European Commission

JRC TECHNICAL REPORTS

Hydro-related modelling for the WATERFLEX Exploratory Research Project

Version 0

Ricardo FERNANDEZ-BLANCO CARRAMOLINO

Konstantinos KAVVADIAS

Ignacio HIDALGO GONZALEZ

2016

Joint Research Centre

EUR 28419 EN

Contact information
Name: Ricardo FERNANDEZ BLANCO CARRAMOLINO
Address: Joint Research Centre, P.O. Box 2, NL-1755 ZG Petten, The Netherlands
E-mail: Ricardo.FERNANDEZ-BLANCO-CARRAMOLINO@ec.europa.eu
Tel.: +31 22456-5215

JRC Science Hub
https://ec.europa.eu/jrc

JRC104640

EUR 28419 EN

PDF	ISBN 978-92-79-65073-4	ISSN 1831-9424	doi: 10.2760/386964

Luxembourg: Publications Office of the European Union, 2016

How to cite this report:
Fernandez Blanco Carramolino, R., Kavvadias, K., Hidalgo Gonzalez, I., *Hydro-related modelling for the WATERFLEX Exploratory Research Project,* EUR 28419 EN, doi: 10.2760/386964

Contents

Abstract

In the context of power systems research, the analysis of the water-energy nexus is crucial. The high amount of water required to meet the needs of irrigation, human consumption and other uses may affect to the scheduling and dispatch of the thermal power plants, since they need freshwater for cooling. Power system models worldwide tend to neglect this water-energy interaction in order to reduce mathematical and computational complexity of the models. However, recent generation adequacy-related episodes (in Poland in 2015 and 2016 or France, Germany, and Spain in 2006) show the importance of these interactions for the operation of the power system. Most analyses expect these incidents to occur with increasing frequency due to climate change.

This first report of the WATERFLEX Exploratory Research Project proposes a medium-term hydrothermal coordination problem where the hydro-specific features of the power system are well represented by means of (i) the water balance in each hydropower plant; (ii) the bounds on water release, spillage, and reservoir levels; as well as (iii) the hydraulic network with water time delays for representing cascade hydropower plants. Also, dispatch constraints on thermal generators are also included in the model. The problem is thus formulated as a linear programming problem.

The proposed model is linked to the dispatch and unit commitment Dispa-SET model in which the thermal generators are precisely represented. Dispa-SET provides short-term operational decisions on aggregated hydropower and disaggregated thermal power plants. These two models are linked to the hydrological LISFLOOD model in order to accurately capture the water-power interactions. LISFLOOD could provide not only the water inflows of the hydropower plants but also the water needs of thermal power plants for a given plan of reservoir levels.

1 Introduction

The objective of the WATERFLEX Exploratory Research Project is to assess the potential of hydropower as a source of flexibility to the European power system, as well as analysing the Water-Energy nexus against the background of the EU initiatives towards a low-carbon energy system. To this purpose, the method proposed in WATERFLEX for better representing and analysing the complex interdependencies between the power and the water sectors consists of combining two of the modelling tools available at the JRC, the LISFLOOD hydrological model [1] and the Dispa-SET unit commitment and dispatch model [2], with a medium-term hydrothermal coordination model (MTHC), as shown in Figure 1. This report describes the objective, structure, underlying concepts, and assumptions of the latter.

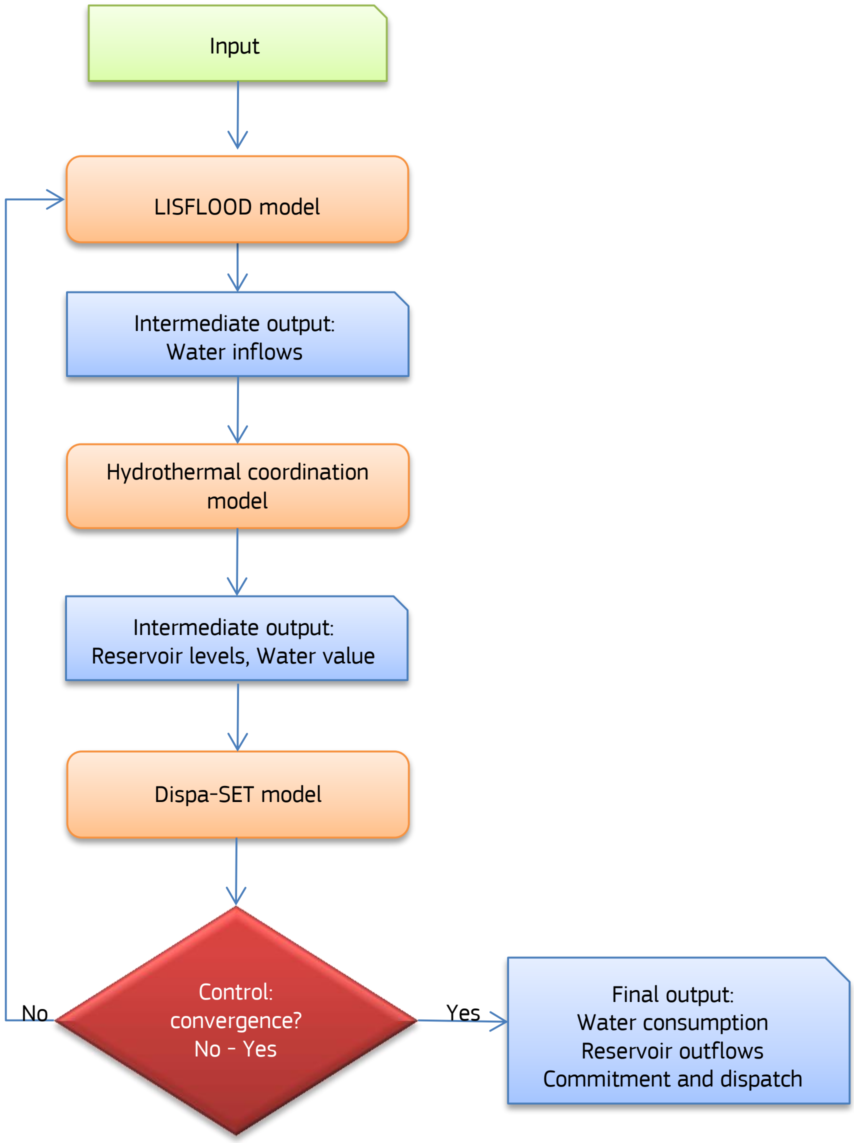


Figure 1. Interactions between LISFLOOD, MTHC, and Dispa-SET models

The MTHC problem takes into account the techno-economic features of hydropower plants and its associated reservoirs as well as thermal power plants in the medium term (time horizon from 1 year to several years). The outcome of this coordination problem is the operation planning of hydro and thermal power plants in weekly or monthly time steps (although daily time steps can also be assumed). This problem can be tackled from two perspectives: (i) the extensive form (also known as deterministic equivalent) or (ii) the stochastic form.

The deterministic MTHC problem basically assumes fixed water inflows and the problem can be formulated by linear programming, nonlinear programming, or mixed-integer linear programming, depending on how the hydro-related and thermal-related technical features are modelled. The deterministic problem could be useful to perform a scenario analysis based on representative time periods, e.g., years.

Regarding the stochastic form, the uncertainty is presented as hydrological scenarios for each planning stage. A hydrological scenario consists of the amount of water (in cubic metres) available to generate energy at each stage through the horizon. These scenarios are built with information from previous years. When considering all the historical data to generate the scenarios, the problem becomes extremely large. However, the number of scenarios can be reduced to a reasonable number of scenarios representing the uncertainty in an accurate way by using scenario reduction techniques. Since the problem is still too large to be solved by traditional methods, a decomposition technique is needed.

Based on the technical literature, there are two ways to tackle the stochastic problem:

1. Vertical (by stage/time), e.g., Stochastic Dual Dynamic Programming (SDDP) which is a Benders decomposition-based algorithm. It is widely used in the open literature but there could have multi-stage difficulties [3], [4], [5], [6].
2. Horizontal (by scenario), e.g., Progressive Hedging (PH) which is an Augmented Lagrangian-based algorithm [7]. PH solves each scenario separately and then finds an optimal solution by penalizing iteratively scenario solutions that do not respect non-anticipativity. Their popularity increased after 2010 for multistage problems.

Also, there are some statistical approaches (external sampling based) such as sample average approximation (SAA) [8, 9] which can be used when the stochastic problem is too large to be solved by exact solution techniques. However, the approach of random generation of scenarios is computationally intractable for solving multistage stochastic programs because of the exponential growth of the number of scenarios when increasing the number of stages [9].

1.1 Computational aspects

The computational aspects are mainly related to the stochastic form of the MTHC problem. One main concern relies on which method is more suitable for the stochastic hydrothermal coordination problem. Traditionally, SDDP has been used to solve such problem but computational difficulties could come up when solving for large-scale systems. Currently, PH is becoming more popular to solve stochastic programming problems since it can be parallelized [10] with minimum amount of communication between each instance. It is also more stable than the Nested Decomposition, allowing for good solutions with less computational time; and it may scale better for large-scale systems [11].

Another computational challenge regardless of the chosen method is to find the best trade-off between complexity, time step, and clustering of plants. The complexity is expected to grow as the scenarios increase. Uhr *et al.* [12] state that "*the exponential growth of the problem size is the limiting factor for the maximum time horizon. It is now clear that the original goal of having a time horizon of one year with planning periods of one month is practically infeasible except for the simplest cases with two or perhaps three scenarios*." The number of scenarios is strongly related to the number of stages that are assumed in the scenario generation and it is crucial to appropriately reduce this number of scenarios with scenario reduction techniques. For the particular case of the MHTC problem, smaller time steps are considered in the first stages, and more nodes are created at the last stages. The reader is referred to references [5], [13], [14] for further information on particular instances. Pereira *et al.* [5] focused on the Brazilian case comprising 39 hydroelectric plants and an aggregated thermal unit. It is assumed 10 stages which results in 512 scenarios. Gonçalves *et al.* [13] assume different realizations at different stages resulting in 1440 scenarios. Tilmant *et al.* [14] analyse the Turkish case study with 2 cascades, 11 hydro plants, and 50 synthetic hydro inflows by considering 20 stages in a time horizon of 60 months.

1.2 Data sources

The ideal dataset needed to solve the MTHC problem, comprising the more relevant information for hydropower plants, thermal power plants, and time series, is summarised in Table 1. For hydropower plants, it is important to collect the plant name, the location of the dam, installed capacity, plant type (reservoir, pumped storage, or run of river), dam height and head, reservoir capacity in volume units, the bounds on storage levels and outflows, as well as the incidence matrix for the connected dams.

For the thermal power plants, in addition to the plant name, location, and plant type, it is also necessary to know the water withdrawal and consumption factors as well as the cooling method used by each plant. These two fields are essential to analyse the water-power nexus and to propose improvements in the modelling of the hydrothermal coordination problem.

Finally, time series regarding generation, inflows, reservoir levels, or run-of-river profiles are crucial to realistically simulate the MTHC problem and to validate the output.

Table 1. Ideal dataset.

Data	Information
Hydropower plant	Plant name Location (longitude and latitude of the dam) Installed capacity Plant type (reservoir, pumped storage, or run of river) Dam height Head Reservoir capacity (volume or energy) Minimum and maximum storage levels (m^3) Minimum and maximum outflow (m^3/s) Network data (connected dams)
Thermal power plant	Plant name Location (longitude and latitude) Plant type Water withdrawal and consumption factors Cooling method
Time series	Generation Inflows, outflows, spillages Minimum and maximum flows Reservoir levels (or filling rates) Run of river profiles

The data related to hydropower plants and thermal power plants are partially covered by Platts([1]) and GlobalData([2]). Aggregated time series have been compiled from public sources for some countries. Times series for discharges, inflows, outflows, and reservoir filling rates from the LISFLOOD model [1] have been provided by unit D2 (Water and Marine Resources) from the Joint Research Centre.

The gathering of hydro-related information for the power plants is a complex task and there is a general problem of matching reservoirs with hydropower plants. Two main sources of information can be used in this work:

— LISFLOOD model [1]: LISFLOOD uses a dataset of reservoirs comprising 1445 units in Europe and North Africa. All of them have location (latitude/longitude) and storage

([1]) World Electric Power Plant database: http://www.platts.com/products/world-electric-power-plants-database
([2]) GlobalData: http://www.energy.globaldata.com

capacity (m^3). 1272 reservoirs have dam-height information and 298 of them have catchment information. However, 29 of them do not have dam-height information. For the aggregate catchments, the LISFLOOD model is able to provide average daily inflows from 1990 till 2014.

— ENTSOE [15]: we have gathered information about the total installed capacity for all hydro-related plants per control zone that have an Energy Information Code (EIC)([3]). Figure 2 shows the aggregated installed capacity per control zone and type of hydropower plant, namely hydro water reservoir, hydro run-of-river and poundage, and hydro pumped storage.

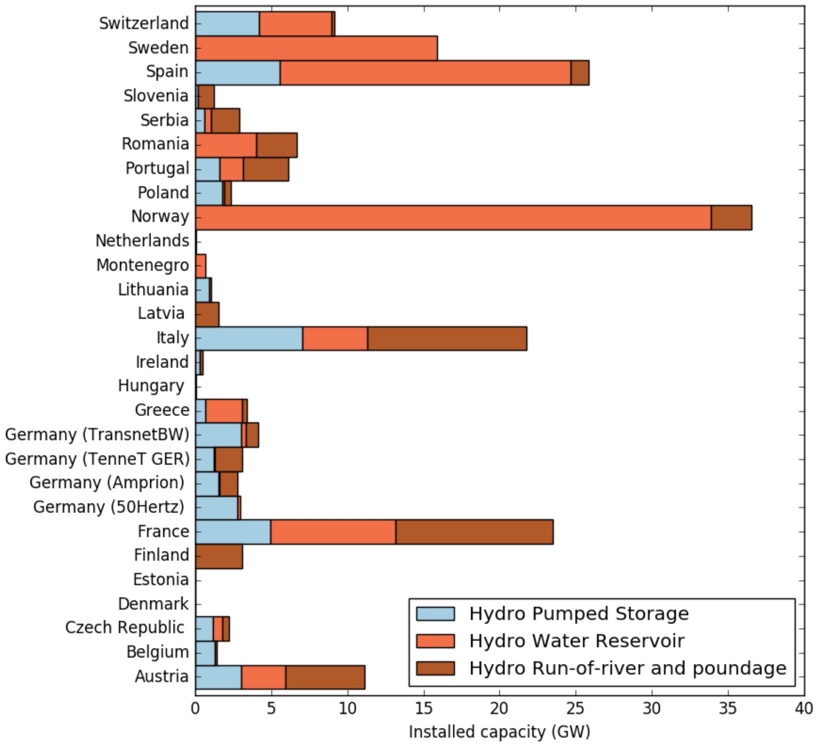


Figure 2. Total installed capacity per type of (hydro) plant and control zone

The hydrological data can also be gathered from other sources such as those listed below:

— The European catchments and Rivers network system (Ecrins) [16], which is a geographical information system of the European hydrographical systems with full topological information. It contains information on dams with reservoirs throughout Europe.

— The Waterbase – Rivers database [17], which contains information with mean river discharge, and cooling pressures.

— The Service for Water Indicators in Climate Change Adaptation (SWICCA [18], [19]) and the Operational Pan-European River Runoff (OPERR [20]) projects([4]), which provide forecasts for river flows, flow duration curve, or water temperature.

— The JRC's Catchment Characterisation and Modelling database [21], which includes a hierarchical set of river segments and catchments based on the Strahler order, a lake layer and structured hydrological feature codes based on the Pfafstetter system.

([3]) Units with a capacity equal or greater than 100 MW, according to Commission Regulation (EU) No 543/213, available at https://www.entsoe.eu/data/entso-e-transparency-platform/.
([4]) From the European Earth observation programme Copernicus http://copernicus.eu/

— The AQUASTAT-FAO [22]geo-referenced database of dams, very similar to the GRanD database used by LISFLOOD. Almost all the European dams included in this database have information regarding dam height, reservoir capacity (in volume), main uses, and geographical coordinates.

— US DOA FA service maintain a public database of reservoir water level from radar altimetry [23].

— Geth *et al.* [24] present an overview of large-scale electricity storage plants in Europe.

2 Mathematical model

The mathematical model is explained below. The notation is first provided in Section 2.1 and subsequently the formulation is described in Section 2.2. In Section 2.4, a description of the Monte Carlo simulations is explained. Finally, Section 2.4 describes how the input data should be given in the model. Note that Annex 1 provides the program documentation and Annex 2 lists the main model module source code for the interested reader.

2.1 Notation

The main notation used throughout this report is listed in Table 2.

Table 2. Model notation.

A. Indices	
h	Index of time (stage)
j	Auxiliary index
u	Index of units
B. Sets	
H	Set of time periods
U	Set of units
Ω_{hydro}	Set of hydro units
Ω_u	Set of upstream reservoirs of plant u
C. Parameters	
c_u	Variable cost (k€/GWh)
d_h	Demand (GW)
f^1	Conversion factor to convert m^3/s into Hm^3
f^2	Conversion factor to convert m^3/s into GWh
G_u^{max}	Maximum generation level (GW)
$head_u$	Nominal head (m)
N_H	Number of time periods
q_{hu}	Natural inflow (m^3/s)
RES_u^0	Initial water content (Hm^3)
RES_u^{max}	Maximum water content (Hm^3)
RES_u^{min}	Minimum water content (Hm^3)
Δt	Time step (h)
τ_u	Water transport delay
η_u	Roundtrip pumping efficiency
D. Variables	
CH_{hu}	Water charge (m^3/s)
$COST$	Objective function value (k€)
DIS_{hu}	Water discharge (m^3/s)
G_{hu}	Generation (GWh)
$PUMP_{hu}$	Pumped energy (GWh)
RES_{hu}	Reservoir level or water content (Hm^3)
$SPILL_{hu}$	Water spillage (m^3/s)
W_{hu}	Water value (€/Hm^3)

2.2 Formulation

The problem can be formulated as the following mathematical program:

$$\text{Minimize} \qquad COST = \sum_{h \in H} \sum_{u \in U} G_{hu} c_u \tag{1}$$

$$\sum_{u \in U} G_{hu} - PUMP_{hu} \geq d_h; \forall h \in H \tag{2}$$

$$RES_{hu} - RES_{h-1,u} = f^1 \left(q_{hu} + \eta_u CH_{hu} - DIS_{hu} - SPILL_{hu} + \sum_{j \in \Omega_u} \left(DIS_{h-\tau_u,j} + SPILL_{h-\tau_u,j} \right) \right)$$
$$: (W_{hu}); \forall u \in \Omega_{hydro}, \forall h \in H \tag{3}$$

$$G_{hu} = DIS_{hu} f^2 head_u; \forall u \in \Omega_{hydro}, \forall h \in H \quad (4)$$

$$PUMP_{hu} = CH_{hu} f^2 head_u; \forall u \in \Omega_{hydro}, \forall h \in H \quad (5)$$

$$RES_u^{min} \leq RES_{hu} \leq RES_u^{max}; \forall u \in \Omega_{hydro}, \forall h \in H \quad (6)$$

$$RES_{N_H,u} = RES_u^0; \forall u \in \Omega_{hydro} \quad (7)$$

$$0 \leq G_{hu} \leq G_u^{max} \Delta t; \forall u \in U, \forall h \in H \quad (8)$$

$$SPILL_{hu} \geq 0; \forall u \in \Omega_{hydro}, \forall h \in H. \quad (9)$$

The objective function (1) represents the total cost of operating the power system during the whole simulation period and is expressed as the sum of the variable costs of the generating units.

The generation-load balance is enforced in (2) so that the power produced by thermal, hydro, and renewable units minus the power that is pumped to the reservoirs (if available) must be greater than the demand. A slack power plant should be added in the data file to capture infeasibilities.

Constraint (3) represents the continuity equation by which the water balance is enforced for each hydropower plant and each time period. This balance takes into account the difference on the water volume of each reservoir, its natural inflow, the energy pumped (if any), the water release (production and spillage), and the water release from upstream reservoirs. The dual variables W_{hu} associated with these constraints represent the water value of each hydropower plant for each time period. Note that, to convert m^3/s into Hm^3, the factor f^1 is equal to $0.0036\,\Delta t$.

Equations (4) and (5) set the water-energy conversion for hydropower discharges and pumped power. A simple conversion unit approach is adopted by means of the conversion factor $f^2 = g\rho\Delta t/10^9$ to convert m^3/s into GWh. This would be modified to incorporate the water head effect of hydro reservoirs. This water head effect is represented in the Hill chart and links the water discharge, the reservoir level, and the power production. This effect is highly nonlinear and a precise model would be needed to accurately represent the Hill chart. Although a simple linear model could be adopted, the lack of publicly available data is a barrier to model this feature.

The lower and upper bounds on reservoir levels are imposed in (6) for each hydropower plant. The border condition is enforced in (7). Generation bounds on generation energy are imposed in (8) for each power plant and time period. Finally, the non-negativity of the water spillage is enforced in (9). Needless to say, the water spillage could be easily bounded by minimum and maximum limits representing regulations in force aimed at protecting the fauna and flora of the water channel.

This problem is characterized as a large-scale linear program and is solved by using the solver GLPK [24] in Pyomo [25], [26]. The formulated model is fully compatible with proprietary solver like CPLEX, GUROBI which are preferred for larger problems.

2.3 Monte Carlo simulations

The model (1)–(9) can be solved using historical years as input data. Individual deterministic runs can then be made for mean or extreme years, e.g. very dry or wet years. While deterministic scenario runs are sufficient for scenario analysis showing the expected estimation divergence among extreme scenarios, they cannot demonstrate the frequency of the incidents. The use of probabilistic analysis such as Monte Carlo can define with a known degree of confidence both the most possible results and the level of risk. The key idea behind the Monte Carlo simulation is to evaluate the model with a set of random parameters as inputs. These parameters are generated from the probability functions of the variables, thus mimicking the sampling procedure of the actual phenomena.

A typical Monte Carlo analysis based on the following steps is carried out:

— Extraction of inputs from a probability distribution according to the nature of the variable. If satisfying historical data, that could reproduce the behaviour of the variable in the future, are available then they can be used to fit an appropriate distribution function. Otherwise a more generic probability function (e.g. Normal, Lognormal, triangular etc) based on expert judgment is used to simulate the probability of such events.

— Calculation of desired outputs for many samples according to the desired confidence level.

— Illustration of the results in a probability distribution function and justification of the uncertainty.

Normally, given the set of input parameters and the accompanying equations, the output can be obtained. This means that, in each hour, some numbers of random values were chosen from the normal distribution of the variable. However, other statistical characteristics have to be usually taken into account for time series such as autocorrelation.

For the MTHC problem, the main source of uncertainty is related to the hydrological inflows. Figure 3 presents an example of historical analysis of inflows with a weekly time step.

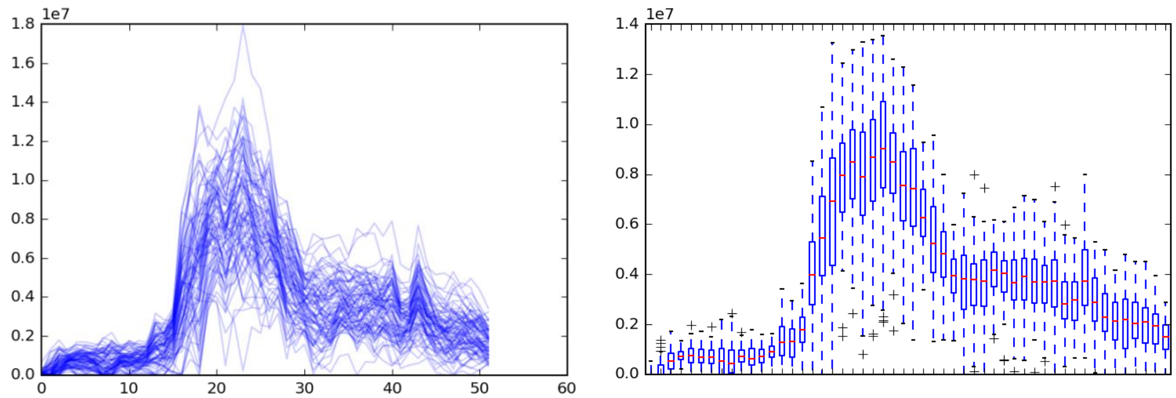


Figure 3. Example of analysis of weekly time series on the left plot. Export of means and standard deviation per time step on the right plot.

Different realization of inflows based on historical means and deviations can be generated by means of equation (10).

$$L_i = \mu_{L_i} + \sigma^2_{L_i} \mathbb{N}_i(0,1), \quad (10)$$

where *i* is the time step, *N* is the length of the time series, *L* is the inflow, *μ* the expected (mean) inflow for time step *i*, σ^2_i the expected variance for time step I and $\mathbb{N}(0,1)$ is a random number generated by a gaussian distribution having a mean of 0 and a standard deviation of 1.

However the above stochastic process does take into account the autocorrelations between two time steps producing as a result unrealistic time series. For that reason, a simple periodic autoregressive (PAR) model was used [28]. Currently a PAR(1) is implemented (also known as Thomas-Fiering or Gauss Markov) for stochastic description of water inflows. In order to do that, the following information is needed: lag-one autocorrelation, historical mean, and standard deviation. It has been proven that this stochastic process simulated this short term memory behaviour better than other processes of higher degree.

The Gauss-Markov stochastic process is formulated as in equation (11).

$$L_i = \begin{cases} \mu_{L_i} + \sigma^2_{L_i}\ \mathbb{N}_i(0,1); i = 1 \\ \mu_{L_i} + \rho \frac{\sigma^2_{L_i}}{\sigma^2_{L_{i-1}}} \left(L_{i-1} - \mu_{L_i}\right) + \sigma^2_{L_i} \sqrt{1-\rho^2} \mathbb{N}_i(0,1);\ i = 1..N' \end{cases} \quad (11)$$

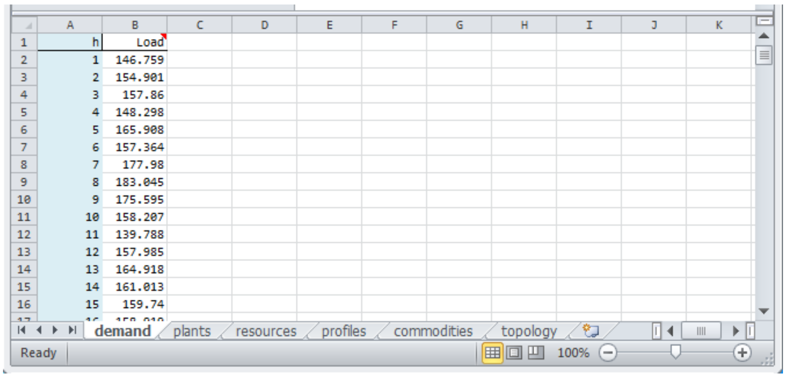
where i is the time step, N is the length of the time series, L is the inflow, μ the expected (mean) inflow for time step i, σ^2_i the expected variance for time step I, $\mathbb{N}(0,1)$ is a random number generated by a gaussian distribution having a mean of 0 and a standard deviation of 1 and ρ the autocorrelation coefficient (AR(1)).

2.4 Input files

An excel file is used as data input to the model. This file has the following sheets: demand, plants, resources, profiles, and topology. The first column is usually the time or the plant index. A YAML file is also used for the configuration of the model

2.4.1 Demand sheet

The demand spreadsheet (see Figure 4) has h rows, each row corresponds to a time step. The first column is the time index (has to be integer) and the second column the actual demand (GW).



h	Load
1	146.759
2	154.901
3	157.86
4	148.298
5	165.908
6	157.364
7	177.98
8	183.045
9	175.595
10	158.207
11	139.788
12	157.985
13	164.918
14	161.013
15	159.74

Figure 4. Screenshot of an example for the demand spreadsheet

2.4.2 Plants sheet

The plants spreadsheet (see Figure 5) has u rows and each row corresponds to each power plant. The following details are included per power plant:

— Plant id: name of the plant.

— Type: it has to be a member among 'Thermal', 'Hydro', 'Solar', 'Wind', 'Slack', or 'Other'. Note that 'Other' stands for other renewable generation. It is used for visualization and model building.

— Type2: it indicates a more specific type within the members in Type. In other words, 'Thermal' can be categorized within 'Fossil Brown coal/Lignite', 'Fossil Gas', or 'Fossil Oil'; and Type 2 for 'Hydro' should be 'Hydro Pumped Storage', 'Hydro Run-of-river and poundage', or 'Hydro Water Reservoir'. The rest of the members in Type 2 should adopt the same string.

— Pmin (GW): minimum power output.

— Pmax (GW): maximum power output.

— VarCost (k€/GWh): variable (operating) costs of each power plant.

For hydropower plants, the following information should also be added:

— Stmin (Hm^3): minimum water content.

— Stmax (Hm^3): maximum water content.

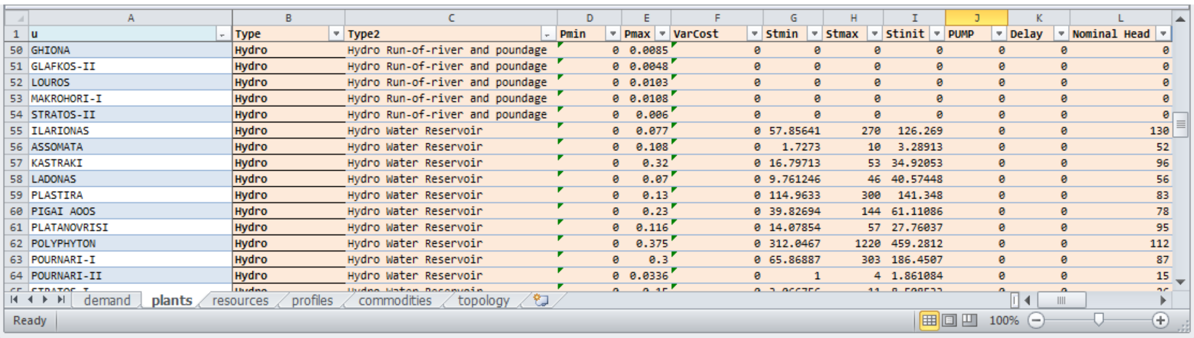
— Stinit (Hm^3): initial water content in the simulation. The model has flexibility whether imposing a border condition in which the final amount of water is going to be the initial water content or not.

— PUMP: roundtrip pumping efficiency for the hydro pumped storage.

— Delay: water transport delay from the hydro unit. It should be expressed in the units of the time step.

— Nominal Head (m): nominal head of the hydro unit.

The model is expressed in volume (Hm^3) and energy units (GWh).



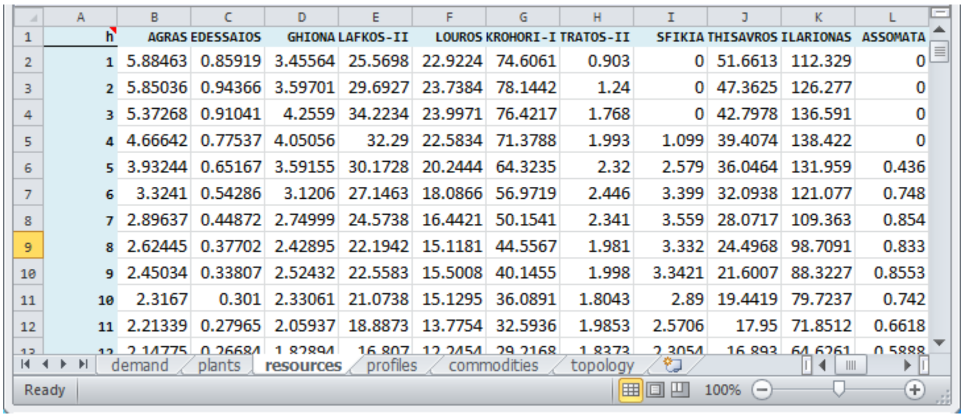
u	Type	Type2	Pmin	Pmax	VarCost	Stmin	Stmax	Stinit	PUMP	Delay	Nominal Head
GHIONA	Hydro	Hydro Run-of-river and poundage	0	0.0085	0	0	0	0	0	0	0
GLAFKOS-II	Hydro	Hydro Run-of-river and poundage	0	0.0048	0	0	0	0	0	0	0
LOUROS	Hydro	Hydro Run-of-river and poundage	0	0.0103	0	0	0	0	0	0	0
MAKROHORI-I	Hydro	Hydro Run-of-river and poundage	0	0.0108	0	0	0	0	0	0	0
STRATOS-II	Hydro	Hydro Run-of-river and poundage	0	0.006	0	0	0	0	0	0	0
ILARIONAS	Hydro	Hydro Water Reservoir	0	0.077	0	57.85641	270	126.269	0	0	130
ASSOMATA	Hydro	Hydro Water Reservoir	0	0.108	0	1.7273	10	3.28913	0	0	52
KASTRAKI	Hydro	Hydro Water Reservoir	0	0.32	0	16.79713	53	34.92053	0	0	96
LADONAS	Hydro	Hydro Water Reservoir	0	0.07	0	9.761246	46	40.57448	0	0	56
PLASTIRA	Hydro	Hydro Water Reservoir	0	0.13	0	114.9633	300	141.348	0	0	83
PIGAI AOOS	Hydro	Hydro Water Reservoir	0	0.23	0	39.82694	144	61.11086	0	0	78
PLATANOVRISI	Hydro	Hydro Water Reservoir	0	0.116	0	14.07854	57	27.76037	0	0	95
POLYPHYTON	Hydro	Hydro Water Reservoir	0	0.375	0	312.0467	1220	459.2812	0	0	112
POURNARI-I	Hydro	Hydro Water Reservoir	0	0.3	0	65.86887	303	186.4507	0	0	87
POURNARI-II	Hydro	Hydro Water Reservoir	0	0.0336	0	1	4	1.861084	0	0	15

Figure 5. Screenshot of an example for the plants spreadsheet

2.4.3 Resources sheet

The resources spreadsheet (see Figure 6) should have h rows and as many columns as hydropower plants. All plants with a 'Hydro' type must be included here.

Each column has an inflow time series in m3/s linked to the hydropower plants indicated in the header. The header names have to match those in the plants spreadsheet.



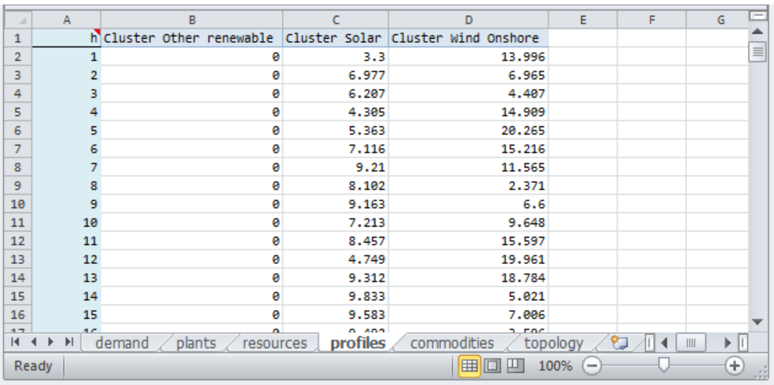
h	AGRAS	EDESSAIOS	GHIONA	LAFKOS-II	LOUROS	KROHORI-I	TRATOS-II	SFIKIA	THISAVROS	ILARIONAS	ASSOMATA
1	5.88463	0.85919	3.45564	25.5698	22.9224	74.6061	0.903	0	51.6613	112.329	0
2	5.85036	0.94366	3.59701	29.6927	23.7384	78.1442	1.24	0	47.3625	126.277	0
3	5.37268	0.91041	4.2559	34.2234	23.9971	76.4217	1.768	0	42.7978	136.591	0
4	4.66642	0.77537	4.05056	32.29	22.5834	71.3788	1.993	1.099	39.4074	138.422	0
5	3.93244	0.65167	3.59155	30.1728	20.2444	64.3235	2.32	2.579	36.0464	131.959	0.436
6	3.3241	0.54286	3.1206	27.1463	18.0866	56.9719	2.446	3.399	32.0938	121.077	0.748
7	2.89637	0.44872	2.74999	24.5738	16.4421	50.1541	2.341	3.559	28.0717	109.363	0.854
8	2.62445	0.37702	2.42895	22.1942	15.1181	44.5567	1.981	3.332	24.4968	98.7091	0.833
9	2.45034	0.33807	2.52432	22.5583	15.5008	40.1455	1.998	3.3421	21.6007	88.3227	0.8553
10	2.3167	0.301	2.33061	21.0738	15.1295	36.0891	1.8043	2.89	19.4419	79.7237	0.742
11	2.21339	0.27965	2.05937	18.8873	13.7754	32.5936	1.9853	2.5706	17.95	71.8512	0.6618

Figure 6. Screenshot of an example for the resources spreadsheet

2.4.4 Profiles sheet

The profiles spreadsheet (see Figure 7) should have h rows and as many columns as renewable units or clusters. All plants with 'Solar', 'Wind', or 'Other' type must be included here.

Each column has a profile time series in GWh linked to the units or clusters indicated in the header.

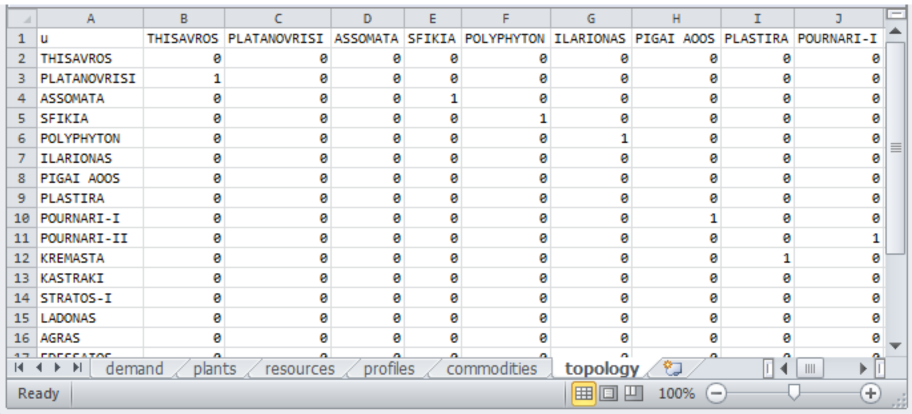


h	Cluster Other renewable	Cluster Solar	Cluster Wind Onshore
1	0	3.3	13.996
2	0	6.977	6.965
3	0	6.207	4.407
4	0	4.305	14.909
5	0	5.363	20.265
6	0	7.116	15.216
7	0	9.21	11.565
8	0	8.102	2.371
9	0	9.163	6.6
10	0	7.213	9.648
11	0	8.457	15.597
12	0	4.749	19.961
13	0	9.312	18.784
14	0	9.833	5.021
15	0	9.583	7.006

Figure 7. Screenshot of an example for the profiles spreadsheet

2.4.5 Topology sheet

The topology spreadsheet (see Figure 8) includes the adjacency matrix of the hydrological network. It should have u rows and u columns corresponding only to the hydro units. Each element of the matrix has a value equal to 1 if the unit in row u is located downstream of the unit in column u; otherwise the value is equal to 0.



u	THISAVROS	PLATANOVRISI	ASSOMATA	SFIKIA	POLYPHYTON	ILARIONAS	PIGAI AOOS	PLASTIRA	POURNARI-I
THISAVROS	0	0	0	0	0	0	0	0	0
PLATANOVRISI	1	0	0	0	0	0	0	0	0
ASSOMATA	0	0	0	1	0	0	0	0	0
SFIKIA	0	0	0	0	1	0	0	0	0
POLYPHYTON	0	0	0	0	0	1	0	0	0
ILARIONAS	0	0	0	0	0	0	0	0	0
PIGAI AOOS	0	0	0	0	0	0	0	0	0
PLASTIRA	0	0	0	0	0	0	0	0	0
POURNARI-I	0	0	0	0	0	0	1	0	0
POURNARI-II	0	0	0	0	0	0	0	0	1
KREMASTA	0	0	0	0	0	0	0	1	0
KASTRAKI	0	0	0	0	0	0	0	0	0
STRATOS-I	0	0	0	0	0	0	0	0	0
LADONAS	0	0	0	0	0	0	0	0	0
AGRAS	0	0	0	0	0	0	0	0	0

Figure 8. Screenshot of an example for the topology spreadsheet

2.4.6 Configuration file

A machine readable configuration file (YAML format) is also used in order to customize the model with the necessary assumptions (see Figure 9). There are options related to the model formulation:

- Time step duration
- Flag: The reservoir levels should match the initial reservoir levels
- Flag: Consider hydrological network

And options related to the solver itself:

- Solver type: free solvers (GLPK, CBC) or proprietary solvers can be used (CPLEX, GUROBI)
- Solver manager: Single instance or parallel
- Symbolic labels: Export formulate linear programming code with meaningful names
- tee: Output all iteration of solver solution
- Other solver-specific options (e.g. maximum time to run). These options will be passed directly to the solver depend on the solver used.

```yaml
casename: 'Greece_avg'

problem_type: 'deterministic'  #or monte_carlo

model_flags:
  timestep_duration: 24 # seconds
  isLastEqFirstStage: True
  network: True

solver_cfg:
  solver: 'glpk'
  solver_manager: 'serial'  # or parallel
  symbolic_solver_labels: True  # better output in lp file but a but slower
  tee: True  # output solver
  options_string: "tmlim=100 "

monte_carlo:
  N: 2  # number of iterations

log_file: "" #keep empty for screen output

```

Figure 9. Screenshot of the options available in the configuration file

3 Conclusion

This work presents a deterministic single-bus model for the medium-term hydrothermal coordination problem with daily, weekly or monthly time steps. The optimization horizon can range from 1 year to several years. The model includes hydro-specific features such as (i) the continuity equation in water units, (ii) bounds on water release, spillage, and reservoir levels, as well as (iii) the consideration of the hydraulic network with water time delays.

Thermal generation can be either aggregated or disaggregated in the model and it allows the modelling of dispatch constraints. These constraints are limited to generation bounds only because the hydrothermal coordination model is linked to the dispatch and unit commitment Dispa-SET model, which accurately reflects the detailed technical features of those units. Also, the proposed single-bus model enforces the power balance in energy units and the link between energy and water units. Therefore, the problem is characterized as a linear programming problem.

This model provides the generation dispatch of thermal, renewable, and hydropower units, as well as the reservoir levels of the hydropower plants for all periods in the medium term. The operation of hydropower plants is passed on to the Dispa-SET model in order to compute the daily production planning of the system. These results could be useful to analyse not only the production planning under different scenarios of water inflows in a hydro-dominant power system, but also the water-power nexus by complementing the previous models with the hydrological LISFLOOD model.

Further work will be devoted to the following directions:

— The explicit separation of constraints in water and energy units would allow for a representation of the water head effect of hydro reservoirs. In other words, the Hill chart linking the water discharge, reservoir level, and power production could be modelled as long as data are publicly available.

— The medium-term hydrothermal coordination problem is essentially stochastic due to the uncertain water inflows. Thus, a suitable scenario generation method and scenario reduction techniques need to be implemented. Also, the method will be extended to incorporate stochastic inflows in the problem formulation.

— Aggregation/disaggregation of hydropower plants belonging to the same river basin to deal with the tractability of the problem in large-scale hydro-dominant power systems.

— Link to the hydrological LISFLOOD model so that the water-power interactions could be taken into account.

References

[1] P. Burek, J. van der Knijff and A. de Roo, "LISFLOOD: Distributed Water Balance And Flood Simulation Model: Revised User Manual," European Commission, Joint Research Centre, Ispra (Italy), 2013.

[2] I. Hidalgo González, S. Quoilin and A. Zucker, "Dispa-SET 2.0: Unit commitment and power dispatch model," European Commission, Joint Research Centre, Petten (The Netherlands), 2014.

[3] M. V. F. Pereira and L. M. V. G. Pinto, "Stochastic optimization of a multireservoir hydroelectric system: A decomposition approach," *Water Resources Research,* vol. 21, no. 6, pp. 779-792, 1985.

[4] M. V. F. Pereira, "Optimal stochastic operations scheduling of large hydroelectric systems," *CEPEL - Centro de Pesquisas de Energia Elétrica,* vol. 11, no. 3, pp. 161-169, 1989.

[5] M. V. F. Pereira and L. M. V. G. Pinto, "Multi-stage stochastic optimization applied to energy planning," *Mathematical Programming,* vol. 52, no. 1, pp. 359-375, 1991.

[6] A. Gjelsvik, B. Mo and A. Haugstad, "Long- and medium-term operations planning and stochastic modelling in hydro-dominated power systems based on stochastic dual dynamic programming," in *Handbook of Power Systems I*, Berlin, Springer-Verlag, 2010, pp. 33-55.

[7] J.-P. Watson and D. L. Woodruff, "Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems," *Computational Management Science,* vol. 8, no. 4, pp. 355-370, 2011.

[8] A. R. d. Queiroz, "Stochastic hydro-thermal scheduling optimization: An overview," *Renewable and Sustainable Energy Reviews,* vol. 62, pp. 382-395, 2016.

[9] A. Shapiro, "Analysis of stochastic dual dynamic programming method," *European Journal of Operational Research,* vol. 209, no. 1, pp. 63-72, 2011.

[10] D. M. Falcao, C. L. T. Borges and G. N. Taranto, "High Performance Computing in Electrical Energy Systems Applications," in *High Performance Computing in Power and Energy Systems*, Springer Berlin Heidelberg, 2013, pp. 1-42.

[11] R. E. C. Gonçalves, E. C. Finardi, E. L. da Silva and M. L. L. dos Santos, "Comparing stochastic optimization methods to solve the medium-term operation planning problem," *Computational & Applied Mathematics,* vol. 30, no. 2, pp. 289-313, 2011.

[12] M. Uhr and M. Morari, *Optimal Operation of a Hydroelectric Power System Subject to Stochastic Inflows and Load,* 2006.

[13] R. E.C. Gonçalves, E. Cristian Finardi and E. Luiz da Silva, "Applying different decomposition schemes using the progressive hedging algorithm to the operation planning problem of a hydrothermal system," *Electric Power Systems Research,* vol. 83, no. 1, pp. 19-27, 2012.

[14] A. Tilmant and R. Kelman, "A stochastic approach to analyze trade-offs and risks associated with large-scale water resources systems," *Water Resources Research,* vol. 43, no. 6, 2007.

[15] "European Network of Transmission System Operators for Electricity," [Online]. Available: https://www.entsoe.eu/. [Accessed November 2016].

[16] "European catchments and Rivers network system (Ecrins)," [Online]. Available: http://www.eea.europa.eu/data-and-maps/data/european-catchments-and-rivers-network. [Accessed November 2016].

[17] E. E. Agency, "Waterbase – Rivers," [Online]. Available: http://www.eea.europa.eu/data-and-maps/data/waterbase-rivers-10#tab-european-data. [Accessed November 2016].

[18] Copernicus, "Service for Water Indicators in Climate Change Adaptation (SWICCA)," [Online]. Available: http://www.swicca.eu/start/climate-indicators/. [Accessed November 2016].

[19] Copernicus, "Service for Water Indicators in Climate Change Adaptation (SWICCA)," [Online]. Available: http://www.swicca.eu/indicator-interface/maps/ . [Accessed November 2016].

[20] "Operational Pan-European River Runoff (OPERR)," [Online]. Available: http://www.smhi.se/en/research/research-departments/oceanography/operr-operational-pan-european-river-runoff-1.16820. [Accessed November 2016].

[21] J. R. C. CCM2, "European rivers and catchments," [Online]. Available: http://data.jrc.ec.europa.eu/dataset/fe1878e8-7541-4c66-8453-afdae7469221/resource/a433cff9-995d-409b-85ff-10de27d80641. [Accessed November 2016].

[22] AQUASTAT-FAO. [Online]. Available: http://www.fao.org/nr/water/aquastat/dams/index.stm . [Accessed November 2016].

[23] U. S. D. o. A. -. F. A. Service, "Global reservoirs/lakes (G-REALM)," [Online]. Available: http://www.pecad.fas.usda.gov/cropexplorer/global_reservoir/ . [Accessed November 2016].

[24] F. Geth, T. Brijs, J. Kathan, J. Driesen and R. Belmans, "An overview of large-scale stationary electricity storage plants in Europe: Current status and new developments," *Renewable and Sustainable Energy Reviews,* vol. 52, pp. 1212-1227, 2015.

[25] "GLPK (GNU Linear Programming Kit)," [Online]. Available: https://www.gnu.org/software/glpk/. [Accessed November 2016].

[26] W. E. Hart, C. Laird, J.-P. Watson and D. L. Woodruff, Pyomo - Optimization Modeling in Python, Springer, 2012.

[27] W. E. Hart, J.-P. Watson and D. L. Woodruff, "Pyomo: modeling and solving mathematical programs in Python," *Mathematical Programming Computation,* vol. 3,

no. 3, pp. 219-260, 2011.

[28] F. L. C. Oliveiraa, R. C. Souzab and A. L. M. Marcatoc, "A time series model for building scenarios trees applied to stochastic optimisation," *International Journal of Electrical Power & Energy Systems,* vol. 67, pp. 315-323, 2015.

[29] A. M. Breipohl, F. N. Lee, D. Zhai and R. Adapa, "A Gauss-Markov load model for the application in risk evaluation and production simulation," *IEEE Transactions on Power Systems,* vol. 7, no. 4, pp. 1493-1499, 1992.

List of abbreviations and definitions

Dispa-SET	Unit commitment and dispatch model.
LISFLOOD	Hydrological model.
MTHC	Medium-term hydrothermal coordination model.
PS	Progressive hedging.
SAA	Sample average approximation.
SDDP	Stochastic dual dynamic programming.
WATERFLEX	Exploratory research project.

List of figures

List of tables

Annex 1: Documentation

This annex provides the programmable interface documentation for the model application. This documentation encompasses seven modules that respectively contains the deterministic mathematical model, the Gauss-Markov load model to generate inflows scenarios, the main run module, the input/output module, the module for post-processing results, and the helper functions.

Model module

waterflex.model.create_model(*data*, *conf*)[source]

Create Pyomo object based on input.

Parameters:

(1) **data** (*dict*) – Dictionary with pandas dataframes with all data

(2) **conf** (*dict*) – Dictionary with model options

Returns: Pyomo model instance

waterflex.model.run_solver(*instance*, *conf*)[source]

Method to solve a pyomo instance.

Parameters:

(1) **instance** – Pyomo unsolved instance

(2) **solver** (*str*) – solver to use. Select between (glpk, cplex, gurobi, cbc et.c)

(3) **solver_manager** (*str*) – serial or pyro

(4) **tee** (*bool*) – if True a detailed solver output will be printed on screen

(5) **options_string** – options to pass to solver

Stochastic module

waterflex.stochastics.GaussMarkov(*mu*, *st*, *r*)

A simple PAR(1) [29].

Parameters:

(1) **mu** – vector of historical time series means

(2) **st** – vector of historical standard deviations

(3) **r** – lag one correlation factor

Returns: Realization of time series

Run module

Main script to run routines from WATERFLEX library.

`run.run_monte_carlo`(*data_filename*, *config*, *results_dir='./results/'*)

Run Monte Carlo (SES) with Gauss-Markov inflow generation.

Parameters:

(1) **data_filename** (*str*) – excel input data

(2) **N** (*int*) – number of scenarios to generate

(3) **results_dir** (*str*) – directory to store results

(4) **solver** (*str*) – solver to use (CPLEX, glpk, etc.)

`run.run_once`(*data_filename*, *config*, *results_dir='./results/'*, *solver='cplex'*)

Run one instance of WATERFLEX model.

Parameters:

(1) **data_filename** – excel input data

(2) **solver** – solver to use (CPLEX, glpk, etc.)

Input/Output module

`waterflex.io.consistency_check`(*data*)[source]

Check input file for consistency errors.

Parameters: **data** – data dictionary to be passed in pyomo model

`waterflex.io.load`(*filename*)[source]

Load a model instance from a pickle file.

Parameters: **filename** – pickle file

Returns: the unpickled model instance

`waterflex.io.parse_excel`(*filename*)[source]

Read Excel and prepare input pandas.

Parameters: **filename** – filename of excel file according to template

Returns: Dictionary with input ready to be processed by Pyomo

`waterflex.io.read_yaml`(*filename*)[source]

Loads YAML file to dictionary.

`waterflex.io.save`(*instance*, *filename*)[source]

Save model instance to pickle file.

Parameters:

(1) **instance** – a model instance

(2) **filename** – pickle file to be written

Postprocess module

`waterflex.postprocess.calc_costs`(*g*, *var_costs*)[source]

Calculate specific system cost per time step (€/kWh). Weighted average of costs for all dispatched generation units.

Parameters:

(1) **g** (*pandas*) – Generation matrix per time step and power plant

(2) **var_costs** (*pandas*) – (Average) variables costs per plant

Returns: System cost time series

`waterflex.postprocess.calc_prices`(*g*, *var_costs*)

Calculate marginal price per time step (€/kWh). The variable cost of the most expensive technology dispatched is considered per time step.

Parameters:

(1) **g** (*pandas*) – Generation matrix per time step and power plant

(2) **var_costs** (*pandas*) – (Average) variables costs per plant

Returns: Marginal price time series

Returns: Installed capacity per type

Return type: Figure with 1 subplot

`waterflex.postprocess.generate_plot_monte`(*result_dict_RES*, *dir='./results/'*, *fontsize=12*, *cbrewer_palette='Set2'*)

Plot that shows all generation realizations and the expected value.

Parameters:

(1) **result_dict_RES** (*dict*) – dictionary with Reservoir levels per unit and per scenario

(2) **dir** (*dir*) – directory to save plot

(3) **fontsize** (*int*) – Size of fonts

(4) **cbrewer_palette** (*str*) – Palette for plot. Works only if seaborn is installed. Check http://colorbrewer2.org/ for nicely looking palettes.

waterflex.postprocess.generate_plot_once(*instance*, *grouped=True*, *dir='./results/'*, *fontsize=12*, *cbrewer_palette='Set2'*)

Create plots from solved instance.

Parameters:

(1) **instance** (*pyomo*) – Solved model instance

(2) **grouped** (*bool*) – if True then it will be plotted by plant type (column 'Type' of input spreadsheet)

(3) **dir** (*str*) – Directory to save plt

(4) **fontsize** (*int*) – Size of fonts

(5) **cbrewer_palette** (*str*) – Palette for plot. Works only if seaborn is installed. Check http://colorbrewer2.org/ for nicely looking palettes.

Returns: Generation mix, Reservoir levels, variable costs

Return type: Figure with 3 subplots

waterflex.postprocess.write_result_spreadsheet(*data*, *instance*, *solver_status*, *dir='./results/'*)

Create report from solved instance.

Parameters:

(1) **instance** (*pyomo*) – Pyomo solved instance

(2) **solver_status** – Solver status information

(3) **dir** (*str*) – directory for report

Returns: Path of Resultfile

Helper functions module

waterflex.helpers.dict_pandas_to_excel(*dictionary*, *dir='./results/'*, *filename='dict.xlsx'*)

Convert a dictionary of pandas to excel file.

Parameters:

(1) **dictionary** (*dict*) – dictionary to be exported

(2) **dir** (*str*) – directory to store the excel file

(3) **filename** (*str*) – filename of excelfile

waterflex.helpers.dict_to_excel(*dictionary*, *dir='./results/'*, *filename='dict.xlsx'*)

Convert a dictionary of scalars to excel file.

Parameters:

(1) **dictionary** (*dict*) – dictionary to be exported

(2) **dir** (*str*) – directory to store the excel file

(3) **filename** (*str*) – filename of excel file

waterflex.helpers.get_set_members(*instance*, *sets*)

Get set members that belong to this set.

Parameters:

(1) **instance** – Pyomo Instance

(2) **sets** – Pyomo Set

Returns: A list with the set members

waterflex.helpers.get_sets(*instance*, *var*)

Get sets that belong to a pyomo Variable or Param.

Parameters:

(1) **instance** – Pyomo Instance

(2) **var** – Pyomo Var (or Param)

Returns: A list with the sets that belong to this Param

waterflex.helpers.pyomo_to_pandas(*instance*, *var*)

Function converting a pyomo variable or parameter into a pandas dataframe. The variable must have one or two dimensions and the sets must be provided as a list of lists.

Parameters:

(1) **instance** – Pyomo Instance

(2) **var** – Pyomo variable

Returns: Instance in pandas Dataframe format

`waterflex.helpers.pyomo_to_pandas_const`(*instance*, *const*)

Function converting a dual variable associated with a constraint into a pandas dataframe. The dual variable must have one or two dimensions and the sets must be provided as a list of lists.

Parameters:

(1) **instance** – Pyomo Instance

(2) **const** – Pyomo constraint

Returns: Instance in pandas Dataframe format

Annex 2: Main model module source code

In this annex, the main model module source code is provided below.

```
import logging
import pyomo.environ as pe


def create_model(data, conf):
    """ Create Pyomo object based on input

    Parameters:
        data (dict): Dictionary with pandas dataframes with all data
        conf (dict): Dictionary with model options

    Returns:
        Pyomo model instance

    """
    m = pe.ConcreteModel('WaterFlex')

    m.demand = data['demand']
    m.plants = data['plants']
    m.inflows = data['resources']
    m.profiles = data['profiles']
    m.topology = data['topology']
    m.spmin = data['spillage_min']    m.spmax = data['spillage_max']
    m.filename = data['info']['filename']
    m.created_time = data['info']['created_time']

    #Get data from config file
    m.casename = conf['casename']

    m.isLastEqFirstStage =  conf['model_flags']['isLastEqFirstStage']
    m.network = conf['model_flags']['network']
    m.dt = conf['model_flags']['timestep_duration']

    # Sets
    m.h = pe.Set(initialize=m.demand.index.get_level_values('h').unique(),
                 ordered=True, doc='Time')
    m.u = pe.Set(initialize=m.plants.index.get_level_values('u').unique(),
                 doc='Plants')

    # Parameters
    m.gravity = pe.Param(initialize=9.81, doc='Gravity constant (m/s2)')
    m.density = pe.Param(initialize=1000, doc='Water density (kg/m3)')
    m.factor1 = pe.Param(initialize=(0.0036)*m.dt, doc='Conversion factor from m3/s to Hm3')
    m.factor2 = pe.Param(initialize=(m.gravity * m.density * 3600 * m.dt)/(3.6*10**12), doc='Conversion
factor to convert m3/s into GWh')

    # Variables
    m.G = pe.Var(m.h, m.u,
                 bounds=gen_bounds, within=pe.NonNegativeReals,
                 doc='Generated energy in timestep h by plant u (GWh)')
    m.PUMP = pe.Var(m.h, m.u,  # bounds?
                    within=pe.NonNegativeReals,
                    doc='Pumping(storing) of energy to reservoir (GWh)')
    m.DIS = pe.Var(m.h, m.u, # bounds?
                   within=pe.NonNegativeReals,
                   doc='Water discharge in timestep h by plant u (m3/s)')
    m.CH = pe.Var(m.h, m.u,  # bounds?
                  within=pe.NonNegativeReals,
                  doc='Water charge at hour h to reservoir u  (m3/s)')
    m.RES = pe.Var(m.h, m.u,
                   bounds=storage_bounds, within=pe.NonNegativeReals,
                   doc='Storage of hour h and plant u (Hm3)')
    m.SPILL = pe.Var(m.h, m.u,
                     within=pe.NonNegativeReals,
                     doc='Spill of hour h and plant u (m3/s)')
    m.FILL = pe.Var(m.h, m.u,
                    within=pe.NonNegativeReals,
                    doc='Slack Var: Fill of hour h and plant u (m3/s)')
    m.RELAX = pe.Var(m.h, m.u,
```

```
                      within=pe.NonNegativeReals,
                      doc='Relaxation of max spill bound at hour h and plant u (m3/s)')

    # Import dual variables into suffix data
    m.dual = pe.Suffix(direction=pe.Suffix.IMPORT)

    # Constraints
    m.cont = pe.Constraint(m.h, m.u, rule=cont_rule, doc='Continuity Equation')
    m.demSat = pe.Constraint(m.h, rule=dem_sat_rule, doc='Demand Satisfaction')
    m.conv_gen = pe.Constraint(m.h, m.u, rule=conv_gen_rule, doc='Conversion of units')
    m.conv_pump = pe.Constraint(m.h, m.u, rule=conv_pump_rule, doc='Conversion of units')

    m.obj = pe.Objective(rule=obj_rule, sense=pe.minimize, doc='minimize(cost = sum of all costs)')

    logging.info("Model prepared")
    return m


####################################################
# Equations
####################################################


# Constraints
# Min Max generation
def gen_bounds(m, h, u):
    variable_res = [u'Wind', u'Solar', u'Other']  # have to be unicode!
    if m.plants.at[u, "Type"] in variable_res:
        max_gen = min(m.plants.at[u, "Pmax"] * m.dt, m.profiles.at[h, u])
    else:
        max_gen = m.plants.at[u, "Pmax"] * m.dt
    return m.plants.at[u, "Pmin"] * m.dt * 0, max_gen

# Min Max storage
def storage_bounds(m, h, u):
    return m.plants.at[u, "Stmin"], m.plants.at[u, "Stmax"]

# Max spillage
def max_spill_bound_rule(m, h, u):
    if m.plants.at[u, "Type"] == 'Hydro':
        return m.SPILL[h, u] <= m.spmax.at[h, u] + m.RELAX[h, u]
    else:
        return pe.Constraint.Skip

# Min spillage
def min_spill_bound_rule(m, h, u):
    if m.plants.at[u, "Type"] == 'Hydro':
        return m.SPILL[h, u] >= m.spmin.at[h, u]
    else:
        return pe.Constraint.Skip

# Continuity rule
def cont_rule(m, h, u):
    if m.plants.at[u, "Type"] == 'Hydro':  # TODO if u in m.hydro(subset of m.u)
        if "Pump" not in m.plants.columns:
            eta_pump = 0  # 0 efficiency if it cannot pump
        else:
            eta_pump = m.plants.at[u, "Pump"]
        # Set reservoir boundary conditions
        if m.network == False:
            balance_rhs = m.factor1 * (m.inflows.at[h, u] - m.DIS[h, u] + eta_pump * m.CH[h, u] -
m.SPILL[h, u] + m.FILL[h, u])
        else:
            balance_rhs = m.factor1 * (m.inflows.at[h, u] + sum((m.DIS[h-m.plants.at[k, "Delay"], k] +
m.SPILL[h-m.plants.at[k, "Delay"], k]) * m.topology.at[u, k]
                                                                 if m.plants.at[k,"Type"] == 'Hydro'
else 0 for k in m.u)
                           + eta_pump * m.CH[h, u] - m.DIS[h, u] - m.SPILL[h, u] + m.FILL[h, u])

        if h == m.h[1]:  # First period
            return m.RES[h, u] - m.plants.at[u, "Stinit"] == balance_rhs
        elif h == m.h[len(m.h)] and m.isLastEqFirstStage:  # Last period storage level same as last?
            m.RES[h, u] = m.plants.at[u, "Stinit"]  # necessary to assign so that it is included in
results
```

```python
            return m.plants.at[u, "Stinit"] - m.RES[h - 1, u] == balance_rhs
        else:
            return m.RES[h, u] - m.RES[h-1, u] == balance_rhs
    else:
        return pe.Constraint.Skip


# Demand satisfaction rule
def dem_sat_rule(m, h):
    return (sum(m.G[h, u] for u in m.u) -
            sum(m.PUMP[h, u] if m.plants.at[u, "Type"] == 'Hydro' else 0
                for u in m.u) >=  # CHECK! only 'Hydro' can pump
            m.demand.at[h, "Load"])

# Conversion of units rule for hydropower generation
def conv_gen_rule(m, h, u):
    if m.plants.at[u, "Type"] == 'Hydro':
        if m.plants.at[u, "Nominal Head"] != 0:
            return m.G[h, u] == m.DIS[h,u] * m.plants.at[u, "Nominal Head"] * m.factor2
        # We assume nominal head equal to 1 for run of river
        else:
            return m.G[h, u] == m.DIS[h,u] * m.factor2
    else:
        return pe.Constraint.Skip

# Conversion of units rule for pumping
def conv_pump_rule(m, h, u):
    if m.plants.at[u, "Type"] == 'Hydro' and m.plants.at[u, "PUMP"] != 0:
        return m.PUMP[h, u] == m.CH[h,u] * m.plants.at[u, "Nominal Head"] * m.factor2
    else:
        return pe.Constraint.Skip

# Objective Function
def obj_rule(m):
    return sum(m.G[l, k] * m.plants.at[k,"VarCost"] + (m.FILL[l, k] + m.RELAX[l, k])*1000000+m.SPILL[l,
k]*0.001
               for k in m.u
               for l in m.h)
```

KJ-NA-28419-EN-N

JRC Mission

As the science and knowledge service of the European Commission, the Joint Research Centre's mission is to support EU policies with independent evidence throughout the whole policy cycle.

EU Science Hub
ec.europa.eu/jrc

@EU_ScienceHub

EU Science Hub - Joint Research Centre

Joint Research Centre

EU Science Hub

Publications Office

doi: 10.2760/386964

ISBN 978-92-79-65073-4