsol(qprodunit(oo,tt,ss,uu,hh));
            StepwiseStartIndUnit(oo,tt,ss,uu,hh):= getsol(startindunit(oo,tt,ss,uu,hh));
            StepwisePEndRes(oo,tt,rr):= getsol(pendres(oo,tt,rr));
            StepwiseQEndRes(oo,tt,rr):= getsol(qendres(oo,tt,rr));
        end-do

    end-do !Ending obligations

    if (tt = getsize(Steps)) then

        ! Emulating the market for the last step
        Dice(tt):= random;

        DiceAdjusted(tt):= Dice(tt)*RealizedProbability(tt);

        forall(ee in Scenarios | StepPredecessorDynamic(ee,tt) = RealizedPredecessor(tt)) do
                if (DiceAdjusted(tt) < sum(pp in
RealizedPredDynCoord(tt)..ee)(StepProbabilityDynamic(pp,tt))) then
                    RealizedProbDynCoord(tt):=ee;
                forall (hh in (24*(tt-1) .. 24*tt)) do
                        ActualPrice(hh):= ScenarioSpotPrice(ee,hh);
                    end-do
                end-if
        end-do

        !Storing the price optimized upon for the last step
        forall (hh in (24*tt-23) .. 24*getsize(Steps)) do
            ModelPrice(hh):= sum(ee in Scenarios | StepPredecessorFull(ee,tt) = RealizedPredecessor(tt))
            ((ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee))
            /(sum(eee in Scenarios | StepPredecessorFull(eee,tt) =
RealizedPredecessor(tt))(EndScenarioProbability(eee))));
        end-do

    end-if

end-do !Ending steps

!Storing the ModelPrice for each step
forall (tt in Steps, hh in (24*tt-23) .. 24*getsize(Steps)) do
    if (tt=1) then
        StepwiseModelPrice(tt,hh):= sum(ee in
Scenarios)(ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee));
    else
        StepwiseModelPrice(tt,hh):=sum(ee in Scenarios | StepPredecessorFull(ee,tt) =
RealizedPredecessor(tt))
            ((ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee))
            /(sum(eee in Scenarios | StepPredecessorFull(eee,tt) =
RealizedPredecessor(tt))(EndScenarioProbability(eee))));
    end-if
end-do

!Calculating and storing the actual realized objective value for each obligation
forall (oo in ObligationIndex) do
    ActualObjectiveValue(oo):= sum(hh in Hours)(ActualPrice(hh)*RealizedPTotSpot(oo,hh))
    -sum(ss in Stations, uu in Units, hh in Hours)(StartUpCost(ss,uu)*RealizedStartIndUnit(oo,ss,uu,hh))
    +sum(rr in Reservoirs)(WaterValue(rr)*RealizedPEndRes(oo,rr));
end-do

!Calculating and storing the objective value optimized upon for each obligation
forall (oo in ObligationIndex) do
    ModelObjectiveValue(oo):= sum(hh in Hours)(ModelPrice(hh)*RealizedPTotSpot(oo,hh))
    -sum(ss in Stations, uu in Units, hh in Hours)(StartUpCost(ss,uu)*RealizedStartIndUnit(oo,ss,uu,hh))
    +sum(rr in Reservoirs)(WaterValue(rr)*RealizedPEndRes(oo,rr));
end-do

!Calculating and storing the actual realized break-even price optimized upon for attending regulation for
each obligation
forall (oo in 2.. getsize(ObligationIndex)) do
    ActualBreakEvenRegPrice(oo):= (ActualObjectiveValue(1)-
ActualObjectiveValue(oo))/(getsize(Hours)*SymRegObligation(oo));
end-do

!Calculating and storing the actual break-even price optimized upon for attending regulation for each
obligation
forall (oo in 2.. getsize(ObligationIndex)) do
    ModelBreakEvenRegPrice(oo):= (ModelObjectiveValue(1)-
ModelObjectiveValue(oo))/(getsize(Hours)*SymRegObligation(oo));
end-do

!Calculating and storing the actual realized marginal cost for attending regulation for each obligation
forall (oo in 2.. getsize(ObligationIndex)) do
    ActualMarginalRegCost(oo):= (ActualObjectiveValue(oo-1)-ActualObjectiveValue(oo))/(getsize(Hours));
end-do

!Calculating and storing the actual realized marginal cost for attending regulation for each obligation
forall (oo in 2.. getsize(ObligationIndex)) do
    ModelMarginalRegCost(oo):= (ModelObjectiveValue(oo-1)-ModelObjectiveValue(oo))/(getsize(Hours));
end-do

forall (oo in ObligationIndex, ss in Stations, hh in Hours) do
    RealizedRunningUnits(oo,ss,hh):= (sum(uu in Units)RealizedRunIndUnit(oo,ss,uu,hh));
end-do

!Calculating the objective values seen (and realized) for each Stepwise step
forall (oo in ObligationIndex, tt in Steps) do
        if (tt=1) then
            StepwiseModelTotObjVal(oo,tt):=StepwiseObjectiveValue(oo,tt);
            StepwiseActualTotObjVal(oo,tt):=StepwiseObjectiveValue(oo,tt);
        else
            StepwiseModelTotObjVal(oo,tt):= sum(hh in 1..24*(tt-1))(ModelPrice(hh)*RealizedPTotSpot(oo,hh)-
sum(ss in Stations, uu in
Units)(StartUpCost(ss,uu)*RealizedStartIndUnit(oo,ss,uu,hh)))+StepwiseObjectiveValue(oo,tt);
            StepwiseActualTotObjVal(oo,tt):= sum(hh in 1..24*(tt-1))(ActualPrice(hh)*RealizedPTotSpot(oo,hh)-
sum(ss in Stations, uu in
Units)(StartUpCost(ss,uu)*RealizedStartIndUnit(oo,ss,uu,hh)))+StepwiseObjectiveValue(oo,tt);
        end-if
end-do

!Calculating the needed regulation price for each Stepwise step
forall (oo in 2.. getsize(ObligationIndex), ObligationIndex, tt in Steps) do
    StepwiseModelBreakEvenRegPrice(oo,tt):= (StepwiseModelTotObjVal(1,tt)-
StepwiseModelTotObjVal(oo,tt))/(SymRegObligation(oo)*getsize(Hours));
    StepwiseActualBreakEvenRegPrice(oo,tt):= (StepwiseActualTotObjVal(1,tt)-
StepwiseActualTotObjVal(oo,tt))/(SymRegObligation(oo)*getsize(Hours));
end-do

!Calculating the marginal regulation cost for each Stepwise step
forall (oo in 2.. getsize(ObligationIndex), ObligationIndex, tt in Steps) do
    StepwiseModelMarginalRegCost(oo,tt):= (StepwiseModelTotObjVal(oo-1,tt)-
StepwiseModelTotObjVal(oo,tt))/getsize(Hours);
    StepwiseActualMarginalRegCost(oo,tt):= (StepwiseActualTotObjVal(oo-1,tt)-
StepwiseActualTotObjVal(oo,tt))/getsize(Hours);
end-do
```

```
OptimizationStartTime:= datetime(SYS_NOW);
!Writes all output data to Excel file
writeln("Writing output data to Excel file");
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\output.xlsx');
    !Writing actual and realized results
    SQLexecute("insert into XACTUALPRICE (HOURS,ACTUALPRICE) values (?,?)",ActualPrice);
    SQLexecute("insert into XPTOTSPOT (OBLIGATIONINDEX,HOURS,PTOTSPOT) values (?,?,?)",RealizedPTotSpot);
    SQLexecute("insert into XQINRES (OBLIGATIONINDEX,RESERVOIRS,HOURS,QINRES) values
(?,?,?,?)",RealizedQInRes);
    SQLexecute("insert into XRUNINDUNIT (OBLIGATIONINDEX,STATIONS, UNITS,HOURS,RUNINDUNIT) values
(?,?,?,?,?)",RealizedRunIndUnit);
    SQLexecute("insert into XPUNIT (OBLIGATIONINDEX,STATIONS,UNITS,HOURS,PUNIT) values
(?,?,?,?,?)",RealizedPUnit);
    SQLexecute("insert into XQBYPASSRES (OBLIGATIONINDEX,RESERVOIRS,HOURS,QBYPASSRES) values
(?,?,?,?)",RealizedQBypassRes);
    SQLexecute("insert into XQFLOODRES (OBLIGATIONINDEX,RESERVOIRS,HOURS,QFLOODRES) values
(?,?,?,?)",RealizedQFloodRes);
    SQLexecute("insert into XQPRODUNIT (OBLIGATIONINDEX,STATIONS,UNITS,HOURS,QPRODUNIT) values
(?,?,?,?,?)",RealizedQProdUnit);
    SQLexecute("insert into XSTARTINDUNIT (OBLIGATIONINDEX,STATIONS, UNITS,HOURS,STARTINDUNIT) values
(?,?,?,?,?)",RealizedRunIndUnit);
    SQLexecute("insert into XPENDRES (OBLIGATIONINDEX,RESERVOIRS,PENDRES) values (?,?,?)",RealizedPEndRes);
    SQLexecute("insert into XQENDRES (OBLIGATIONINDEX,RESERVOIRS,QENDRES) values (?,?,?)",RealizedQEndRes);
    SQLexecute("insert into XACTUALOBJECTIVEVALUE (OBLIGATIONINDEX,ACTUALOBJECTIVEVALUE) values
(?,?)",ActualObjectiveValue);
    SQLexecute("insert into XACTUALBREAKEVENREGPRICE (OBLIGATIONINDEX,ACTUALBREAKEVENREGPRICE) values
(?,?)",ActualBreakEvenRegPrice);
    SQLexecute("insert into XSYMREGOBLIGATION (OBLIGATIONINDEX,SYMREGOBLIGATION) values
(?,?)",SymRegObligation);
    SQLexecute("insert into XTURBEFF (OBLIGATIONINDEX,STATIONS,UNITS,HOURS,TURBEFF) values
(?,?,?,?,?)",RealizedTurbEff);
    SQLexecute("insert into XMODELPRICE (HOURS,MODELPRICE) values (?,?)",ModelPrice);
    SQLexecute("insert into XMODELOBJECTIVEVALUE (OBLIGATIONINDEX,MODELOBJECTIVEVALUE) values
(?,?)",ModelObjectiveValue);
    SQLexecute("insert into XMODELBREAKEVENREGPRICE (OBLIGATIONINDEX,MODELBREAKEVENREGPRICE) values
(?,?)",ModelBreakEvenRegPrice);
    SQLexecute("insert into XMODELMARGINALREGCOST (OBLIGATIONINDEX,MODELMARGINALREGCOST) values
(?,?)",ModelMarginalRegCost);
    SQLexecute("insert into XACTUALMARGINALREGCOST (OBLIGATIONINDEX,ACTUALMARGINALREGCOST) values
(?,?)",ActualMarginalRegCost);


    SQLexecute("insert into XRUNNINGUNITS (OBLIGATIONINDEX,STATIONS,HOURS,RUNNINGUNITS) values
(?,?,?,?)",RealizedRunningUnits);
    !Writing stepwise results
    SQLexecute("insert into XIPTOTSPOT (OBLIGATIONINDEX,STEPS,HOURS,IPTOTSPOT) values
(?,?,?,?)",StepwisePTotSpot);
    SQLexecute("insert into XIQINRES (OBLIGATIONINDEX,STEPS,RESERVOIRS,HOURS,IQINRES) values
(?,?,?,?,?)",StepwiseQInRes);
    SQLexecute("insert into XIRUNINDUNIT (OBLIGATIONINDEX,STEPS,STATIONS,UNITS,HOURS,IRUNINDUNIT) values
(?,?,?,?,?,?)",StepwiseRunIndUnit);
    SQLexecute("insert into XISTARTINDUNIT (OBLIGATIONINDEX,STEPS,STATIONS,UNITS,HOURS,ISTARTINDUNIT) values
(?,?,?,?,?,?)",StepwiseStartIndUnit);
    SQLexecute("insert into XIPUNIT (OBLIGATIONINDEX,STEPS,STATIONS,UNITS,HOURS,IPUNIT) values
(?,?,?,?,?,?)",StepwisePUnit);
    SQLexecute("insert into XIQBYPASSRES (OBLIGATIONINDEX,STEPS,RESERVOIRS,HOURS,IQBYPASSRES) values
(?,?,?,?,?)",StepwiseQBypassRes);
    SQLexecute("insert into XIQFLOODRES (OBLIGATIONINDEX,STEPS,RESERVOIRS,HOURS,IQFLOODRES) values
(?,?,?,?,?)",StepwiseQFloodRes);
    SQLexecute("insert into XIQPRODUNIT (OBLIGATIONINDEX,STEPS,STATIONS,UNITS,HOURS,IQPRODUNIT) values
(?,?,?,?,?,?)",StepwiseQProdUnit);
    SQLexecute("insert into XIPENDRES (OBLIGATIONINDEX,STEPS,RESERVOIRS,IPENDRES) values
(?,?,?,?)",StepwisePEndRes);
    SQLexecute("insert into XIQENDRES (OBLIGATIONINDEX,STEPS,RESERVOIRS,IQENDRES) values
(?,?,?,?)",StepwiseQEndRes);
    SQLexecute("insert into XIOBJECTIVEVALUE (OBLIGATIONINDEX,STEPS,IOBJECTIVEVALUE) values
(?,?,?)",StepwiseObjectiveValue);
    SQLexecute("insert into XIMODELTOTOBJVAL (OBLIGATIONINDEX,STEPS,IMODELTOTOBJVAL) values
(?,?,?)",StepwiseModelTotObjVal);
    SQLexecute("insert into XIMODELBREAKEVENREGPRICE (OBLIGATIONINDEX,STEPS,IMODELBREAKEVENREGPRICE) values
(?,?,?)",StepwiseModelBreakEvenRegPrice);
    SQLexecute("insert into XIACTUALTOTOBJVAL (OBLIGATIONINDEX,STEPS,IACTUALTOTOBJVAL) values
(?,?,?)",StepwiseActualTotObjVal);
    SQLexecute("insert into XIACTUALBREAKEVENREGPRICE (OBLIGATIONINDEX,STEPS,IACTUALBREAKEVENREGPRICE) values
(?,?,?)",StepwiseActualBreakEvenRegPrice);
    SQLexecute("insert into XIACTUALMARGINALREGCOST (OBLIGATIONINDEX,STEPS,IACTUALMARGINALREGCOST) values
(?,?,?)",StepwiseActualMarginalRegCost);
    SQLexecute("insert into XIMODELMARGINALREGCOST (OBLIGATIONINDEX,STEPS,IMODELMARGINALREGCOST) values
(?,?,?)",StepwiseModelMarginalRegCost);
    SQLexecute("insert into XIMODELPRICE (STEPS,HOURS,IMODELPRICE) values (?,?,?)",StepwiseModelPrice);


SQLdisconnect;

OptimizationEndTime:= datetime(SYS_NOW);
writeln("Successfully written data to Excel file");
writeln("Time used to write output data: ",(OptimizationEndTime-OptimizationStartTime)," seconds");
writeln("Model run complete");
ModelEndTime:= datetime(SYS_NOW);
writeln("Total elapsed time of model: ",(ModelEndTime-ModelStartTime)," seconds");
end-model;
```

E.2 The stochastic multi-stage optimization model with Monte Carlo-properties

```
model stochastic_montecarlo !Model name
options explterm !All statements must be terminated by a semicolon
options noimplicit !No implicit declarations
uses "mmxprs"; !Gain access to the Xpress-Optimizer library
uses "mmodbc"; !Gain access to SQL for Excel input
uses "mmsystem"; !Gain access to datetime

!Declaration of tic/toc
declarations
    ModelStartTime: datetime;
    ModelEndTime: datetime;
    OptimizationStartTime: datetime;
    OptimizationEndTime: datetime;
end-declarations

ModelStartTime:= datetime(SYS_NOW);

!Declaration of sets
declarations
    Hours:          set of integer;
    Days:           set of integer;
    Units:          set of integer;
    Stations:       set of integer;
    Reservoirs:     set of integer;
    Index:          set of integer;
    ObligationIndex:set of integer;
    Scenarios:      set of integer;
    Steps:          set of integer;
    Nodes:          set of integer;
    ModelIndex:     set of integer;
end-declarations

!Reading input data of sets from excel
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\input_216.xlsx');
    SQLexecute("select HOURS from HH", Hours);
    SQLexecute("select UNITS from UU", Units);
    SQLexecute("select STATIONS from SS", Stations);
    SQLexecute("select RESERVOIRS from RR", Reservoirs);
    SQLexecute("select INDEX from II", Index);
    SQLexecute("select OBLIGATIONINDEX from OO", ObligationIndex);
    SQLexecute("select MODELINDEX from MM", ModelIndex);
SQLdisconnect;

SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\stoch_input_216.xlsx');
    SQLexecute("select SCENARIOS from EE", Scenarios);
    SQLexecute("select STEPS from TT", Steps);
    SQLexecute("select NODES from NN", Nodes);
SQLdisconnect;

!Declaration of general parameters
declarations
    !Spotprice
    SpotPrice:                  array(Hours)                of real;
    !Symmetrical regulation obligation
    SymRegObligation:           array(ObligationIndex)      of real;
    !Water value
    WaterValue:                 array(Reservoirs)           of real;
    !Intial reservoar volum
    InitialWaterVolume:         array(Reservoirs)           of real;
    !Inflow
    InflowRes:                  array(Reservoirs,Hours)     of real;
    !Max production on each unit
    MaxProduction:              array(Stations,Units)       of real;
    !Min production on each unit
    MinProduction:              array(Stations,Units)       of real;
    !Max bypass m3/s
    MaxBypass:                  array(Reservoirs)           of real;
    !Min bypass m3/s
    MinBypass:                  array(Reservoirs)           of real;
    !Max reservoar volume m3
    MaxReservoirVolume:         array(Reservoirs)           of real;
    !Min reservoar volume m3
    MinReservoirVolume:         array(Reservoirs)           of real;
    !Max regulation of reservoar per hour m3/h
    MaxRegulatingPerHour:       array(Reservoirs)           of real;
     !Start-up cost at unit u at station s
    StartUpCost:                array(Stations,Units)       of real;
    !Max ramping at unit u at station s
    MaxRamping:                 array(Stations,Units)       of real;
    !Slopes of the piecewise linear PQ-curve
    Slope:                      array(Index)                of real;
    !Interceptions of the piecewise linear PQ-curve
    Intercept:                  array(Index)                of real;
    !Head for each unit
    HeadUnit:                   array(Stations,Units)       of real;
    !System head for each unit
    TotHeadUnit:                array(Stations,Units)       of real;
    !Max discharge through unit u at station s
    MaxDiscUnit:                array(Stations,Units)       of real;
    !Min discharge through unit u at station s
    MinDiscUnit:                array(Stations,Units)       of real;
    !Best point efficiency for each unit
    EffBPUnit:                  array(Stations,Units)       of real;

    !Declaration of parameters used by the stochastic part of the model

    !The spot prices for all possible Scenarios
    ScenarioSpotPrice:          array(Scenarios, Hours) of real;
    !The distribution of probabilities for each Scenario to occur
    EndScenarioProbability:     array(Scenarios)        of real;
    !The predecessor node given each node
    NodePredecessor:            array(Nodes)            of integer;
    !The probability for each node in the scenario tree
    NodeProbability:            array(Nodes)            of real;
    !The number of branches in each step in the scenario tree
    StepNumberOfBranch:         array(Steps)            of integer;
    !The matrices created by the model in order to use Scenred data
    StepProbabilityDynamic:             array(Scenarios,Steps) of real;
    StepPredecessorDynamic:     array(Scenarios,Steps) of integer;
    StepPredecessorFull:        array(Scenarios,Steps) of real;
    !Counters needed in order to create the above matrices
    EqualNumbersCounter:        integer;
    IteratorCounter:            integer;
    Counter:                    integer;
    CoordCounter:               integer;
    Indicator:                  integer;
    !Storing the random function deciding the stochastic outcomes
    Dice:                       array(Steps) of real;
    DiceAdjusted:               array(Steps) of real;
    !Declaration of parameters used to store stochastic results
    RealizedPredecessor:        array(Steps) of real;
    RealizedPredDynCoord:       array(Steps) of integer;
    RealizedProbDynCoord:       array(Steps) of integer;
    RealizedProbability:        array(Steps) of real;

    !Declaration of parameters used to store variables of stepwise runs
    StepwisePTotSpot:               array(ObligationIndex,Steps,Hours)                  of real;
    StepwiseQInRes:                 array(ObligationIndex,Steps,Reservoirs,Hours)       of real;
    StepwiseRunIndUnit:             array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwisePUnit:                  array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwiseQBypassRes:             array(ObligationIndex,Steps,Reservoirs,Hours)       of real;
    StepwiseQFloodRes:              array(ObligationIndex,Steps,Reservoirs,Hours)       of real;
    StepwiseQProdUnit:              array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwiseStartIndUnit:           array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwisePEndRes:                array(ObligationIndex,Steps,Reservoirs)             of real;
    StepwiseObjectiveValue:         array(ObligationIndex,Steps)                        of real;
    StepwiseQEndRes:                array(ObligationIndex,Steps,Reservoirs)             of real;
    StepwiseBreakEvenRegPrice:      array(ObligationIndex,Steps)                        of real;
    StepwiseTurbEff:                array(ObligationIndex,Steps,Stations,Units,Hours)   of real;
    StepwiseModelTotObjVal:         array(ObligationIndex,Steps)                        of real;
    StepwiseModelBreakEvenRegPrice: array(ObligationIndex,Steps)                        of real;
    StepwiseActualTotObjVal:        array(ObligationIndex,Steps)                        of real;
    StepwiseActualBreakEvenRegPrice:array(ObligationIndex,Steps)                        of real;
    StepwiseModelMarginalRegCost:   array(ObligationIndex,Steps)                        of real;
    StepwiseActualMarginalRegCost:  array(ObligationIndex,Steps)                        of real;


    !Declaration of parameters used to store variables of realized runs
    ActualPrice:                array(ModelIndex,Hours)                     of real;
    ActualObjectiveValue:       array(ModelIndex,ObligationIndex)           of real;
    ActualBreakEvenRegPrice:    array(ModelIndex,ObligationIndex)           of real;
    ActualMarginalRegCost:      array(ModelIndex,ObligationIndex)           of real;
    RealizedPTotSpot:           array(ObligationIndex,Hours)                of real;
    RealizedQInRes:             array(ObligationIndex,Reservoirs,Hours)     of real;
    RealizedRunIndUnit:         array(ObligationIndex,Stations,Units,Hours) of real;
    RealizedPUnit:              array(ObligationIndex,Stations,Units,Hours) of real;
    RealizedQBypassRes:         array(ObligationIndex,Reservoirs,Hours)     of real;
    RealizedQFloodRes:          array(ObligationIndex,Reservoirs,Hours)     of real;
    RealizedQProdUnit:          array(ObligationIndex,Stations,Units,Hours) of real;
    RealizedStartIndUnit:       array(ObligationIndex,Stations,Units,Hours) of real;
    RealizedPEndRes:            array(ObligationIndex,Reservoirs)           of real;
    RealizedQEndRes:            array(ObligationIndex,Reservoirs)           of real;
    RealizedTurbEff:            array(ObligationIndex,Stations,Units,Hours) of real;
    RealizedRunningUnits:       array(ObligationIndex, Stations,Hours)      of real;

    !Declaration of parameters used to store the price and marginal cost for regulation
    !(and correspondig objective value) that the model sees when optimizing
    ModelPrice:                 array(ModelIndex,Hours)             of real;
    ModelObjectiveValue:        array(ModelIndex,ObligationIndex)   of real;
    ModelBreakEvenRegPrice:     array(ModelIndex,ObligationIndex)   of real;
    ModelMarginalRegCost:       array(ModelIndex,ObligationIndex)   of real;
end-declarations

    !Reading input data of parameters from excel
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\input_216.xlsx');
    SQLexecute("select SPOTPRICE from HH", SpotPrice);
    SQLexecute("select SYMREGOBLIGATION from OO", SymRegObligation);
    SQLexecute("select WATERVALUE from RR", WaterValue);
    SQLexecute("select INITIALWATERVOLUME from RR", InitialWaterVolume);
    SQLexecute("select INFLOWRES from RRHH", InflowRes);
    SQLexecute("select MAXPRODUCTION from SSUU", MaxProduction);
    SQLexecute("select MINPRODUCTION from SSUU", MinProduction);
    SQLexecute("select MAXBYPASS from RR", MaxBypass);
    SQLexecute("select MINBYPASS from RR", MinBypass);
    SQLexecute("select MAXRESERVOIRVOLUME from RR", MaxReservoirVolume);
    SQLexecute("select MINRESERVOIRVOLUME from RR", MinReservoirVolume);
    SQLexecute("select MAXREGULATINGPERHOUR from RR", MaxRegulatingPerHour);
    SQLexecute("select STARTUPCOST from SSUU", StartUpCost);
    SQLexecute("select MAXRAMPING from SSUU", MaxRamping);
    SQLexecute("select HEADUNIT from SSUU", HeadUnit);
    SQLexecute("select MAXDISCUNIT from SSUU", MaxDiscUnit);
    SQLexecute("select MINDISCUNIT from SSUU", MinDiscUnit);
    SQLexecute("select TOTHEADUNIT from SSUU", TotHeadUnit);
    SQLexecute("select EFFBPUNIT from SSUU", EffBPUnit);
    !PQ curve
    SQLexecute("select SLOPE from II", Slope);
    SQLexecute("select INTERCEPT from II", Intercept);
    SQLdisconnect;

    SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\stoch_input_216.xlsx');
    SQLexecute("select SCENARIOSPOTPRICE from EEHH", ScenarioSpotPrice);
    SQLexecute("select ENDSCENARIOPROBABILITY from EE", EndScenarioProbability);
    SQLexecute("select NODEPREDECESSOR from NN", NodePredecessor);
    SQLexecute("select NODEPROBABILITY from NN", NodeProbability);
    SQLexecute("select STEPNUMBEROFBRANCH from TT", StepNumberOfBranch);
SQLdisconnect;

writeln("All input data successfully read from Excel file");

!Declaration of variables

declarations

    !Total spot volume in hour h
    ptotspot:       array(ObligationIndex,Steps,Hours)                of mpvar;
    !Reservoar volume at the end of the periode
    qendres:        array(ObligationIndex,Steps,Reservoirs)           of mpvar;
    !Energy in the reservoar at the end of the periode
    pendres:        array(ObligationIndex,Steps,Reservoirs)           of mpvar;
    !Discharge thruogh unit u at station s in hour h
    qprodunit:      array(ObligationIndex,Steps,Stations,Units,Hours) of mpvar;
    !Flood from reservoar r in hour h
    qfloodres:      array(ObligationIndex,Steps,Reservoirs,Hours)     of mpvar;
    !Bypass from reservoar r in hour h
    qbypassres:     array(ObligationIndex,Steps,Reservoirs,Hours)     of mpvar;
    !reservoar volume in reservoar r in hour h
    qinres:         array(ObligationIndex,Steps,Reservoirs,Hours)     of mpvar;
    !Spot production from unit u at station s in hour h
    punit:          array(ObligationIndex,Steps,Stations,Units,Hours) of mpvar;
    !start indicator
    startindunit:   array(ObligationIndex,Steps,Stations,Units,Hours) of mpvar;
    !Run indicator, 1 if unit u runs in hour h
    runindunit:     array(ObligationIndex,Steps,Stations,Units,Hours) of mpvar;


    ! Declaration of the objective function
maxz:                   linctr;

!Declaration of constranits

! Sets initial reservoir volume in qinres for all reservoirs
restInitWaterBalance:       array(Reservoirs)               of linctr;
!Water balances
restWaterBalance1:          array(Reservoirs,Hours)         of linctr;
restWaterBalance2:          array(Reservoirs,Hours)         of linctr;
restWaterBalance3:          array(Reservoirs,Hours)         of linctr;
restWaterBalance4:          array(Reservoirs,Hours)         of linctr;
restWaterBalance5:          array(Reservoirs,Hours)         of linctr;
restWaterBalance6:          array(Reservoirs,Hours)         of linctr;
restWaterBalance7:          array(Reservoirs,Hours)         of linctr;
restWaterBalance8:          array(Reservoirs,Hours)         of linctr;
!Total spot production from unit to system level
restTotSpotProd:            array(Hours)                    of linctr;
!Max bypass
restMaxBypassRes:           array(Reservoirs,Hours)         of linctr;
!Min bypass
restMinBypassRes:           array(Reservoirs,Hours)         of linctr;
!Max reservoir volume
restMaxVolumeRes:           array(Reservoirs,Hours)         of linctr;
!Min reservoir volume
restMinVolumeRes:           array(Reservoirs,Hours)         of linctr;
!Max reservoir regulation  equation 1
restMaxRegRes1:             array(Reservoirs,Hours)         of linctr;
!Max reservoir regulation equation 2
restMaxRegRes2:             array(Reservoirs,Hours)         of linctr;
!Start-up cost
restStartupCostUnit:        array(Stations,Units,Hours)     of linctr;
!Max ramping equation 1
restMaxRampUnit1:           array(Stations,Units)           of linctr;
!Max ramping equation 2
restMaxRampUnit2:           array(Stations,Units)           of linctr;
!Max production on unit u at station s in hour h
restMaxProdUnit:            array(Stations,Units,Hours)     of linctr;
!Min production on unit u at station s in hour h
restMinProdUnit:            array(Stations,Units,Hours)     of linctr;
!Production indicator for each unit in each hour
restProdIndUnit:            array(Stations,Units,Hours)     of linctr;
```

```
!Max reservoar volume at the end of the period
restMaxVolumeEndRes:        array(Reservoirs)               of linctr;
!Min reservoar volume at the end of the period
restMinVolumeEndRes:        array(Reservoirs)               of linctr;
!Calculates the equivalent energy of the water in the reservoir at the end of the
!period in MWh
restEndResEnergy:           array(Reservoirs)               of linctr;
!Max discharge thruogh unit u at station s in hour h
restMaxDiscUnit:            array(Stations,Units,Hours)     of linctr;
!Min discharge thruogh unit u at station s in hour h
restMinDiscUnit:            array(Stations,Units,Hours)     of linctr;
!PQ-curve
restPQCurve1:               array(Stations, Units,Hours)    of linctr;
!PQ-curve
restPQCurve2:               array(Stations, Units,Hours)    of linctr;
!PQ-curve
restPQCurve3:               array(Stations, Units,Hours)    of linctr;
!PQ-curve
restPQCurve4:               array(Stations, Units,Hours)    of linctr;
!PQ-curve
restPQCurve5:               array(Stations, Units,Hours)    of linctr;
!PQ-curve
restPQCurve6:               array(Stations, Units,Hours)    of linctr;
!PQ-curve
restPQCurve7:               array(Stations, Units,Hours)    of linctr;
!Reserving upward regulation capacity
restMaxProdObligation:      array(Hours)                    of linctr;
!Reserving downward regulation capacity
restMinProdObligation:      array(Hours)                    of linctr;

restStepChangeQInRes:       array(Reservoirs)               of linctr;

restStepChangeQProdUnit:    array (Stations,Units)          of linctr;

restStepChangeQFloodRes:    array(Reservoirs)               of linctr;

restStepChangeQBypassRes:   array(Reservoirs)               of linctr;

restStepChangePUnit:        array (Stations,Units)          of linctr;

restStepChangeRunIndUnit:   array (Stations,Units)          of linctr;

restStepChangeStartIndUnit: array (Stations,Units)          of linctr;

end-declarations


! Creates binary variable runindunit
forall (oo in ObligationIndex, tt in Steps,ss in Stations,uu in Units,hh in Hours) do
    runindunit(oo,tt,ss,uu,hh) is_binary;
end-do

!forall (oo in ObligationIndex, tt in Steps,ss in Stations,uu in Units,hh in Hours) do
!   startindunit(oo,tt,ss,uu,hh) is_binary;
!end-do

writeln("All restrictions and variables successfully created");
writeln("Creating stochastic matrices needed by the optimization model");

! Creation of needed matrices for the stochastic calculations

! Creating the dynamic matrices from excel input data

forall (tt in Steps) do

    if (tt =1) then

        forall (ee in 1.. StepNumberOfBranch(tt)) do

            StepProbabilityDynamic(ee,tt):= NodeProbability(1+ee);
            StepPredecessorDynamic(ee,tt):= NodePredecessor(1+ee);

        end-do

    else

        forall (ee in 1.. StepNumberOfBranch(tt)) do

            StepProbabilityDynamic(ee,tt) := NodeProbability(1+sum(ttt in 2.. tt)(24*StepNumberOfBranch(ttt-
1))+ee);
            StepPredecessorDynamic(ee,tt) := NodePredecessor(1+sum(ttt in 2.. tt)(24*StepNumberOfBranch(ttt-
1))+ee);

        end-do

    end-if

end-do

    !Creating the explicit matrix from the dynamic matrix

    !Creating last column of the explicit matrix
    forall (ee in Scenarios) do

        StepPredecessorFull(ee,getsize(Steps)):=StepPredecessorDynamic(ee,getsize(Steps));
    end-do


    ! Creating the remaining columns of the explicit matrix
    forall (tt in 1.. getsize(Steps)-1) do

        EqualNumbersCounter := 0;
        IteratorCounter:= 1;
        Counter:= 1;

        forall (ee in 2.. getsize(Scenarios)) do

            if (StepPredecessorFull(ee-1,1+getsize(Steps)-tt) = StepPredecessorFull(ee,1+getsize(Steps)-tt))
then

                EqualNumbersCounter:= EqualNumbersCounter + 1;

            else

                forall(pp in IteratorCounter.. IteratorCounter+EqualNumbersCounter) do

                        StepPredecessorFull(pp,(getsize(Steps)-tt)) :=
StepPredecessorDynamic(Counter,getsize(Steps)-tt);
                end-do

                Counter:= Counter + 1;
                IteratorCounter := IteratorCounter + EqualNumbersCounter +1;
                EqualNumbersCounter :=0;

            end-if

        end-do

    end-do

        forall (tt in 1.. getsize(Steps)-1) do

            StepPredecessorFull(getsize(Scenarios),getsize(Steps)-tt):=
            StepPredecessorDynamic(StepNumberOfBranch(getsize(Steps)-tt),getsize(Steps)-tt);

        end-do


            forall (mm in ModelIndex) do

    writeln("Model run start for ModelIndex ", mm);

    ! Model algoritm begins

forall (tt in Steps) do ! Iterating over all steps

        if (tt =1) then ! For the first step only

        !Establishing initial values

            RealizedProbability(tt):= 1;
            RealizedPredecessor(tt):=1;
            RealizedPredDynCoord(tt):=1;

            !Optimizing the entire period with all scenarios weighted according to corresponding probability



            !Saving the price optimized upon for the first 24 hours
            forall (hh in 1..24) do
                ModelPrice(mm,hh) :=sum(ee in
Scenarios)(ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee));
            end-do

        else    ! for all other steps

                ! Emulating what actually happend from the previous step

            Dice(tt-1):= random;    ! Emulating which way the market will go
            DiceAdjusted(tt-1):= Dice(tt-1)*RealizedProbability(tt-1);  ! Adjusting it to the same size as
corresponding


            ! Storing the coordinate to the probability that was realized in the previous step
            Indicator:=0;

            forall(ee in Scenarios | StepPredecessorDynamic(ee,tt-1) = RealizedPredecessor(tt-1)) do

                if (DiceAdjusted(tt-1) < sum(pp in RealizedPredDynCoord(tt-
1)..ee)(StepProbabilityDynamic(pp,tt-1))) then

                    if (Indicator = 0) then

                    RealizedProbDynCoord(tt-1):=ee;
                    RealizedProbability(tt):= StepProbabilityDynamic(ee,tt-1);
                    Indicator:=1;

                    end-if
                end-if

            end-do

            ! Storing the predecessor for the probabilities that shall be weighted

            Counter:=1;
            CoordCounter:=1;
            while (Counter < RealizedProbDynCoord(tt-1)) do

                if (StepPredecessorDynamic(CoordCounter,tt) <>  StepPredecessorDynamic(CoordCounter+1,tt))
then

                    Counter:= Counter +1;

                end-if

                CoordCounter:= CoordCounter +1;

            end-do

            RealizedPredDynCoord(tt):= CoordCounter;
            RealizedPredecessor(tt):= StepPredecessorDynamic(CoordCounter,tt);

            ! Storing the actual realized spot price (by picking the first price series with the
corresponding predecessor)

            Counter:=1;

            while (StepPredecessorFull(Counter,tt) <> RealizedPredecessor(tt)) do

                Counter:= Counter + 1;

            end-do

            forall (hh in (24*tt-47) .. 24*(tt-1)) do

                ActualPrice(mm,hh):= ScenarioSpotPrice(Counter,hh);

            end-do

            ! Maximizing over all periods remaining considering valid probabilities and price series

            !Storing the price optimized upon for each step
            forall (hh in 24*(tt-1)..24*tt) do
                ModelPrice(mm,hh) :=sum(ee in Scenarios | StepPredecessorFull(ee,tt) =
RealizedPredecessor(tt))
                 ((ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee))
                 /(sum(eee in Scenarios | StepPredecessorFull(eee,tt) =
RealizedPredecessor(tt))(EndScenarioProbability(eee))));
            end-do

        end-if

        forall(oo in ObligationIndex)do !Iterating over all obligations

        if (tt = 1) then
            !Setting initial volume of reservoirs
            forall (rr in Reservoirs) do
                restInitWaterBalance(rr) := qinres(oo,tt,rr,1) = InitialWaterVolume(rr);
            end-do

            !Water balance upper reservoir for first step
            forall (rr in 1..1, hh in (24*tt-23)..(24*getsize(Steps)-1)) do
                restWaterBalance1(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)
                -sum(ss in 1..1, uu in Units)(qprodunit(oo,tt,ss,uu,hh)*3600)-qfloodres(oo,tt,rr,hh)-
qbypassres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) =0;
            end-do

            !Water balance other reservoirs for first step
            forall (rr in 2.. getsize(Reservoirs), hh in 24*tt-23 .. (24*getsize(Steps)-1)) do
                restWaterBalance2(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)+ sum(uu in
Units)(qprodunit(oo,tt,(rr-1),uu,hh)*3600)
                -sum(uu in Units)(qprodunit(oo,tt,rr,uu,hh)*3600)-
qfloodres(oo,tt,rr,hh)+qbypassres(oo,tt,(rr-1),hh)-qbypassres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) =0;
            end-do

            !Max ramping equation 1
            forall (ss in Stations, uu in Units, hh in 24*tt-22 .. 24*getsize(Steps)) do
                restMaxRampUnit1(ss,uu) := punit(oo,tt,ss,uu,hh) - punit(oo,tt,ss,uu,(hh-1)) <=
MaxRamping(ss,uu);
            end-do

            !Max ramping equation 2
            forall (ss in Stations, uu in Units, hh in 24*tt-22 .. 24*getsize(Steps)) do
                restMaxRampUnit2(ss,uu) := punit(oo,tt,ss,uu,(hh-1)) - punit(oo,tt,ss,uu,hh) >= -
MaxRamping(ss,uu);
            end-do

            !Max reservoir regulation equation 1
            forall (rr in Reservoirs, hh in 24*tt-23 .. (24*getsize(Steps)-1)) do
                restMaxRegRes1(rr,hh) := qinres(oo,tt,rr,(hh+1))-qinres(oo,tt,rr,hh) <=
MaxRegulatingPerHour(rr);
            end-do
            !Max reservoir regulation equation 2
            forall (rr in Reservoirs, hh in 24*tt-23 .. (24*getsize(Steps)-1)) do
                restMaxRegRes2(rr,hh) := qinres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) <=
MaxRegulatingPerHour(rr);
            end-do

                !Start-up costs
            forall (ss in Stations, uu in Units,hh in 24*tt-22 .. 24*getsize(Steps)) do
                restStartupCostUnit(ss,uu,hh) := startindunit(oo,tt,ss,uu,hh) - (runindunit(oo,tt,ss,uu,hh) -
runindunit(oo,tt,ss,uu,(hh-1))) >= 0 ;
            end-do

            !Start-up costs
            forall (ss in Stations, uu in Units, hh = 24*tt-23) do
                restStartupCostUnit(ss,uu,hh) := startindunit(oo,tt,ss,uu,hh) -
(runindunit(oo,tt,ss,uu,(hh+1)) - runindunit(oo,tt,ss,uu,hh)) >= 0 ;
            end-do

            !Objective value for first step
            maxz:= sum(ee in Scenarios, hh in (24*tt-23) .. 24*getsize(Steps))
            (ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee)*ptotspot(oo,tt,hh))
            + sum(rr in Reservoirs)(WaterValue(rr)*pendres(oo,tt,rr))
            - sum(ss in Stations, uu in Units, hh in (24*tt-23) .. 24*getsize(Steps))
            (StartUpCost(ss,uu)*startindunit(oo,tt,ss,uu,hh));

            else

        ! Setting all variables in current step for hour 24 equal to hour 24 of previous step
        forall (rr in Reservoirs, ss in Stations, uu in Units) do
            restStepChangeQInRes(rr) := qinres(oo,tt,rr,(24*(tt-1))) = StepwiseQInRes(oo,tt-1,rr,(24*(tt-
1)));
            restStepChangeQProdUnit(ss,uu):= qprodunit(oo,tt,ss,uu,(24*(tt-1))) = StepwiseQProdUnit(oo,tt-
1,ss,uu,(24*(tt-1)));
            restStepChangeQFloodRes(rr) := qfloodres(oo,tt,rr,(24*(tt-1))) = StepwiseQFloodRes(oo,tt-
1,rr,(24*(tt-1)));
            restStepChangeQBypassRes(rr) := qbypassres(oo,tt,rr,(24*(tt-1))) = StepwiseQBypassRes(oo,tt-
1,rr,(24*(tt-1)));
            restStepChangePUnit(ss,uu):= punit(oo,tt,ss,uu,(24*(tt-1))) = StepwisePUnit(oo,tt-
1,ss,uu,(24*(tt-1)));
            restStepChangeRunIndUnit(ss,uu) := runindunit(oo,tt,ss,uu,(24*(tt-1))) =
StepwiseRunIndUnit(oo,tt-1,ss,uu,(24*(tt-1)));
            restStepChangeStartIndUnit(ss,uu) := startindunit(oo,tt,ss,uu,(24*(tt-1))) =
StepwiseStartIndUnit(oo,tt-1,ss,uu,(24*(tt-1)));
        end-do


            !Water balance upper reservoir for other steps
            forall (rr in 1..1, hh in (24*(tt-1))..(24*getsize(Steps)-1)) do
                restWaterBalance3(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)
                -sum(ss in 1..1, uu in Units)(qprodunit(oo,tt,ss,uu,hh)*3600)-qfloodres(oo,tt,rr,hh)-
qbypassres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) =0;
            end-do

            !Water balance other reservoirs for other steps
            forall (rr in 2.. getsize(Reservoirs), hh in (24*(tt-1)) .. (24*getsize(Steps)-1)) do
                restWaterBalance4(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)+ sum(uu in
Units)(qprodunit(oo,tt,(rr-1),uu,hh)*3600)
                -sum(uu in Units)(qprodunit(oo,tt,rr,uu,hh)*3600)-
qfloodres(oo,tt,rr,hh)+qbypassres(oo,tt,(rr-1),hh)-qbypassres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) =0;
            end-do

            !Max ramping equation 1
            forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
                restMaxRampUnit1(ss,uu) := punit(oo,tt,ss,uu,hh) - punit(oo,tt,ss,uu,(hh-1)) <=
MaxRamping(ss,uu);
            end-do

            !Max ramping equation 2
            forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
                restMaxRampUnit2(ss,uu) := punit(oo,tt,ss,uu,(hh-1)) - punit(oo,tt,ss,uu,hh) >= -
MaxRamping(ss,uu);
            end-do

            !Max reservoir regulation equation 1
            forall (rr in Reservoirs, hh in 24*(tt-1) .. (24*getsize(Steps)-1)) do
                restMaxRegRes1(rr,hh) := qinres(oo,tt,rr,(hh+1))-qinres(oo,tt,rr,hh) <=
MaxRegulatingPerHour(rr);
            end-do

            !Max reservoir regulation equation 2
            forall (rr in Reservoirs, hh in 24*(tt-1) .. (24*getsize(Steps)-1)) do
                restMaxRegRes2(rr,hh) := qinres(oo,tt,rr,hh)-qinres(oo,tt,rr,(hh+1)) <=
MaxRegulatingPerHour(rr);
            end-do

            !Start-up costs
            forall (ss in Stations, uu in Units,hh in 24*tt-23 .. 24*getsize(Steps)) do
                restStartupCostUnit(ss,uu,hh) := startindunit(oo,tt,ss,uu,hh) - (runindunit(oo,tt,ss,uu,hh) -
runindunit(oo,tt,ss,uu,(hh-1))) >= 0 ;
            end-do

        ! The maximization function for all steps except tt=1
            maxz:=
            sum(ee in Scenarios | StepPredecessorFull(ee,tt) = RealizedPredecessor(tt), hh in (24*tt-23) ..
24*getsize(Steps))
            ((ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee))
            /(sum(eee in Scenarios | StepPredecessorFull(eee,tt) =
RealizedPredecessor(tt))(EndScenarioProbability(eee)))
            *ptotspot(oo,tt,hh))
            + sum(rr in Reservoirs)(WaterValue(rr)*pendres(oo,tt,rr))
            - sum(ss in Stations, uu in Units, hh in (24*tt-23) .. 24*getsize(Steps))
            (StartUpCost(ss,uu)*startindunit(oo,tt,ss,uu,hh));

        end-if

        !Water balance last hour upper reservoir
        forall (rr = 1, hh = 24*getsize(Steps)) do
            restWaterBalance5(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)
            -sum(ss in 1..1, uu in Units)(qprodunit(oo,tt,ss,uu,hh)*3600)-qfloodres(oo,tt,rr,hh)-
qbypassres(oo,tt,rr,hh)-qendres(oo,tt,rr) =0;
        end-do

        !Water balance last hour other reservoirs
        forall (rr in 2.. getsize(Reservoirs), hh = 24*getsize(Steps)) do
            restWaterBalance6(rr,hh) := qinres(oo,tt,rr,hh)+InflowRes(rr,hh)+ sum(uu in
Units)(qprodunit(oo,tt,(rr-1),uu,hh)*3600)
            -sum(uu in Units)(qprodunit(oo,tt,rr,uu,hh)*3600)-qfloodres(oo,tt,rr,hh)+qbypassres(oo,tt,(rr-
1),hh)-qbypassres(oo,tt,rr,hh)-qendres(oo,tt,rr) =0;
        end-do

        !Calculates the equivalent energy of the water in the reservoar at the end of the
        !period in MWh
        forall(rr in Reservoirs) do
            restEndResEnergy(rr) := pendres(oo,tt,rr) =
qendres(oo,tt,rr)*TotHeadUnit(rr,1)*1000*9.81*EffBPUnit(rr,1)/(3600*1000000);
        end-do

        !Cuts of PQ-curves
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps) |MaxDiscUnit(ss,uu) > 0) do
            restPQCurve1(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(1)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(1))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve2(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(2)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(2))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve3(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(3)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(3))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve4(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(4)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(4))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve5(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(5)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(5))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
            restPQCurve6(ss,uu,hh) := punit(oo,tt,ss,uu,hh) <=
(Slope(6)*qprodunit(oo,tt,ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(6))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
        end-do

        !Max bypass
        forall (rr in Reservoirs,  hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxBypassRes(rr,hh) := qbypassres(oo,tt,rr,hh)<= MaxBypass(rr);
        end-do

        !Min bypass
        forall (rr in Reservoirs, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMinBypassRes(rr,hh) := qbypassres(oo,tt,rr,hh)>= MinBypass(rr);
        end-do

        !Max resevoir volume
        forall (rr in Reservoirs, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxVolumeRes(rr,hh) := qinres(oo,tt,rr,hh) <= MaxReservoirVolume(rr);
        end-do

        !Min reservoir volume
        forall (rr in Reservoirs, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMinVolumeRes(rr,hh) := qinres(oo,tt,rr,hh) >= MinReservoirVolume(rr);
        end-do

        !Max resevoir volume at the end of the period
        forall (rr in Reservoirs) do
            restMaxVolumeEndRes(rr) := qendres(oo,tt,rr) <= MaxReservoirVolume(rr);
        end-do

        !Min resevoir volume at the end of the period
        forall (rr in Reservoirs) do
            restMinVolumeEndRes(rr) := qendres(oo,tt,rr) >= MinReservoirVolume(rr);
        end-do

        !Max Discharge
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxDiscUnit(ss,uu,hh) := qprodunit(oo,tt,ss,uu,hh) <= MaxDiscUnit(ss,uu);
        end-do

        !Min Discharge
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
        restMinDiscUnit(ss,uu,hh) := qprodunit(oo,tt,ss,uu,hh) >= MinDiscUnit(ss,uu);
        end-do

        !Total production in the spot market from unit level to system level
        forall (hh in 24*tt-23 .. 24*getsize(Steps)) do
            restTotSpotProd(hh) := sum(ss in Stations, uu in Units)(punit(oo,tt,ss,uu,hh)) -
ptotspot(oo,tt,hh) = 0 ;
        end-do
        !Max production on each unit
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxProdUnit(ss,uu,hh) := punit(oo,tt,ss,uu,hh)-
MaxProduction(ss,uu)*runindunit(oo,tt,ss,uu,hh) <= 0;
        end-do

        !Min production on each unit
        forall (ss in Stations, uu in Units, hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMinProdUnit(ss,uu,hh) := punit(oo,tt,ss,uu,hh)-
MinProduction(ss,uu)*runindunit(oo,tt,ss,uu,hh) >= 0;
        end-do

        !Reserving upward regulation capacity
        forall(hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMaxProdObligation(hh) := ptotspot(oo,tt,hh)+SymRegObligation(oo)-sum(ss in Stations, uu in
Units)MaxProduction(ss,uu)*runindunit(oo,tt,ss,uu,hh) <= 0;
        end-do

        !Reserving downward regulation capacity
        forall(hh in 24*tt-23 .. 24*getsize(Steps)) do
            restMinProdObligation(hh) := ptotspot(oo,tt,hh)-SymRegObligation(oo)-sum(ss in Stations, uu in
Units)MinProduction(ss,uu)*runindunit(oo,tt,ss,uu,hh) >= 0;
        end-do

        !Running optimizations for all steps and obligations
        writeln("Optimizing Step: ",tt,", ObligationIndex: ",oo);
        OptimizationStartTime:= datetime(SYS_NOW);
        maximize(maxz);

        StepwiseObjectiveValue(oo,tt):= getobjval;
        OptimizationEndTime:= datetime(SYS_NOW);

        !Check for optimiality / feasibility
        if  (getprobstat = XPRS_INF) then
            writeln("Infeasible solution");
            writeln("Step: ",tt,", ObligationIndex: ",oo);
        end-if
        if (not getprobstat = XPRS_OPT) then
            writeln("Not solved to optimality");
            writeln("Step: ",tt,", ObligationIndex: ",oo);
        else
            writeln("Step: ",tt,", ObligationIndex: ",oo," is optimized successfully");
            writeln("Elapsed time of optimization: ",OptimizationEndTime-OptimizationStartTime," seconds");

        end-if

        !Saving the realized  solutions from each step to show the actual behavior throughout the step
        forall(rr in Reservoirs, ss in Stations, uu in Units, hh in 24*tt-23 .. 24*tt) do

            RealizedPTotSpot(oo,hh):= getsol(ptotspot(oo,tt,hh));
            RealizedQInRes(oo,rr,hh):= getsol(qinres(oo,tt,rr,hh));
            RealizedRunIndUnit(oo,ss,uu,hh):= getsol(runindunit(oo,tt,ss,uu,hh));
            RealizedPUnit(oo,ss,uu,hh):= getsol(punit(oo,tt,ss,uu,hh));
            RealizedQBypassRes(oo,rr,hh):= getsol(qbypassres(oo,tt,rr,hh));
            RealizedQFloodRes(oo,rr,hh):= getsol(qfloodres(oo,tt,rr,hh));
            RealizedQProdUnit(oo,ss,uu,hh):= getsol(qprodunit(oo,tt,ss,uu,hh));
            RealizedStartIndUnit(oo,ss,uu,hh):= getsol(startindunit(oo,tt,ss,uu,hh));
            RealizedPEndRes(oo,rr):= getsol(pendres(oo,tt,rr));
            RealizedQEndRes(oo,rr):= getsol(qendres(oo,tt,rr));
            if (getsol(qprodunit(oo,tt,ss,uu,hh)) >0) then
                RealizedTurbEff(oo,ss,uu,hh):=
(1000000*getsol(punit(oo,tt,ss,uu,hh))/(getsol(qprodunit(oo,tt,ss,uu,hh))*9.81*1000*HeadUnit(ss,uu)));
            end-if
        end-do

        !Saving the Stepwise solutions from each step to show the Stepwise behavior throughout the step
        forall (rr in Reservoirs, ss in Stations, uu in Units, hh in Hours) do
            StepwisePTotSpot(oo,tt,hh):= getsol(ptotspot(oo,tt,hh));
            StepwiseQInRes(oo,tt,rr,hh):= getsol(qinres(oo,tt,rr,hh));
            StepwiseRunIndUnit(oo,tt,ss,uu,hh):= getsol(runindunit(oo,tt,ss,uu,hh));
            StepwisePUnit(oo,tt,ss,uu,hh):= getsol(punit(oo,tt,ss,uu,hh));
            StepwiseQBypassRes(oo,tt,rr,hh):= getsol(qbypassres(oo,tt,rr,hh));
            StepwiseQFloodRes(oo,tt,rr,hh):= getsol(qfloodres(oo,tt,rr,hh));
            StepwiseQProdUnit(oo,tt,ss,uu,hh):= getsol(qprodunit(oo,tt,ss,uu,hh));
            StepwiseStartIndUnit(oo,tt,ss,uu,hh):= getsol(startindunit(oo,tt,ss,uu,hh));
            StepwisePEndRes(oo,tt,rr):= getsol(pendres(oo,tt,rr));
            StepwiseQEndRes(oo,tt,rr):= getsol(qendres(oo,tt,rr));
        end-do

    end-do !Ending obligations

    if (tt = getsize(Steps)) then

        ! Emulating the market for the last step
        Dice(tt):= random;

        DiceAdjusted(tt):= Dice(tt)*RealizedProbability(tt);


        forall(ee in Scenarios | StepPredecessorDynamic(ee,tt) = RealizedPredecessor(tt)) do
                if (DiceAdjusted(tt) < sum(pp in
RealizedPredDynCoord(tt)..ee)(StepProbabilityDynamic(pp,tt))) then
                    RealizedProbDynCoord(tt):=ee;
                forall (hh in (24*(tt-1) .. 24*tt)) do
                        ActualPrice(mm,hh):= ScenarioSpotPrice(ee,hh);

                    end-do
                end-if
        end-do

        !Storing the price optimized upon for the last step
        forall (hh in (24*tt-23) .. 24*getsize(Steps)) do
            ModelPrice(mm,hh):= sum(ee in Scenarios | StepPredecessorFull(ee,tt) = RealizedPredecessor(tt))
            ((ScenarioSpotPrice(ee,hh)*EndScenarioProbability(ee))
            /(sum(eee in Scenarios | StepPredecessorFull(eee,tt) =
RealizedPredecessor(tt))(EndScenarioProbability(eee))));
        end-do

    end-if




end-do !Ending steps



!Calculating and storing the actual realized objective value for each obligation
forall (oo in ObligationIndex) do
    ActualObjectiveValue(mm,oo):= sum(hh in Hours)(ActualPrice(mm,hh)*RealizedPTotSpot(oo,hh))
    -sum(ss in Stations, uu in Units, hh in Hours)(StartUpCost(ss,uu)*RealizedStartIndUnit(oo,ss,uu,hh))
    +sum(rr in Reservoirs)(WaterValue(rr)*RealizedPEndRes(oo,rr));
end-do

!Calculating and storing the objective value optimized upon for each obligation
forall (oo in ObligationIndex) do
    ModelObjectiveValue(mm,oo):= sum(hh in Hours)(ModelPrice(mm,hh)*RealizedPTotSpot(oo,hh))
    -sum(ss in Stations, uu in Units, hh in Hours)(StartUpCost(ss,uu)*RealizedStartIndUnit(oo,ss,uu,hh))
    +sum(rr in Reservoirs)(WaterValue(rr)*RealizedPEndRes(oo,rr));
end-do

!Calculating and storing the actual realized marginal price for attending regulation for each obligation
forall (oo in 2.. getsize(ObligationIndex)) do
    ActualBreakEvenRegPrice(mm,oo):= (ActualObjectiveValue(mm,1)-
ActualObjectiveValue(mm,oo))/(getsize(Hours)*SymRegObligation(oo));
end-do

!Calculating and storing the break-even price optimized upon for attending regulation for each obligation
forall (oo in 2.. getsize(ObligationIndex)) do
    ModelBreakEvenRegPrice(mm,oo):= (ModelObjectiveValue(mm,1)-
ModelObjectiveValue(mm,oo))/(getsize(Hours)*SymRegObligation(oo));
end-do

!Calculating and storing the actual break-even price optimized upon for attending regulation for each
obligation
forall (oo in 2.. getsize(ObligationIndex)) do
    ActualMarginalRegCost(mm,oo):= (ActualObjectiveValue(mm,oo-1)-
ActualObjectiveValue(mm,oo))/(getsize(Hours));
end-do

!Calculating and storing the actual realized marginal cost for attending regulation for each obligation
forall (oo in 2.. getsize(ObligationIndex)) do
    ModelMarginalRegCost(mm,oo):= (ModelObjectiveValue(mm,oo-1)-ModelObjectiveValue(mm,oo))/(getsize(Hours));
end-do

forall (oo in ObligationIndex, ss in Stations, hh in Hours) do
    RealizedRunningUnits(oo,ss,hh):= (sum(uu in Units)RealizedRunIndUnit(oo,ss,uu,hh));
end-do

!Calculating the objective values seen (and realized) for each Stepwise step
forall (oo in ObligationIndex, tt in Steps) do
        if (tt=1) then
            StepwiseModelTotObjVal(oo,tt):=StepwiseObjectiveValue(oo,tt);
            StepwiseActualTotObjVal(oo,tt):=StepwiseObjectiveValue(oo,tt);
        else
            StepwiseModelTotObjVal(oo,tt):= sum(hh in 1..24*(tt-
1))(ModelPrice(mm,hh)*RealizedPTotSpot(oo,hh)-sum(ss in Stations, uu in
Units)(StartUpCost(ss,uu)*RealizedStartIndUnit(oo,ss,uu,hh)))+StepwiseObjectiveValue(oo,tt);
            StepwiseActualTotObjVal(oo,tt):= sum(hh in 1..24*(tt-
1))(ActualPrice(mm,hh)*RealizedPTotSpot(oo,hh)-sum(ss in Stations, uu in
Units)(StartUpCost(ss,uu)*RealizedStartIndUnit(oo,ss,uu,hh)))+StepwiseObjectiveValue(oo,tt);
        end-if
end-do

!Calculating the needed regulation price for each Stepwise step
forall (oo in 2.. getsize(ObligationIndex), ObligationIndex, tt in Steps) do
    StepwiseModelBreakEvenRegPrice(oo,tt):= (StepwiseModelTotObjVal(1,tt)-
StepwiseModelTotObjVal(oo,tt))/(SymRegObligation(oo)*getsize(Hours));
    StepwiseActualBreakEvenRegPrice(oo,tt):= (StepwiseActualTotObjVal(1,tt)-
StepwiseActualTotObjVal(oo,tt))/(SymRegObligation(oo)*getsize(Hours));
end-do

!Calculating the marginal regulation cost for each Stepwise step
forall (oo in 2.. getsize(ObligationIndex), ObligationIndex, tt in Steps) do
    StepwiseModelMarginalRegCost(oo,tt):= (StepwiseModelTotObjVal(oo-1,tt)-
StepwiseModelTotObjVal(oo,tt))/getsize(Hours);
```

```
    StepwiseActualMarginalRegCost(oo,tt):= (StepwiseActualTotObjVal(oo-1,tt)-
StepwiseActualTotObjVal(oo,tt))/getsize(Hours);
end-do

end-do !Ending ModelIndex

OptimizationStartTime:= datetime(SYS_NOW);
!Writes all output data to Excel file
writeln("Writing output data to Excel file");

    !Writing out results
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\output_mm.xlsx');
    SQLexecute("insert into XACTUALPRICE (MODELINDEX,HOURS,ACTUALPRICE) values (?,?,?)",ActualPrice);
    SQLexecute("insert into XMODELPRICE (MODELINDEX,HOURS,MODELPRICE) values (?,?,?)",ModelPrice);
    SQLexecute("insert into XACTUALOBJECTIVEVALUE (MODELINDEX,OBLIGATIONINDEX,ACTUALOBJECTIVEVALUE) values
(?,?,?)",ActualObjectiveValue);
    SQLexecute("insert into XMODELOBJECTIVEVALUE (MODELINDEX,OBLIGATIONINDEX,MODELOBJECTIVEVALUE) values
(?,?,?)",ModelObjectiveValue);
    SQLexecute("insert into XACTUALBREAKEVENREGPRICE (MODELINDEX,OBLIGATIONINDEX,ACTUALBREAKEVENREGPRICE)
values (?,?,?)",ActualBreakEvenRegPrice);
    SQLexecute("insert into XMODELBREAKEVENREGPRICE (MODELINDEX,OBLIGATIONINDEX,MODELBREAKEVENREGPRICE)
values (?,?,?)",ModelBreakEvenRegPrice);
    SQLexecute("insert into XACTUALMARGINALREGCOST (MODELINDEX,OBLIGATIONINDEX,ACTUALMARGINALREGCOST) values
(?,?,?)",ActualMarginalRegCost);
    SQLexecute("insert into XMODELMARGINALREGCOST (MODELINDEX,OBLIGATIONINDEX,MODELMARGINALREGCOST) values
(?,?,?)",ModelMarginalRegCost);

SQLdisconnect;

OptimizationEndTime:= datetime(SYS_NOW);
writeln("Successfully written data to Excel file");
writeln("Time used to write output data: ",(OptimizationEndTime-OptimizationStartTime)," seconds");
writeln("Model run complete");
ModelEndTime:= datetime(SYS_NOW);
writeln("Total elapsed time of model: ",(ModelEndTime-ModelStartTime)," seconds");
end-model;
```

E.3 The deterministic model

```
model deterministic
options explterm
options noimplicit
uses "mmxprs";
uses "mmodbc"; !gain access to SQL for Excel input

!Declaration of sets
declarations
    Hours:          set of integer;
    Days:           set of integer;
    Units:          set of integer;
    Stations:       set of integer;
    Reservoirs:     set of integer;
    Index:          set of integer;
    ObligationIndex:set of integer;
end-declarations

!Reading input data from excel
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\input.xlsx');
    SQLexecute("select HOURS from HH", Hours);
    SQLexecute("select DAYS from DD", Days);
    SQLexecute("select UNITS from UU", Units);
    SQLexecute("select STATIONS from SS", Stations);
    SQLexecute("select RESERVOIRS from RR", Reservoirs);
    SQLexecute("select INDEX from II", Index);
    SQLexecute("select OBLIGATIONINDEX from OO", ObligationIndex);
SQLdisconnect;


!Declaration of parameters
declarations
    !Spotpris
    SpotPrice:                  dynamic array(Hours)            of real;
    !Symmetrical regulation obligation
    SymRegObligation:           dynamic array(ObligationIndex)  of real;
   !Water value
    WaterValue:                 dynamic array(Reservoirs)       of real;
    !Intial reservoar volume m3
    InitialWaterVolume:         dynamic array(Reservoirs)       of real;
    !Inflow
    InflowRes:                  dynamic array(Reservoirs,Hours) of real;
    !Max production on each unit
    MaxProduction:              dynamic array(Stations,Units)   of real;
    !Min production on each unit
    MinProduction:              dynamic array(Stations,Units)   of real;
    !Max bypass m3/s
    MaxBypass:                  dynamic array(Reservoirs)       of real;
    !Min bypass m3/s
    MinBypass:                  dynamic array(Reservoirs)       of real;
    !Max reservoar volume m3
    MaxReservoirVolume:         dynamic array(Reservoirs)       of real;
    !Min reservoar volume m3
    MinReservoirVolume:         dynamic array(Reservoirs)       of real;
    !Max regulation of reservoar per hour m3/h
    MaxRegulatingPerHour:       dynamic array(Reservoirs)       of real;
     !Start-up cost at unit u at station s
    StartUpCost:                array(Stations,Units)           of real;
    !Max ramping at unit u at station s
    MaxRamping:                 array(Stations,Units)           of real;
    !Slopes of the oiecewize linear PQ-curve
    Slope:                      array(Index)                    of real;
    !Interceptions of the piecewize linear PQ-curve
    Intercept:                  array(Index)                    of real;
    !Head for each unit
    HeadUnit:                   array(Stations,Units)           of real;
    !System head for each unit
    TotHeadUnit:                array(Stations,Units)           of real;
    !Max discharge through unit u at station s
    MaxDiscUnit:                array(Stations,Units)           of real;
    !Min discharge through unit u at station s
    MinDiscUnit:                dynamic array(Stations,Units)   of real;
    !Best point efficiency for each unit
    EffBPUnit:                  array(Stations,Units)           of real;


   ! Defining output parameters(variables)

    OutObjectiveValue:  array(ObligationIndex)                          of real;
    OutMarginalRegPrice:array(ObligationIndex)                          of real;
    OutHours:           array(Hours)                                    of real;
    OutSpotPrice:       array(Hours)                                    of real;
    OutPTotSpot:        array (ObligationIndex,Hours)                   of real;
    OutQInRes:          array(ObligationIndex,Reservoirs,Hours)         of real;
    OutInflowRes:       array(Reservoirs,Hours)                         of real;
    OutPUnit:           array(ObligationIndex,Stations,Units,Hours)     of real;
    OutWaterValue:      array(Reservoirs)                               of real;
    OutAvailableUpReg:  array(ObligationIndex,Hours)                    of real;
    OutAvailableDownReg:array(ObligationIndex,Hours)                    of real;
    OutQFloodRes:       array(ObligationIndex,Reservoirs,Hours)         of real;
    OutQBypassRes:      array(ObligationIndex,Reservoirs,Hours)         of real;
```

```
end-declarations

    !Reading input data from excel
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\input.xlsx');
    SQLexecute("select SPOTPRICE from HH", SpotPrice);
    SQLexecute("select SYMREGOBLIGATION from OO", SymRegObligation);
    SQLexecute("select WATERVALUE from RR", WaterValue);
    SQLexecute("select INITIALWATERVOLUME from RR", InitialWaterVolume);
    SQLexecute("select INFLOWRES from RRHH", InflowRes);
    SQLexecute("select MAXPRODUCTION from SSUU", MaxProduction);
    SQLexecute("select MINPRODUCTION from SSUU", MinProduction);
    SQLexecute("select MAXBYPASS from RR", MaxBypass);
    SQLexecute("select MINBYPASS from RR", MinBypass);
    SQLexecute("select MAXRESERVOIRVOLUME from RR", MaxReservoirVolume);
    SQLexecute("select MINRESERVOIRVOLUME from RR", MinReservoirVolume);
    SQLexecute("select MAXREGULATINGPERHOUR from RR", MaxRegulatingPerHour);
    SQLexecute("select STARTUPCOST from SSUU", StartUpCost);
    SQLexecute("select MAXRAMPING from SSUU", MaxRamping);
    SQLexecute("select HEADUNIT from SSUU", HeadUnit);
    SQLexecute("select MAXDISCUNIT from SSUU", MaxDiscUnit);
    SQLexecute("select MINDISCUNIT from SSUU", MinDiscUnit);
    SQLexecute("select TOTHEADUNIT from SSUU", TotHeadUnit);
    SQLexecute("select EFFBPUNIT from SSUU", EffBPUnit);
    !PQ curve
    SQLexecute("select SLOPE from II", Slope);
    SQLexecute("select INTERCEPT from II", Intercept);
    SQLdisconnect;


!Declaration of variables

declarations

    !Total spot volume in hour h
    ptotspot:       array(Hours)                of mpvar;
    !Reservoar volume at the end of the periode
    qendres:        array(Reservoirs)           of mpvar;
    !Energy in the reservoar at the end of the periode
    pendres:        array(Reservoirs)           of mpvar;
    !Discharge thruogh unit u at station s in hour h
    qprodunit:      array(Stations,Units,Hours) of mpvar;
    !Flood from reservoar r in hour h
    qfloodres:      array(Reservoirs,Hours)     of mpvar;
    !Bypass from reservoar r in hour h
    qbypassres:     array(Reservoirs,Hours)     of mpvar;
    !reservoar volume in reservoar r in hour h
    qinres:         array(Reservoirs,Hours)     of mpvar;
    !Spot production from unit u at station s in hour h
    punit:          array(Stations,Units,Hours) of mpvar;
    !start indicator
    startindunit:   array(Stations,Units,Hours) of mpvar;
    !Run indicator, 1 if unit u runs in hour h
    runindunit:     array(Stations,Units,Hours) of mpvar;



    ! Declaration of objective function
maxz:   linctr;


!Declaration of constranits

! Sets initial reservoir volume in qinres for all reservoirs
restInitWaterBalance:   array(Reservoirs)           of linctr;
!Water balance upper reservoir
restWaterBalance1:      array(Reservoirs,Hours)     of linctr;
!Water balance other reservoirs
restWaterBalance2:      array(Reservoirs,Hours)     of linctr;
!Total production i elspot from unit to system level
restTotSpotProd:        array(Hours)                of linctr;
!Max bypass
restMaxBypassRes:       array(Reservoirs,Hours)     of linctr;
!Min bypass
restMinBypassRes:       array(Reservoirs,Hours)     of linctr;
!Max reservoar volume
restMaxVolumeRes:       array(Reservoirs,Hours)     of linctr;
!Min reservoar volume
restMinVolumeRes:       array(Reservoirs,Hours)     of linctr;
!Max reservoar regulation m3 equation 1
restMaxRegRes1:         array(Reservoirs,Hours)     of linctr;
!Max reservoar regulation equation 2
restMaxRegRes2:         array(Reservoirs,Hours)     of linctr;
!Start-up cost
restStartupCostUnit:    array(Stations,Units,Hours  of linctr;
!Max ramping equation 1
restMaxRampUnit1:       array(Stations,Units)       of linctr;
!Max ramping equation 2
restMaxRampUnit2:       array(Stations,Units)       of linctr;
!Max production on unit u at station s in hour h
restMaxProdUnit:        array(Stations,Units,Hours) of linctr;
!Min production on unit u at station s in hour h
restMinProdUnit:        array(Stations,Units,Hours) of linctr;
!Production indicator for each unit in each hour
restProdIndUnit:        array(Stations,Units,Hours) of linctr;
!Max reservoar volume at the end of the periode
```

```
restMaxVolumeEndRes:    array(Reservoirs)           of linctr;
!Min reservoar volume at the end of the periode
restMinVolumeEndRes:    array(Reservoirs)           of linctr;
!Calculates the equivalent energy of the water in the reservoar at the end of the
!periode in MWh
restEndResEnergy:       array(Reservoirs)           of linctr;
!Max discharge thruogh unit u at station s in hour h
restMaxDiscUnit:        array(Stations,Units,Hours) of linctr;
!Min discharge thruogh unit u at station s in hour h
restMinDiscUnit:        array(Stations,Units,Hours) of linctr;
!PQ-curve
restPQCurve1:           array(Stations,Units,Hours) of linctr;
!PQ-curve
restPQCurve2:           array(Stations,Units,Hours) of linctr;
!PQ-curve
restPQCurve3:           array(Stations,Units,Hours) of linctr;
!PQ-curve
restPQCurve4:           array(Stations,Units,Hours) of linctr;
!PQ-curve
restPQCurve5:           array(Stations,Units,Hours) of linctr;
!PQ-curve
restPQCurve6:           array(Stations,Units,Hours) of linctr;
!PQ-curve
restPQCurve7:           array(Stations,Units,Hours) of linctr;
!check if there is enoght rotating reserves available on system level for upward regulation
restMaxProdObligation:  array(Hours)                of linctr;
!check if there is enoght rotating reserves available on system level for downward regulation
restMinProdObligation:  array(Hours)                of linctr;

end-declarations

! Creates binary variable runindunit
forall (ss in Stations,uu in Units,hh in Hours) do
runindunit(ss,uu,hh) is_binary;
end-do

! Object function

maxz := sum(hh in Hours)(SpotPrice(hh)*ptotspot(hh))
+ sum(rr in Reservoirs)(WaterValue(rr)*pendres(rr))
- sum(ss in Stations, uu in Units, hh in Hours)(StartUpCost(ss,uu)*startindunit(ss,uu,hh));



!Constranits

! Sets initial reservoir volume in qinres for all reservoirs
forall (rr in Reservoirs) do
    restInitWaterBalance(rr) := qinres(rr,1) = InitialWaterVolume(rr);
end-do

 !Water balance upper reservoir
forall (rr in 1..1, hh in 1..(getsize(Hours)-1)) do
    restWaterBalance1(rr,hh) := qinres(rr,hh)+InflowRes(rr,hh)
    -sum(ss in 1..1, uu in Units)(qprodunit(ss,uu,hh)*3600)-qfloodres(rr,hh)-qbypassres(rr,hh)-
qinres(rr,(hh+1)) =0;
end-do

!Water balance other reservoirs other hours
forall (rr in 2.. getsize(Reservoirs), hh in 1..(getsize(Hours)-1)) do
    restWaterBalance2(rr,hh) := qinres(rr,hh)+InflowRes(rr,hh)+ sum(uu in Units)(qprodunit((rr-
1),uu,hh)*3600)
    -sum(uu in Units)(qprodunit(rr,uu,hh)*3600)-qfloodres(rr,hh)+qbypassres((rr-1),hh)-qbypassres(rr,hh)-
qinres(rr,(hh+1)) =0;
end-do

!Water balance last hour upper reservoar
forall (rr = 1, hh = getsize(Hours)) do
restWaterBalance1(rr,hh) := qinres(rr,hh)+InflowRes(rr,hh)
-sum(ss in 1..1, uu in Units)(qprodunit(ss,uu,hh)*3600)-qfloodres(rr,hh)-qbypassres(rr,hh)-qendres(rr) =0;
end-do

!Water balance other reservoirs last hour
forall (rr in 2.. getsize(Reservoirs), hh = getsize(Hours)) do
    restWaterBalance2(rr,hh) := qinres(rr,hh)+InflowRes(rr,hh)+ sum(uu in Units)(qprodunit((rr-
1),uu,hh)*3600)
    -sum(uu in Units)(qprodunit(rr,uu,hh)*3600)-qfloodres(rr,hh)+qbypassres((rr-1),hh)-qbypassres(rr,hh)-
qendres(rr) =0;
end-do

!Calculates the equivalent energy of the water in the reservoar at the end of the
!periode in MWh
forall(rr in Reservoirs) do
    restEndResEnergy(rr) := pendres(rr) =
qendres(rr)*TotHeadUnit(rr,1)*1000*9.81*EffBPUnit(rr,1)/(3600*1000000);
end-do

!Cuts of PQ-curves
forall (ss in Stations, uu in Units, hh in Hours |MaxDiscUnit(ss,uu) > 0) do
    restPQCurve1(ss,uu,hh) := punit(ss,uu,hh) <=
(Slope(1)*qprodunit(ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(1))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
    restPQCurve2(ss,uu,hh) := punit(ss,uu,hh) <=
(Slope(2)*qprodunit(ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(2))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
    restPQCurve3(ss,uu,hh) := punit(ss,uu,hh) <=
(Slope(3)*qprodunit(ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(3))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
```

```
    restPQCurve4(ss,uu,hh) := punit(ss,uu,hh) <=
(Slope(4)*qprodunit(ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(4))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
    restPQCurve5(ss,uu,hh) := punit(ss,uu,hh) <=
(Slope(5)*qprodunit(ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(5))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
    restPQCurve6(ss,uu,hh) := punit(ss,uu,hh) <=
(Slope(6)*qprodunit(ss,uu,hh)/MaxDiscUnit(ss,uu)+Intercept(6))*MaxDiscUnit(ss,uu)*HeadUnit(ss,uu);
end-do

!Max bypass
forall (rr in Reservoirs,  hh in Hours) do
    restMaxBypassRes(rr,hh) := qbypassres(rr,hh)<= MaxBypass(rr);
end-do

!Min bypass
forall (rr in Reservoirs, hh in Hours) do
    restMinBypassRes(rr,hh) := qbypassres(rr,hh)>= MinBypass(rr);
end-do

!Max resevoir volume
forall (rr in Reservoirs, hh in Hours) do
    restMaxVolumeRes(rr,hh) := qinres(rr,hh) <= MaxReservoirVolume(rr);
end-do

!Min reservoir volume
forall (rr in Reservoirs, hh in Hours) do
    restMinVolumeRes(rr,hh) := qinres(rr,hh) >= MinReservoirVolume(rr);
end-do

!Max resevoir volume at the end of the periode
forall (rr in Reservoirs) do
    restMaxVolumeEndRes(rr) := qendres(rr) <= MaxReservoirVolume(rr);
end-do

!Min resevoir volume at the end of the periode
forall (rr in Reservoirs) do
    restMinVolumeEndRes(rr) := qendres(rr) >= MinReservoirVolume(rr);
end-do

! Max Discharge
forall (ss in Stations, uu in Units, hh in Hours) do
    restMaxDiscUnit(ss,uu,hh) := qprodunit(ss,uu,hh) <= MaxDiscUnit(ss,uu);
end-do

!Min Discharge
forall (ss in Stations, uu in Units, hh in Hours) do
    restMinDiscUnit(ss,uu,hh) := qprodunit(ss,uu,hh) >= MinDiscUnit(ss,uu);
end-do

!Max ramping equation 1
forall (ss in Stations, uu in Units, hh in 2..getsize(Hours)) do
    restMaxRampUnit1(ss,uu) := punit(ss,uu,hh) - punit(ss,uu,(hh-1)) <= MaxRamping(ss,uu);
end-do

!Max ramping equation 1
forall (ss in Stations, uu in Units, hh in 1..1) do
    restMaxRampUnit1(ss,uu) := punit(ss,uu,(hh+1)) - punit(ss,uu,hh) <= MaxRamping(ss,uu);
end-do

!Max ramping equation 2
forall (ss in Stations, uu in Units, hh in 2..getsize(Hours)) do
    restMaxRampUnit2(ss,uu) := punit(ss,uu,(hh-1)) - punit(ss,uu,hh) >= -MaxRamping(ss,uu);
end-do

!Max ramping equation 2
forall (ss in Stations, uu in Units, hh in 1..1) do
    restMaxRampUnit2(ss,uu) := punit(ss,uu,hh) - punit(ss,uu,(hh+1)) >= -MaxRamping(ss,uu);
end-do

!Max reservoir regulation equation 1
forall (rr in Reservoirs, hh in 1..(getsize(Hours)-1)) do
    restMaxRegRes1(rr,hh) := qinres(rr,(hh+1))-qinres(rr,hh) <= MaxRegulatingPerHour(rr);
end-do

!Max reservoir regulation equation 2
forall (rr in Reservoirs, hh in 1..(getsize(Hours)-1)) do
    restMaxRegRes2(rr,hh) := qinres(rr,hh)-qinres(rr,(hh+1)) <= MaxRegulatingPerHour(rr);
end-do

!Start-up costs
forall (ss in Stations, uu in Units, hh in 2..getsize(Hours)) do
    restStartupCostUnit(ss,uu,hh) := startindunit(ss,uu,hh) - (runindunit(ss,uu,hh) - runindunit(ss,uu,(hh-
1))) >= 0 ;
end-do

!Start-up costs
forall (ss in Stations, uu in Units, hh in 1..1) do
    restStartupCostUnit(ss,uu,hh) := startindunit(ss,uu,hh) - (runindunit(ss,uu,(hh+1)) -
runindunit(ss,uu,hh)) >= 0 ;
end-do

!Total production in elspot from unit to system level
forall (hh in Hours) do
    restTotSpotProd(hh) := sum(ss in Stations, uu in Units)(punit(ss,uu,hh)) - ptotspot(hh) = 0 ;
end-do
```

```
!Max production on each unit
forall (ss in Stations, uu in Units, hh in Hours) do
    restMaxProdUnit(ss,uu,hh) := punit(ss,uu,hh)-MaxProduction(ss,uu)*runindunit(ss,uu,hh) <= 0;
end-do

!Min production on each unit
forall (ss in Stations, uu in Units, hh in Hours) do
    restMinProdUnit(ss,uu,hh) := punit(ss,uu,hh)-MinProduction(ss,uu)*runindunit(ss,uu,hh) >= 0;
end-do


!! Run the optimization command for different levels of obligation in the regulation market.

forall(oo in ObligationIndex)do


!check if there is enoght rotating reserves available on system level for upward regulation
forall(hh in Hours) do
    restMaxProdObligation(hh) := ptotspot(hh)+SymRegObligation(oo)-sum(ss in Stations, uu in
Units)MaxProduction(ss,uu)*runindunit(ss,uu,hh) <= 0;
end-do

!check if there is enoght rotating reserves available on system level for downward regulation
forall(hh in Hours) do
    restMinProdObligation(hh) := ptotspot(hh)-SymRegObligation(oo)-sum(ss in Stations, uu in
Units)MinProduction(ss,uu)*runindunit(ss,uu,hh) >= 0;
end-do

!Run the optimization command
maximize(maxz);

!Save the solution from the objective function to the parameter OutObjectiveValue(oo)
OutObjectiveValue(oo):=getobjval;

forall (rr in Reservoirs, ss in Stations, uu in Units, hh in Hours) do
    OutQInRes(oo,rr,hh):= getsol(qinres(rr,hh));
    OutInflowRes(rr,hh):= InflowRes(rr,hh);
    OutQBypassRes(oo,rr,hh):= getsol(qbypassres(rr,hh));
    OutQFloodRes(oo,rr,hh):= getsol(qfloodres(rr,hh));
    OutPTotSpot(oo,hh) := getsol(ptotspot(hh));
    OutPUnit(oo,ss,uu,hh):= getsol(punit(ss,uu,hh));
    OutHours(hh):= hh;
    OutSpotPrice(hh):= SpotPrice(hh);
    OutWaterValue(rr):= WaterValue(rr);
    OutAvailableUpReg(oo,hh):= sum(ss in Stations, uu in
Units)MaxProduction(ss,uu)*getsol(runindunit(ss,uu,hh)) - getsol(ptotspot(hh));
    OutAvailableDownReg(oo,hh):= getsol(ptotspot(hh))-sum(ss in Stations, uu in
Units)MinProduction(ss,uu)*getsol(runindunit(ss,uu,hh));
end-do

end-do  !Ending oo

!Calculate cost/required price in €/MW available reglation capacity
forall (oo in 2..(getsize(ObligationIndex))) do
    OutMarginalRegPrice(oo):=(OutObjectiveValue(1)-
OutObjectiveValue(oo))/(SymRegObligation(oo)*getsize(Hours));
end-d

!Writes output data to Excel file
SQLconnect('Driver={Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm, *xlsb)};DBQ=.\\output.xlsx');
    SQLexecute("insert into XHOURS (HOURS) values (?)",OutHours);
    SQLexecute("insert into XSPOTPRICE (SPOTPRICE) values (?)",OutSpotPrice);
    SQLexecute("insert into XMARGINALREGPRICE (MARGINALREGPRICE) values (?)",OutMarginalRegPrice);
    SQLexecute("insert into XSYMREGOBLIGATION (SYMREGOBLIGATION) values (?)",SymRegObligation);
    SQLexecute("insert into XOBJECTIVEVALUE (OBJECTIVEVALUE) values (?)",OutObjectiveValue);
    SQLexecute("insert into XQINRES (SYMREGOBLIGATION,RESERVOIRS,HOURS,QINRES) values (?,?,?,?)",OutQInRes);
    SQLexecute("insert into XQBYPASSRES (SYMREGOBLIGATION,RESERVOIRS,HOURS,QBYPASSRES) values
(?,?,?,?)",OutQBypassRes);
    SQLexecute("insert into XQFLOODRES (SYMREGOBLIGATION,RESERVOIRS,HOURS,QFLOODRES) values
(?,?,?,?)",OutQFloodRes);
    SQLexecute("insert into XPTOTSPOT (SYMREGOBLIGATION,HOURS,PTOTSPOT) values (?,?,?)",OutPTotSpot);
    SQLexecute("insert into XPUNIT (SYMREGOBLIGATION,STATIONS,UNITS,HOURS, PUNIT) values
(?,?,?,?,?)",OutPUnit);
SQLdisconnect;

end-model;
```

F Probability distribution testing

The tool *Minitab 17 Statistical Software* was used to investigate if any general mathematical probability distribution could describe the results from the Monte Carlo-simulations in a satisfying way. The probability distributions the program analyzed were:

- Normal
- Exponential
- 2-parameter Exponential
- Weibull
- 3-parameter Weibull
- Lognormal
- 3-parameter Lognormal
- Smallest Extreme values
- Largest Extreme values
- Gamma
- 3-parameter Gamma

Detailed output data from the software is presented in appendices F.1 - F.4.

F.1 Probability distribution testing 6 MW regulation obligation

Distribution ID Plot for C1 [6MW regulation obligation]

```
Descriptive Statistics

   N  N*     Mean    StDev   Median  Minimum  Maximum  Skewness  Kurtosis
1000   0  17.1155  7.27321  14.9673  8.03411  51.8352   1.55797   3.39438


Box-Cox transformation: λ = -0.678532


Goodness of Fit Test

Distribution                  AD       P  LRT P
Normal                    34.980  <0.005
Box-Cox Transformation     5.693  <0.005
Lognormal                 11.546  <0.005
3-Parameter Lognormal      3.005       *  0.000
Exponential              182.803  <0.003
2-Parameter Exponential   23.404  <0.010  0.000
Weibull                   28.278  <0.010
3-Parameter Weibull        4.282  <0.005  0.000
Smallest Extreme Value    78.681  <0.010
Largest Extreme Value     14.387  <0.010
Gamma                     17.489  <0.005
3-Parameter Gamma          3.219       *  0.000
Logistic                  27.558  <0.005
Loglogistic               11.709  <0.005
3-Parameter Loglogistic    4.192       *  0.000


ML Estimates of Distribution Parameters

Distribution             Location    Shape     Scale  Threshold
Normal*                  17.11553            7.27321
Box-Cox Transformation*   0.15814            0.03797
Lognormal*                2.76411            0.37784
3-Parameter Lognormal     2.05829            0.72151    7.08076
Exponential                                 17.11553
2-Parameter Exponential                      9.09051    8.02502
Weibull                           2.44880   19.32507
3-Parameter Weibull               1.31072    9.89069    8.02704
Smallest Extreme Value   21.18796           10.12820
Largest Extreme Value    14.02241            4.87433
Gamma                             6.75215    2.53483
3-Parameter Gamma                 1.64411    5.53916    8.00857
Logistic                 16.13938            3.87027
Loglogistic               2.73889            0.22048
3-Parameter Loglogistic   1.96681            0.47034    7.71389

* Scale: Adjusted ML estimate
```

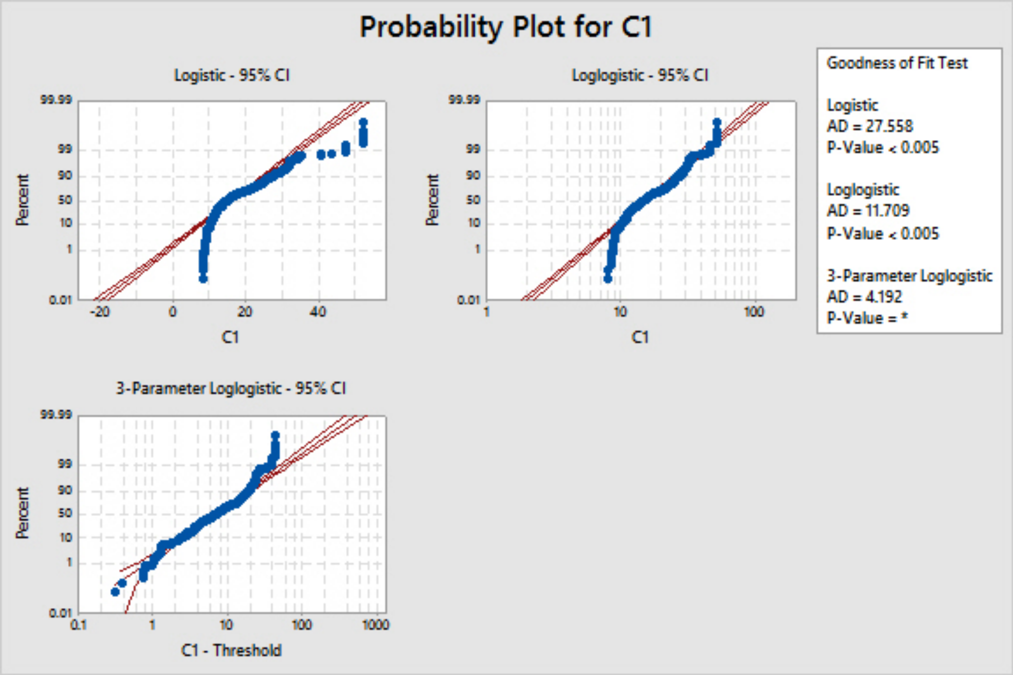


Figure XV: Goodness of Fit Test graph A for 6 MW regulation obligation

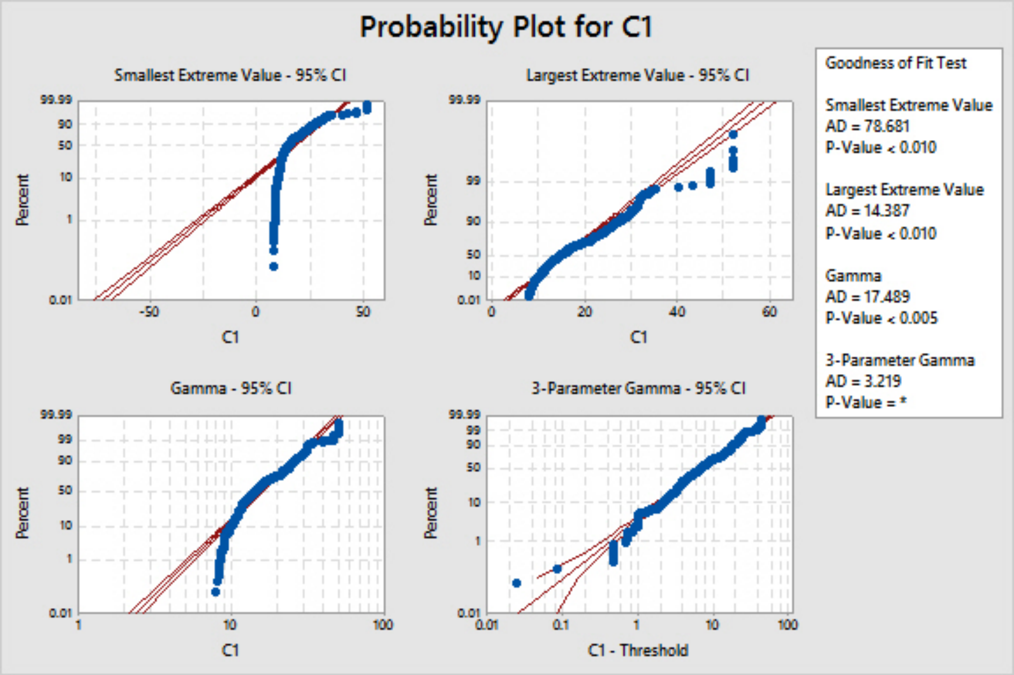


Figure XVI: Goodness of Fit Test graph B for 6 MW regulation obligation

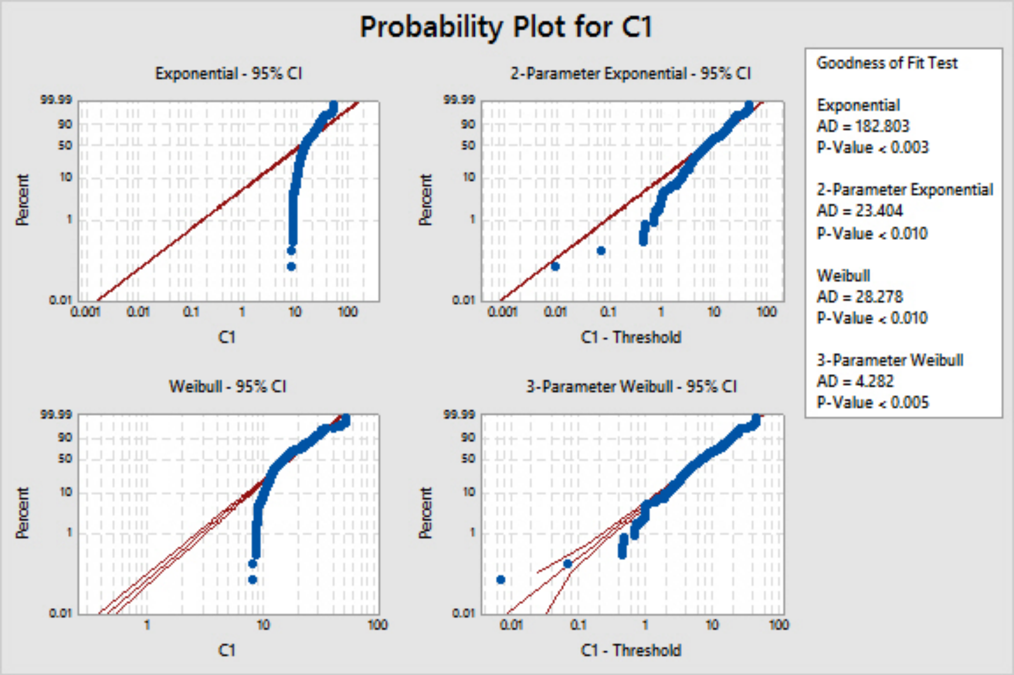


Figure XVII: Goodness of Fit Test graph C for 6 MW regulation obligation

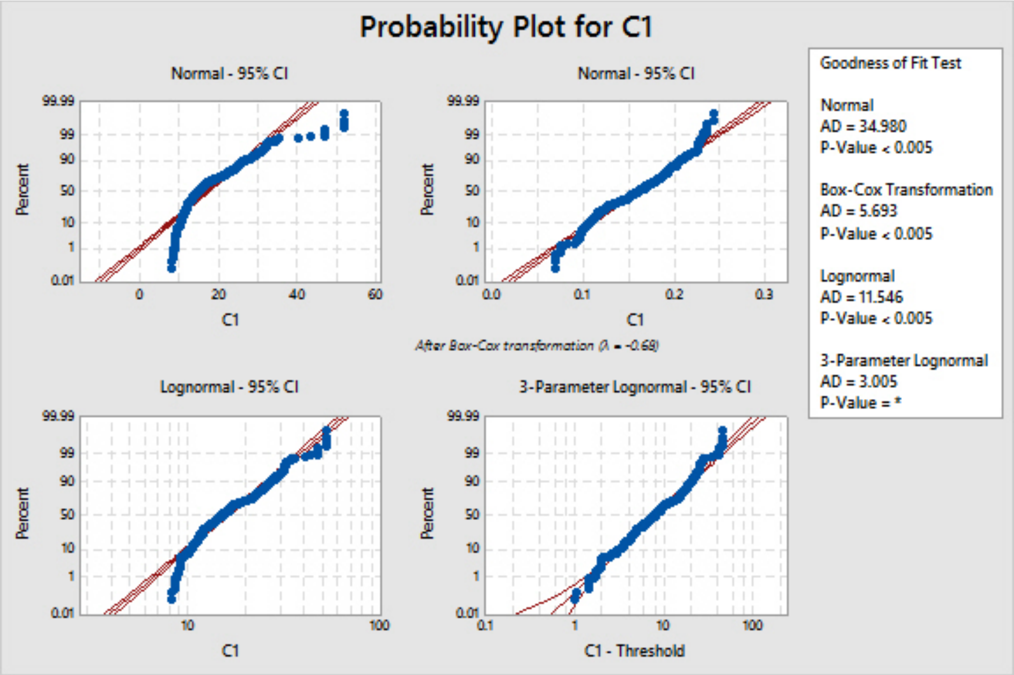


Figure XVIII: Goodness of Fit Test graph D for 6 MW regulation obligation

F.2 Probability distribution testing 12 MW regulation obligation

Distribution ID Plot for C2 [12 MW regulation obligation]

```
Descriptive Statistics

   N  N*     Mean    StDev   Median  Minimum  Maximum  Skewness  Kurtosis
1000   0  22.1906  10.7025  19.4642  8.15650  64.0658   1.02209  0.960906


Box-Cox transformation: λ = 0


Goodness of Fit Test

Distribution                  AD       P  LRT P
Normal                    21.401  <0.005
Box-Cox Transformation     4.440  <0.005
Lognormal                  4.440  <0.005
3-Parameter Lognormal      4.606       *  0.003
Exponential              134.553  <0.003
2-Parameter Exponential   18.107  <0.010  0.000
Weibull                   11.444  <0.010
3-Parameter Weibull        4.486  <0.005  0.000
Smallest Extreme Value    50.365  <0.010
Largest Extreme Value      7.381  <0.010
Gamma                      6.597  <0.005
3-Parameter Gamma          5.888       *  0.000
Logistic                  17.956  <0.005
Loglogistic                5.803  <0.005
3-Parameter Loglogistic    5.950       *  0.000


ML Estimates of Distribution Parameters

Distribution             Location    Shape     Scale  Threshold
Normal*                  22.19062           10.70254
Box-Cox Transformation*   2.98890            0.47221
Lognormal*                2.98890            0.47221
3-Parameter Lognormal     2.78478            0.57635    3.20577
Exponential                                 22.19062
2-Parameter Exponential                     14.04817    8.14245
Weibull                            2.21595  25.16232
3-Parameter Weibull                1.25970  15.14897    8.04964
Smallest Extreme Value   27.97587           13.08542
Largest Extreme Value    17.38527            7.92474
Gamma                              4.67393   4.74774
3-Parameter Gamma                  1.40758  10.10783    7.96301
Logistic                 21.05754            5.99171
Loglogistic               2.98573            0.27876
3-Parameter Loglogistic   2.67234            0.38627    4.91089

* Scale: Adjusted ML estimate
```

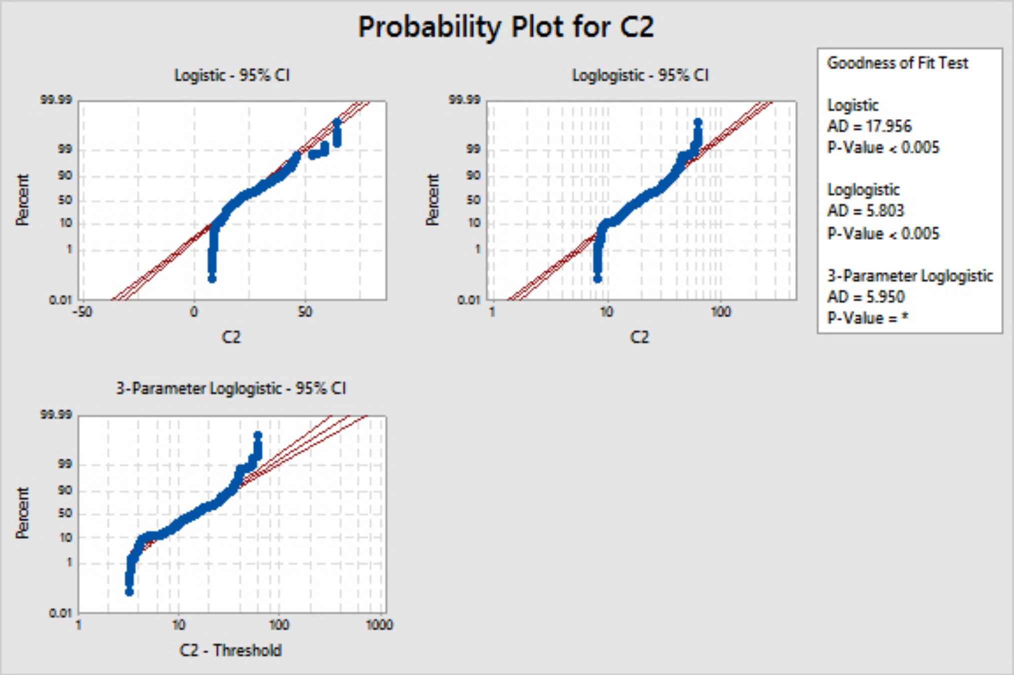


Figure XIX: Goodness of Fit Test graph A for 12 MW regulation obligation

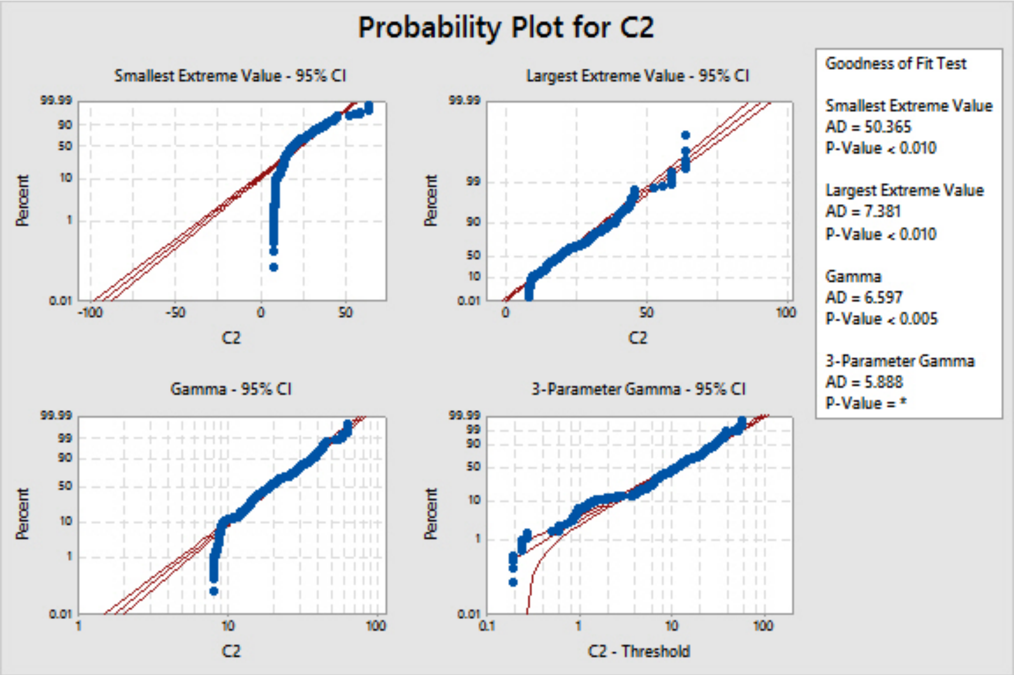


Figure XX: Goodness of Fit Test graph B for 12 MW regulation obligation

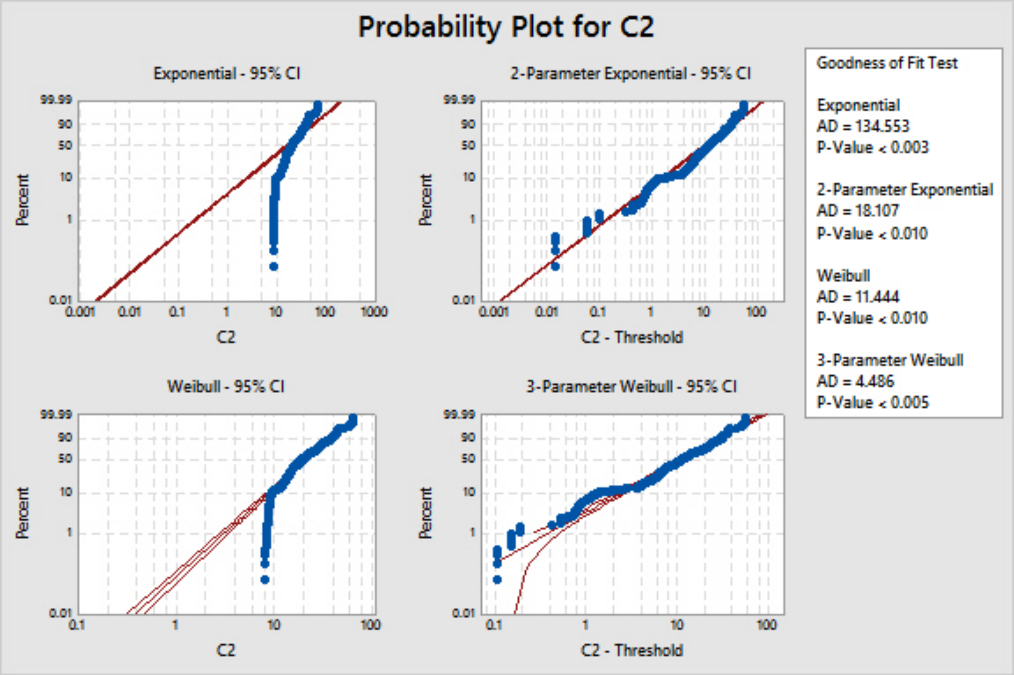


Figure XXI: Goodness of Fit Test graph C for 12 MW regulation obligation

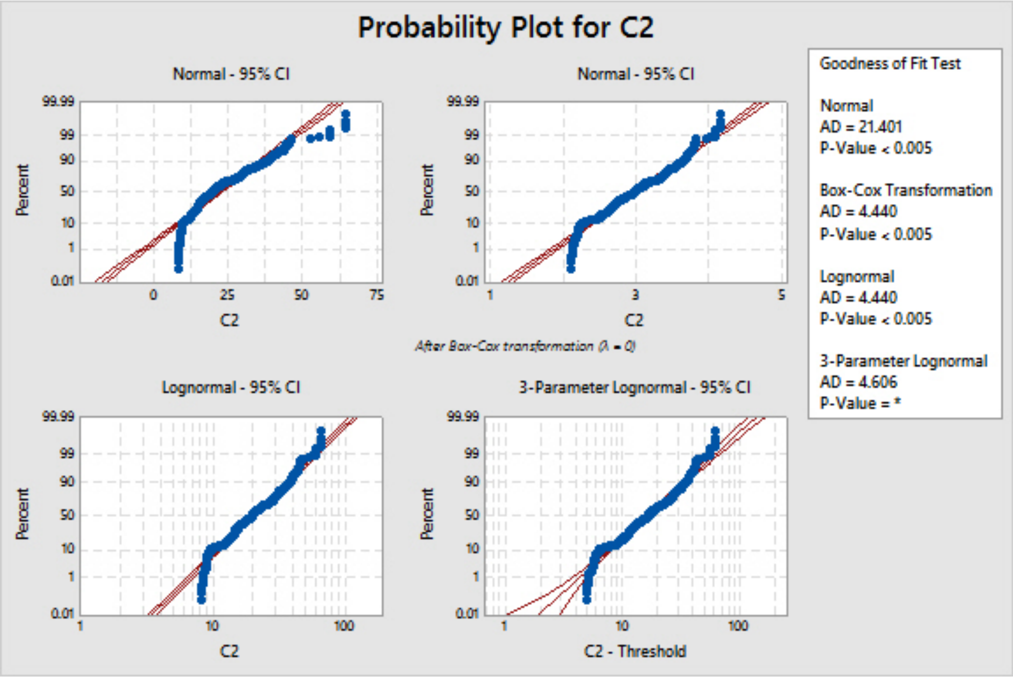


Figure XXII: Goodness of Fit Test graph D for 12 MW regulation obligation

F.3 Probability distribution testing 18 MW regulation obligation

Distribution ID Plot for C3 [18 MW regulation obligation]

```
Descriptive Statistics

   N  N*     Mean    StDev   Median  Minimum  Maximum  Skewness  Kurtosis
1000   0  18.5294  8.08141  16.4674  7.20652  49.7919  0.944627  0.825135


Box-Cox transformation: λ = 0


Goodness of Fit Test

Distribution                  AD       P  LRT P
Normal                    18.203  <0.005
Box-Cox Transformation     4.495  <0.005
Lognormal                  4.495  <0.005
3-Parameter Lognormal      4.673       *  0.074
Exponential              156.482  <0.003
2-Parameter Exponential   29.319  <0.010  0.000
Weibull                   11.246  <0.010
3-Parameter Weibull        4.764  <0.005  0.000
Smallest Extreme Value    46.446  <0.010
Largest Extreme Value      5.920  <0.010
Gamma                      5.987  <0.005
3-Parameter Gamma          5.689       *  0.000
Logistic                  15.677  <0.005
Loglogistic                5.718  <0.005
3-Parameter Loglogistic    5.796       *  0.003


ML Estimates of Distribution Parameters

Distribution             Location    Shape     Scale  Threshold
Normal*                  18.52940            8.08141
Box-Cox Transformation*   2.82788            0.42965
Lognormal*                2.82788            0.42965
3-Parameter Lognormal     2.69416            0.49010    1.90607
Exponential                                 18.52940
2-Parameter Exponential                     11.33421    7.19519
Weibull                           2.43976  20.95428
3-Parameter Weibull               1.38833  12.53156    7.05256
Smallest Extreme Value   22.87386            9.76011
Largest Extreme Value    14.86414            6.14413
Gamma                             5.62711   3.29288
3-Parameter Gamma                 1.80407   6.54130    6.72841
Logistic                 17.74698            4.54665
Loglogistic               2.82682            0.25226
3-Parameter Loglogistic   2.57853            0.32607    3.49439

* Scale: Adjusted ML estimate
```

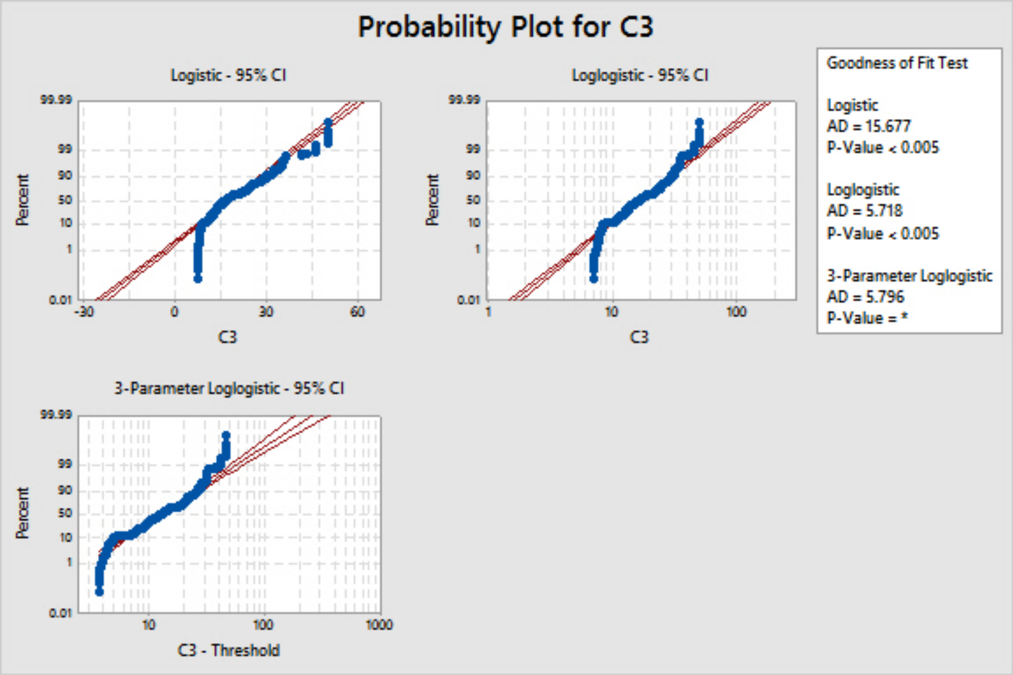


Figure XXIII: Goodness of Fit Test graph A for 18 MW regulation obligation

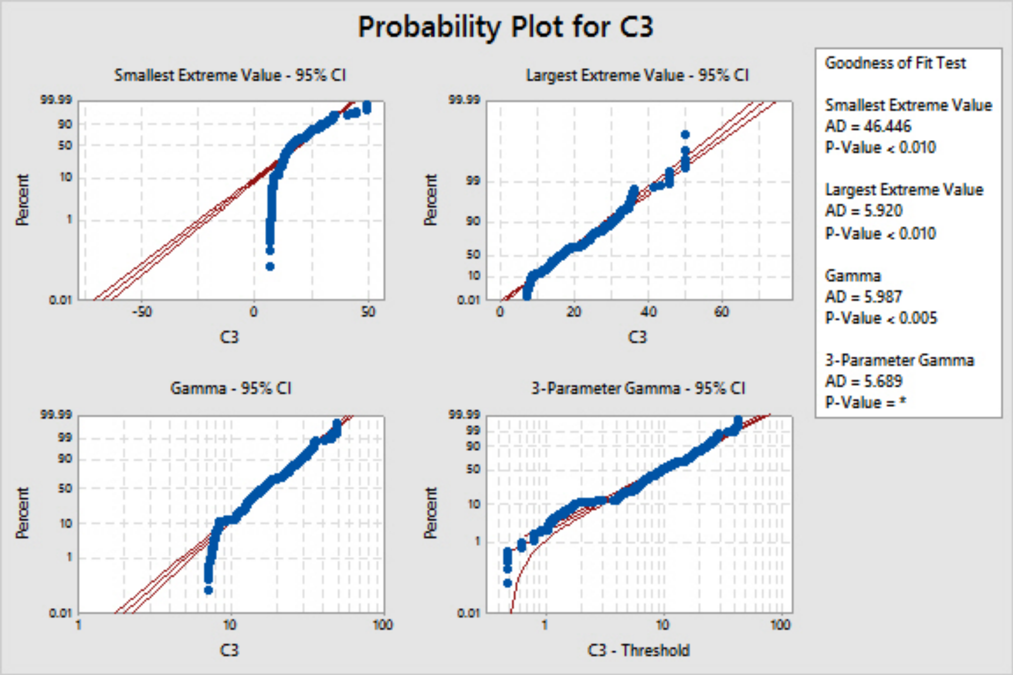


Figure XXIV: Goodness of Fit Test graph B for 18 MW regulation obligation

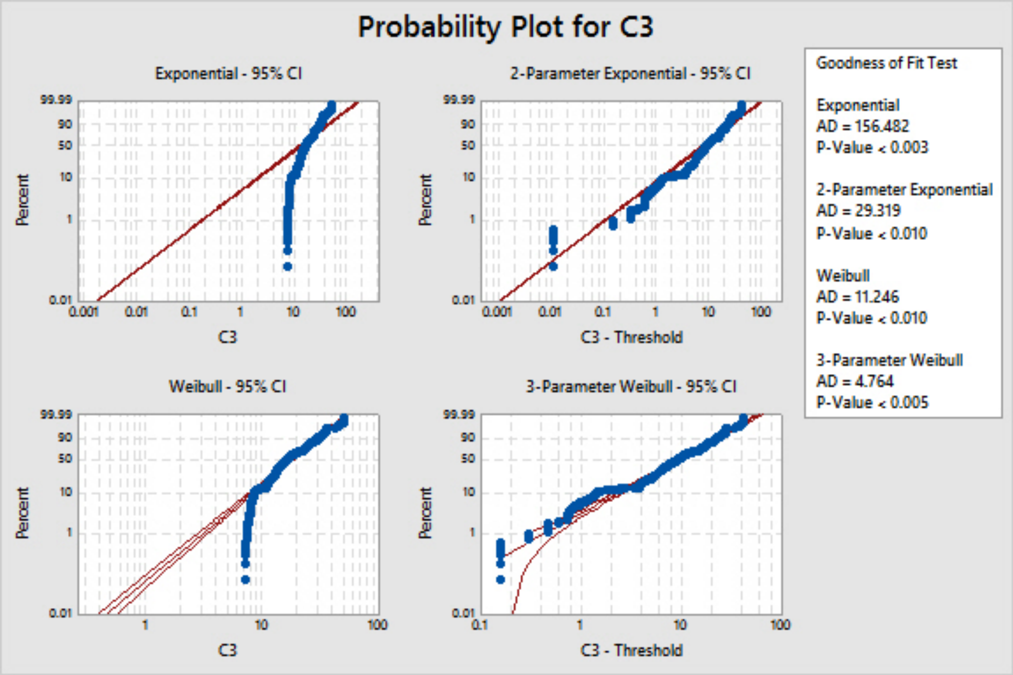


Figure XXV: Goodness of Fit Test graph C for 18 MW regulation obligation

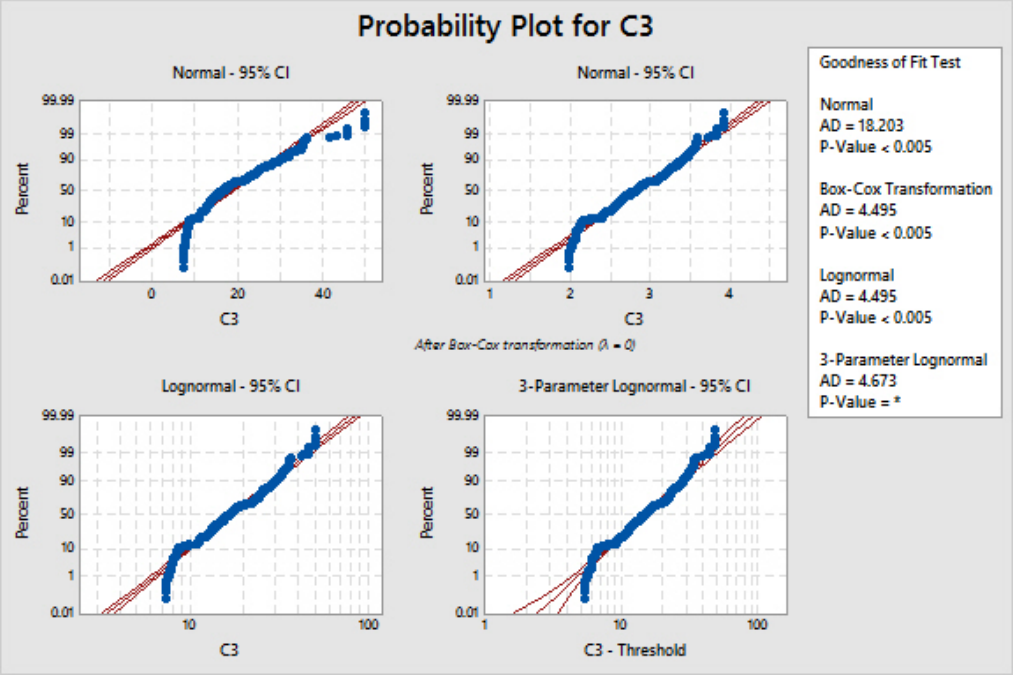


Figure XXVI: Goodness of Fit Test graph D for 18 MW regulation obligation

F.4 Probability distribution testing 24 MW regulation obligation

Distribution ID Plot for C4 [24 MW regulation obligation]

```
Descriptive Statistics

   N  N*     Mean    StDev   Median  Minimum  Maximum  Skewness  Kurtosis
1000   0  51.5334  10.0045  50.7279  30.2352  86.4873  0.562443  0.588250


Box-Cox transformation: λ = 0


Goodness of Fit Test

Distribution                  AD       P  LRT P
Normal                     4.136  <0.005
Box-Cox Transformation     1.399  <0.005
Lognormal                  1.399  <0.005
3-Parameter Lognormal      1.360       *  0.820
Exponential              304.140  <0.003
2-Parameter Exponential  126.271  <0.010  0.000
Weibull                   12.191  <0.010
3-Parameter Weibull        2.258  <0.005  0.000
Smallest Extreme Value    31.389  <0.010
Largest Extreme Value      4.534  <0.010
Gamma                      1.433  <0.005
3-Parameter Gamma          1.459       *  0.022
Logistic                   2.934  <0.005
Loglogistic                1.337  <0.005
3-Parameter Loglogistic    1.345       *  0.927


ML Estimates of Distribution Parameters

Distribution             Location     Shape     Scale  Threshold
Normal*                  51.53343            10.00449
Box-Cox Transformation*   3.92369             0.19283
Lognormal*                3.92369             0.19283
3-Parameter Lognormal     3.95589             0.18663   -1.62608
Exponential                                  51.53343
2-Parameter Exponential                      21.31956   30.21387
Weibull                             5.22196  55.69309
3-Parameter Weibull                 2.36009  25.20265   29.17689
Smallest Extreme Value   56.76928            11.46541
Largest Extreme Value    46.77305             8.73347
Gamma                              27.12854   1.89960
3-Parameter Gamma                  12.76299   2.80184   15.77352
Logistic                 51.06074             5.61607
Loglogistic               3.92484             0.10975
3-Parameter Loglogistic   3.90806             0.11163    0.83582

* Scale: Adjusted ML estimate
```

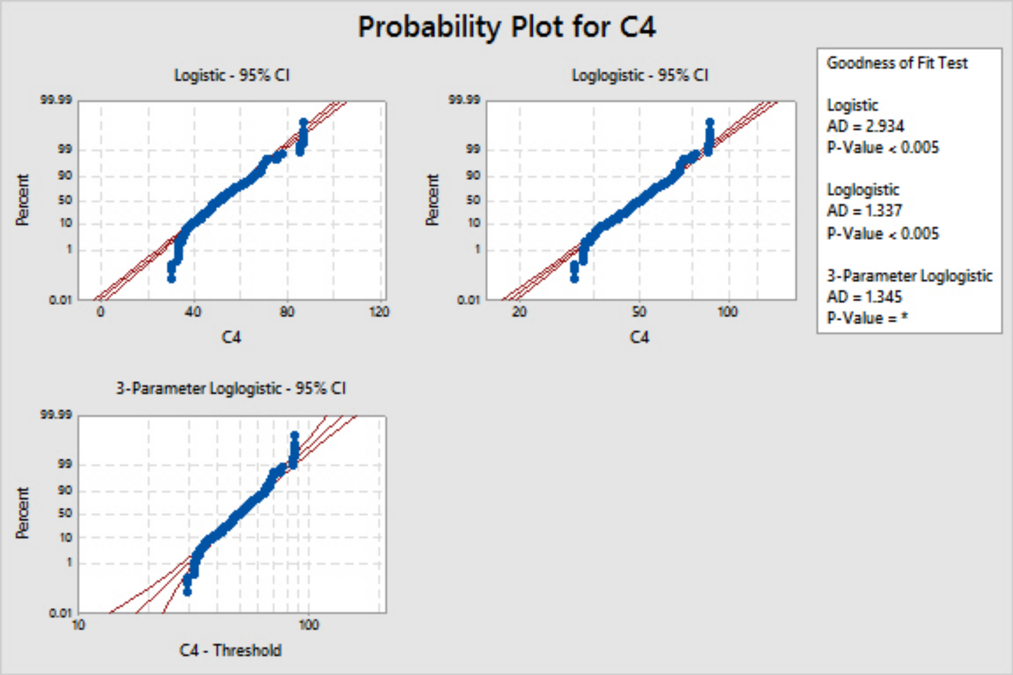


Figure XXVII: Goodness of Fit Test graph A for 24 MW regulation obligation

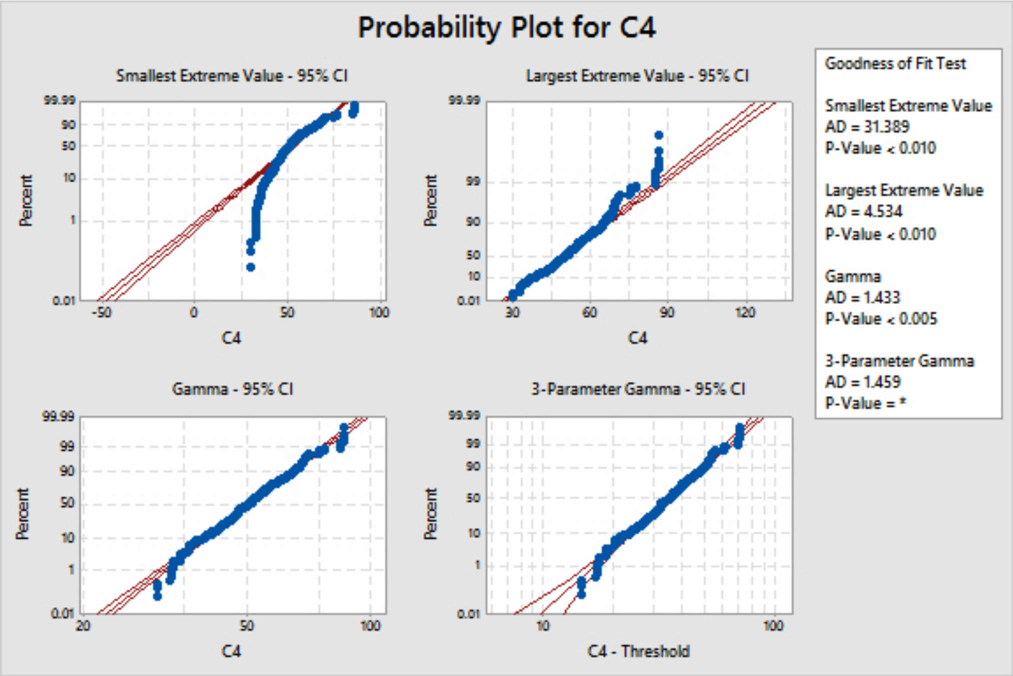


Figure XXVIII: Goodness of Fit Test graph B for 24 MW regulation obligation

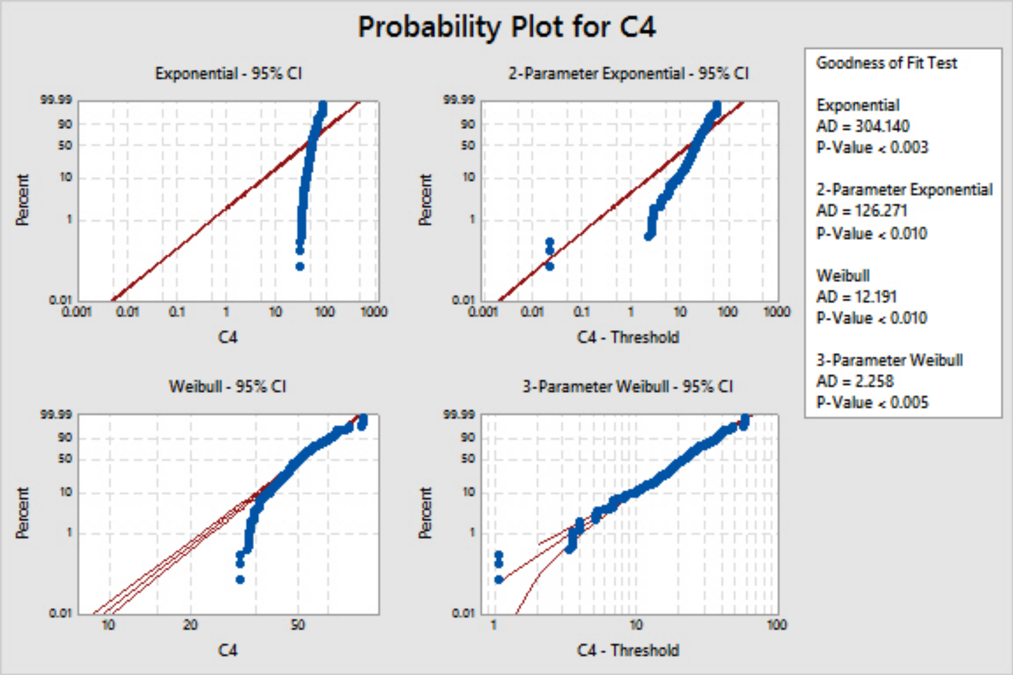


Figure XXIX: Goodness of Fit Test graph C for 24 MW regulation obligation

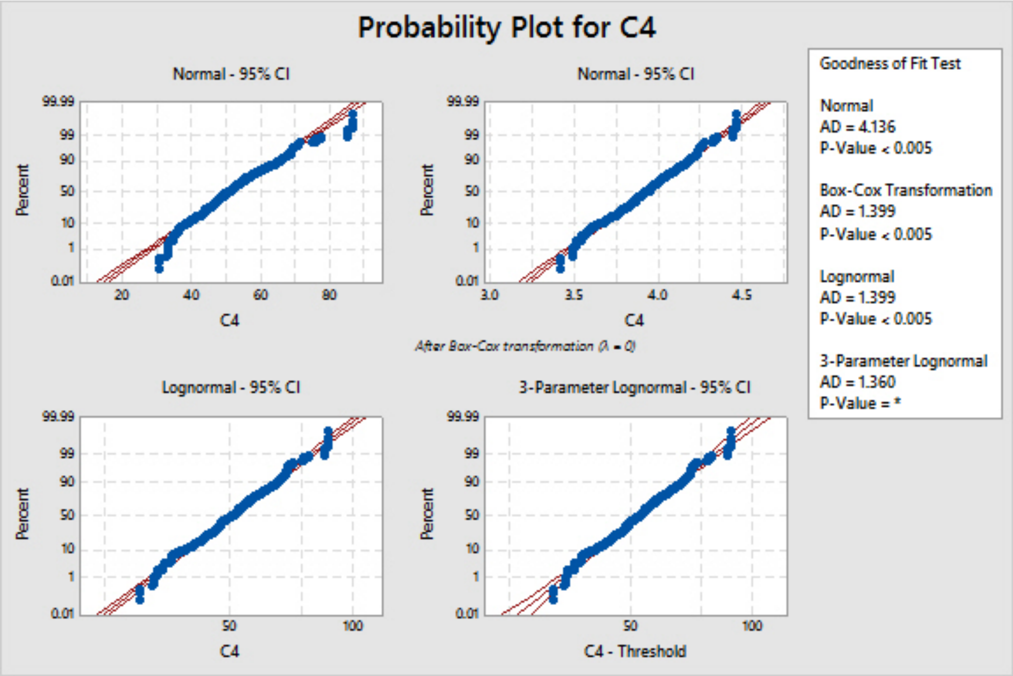


Figure XXX: Goodness of Fit Test graph D for 24 MW regulation obligation

G Table with negative marginal costs referred to in the discussion chapter 6.3

Table III: Table with negative marginal costs

Regulation Obligation [MW]	Actual marginal cost [NOK/MW]	Model marginal cost [NOK/MW]
1	13,6698555	40,8619789
2	2,28760115	9,56889518
3	18,5819412	3,94168358
4	19,2816121	4,78707149
5	2,03427301	5,31193578
6	4,46542154	4,55389672
7	-31,4936646	22,5581812
8	31,0123979	-4,71401221
9	13,6880833	1,32210538
10	7,93175561	9,27834847
11	-6,41188548	13,0383336
12	-50,60616	46,3198001
13	5,99056565	8,34512499
14	9,31618014	8,88471642
15	9,38929116	10,8319632
16	-9,464062	6,91176345
17	6,12469873	8,70337716
18	-0,96369238	7,48270644
19	17,1929615	11,0656367
20	-10,5286478	12,0197685
21	9,34983109	11,0028191
22	13,0067357	12,3828121
23	797,672474	719,829946
24	-10,5130867	1,1709708
25	24,6923386	13,042938
26	19,8456252	17,3415993

H Input data used in SCENRED

H.1 Treecon.opt

```
# tree construction option file

construction_method 1

reduction_method   1
first_branch      1
*red_percentage   0.3
*eps_evolution    0.5
order            1
scaling          0
*eps_growth       1

*read_distance     dist.sd
*write_distance     dist.sd


*section epsilon
  2*4*1  0.2
  5     0.1
end

section num_nodes
2          2
26         4
50         8
74         16
98         32
122        64
146        128
170        256
194        512
end
```

H.2 Scenred.opt

```
reduction_method  1

red_num_leaves    512
*red_percentage    0.40
metric_type      1
order           1
p_norm          1
scaling         1
write_distance   dist.sd
```

I The EEM14 article - Optimal day-ahead electricity market bidding considering different ancillary services

This article was submitted, approved and presented at the 11th International Conference on the European Energy Market in Cracow, 28th-30th of May 2014. It is based on the preliminary specialization project written last fall [2], however, this article was written during the spring as a part of the master thesis.

11th International Conference
"The European Energy Market, EEM14"
28-30 May 2014, Kraków, Poland
Competition

Figure XXXI: Bjørnar (left) and Sjur (right) after an award was received for qualifying to the final of the best PhD-paper competition

Optimal day-ahead electricity market bidding considering different ancillary services

Bjørnar Fjelldal, Sjur Messel Nafstad, Gro Klæboe
Norwegian University of Science and Technology (NTNU),
Department of Electric Power Engineering
Trondheim, Norway, 2014
bjornar@fjelldal.org, sjurmess@stud.ntnu.no,
gro.klaeboe@powel.no

***Abstract* - This paper studies optimal bidding strategies for a hydropower producer in a day-ahead market and different regulation markets. It contains two case studies, analyzing the optimal production pattern considering a pre-spot situation taking different regulation markets into account. Both the optimal bid structure between the spot market and the primary regulation market in Switzerland as well as the spot market and the secondary regulation market in Norway were investigated. Models regarding both case studies were built, and the consistency of the analysis over an entire year was tested. The results illustrate how a small change in the utilization of the watercourse can be very lucrative when considering an additional regulation market, and discusses the importance of correctly calculated water values and profit factors with such ancillary services present.**

***Index Terms* - multi-market optimization, regulation markets, primary regulation, secondary regulation, model development**

I. Introduction

In the recent years it have been established an increased number of regulation markets that hydropower producers need to take into account when optimizing their production and bidding strategy. The time horizons span from seasonal and week-ahead markets to intraday markets that are cleared up to one hour before the delivery. Also, different physical properties between these markets are present, i.e. how fast the regulation capacity needs to be able to contribute to the system, and how much volume it should contribute with [1], [2]. In addition to a large number of markets to choose from, these properties complicate the producers' decision-making concerning which market to enter. Due to this it has become an increasing need for developing multimarket optimization tools in order to help hydropower producers solve these problems.

This paper focuses on two specific market cases:

- Optimal bidding strategies between the spot market and the primary regulation market in Switzerland
- Optimal bidding strategies between the spot market and the secondary regulation market in Norway

Regulation markets are in general rapidly expanding, and hydropower producers are getting increasingly aware of the opportunities such markets offer. This paper studies the change in production patterns and observations related to this, when ancillary services are taken into account. To investigate these problems, two different models were developed. A base model only considering optimization in the spot market was also developed for a comparison of the results.

II. Nomenclature

TABLE I: INDICES

d	Day d
h	Hour h
i	PQ-cut index i
r	Reservoir r
s	Station s
u	Unit u

TABLE II: SETS

D	Number of days
H	Number of hours
I	Number of PQ-cut indices
R	Number of reservoirs
S	Number of stations
U	Number of units at station s

TABLE III: VARIABLES

$b^{\mathrm{r},h}$	Bypass from reservoir r in hour h	m^3/s
$d_{inv}^{s,u,h}$	Droop setting on unit u at station s in hour h (inverse due to a linearization of the model)	%
$f^{\mathrm{r},h}$	Spilled flood from reservoir r in hour h	m^3/s
$q^{s,u,h}$	Discharge through unit u at station s in hour h	m^3/s
$x^{s,u,h}$	Spot production on unit u at station s in hour h	MW
x_{tot}^{h}	Total spot production in hour h	MW
$y_{tot,sym}^{h}$	Total symmetrical primary regulation in hour h	MW
$y_{up}^{s,u,h}$	Upward regulation capacity on unit u at station s in hour h	MW
$y_{down}^{s,u,h}$	Downward regulation capacity on unit u at station s in hour h	MW
$y_{tot,up}^{h}$	Total upward regulation capacity in hour h	MW
$y_{tot,down}^{h}$	Total downward regulation capacity in hour h	MW
α^{r}	Energy equivalent of final level at reservoir r	MWh
$\beta^{s,u,h}$	Start indicator, 1 if unit u at station s starts in hour h, 0 otherwise	{0,1}
$\gamma^{r,h}$	Reservoir level at reservoir r in hour h	m^3
θ^{r}	Final reservoir level at reservoir r	m^3
$\lambda^{s,u,h}$	Max primary regulation on unit u at station s in hour h limited by the droop setting	MW
$\zeta^{s,u,h}$	Run indicator, 1 if unit u at station s runs in hour h, 0 otherwise	{0,1}
$\varphi^{s,u,h}$	Max primary regulation on unit u at station s in hour h	MW
φ_{tot}^{h}	Max total primary regulation in hour h	MW

TABLE IV: PARAMETERS

$A_{\max}^{r}$	Max reservoir regulation per hour at reservoir r	m^3
$B_{\max}^{r}$	Max bypass from reservoir r	m^3/s
$B_{\min}^{r}$	Min bypass from reservoir r	m^3/s
$C^{s,u}$	Startup cost for unit u at station s	€
$D_{const}^{s,u}$	Constant droop setting on unit u at station s when optimizing against a secondary regulation market	%
$D_{inv,\max}^{s,u}$	Max droop setting (inverse due to a linearization of the model)	%
$D_{inv,\min}^{s,u}$	Min droop setting (inverse due to a linearization	%

	of the model)	
F	Nominal system frequency	Hz
$G^{s,u}_{\max}$	Max ramping per hour on unit u at station s	MW
g	Gravity constant	m/s^2
$H^{s,u}$	Head for unit u at station s	m
$H^{s,u}_{tot}$	Total system head for unit u at station s	m
K^{h}_{up}	Delivery factor for upward regulation in a secondary regulation market	%
K^{h}_{down}	Delivery factor for downward regulation in a secondary regulation market	%
$L^{r}_{\max}$	Max reservoir volume in reservoir r	m^3
$L^{r}_{\min}$	Min reservoir volume in reservoir r	m^3
M_{up}	Max upward regulation bidding volume into a secondary regulation market	MW
M_{down}	Max downward regulation bidding volume into a secondary regulation market	MW
M_{sym}	Max regulation bidding volume into a symmetric primary regulation market	MW
$N^{r,h}$	Inflow to reservoir r in hour h	m^3/s
P^{h}_{spot}	Spot price of corresponding market in hour h	€/MWh
$P^{h}_{reg,sym}$	Price for reserved regulation capacity in a symmetric primary regulation market in hour h	€/MWh
$P^{h}_{reg,up}$	Price for reserved upward regulation capacity in a secondary regulation market in hour h	€/MWh
$P^{h}_{reg,down}$	Price for reserved downward regulation capacity in a secondary regulation market in hour h	€/MWh
$P^{h}_{en,up}$	Price for delivered upward regulation volume in a secondary regulation market in hour h	€/MWh
$P^{h}_{en,down}$	Price for delivered downward regulation volume in a secondary regulation market in hour h	€/MWh
$Q^{s,u}_{\max}$	Max discharge through unit u at station s	m^3/s
$Q^{s,u}_{\min}$	Min discharge through unit u at station s	m^3/s
W^{r}	Water value for reservoir r	€/MWh
$X^{s,u}_{\max}$	Max production on unit u at station s	MW
$X^{s,u}_{\min}$	Min production on unit u at station s	MW
Γ^{i}	Slope of cut i in the piecewise linearized PQ-curve	-
Δ_{freq}	Frequency deviation limit for a symmetric primary regulation market	Hz
$\eta^{s,u}_{\max}$	Turbine efficiency at best-point on unit u at station s	%
ρ	Water density	kg/m^3
Ω^{i}	Intercept of cut i in the piecewise linearized PQ-curve	-

III. The Mathematical Formulation of the Model

Three independent models that utilize many of the same properties and equations are presented below. A deterministic day-ahead market model was developed only taking a spot market into account. This model was developed further by taking either a primary or a secondary regulation market into consideration. [3], [4] and [5] are other examples of models built to achieve many of the same purposes as the models considered in this paper.

The efficiency curves of the turbines are implemented using a power-discharge-curve, doing a piecewise linearization due to the optimization solving algorithm's limitations of having a linearized problem.

Some other simplifications regarding the model were also made [6];

- Constant head factor during production
- The same relative efficiency curve for all units
- No change in use of water when regulation is activated
- Cascade-connected reservoirs
- Best-point strategy when calculating end reservoir value
- Water values assumed known and independent
- All markets are assumed perfect

A. The day-ahead market model

$$\max \sum_{h\in H} P^{h}_{spot} x^{h}_{tot} + \sum_{r\in R} W^{r}\alpha^{r} - \sum_{s\in S}\sum_{u\in U}\sum_{h\in H} C^{s,u}\beta^{s,u,h} \qquad (1)$$

The basic spot model - that does not take regulation capacity revenues into account - maximizes the profit from the main spot market and the remaining total value of the water left in the reservoir. The corresponding startup costs is also taken into consideration. See equation (1).

$$\gamma^{r,h} + N^{r,h} - \sum_{s\in S}\sum_{u\in U} q^{s,u,h} - f^{r,h} - b^{r,h} - \gamma^{r,(h+1)} = 0 \qquad r=1,\ s=1, \forall u\in U, \forall h\in H \neq h=H \qquad (2)$$

$$\gamma^{r,h} + N^{r,h} + \sum_{s\in S}\sum_{u\in U} q^{(s-1),u,h} - \sum_{s\in S}\sum_{u\in U} q^{s,u,h} - f^{r,h} - b^{r,h} + b^{(r-1),h} - \gamma^{r,(h+1)} = 0 \qquad \forall r\in R\neq r=1, \forall s\in S\neq s=1, \forall u\in U, \forall h\in H\neq h=H \qquad (3)$$

$$\gamma^{r,h} + N^{r,h} - \sum_{s\in S}\sum_{u\in U} q^{s,u,h} - f^{r,h} - b^{r,h} - \theta^{r} = 0 \qquad r=1,\ s=1, \forall u\in U, h=H \qquad (4)$$

$$\gamma^{r,h} + N^{r,h} + \sum_{s\in S}\sum_{u\in U} q^{(s-1),u,h} - \sum_{s\in S}\sum_{u\in U} q^{s,u,h} - f^{r,h} - b^{r,h} + b^{(r-1),h} - \theta^{r} = 0 \qquad \forall r\in R\neq r=1, \forall s\in S\neq s=1, \forall u\in U, h=H \qquad (5)$$

Equation (2)-(5) describes the water balance of each reservoir in terms of initial water volume, inflow, discharge, flooding and bypass to other lower reservoirs.

$$\alpha^{r} = \frac{\theta^{r} H^{s,u}_{tot}\rho g \eta^{s,u}_{\max}}{3.6\cdot 10^{9}} \qquad \forall r\in R, \forall s\in S, \forall u\in U \qquad (6)$$

The conversion from reservoir volume (m^3) to energy (MWh) is done by equation (6). Since it is unknown how the plant will operate with its future amounts of water, a best point strategy is used.

$$x^{s,u,h} \leq \left(\Gamma^{i}\frac{q^{s,u,h}}{Q^{s,u}_{\max}} + \Omega^{i}\right) Q^{s,u}_{\max} H^{s,\imath} \qquad \forall s\in S, \forall u\in U, \forall h\in H, \forall i\in I \qquad (7)$$

When calculating the power output, data from an efficiency curve has been converted into a PQ-curve. A piecewise linearization of that curve, the linear cuts described by the matrixes in equation (7), determines the power output a unit will get from the amount of discharged water.

$$B^{r}_{\min} \leq b^{r,h} \leq B^{r}_{\max} \qquad \forall r\in R, \forall h\in H \qquad (8)$$

$$L^{r}_{\min} \leq \gamma^{r,h} \leq L^{r}_{\max} \qquad \forall r\in R, \forall h\in H \qquad (9)$$

$$L^{r}_{\min} \leq \theta^{r,h} \leq L^{r}_{\max} \qquad \forall r\in R \qquad (10)$$

$$Q^{s,u}_{\min} \leq q^{s,u,h} \leq Q^{s,u}_{\max} \qquad \forall s\in S, \forall u\in U, \forall h\in H \qquad (11)$$

Equation (8)-(11) simply describes the minimum and maximum allowed levels of the respective bypass from reservoirs, reservoir volumes, end reservoir volumes and discharge from units,

$$\left|x^{s,u,h} - x^{s,u,(h-1)}\right| \leq G^{s,u}_{\max} \qquad \forall s \in S, \forall u \in U, \forall h \in H \neq h = H \qquad (12)$$

$$\left|\gamma^{r,h} - \gamma^{r,(h+1)}\right| \leq A^{r}_{\max} \qquad \forall r \in R, \forall h \in H \neq h = H \qquad (13)$$

whereas (12) and (13) limits the maximum change in reservoir volume (regulation) and production (ramping).

$$x^{s,u,h} - X^{s,u}_{\max}\zeta^{s,u,h} \leq 0 \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (14)$$

$$x^{s,u,h} - X^{s,u}_{\min}\zeta^{s,u,h} \geq 0 \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (15)$$

(14) and (15) limits the maximum and minimum production on each unit - as well as initializing a binary running indicator.

$$\beta^{s,u,h} - \left(\zeta^{s,u,h} - \zeta^{s,u,(h-1)}\right) \geq 0 \qquad \forall s \in S, \forall u \in U, \forall h \in H \neq h = 1 \qquad (16)$$

$$\beta^{s,u,h} - \left(\zeta^{s,u,(h+1)} - \zeta^{s,u,h}\right) \geq 0 \qquad \forall s \in S, \forall u \in U, h = 1 \qquad (17)$$

This binary indicator is again used in equation (16) and (17) to tell if a unit is starting up at hour h. Due to the binary properties of the running indicator, the startup indicator remains continuous, as it will either be 0 or 1 anyway because of its presence in the objective function.

$$\sum_{s \in S}\sum_{u \in U} x^{s,u,h} - x^{h}_{tot} = 0 \qquad \forall h \in H \qquad (18)$$

The spot production from all units is summed up in equation (18).

$$\zeta^{s,u,h} \in \{0,1\} \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (19)$$

$$x^{s,u,h}, x^{h}_{tot}, \alpha^{r}, \beta^{s,u,h}, \gamma^{r,h}, q^{s,u,h}, f^{r,h}, b^{r,h} \geq 0 \qquad \forall r \in R, \forall s \in S, \forall u \in U, \forall h \in H \qquad (20)$$

Equation (19) and (20) states the binary and non-negativity conditions of the variables.

B. Introducing the primary regulation market

When introducing a primary regulation market to the model, equations (2)-(20) are still used. A new objective function taking both the spot market and a symmetric primary regulation market into consideration, replaces equation (1). This objective function is then added along with new equations describing the new market, droop settings and biddable volumes of primary regulation.

$$\max \sum_{h \in H}\left(P^{h}_{spot}x^{h}_{tot} + P^{h}_{reg,sym}\varphi^{h}_{tot}\right) + \sum_{r \in R} W^{r}\alpha^{r} - \sum_{s \in S}\sum_{u \in U}\sum_{h \in H} C^{s,u}\beta^{s,u,h} \qquad (21)$$

The model with the included primary regulation market has an alternate way of income, thus the objective function (21) maximizes revenues from the spot market and the primary regulation market as well as the remaining value of the water in the reservoirs. Startup costs are still taken into account. Since the primary regulation in this case study is symmetrical, only the lowest value of upward regulation, downward regulation and regulation available by the droop setting is applicable.

$$y^{s,u,h}_{up} = X^{s,u}_{\max}\zeta^{s,u,h} - x^{s,u,h} \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (22)$$

$$y^{s,u,h}_{down} = x^{s,u,h} - X^{s,u}_{\min}\zeta^{s,u,h} \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (23)$$

The maximum amount of primary upward and downward regulation capacity is determined by equation (22) and (23), simply calculating the difference between the production to the spot market and the corresponding maximum and minimum production limits of the unit.

$$\lambda^{s,u,h} = X^{s,u}_{\max}\frac{\Delta_{freq}}{F}d^{s,u,h}_{inv} \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (24)$$

But since the droop setting on a unit often limits the maximum change in power output, a third restriction for maximum primary regulation is implemented (24). To avoid a non-linear expression, the droop variable has been inverted, making the equation programmable in the linear optimization language used.

$$D^{s,u}_{inv,\max} \leq d^{s,u,h}_{inv} \leq D^{s,u}_{inv,\min} \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (25)$$

$$\left|d^{s,u,h}_{inv} - d^{s,u,(h-1)}_{inv}\right| \leq D^{s,u}_{inv,reg} \qquad \forall s \in S, \forall u \in U, \forall h \in H \neq h = H \qquad (26)$$

(25) and (26) gives the possible outcomes for the (inverse) droop setting, respectively minimum and maximum values and maximum change from hour to hour.

$$\varphi^{s,u,h} \leq y^{s,u,h}_{up} \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (27)$$

$$\varphi^{s,u,h} \leq y^{s,u,h}_{down} \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (28)$$

$$\varphi^{s,u,h} \leq \lambda^{s,u,h} \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (29)$$

(27)-(29) states that the actual primary regulation capacity is limited to the lowest value of either upward regulation capacity, downward regulation capacity or regulation capacity available with the given droop setting.

$$\sum\sum \varphi^{s,u,h} - \varphi^{h}_{tot} = 0 \qquad \forall h \in H \qquad (30)$$

Equation (30) sums up the total applicable primary regulation capacity for the system.

$$\varphi^{h}_{tot} = \varphi^{h+1}_{tot} \qquad \forall h \in H \neq h = H \qquad (31)$$

The market restriction in Switzerland states that the same amount of primary regulation must be bid in each hour for the whole period, as shown in equation (31).

$$\varphi^{h}_{tot} \leq M_{sym} \qquad \forall h \in H \qquad (32)$$

Equation (32) states a maximum bid volume for the primary regulation determined by the market rules.

$$\varphi^{s,u,h}, \varphi^{h}_{tot}, y^{s,u,h}_{up}, y^{s,u,h}_{down}, \lambda^{s,u,h}, d^{s,u,h}_{inv} \geq 0 \qquad \forall s \in S, \forall u \in U, \forall h \in H \qquad (33)$$

Lastly, the new variables need to be non-negative (33).

C. Introducing the secondary regulation market

Similarly as with the introduction of the primary regulation market, implementing a secondary regulation market yields a new objective function and equations describing the new market, while continuing using equation (2)-(20) as the basis for the technical definition of the watercourse. Note that none of the new equations introduced in the primary regulation market model is used in the model below.

$$\max \sum_{h\in H}\left(P^h_{spot}x^h_{tot}+P^h_{reg,up}y^h_{tot,up}+P^h_{reg,down}y^h_{tot,down}+P^h_{en,up}\mathrm{K}^h_{up}y^h_{tot,up}+P^h_{en,down}\mathrm{K}^h_{down}y^h_{tot,down}\right) \quad (34)$$
$$+\sum_{r\in R}W^r\alpha^r-\sum_{s\in S}\sum_{u\in U}\sum_{h\in H}C^{s,u}\beta^{s,u,h}$$

The secondary regulation market in Norway gives a payment for both reserved capacity and actual delivered energy in the delivery period. This yields that yet another element in the objective function must be added. Equation (34) now maximizes the revenues from the spot market, from the secondary regulation capacity and from the actual energy delivered through secondary regulation in the optimization period. It is not possible to predict the volume of activated secondary regulation; this depends on events during the operation hours. Delivery factors are used as a measurement of the actual delivered amount of secondary regulation energy. These parameters describe the relationship between the reserved upward and downward capacity and the actual volume of upward and downward regulation delivered to the system, that yields an additional income. The value of the remaining water in the reservoir as well as startup costs are also taken into account.

$$y^{s,u,h}_{up} \le \left(X^{s,u}_{\max}-X^{s,u}_{\max}\frac{\frac{\Delta_{freq}}{F}}{D_{const}}\right)\zeta^{s,u,h}-x^{s,u,h} \qquad \forall s\in S,\forall u\in U,\forall h\in H \qquad (35)$$

$$y^{s,u,h}_{down} \le x^{s,u,h}-X^{s,u}_{\min}\zeta^{s,u,h}-\left(X^{s,u}_{\max}\frac{\frac{\Delta_{freq}}{F}}{D_{const}}\right)\zeta^{s,u,h} \qquad \forall s\in S,\forall u\in U,\forall h\in H \qquad (36)$$

Equation (35) and (36) assumes that a droop setting has been decided upon when optimizing against the primary regulation market, yielding the droop setting constant. This gap of power variance is seen upon as unavailable for the secondary regulation market, since the primary market is implicitly taken into account. Accompanied by the actual spot production, the equations determine the maximum secondary upward and downward regulation capacity.

$$\sum_{s\in S}\sum_{u\in U}y^{s,u,h}_{up}-y^h_{tot,up}=0 \qquad \forall h\in H \qquad (37)$$

$$\sum_{s\in S}\sum_{u\in U}y^{s,u,h}_{down}-y^h_{tot,down}=0 \qquad \forall h\in H \qquad (38)$$

For simplicity, equation (37) and (38) sums up the total available secondary upward and downward regulation capacities from each unit.

$$y^{(24(d-1)+h)}_{up,tot}=y^{(24(d-1)+h+1)}_{up,tot} \qquad \forall d\in D, h=1..7 \qquad (39)$$
$$y^{(24(d-1)+h)}_{up,tot}=y^{(24(d-1)+h+1)}_{up,tot} \qquad \forall d\in D, h=9..19 \qquad (40)$$
$$y^{(24(d-1)+h)}_{up,tot}=y^{(24(d-1)+h+1)}_{up,tot} \qquad \forall d\in D, h=21..23 \qquad (41)$$
$$y^{(24(d-1)+h)}_{down,tot}=y^{(24(d-1)+h+1)}_{down,tot} \qquad \forall d\in D, h=1..7 \qquad (42)$$
$$y^{(24(d-1)+h)}_{down,tot}=y^{(24(d-1)+h+1)}_{down,tot} \qquad \forall d\in D, h=9..19 \qquad (43)$$
$$y^{(24(d-1)+h)}_{down,tot}=y^{(24(d-1)+h+1)}_{down,tot} \qquad \forall d\in D, h=21..23 \qquad (44)$$
$$y^{(24(d-1)+h)}_{up,tot}=y^{(24(d)+h)}_{up,tot} \qquad \forall d\in D\neq d=D, h=1..24 \qquad (45)$$
$$y^{(24(d-1)+h)}_{down,tot}=y^{(24(d)+h)}_{down,tot} \qquad \forall d\in D\neq d=D, h=1..24 \qquad (46)$$

The market restrictions for the secondary regulation market in question divides the bidding into three time-blocks, where all secondary regulation must be equal each day of the period. This is taken into consideration by equations (39)-(46).

$$y^h_{tot,up}\le M_{up} \qquad \forall h\in H \qquad (47)$$

$$y^h_{tot,up}\le M_{down} \qquad \forall h\in H \qquad (48)$$

The maximum bid volume into the secondary regulation market is shown in (47) and (48).

$$y^{s,u,h}_{up}, y^{s,u,h}_{down}, y^h_{tot,up}, y^h_{tot,down}\ge 0 \qquad \forall s\in S,\forall u\in U,\forall h\in H \qquad (49)$$

Equation (49) declares the new variables non-negative.

IV. RESULTS

To achieve a more consistent basis for further analysis, the models were executed over a period of one week throughout the months of February, May, August and November. Historical spot prices, regulation market prices and inflow corresponding to the different seasons were used as input data. In three out of four periods, adjusting the production pattern to take a primary regulation market into account yields an increased objective function value. When introducing the secondary regulation market, the same increase was observed for all four periods.

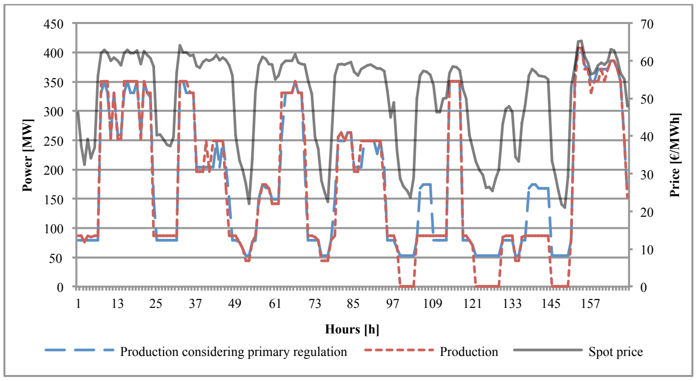


Figure 1: Optimal production patterns in the Swiss spot market and when including the primary regulation market during a week of February

Figure 1 illustrates the change in the production pattern when introducing a symmetric primary regulation market. When only optimizing against the spot market, it is optimal to avoid production in certain time segments. However, with the opportunity to participate in a regulation market with weekly obligations, no production stops are observed, and the amplitude of the maximum and minimum production are reduced in order to be able to deliver regulation if needed.

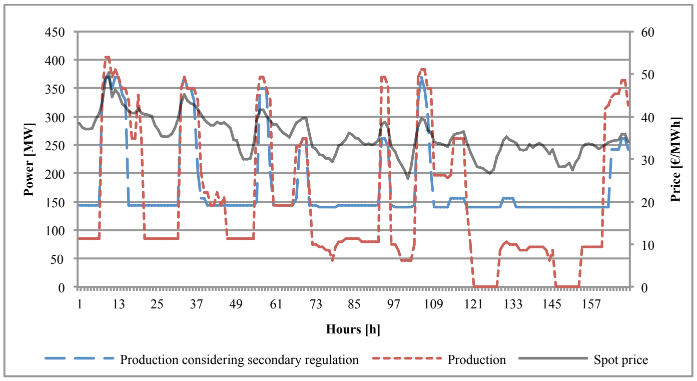


Figure 2: Optimal production patterns in the Norwegian spot market and when including the secondary regulation market during a week of May

The same behavior is observed when introducing a secondary regulation market, as shown in Figure 2.

V. Discussion

A. Possibly inaccurate calculations of water values

Water values are the expression for the expected marginal value of the energy stored in the reservoirs [7].

These are calculated based on reservoir levels, inflow forecasts and price forecasts in the market. However, in most models, only the spot price is taken into account, and with the growing need for regulation, this assumption may not be sufficient.

Because primary and secondary regulation does not necessarily involve using any water – the market pays for reserving the capacity, not necessarily using it - this can be regarded as income not taken into account by water values. Attendance in the primary and secondary regulation market can of course be seen upon as a single incident, but when this occurs more and more often and at a rising regulation market price, it can be considered whether or not the value of the water should be calculated not only by spot price forecasts, but also taking the prices from the regulation markets into account.

One challenge regarding this issue is at which degree to value the impact of these regulation markets, since the attendance is not known and brings yet another uncertainty into the calculations of the water values, if being considered.

B. The influence of delivery factors

In the Norwegian secondary regulation market the producers get paid both for reserved regulation capacity and delivered secondary regulation volume into the system. The reserved regulation capacity can be optimized as a variable in the model but since the amount of delivered secondary regulation volume depends on what actually happens with the frequency during the operating hours this cannot be optimized within the model itself.

However, since the producers get paid for this activated volume, it should be taken into account by the optimization model. This was done by introducing two constant factors in the objective function, called delivery factors. These constants describe the relationship between the reserved capacity and the actual delivered volume of upward and downward regulation delivered to the system, see equation (34). Delivery factors of 5% were used in this study. These were estimated based on tertiary regulation data, stating an average activation proportion approximately 50% of the time, equally distributed between upward and downward regulation. Combining this with a maximum startup time of 15 minutes for the tertiary regulation - which equals the maximum activation time for secondary regulation - this yields a maximum estimate of the delivery factors of 6.25%. If this model should be utilized in practice, historical data could be used to describe this relation between the reserved regulation capacity and actual delivered regulation volume into the system.

Since there was some uncertainty related to this assumption it was tested what happened if the delivery factors were neglected meaning that the producers only would get paid for the reserved secondary capacity and not the activated. This of course had an influence on the increase in profit but it did not influence the decision of going into the secondary market or not, the optimal production patterns were identical [6]. Even without payment of delivered capacity it was optimal to bid a maximum volume into the secondary market according to the market restrictions.

VI. Conclusion

Both case studies are showing that participating in regulation markets is profitable, if the production pattern is optimized taking both a spot market and a regulation market into account beforehand. It is however noteworthy that these case studies assumes that both the spot market and the regulation market are bid into simultaneously, and that the optimal production pattern presupposes all bids to be won by the producer. Also, no extra cost of attending such a regulation market is taken into account, and the possible additional use of water is neglected. Therefore, the increase in profit is somewhat over-exaggerated, and should not be interpreted in any other way than being an indicator of the possible profitability of bidding into a multi market structure.

The optimal production pattern is not that affected by the introduction of regulation markets. For the primary regulation market, a general trend is that the minimum production during the period is raised from zero production to the production needed in order to cover the actual bid into the regulation market. Also, a subsequent effect is a lower possible maximum production when such a production is favorable. The same effect was seen by the introduction of the secondary regulation market, but due to the specific market definitions the production pattern can vary more between the given time-blocks.

Perhaps the most essential and influencing factor of the results in these case studies is the topology and design of the watercourse optimized upon. The restrictions describing the physical properties of the watercourse, i.e. the reservoir size and production capacity, are more limiting than the applied market rules. If the same optimizations were to be done on a watercourse with a different design, other optimal results would most likely be found.

VII. Acknowledgments

We would like to thank our supervisor Olav Bjarte Fosso (Department of Electric Power Engineering, NTNU) for helpful guidance and support during this work.

VIII. References

[1] Ivar Wangensteen, *Power System Economics - the Nordic Electricity Market*, 2nd ed.: Tapir Academic Press, 2011, pp. 275-289.

[2] Gro Klæboe and Olav Bjarte Fosso, "Optimal bidding in sequential physical markets – a literature review and framework discussion," *Proc. 2013 International Conference on the European Energy Market*, 2013.

[3] Anders Lund Eriksrud and Jørgen Braathen, "Spot Market Bidding Taking the Balancing Power Market Into Account," unpublished, Department of Industrial Economics and Technology Management, NTNU, 2012.

[4] M. Olsson and L. Söder, "Hydropower planning including trade-off between energy and reserve markets," *Proc. 2003 IEEE PowerTech Conference*, 2003.

[5] Ellen Krohn Aasgård, Gørild Slettjord Andersen, Stein-Erik Fleten, and Daniel Haugstvedt, "Evaluating a Stochastic Programming Based Bidding Model for a Multireservoir System," *Power Systems, IEEE Transactions Issue 99*, 2013.

[6] Bjørnar Fjelldal and Sjur Messel Nafstad, "Optimal pre-spot bidding considering different regulation markets," unpublished, Department of Electric Power Engineering, NTNU, 2013.

[7] S. Stage and Y. Larsson, "Incremental Cost of Water Power," *IEEE Transactions of Power Apparatus and Systems*, vol. 80, no. 3, pp. 361-364, 1961.