Decomposition methods for a spatial model for long-term energy pricing problem

Philippe Mahey, Jonas Koko, Arnaud Lenoir

Decomposition methods for a spatial model for long-term energy pricing problem

Philippe Mahey[1], Jonas Koko[1], and Arnaud Lenoir[2]

[1]LIMOS, Université Clermont 2 – CNRS UMR 6158, BP 10448, 63000 Clermont-Ferrand, France
[2]EDF R& D, Clamart, France

Abstract

We consider an energy production network with zones of production and transfer links. Each zone representing an energy market (a country, part of a country or a set of countries) has to satisfy the local demand using its hydro and thermal units and possibly importing and exporting using links connecting the zones.

Assuming that we have the appropriate tools to solve a single zonal problem (approximate dynamic programming, dual dynamic programming, etc.), the proposed algorithm allows us to coordinate the productions of all zones.

We propose two reformulations of the dynamic model which lead to different decomposition strategies. Both algorithms are adaptations of known monotone operator splitting methods, namely the Alternating Direction Method of Multipliers and the Proximal Decomposition algorithm which have been proved to be useful to solve convex separable optimization problems. Both algorithms present similar performance in theory but our numerical experimentation on real-size dynamic models have shown that Proximal Decomposition is better suited to the coordination of the zonal subproblems, becoming a natural choice to solve the dynamic optimization of the European electricity market.

1 Introduction

In order to forecast electricity prices for one or several coupled markets, one can simulate the markets behavior. This is usually done by modeling the supply (production units) and the demand (electricity demand forecast) and assuming some market rules. For instance, we will assume in this paper a perfect market situation. From the classical micro-economic theory, this is equivalent to one lead optimizer who is responsible for minimizing the overall cost of the network.

The present model includes energy production planning on a multiperiod horizon projected into a far future (2025) and a network of production zones, interconnected by energy transactions, like in the European Market for electricity. It results in a large-scale multistage optimization problem. In order to simulate the behavior of the production network and to estimate the interzonal prices of electricity in a far future, we need to determine strategies for the management of the yearly cycle of European water dams. Such a strategy can theoretically be computed via dynamic programming where the zonal demands, water inflows and variable renewable generation are random processes. The state variable is the vector of aggregated zonal dams levels. Its size typically lies between 10 and 20 making the computation of the optimal Bellman functions hopeless. To avoid the curse of

dimensionality, a relaxed version of the problem involving a simplification of the information exchanged between zones can be solved by coordination of zonal dynamic programs giving rise to zonal strategies. Indeed, it is well known that the computational cost of Dynamic Programming depends on the number of possible states at each stage (for example, for a 20-dimensional state, if we discretize each component into 50 values, we will need to consider 50^{20} choices !). Thus, decomposing by zone will reduce this huge number to a tractable number of states (no more than 3 states, including the independent noise variables, in our model). On the other hand, if we decide to decompose w.r.t. scenarios, we will face the exponential growth of the scenario tree w.r.t. the number of stages (365 days in our case study). In other words, we aim at solving the dynamic but deterministic spatial model using a fast and decentralized solution strategy which will be embedded in a huge scenario-based model in a future work.

Then we have to think about approximate methods or decomposition schemes in order to solve such problems. Decomposition techniques have been widely used to cope with the inherent difficulties of multistage optimal control problems. In the present model, we are faced with the necessity to decompose the problem into decentralized zonal subproblems for which optimal solutions some efficient Dynamic Programming routine can be used (see [2]). As the aggregate model is in general convex, piecewise linear or quadratic, regularized coordination like in Monotone Operator Splitting techniques is a natural candidate to approximate the global optimum.

Monotone operator splitting methods are designed to solve monotone inclusions including the sum of two monotone operators and the algorithms aim at alternating forward (subgradient) steps and backward (proximal) steps on each operator separately, inducing at the same time decomposition techniques and Augmented Lagrangian techniques. Theoretical results and state-of-the-art of the main splitting algorithms can be found in [13, 1, 12]. In this paper, we will compare the behavior of two classical splitting methods, namely Alternate Direction Method of Multipliers (ADMM) and Proximal Decomposition Algorithm (PDA) applied to a large-scale multistage optimal control problem.

It is well-known that ADMM and PDA are very close algorithms [6], but they may differ thanks to different reformulations of the coupling relations and variables between both operators. For PDA, we will decouple the incidence arcs of the production networks and, for ADMM, production costs and transmission costs are decoupled, decomposing the calculus of the former into zonal subproblems. Both algorithms have produced many studies, variants with inexact proximal calculations, multidimensional and adaptive scaling, as well as applications to diverse fields like nonlinear mechanics [9], stochastic multistage optimization problems (like the Progressive Hedging method of Rockafellar and Wets [18]), multicommodity network flows ([7], [16]), image reconstruction [4] or classification [3].

We compare two different separable formulations, one solved by PDA (indeed similar to Spingarn's Partial Inverse algorithm, see [19]), and the other by ADMM, both closely related to Douglas-Rachford's splitting scheme (see [8]). After presenting the dynamic model in section 3, numerical results on realistic simulations in the last section show similarities and contrasted performance of both algorithms. It will be shown that ADMM is better on smaller and sparse networks, while PDA is better on large and dense networks.

We first begin by presenting, in Section 2, the network based optimal control problem we want to solve. Then we propose decomposition methods, in Section 3, in order to treat this problem. In Section 4, some numerical results are presented to illustrate the behavior of the decomposition methods proposed.

2 The multizonal optimization model

We consider a set of geographical zones Z where there is some demand for a commodity (electricity or gas). The time line is split in T time intervals indexed by $t \in \{0, \dots, T-1\}$ and we denote $d_{z,t}$ the total demand for zone $z \in Z$ at step t. There are several sources for supplying the demand:

1. Each zone has local production units which deliver a quantity $p_{z,t}$ at time step t. The corresponding aggregated cost is $c_z(p_{z,t})$.

2. Commodity can be imported from a neighboring zone z' through transportation line $e = (z, z')$. Interconnections between zones define a directed graph $G = (Z, E \subset Z \times Z)$ with $|Z| = n, |E| = m$, vertices and edges of which are zones and transportation lines. The quantity transported through line e is denoted $f_{e,t}$ and the associated cost is $l_{e,t}(f_{e,t})$.

3. Finally, some part of the production can be stored in a local storage and reused later. The level of stored commodity in zone z at the beginning of time step t is denoted by $x_{z,t}$. The initial level $x_{z,0}$ is given and the final level is denoted $x_{z,T}$. It obeys the dynamics
$$x_{z,t+1} = x_{z,t} - u_{z,t} + i_{z,t} \tag{1}$$
where $u_{z,t}$ is the usage of the storage (positive for withdrawal assuming for the sake of clarity that no pumping is allowed in the present model). Quantity $i_{z,t}$ is an additional input in the storage, assumed to be known : we can think about water inflows in dams.

The network flow balance can be stated for each period t to satisfy the demand $d_{z,t}$ of each zone. In practice, a failure can occur if the balance equation is not satisfied. Then a load shed $\eta_{z,t} \geq 0$ is introduced and penalized by a high quadratic cost $h_z(\eta_{z,t}) = c^{fail}\eta_{z,t}^2$. Thus, the balance between production and demand can be written by the following equations :

$$p_{z,t} + u_{z,t} + \sum_{e \in z^+} f_{e,t} + \eta_{z,t} = d_{z,t} + \sum_{e \in z^-} f_{e,t}, \quad \forall z \in Z, \forall t = 0, \dots, T-1. \tag{2}$$

2.1 Thermal production

The thermal production cost is a piecewise affine and convex function of the production levels $p_{z,t}$. To avoid increasing the number of unknowns, it is replaced by a quadratic approximation (using least-square regression) as shown in Figure 1. Then, in the sequel, we consider the production cost g_{zt} as a quadratic function.

2.2 Hydroelectric production

The level of stored commodity in zone z at the beginning of time period t is denoted by $x_{z,t}$. It will take the role of the *state variable*. The initial level $x_{z,0}$ is given and the final level is denoted $x_{z,T}$. It obeys the dynamics equation (1). Minimal and maximal levels are fixed for each zone and each period, so that $X_{z,t}^{min} \leq x_{z,t} \leq X_{z,t}^{max}, \forall z, t$. We neglect

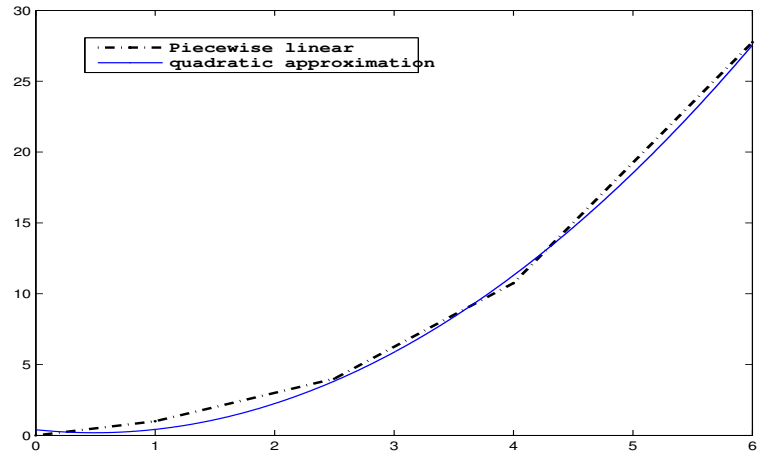

Figure 1: Quadratic approximation of a piecewise linear cost function

here the hydroelectric production cost. On the other hand, we add a cost on the final state $x_{zT} \mapsto \Psi(x_{zT})$ to penalize the excess of water reserves at the end of the horizon :

$$\Psi(x_{z,T}) = c_z^{final} \max\{0, x_{z,0} - x_{z,T}\}. \tag{3}$$

Thus we impose a linear cost on the reduction in storage between the start and the end of the horizon, but we do not include rewards for an increase in the reserves.

2.3 Interzonal transfer costs

For an arc $e = (z, z') \in E$ interconnecting two zones z and z', the flow transfer during period t is the variable $f_{e,t}$ which is bounded by $0 \leq f_{e,t} \leq \kappa_{e,t}$. The transfer cost is linear and denoted by $l_{et}(f_{et}) = c_e^{inter} f_{et}$.

2.4 Dynamic quadratic model

The model consists in minimizing the sum of the production costs under offer-demand equilibrium constraints in each zone including the dynamic equations on the state and control variables.

$$\min_{(p,u,f,x,\eta)} \sum_{t=0}^{T-1} \left[\sum_{z \in Z} [g_{zt}(p_{zt}) + h_z(\eta_{zt})] + \sum_{e \in E} l_{et}(f_{et}) \right] + \sum_{z \in Z} \psi_z(x_{zT}) \tag{4}$$

$$\text{Equations} \quad (1), (2) \quad \forall z \in Z,\ t \in [0, T-1]$$

$$X_{zt}^{min} \leq x_{zt} \leq X_{zt}^{max}, 0 \leq u_{zt} \leq U_z^{max} \delta_h, \quad \forall z \in Z,\ t \in [0, T-1]$$

$$0 \leq f_{et} \leq \kappa_{et}, \quad \forall e \in E,\ t \in [0, T-1]$$

This large-scale model will now be reformulated to focus on the interzonal coupling. To simplify the notations, we will use the 4-dimensional vector $q_{z,t} = (p_{z,t}, u_{z,t}, \eta_{z,t}, x_{z,t})$ in order to build the concatenated vectors $\boldsymbol{q}_z = (q_{z,t},\ t = 1, \ldots, T-1)$ and $\boldsymbol{q}_t = (q_{z,t},\ z \in Z)$. Similarly, we define the vectors $\boldsymbol{d}_z, \boldsymbol{d}_t, \boldsymbol{i}_z, \boldsymbol{i}_t$. For the flow variables $f_{e,t}$, we define

$\boldsymbol{f}_e = (f_{e,t},\, t = 1, \ldots, T-1)$ and $\boldsymbol{f}_t = (f_{e,t},\, e \in E)$. Finally, the whole concatenation of production and flow values on the horizon will be naturally denoted by $\boldsymbol{q}$ and $\boldsymbol{f}$ (the bold face notation is thus reserved for subvectors with one or more running indexes).

The graph G will be represented by its node-arc $(n \times m)$ incidence matrix A, so that $A\boldsymbol{f}_t = \sum_{e \in z^+} f_{e,t} - \sum_{e \in z^-} f_{e,t}$. We will also use the $(n \times 4n)$ matrix $B = \mathbb{I}_n \otimes [1\ 1\ 1\ 0]$, so that the demand satisfaction equations can be globally rewritten as :

$$B\boldsymbol{q}_t + A\boldsymbol{f}_t = \boldsymbol{d}_t, \quad \forall t.$$

Thus, $\boldsymbol{q}$ and $\boldsymbol{f}$ are coupled through the incidence matrix A at each time period t and letting

$$\begin{aligned} G_z(\boldsymbol{q}_z) = &\quad \sum_{t=0}^{T-1} [g_{zt}(p_{z,t}) + h_z(\eta_{z,t})] + \psi_z(x_{z,T}), \\ L_e(\boldsymbol{f}_e) = &\quad \sum_{t=0}^{T-1} l_{e,t}(f_{e,t}), \end{aligned}$$

we can reformulate the model (4) as :

$$\min_{(\boldsymbol{q},\boldsymbol{f})} \sum_{z \in Z} G_z(\boldsymbol{q}_z) + \sum_{e \in E} L_e(\boldsymbol{f}_e) \tag{5}$$

$$B\boldsymbol{q}_t + A\boldsymbol{f}_t = \boldsymbol{d}_t, \quad t = 1, \ldots, T-1 \tag{6}$$

$$\begin{aligned} &x_{z,t+1} = x_{zt} - u_{zt} + i_{zt} \quad \forall z, t \\ &q_{z,t} \in \mathcal{P}_{z,t}, f_{e,t} \in F_{e,t} \quad \forall z, t, e \end{aligned} \tag{7}$$

where

$$\begin{aligned} \mathcal{P}_{z,t} &= \left\{0 \le p_{z,t} \le P_{z,t}^{max}, 0 \le u_{z,t} \le U_{z,t}^{max},\ X_{z,t}^{min} \le x_{z,t} \le X_{z,t}^{max}\right\} \\ F_{e,t} &= \left\{0 \le f_{e,t} \le \kappa_{et}\right\}. \end{aligned}$$

Observe that the dynamic equations (7) are separable with respect to zonal indexes. Later, they will be kept in the zonal subproblems where they are supposed to be treated by Dynamic Programming. The coupling constraints (6) could be easily dualized to induce zonal and network subproblems, but we propose here to use ADMM as a reference for further comparisons with other splitting schemes. ADMM will be compared with PDA as described below, and both algorithms alternate resolutions of the subproblems associated with separable Augmented Lagrangian functions. The difference between both splitting resides in the treatment of the coupling constraints. While ADMM will directly consider the Augmented Lagrangian associated with the coupling constraints (6) in model (5), PDA will decouple the terms $A\boldsymbol{f}_t$ by creating local copies of the ingoing flows $(\boldsymbol{f}_e, e \in z^-)$ as explained in the next section.

3 Decomposition Algorithms

We now study decomposition strategies for our dynamic quadratic programming model (5).

We can reorganize the columns (arcs) of the node-arc incidence matrix A by ordering the zones and the *outgoing arcs* in each zone

$$A = \left[\begin{array}{ccc|ccc|ccc|ccc} 1 & 1 & 1 & -1 & & & -1 & & & -1 & & \\ -1 & & & 1 & 1 & 1 & & -1 & & & -1 & \\ & -1 & & & -1 & & 1 & 1 & 1 & & & -1 \\ & & -1 & & & -1 & & & -1 & 1 & 1 & 1 \end{array}\right]$$

Observe in the above example with 4 zones that the arc ordering gives rise to diagonal blocks (arrays of three 1) denoted hereafter by A_z. We define $\boldsymbol{f}_{z,t}$ to denote the vector of outgoing flows $f_{e,t}$, $e \in z^+$ and we introduce copies $\boldsymbol{\phi}_{z,t}$ of the ingoing flows for each zone z associated with each -1 in row z of the incidence matrix to obtain the following equivalent reformulation :

$$Bq_{z,t} + A_z \boldsymbol{f}_{z,t} - \sum_{e \in z^-} \phi_{ez,t} = d_{z,t}, \forall z, t \tag{8}$$

$$\phi_{ez,t} = f_{ez',t}, \quad \text{for} \quad e = (z', z) \in z'^+ \cap z^-, \forall z, z' \in Z, \forall t \tag{9}$$

Observe that the demand equation (8) is now completely decentralized with additional variables ϕ_{ez}. The set of equations (9) represents a coupling subspace between zones in $\mathbb{R}^{2m(T-1)}$, denoted hereafter by $\mathcal{A}$.

By setting

$$\begin{aligned} \boldsymbol{Q}_z(\boldsymbol{f}_z, \boldsymbol{\phi}_z) \quad = \quad & \min_{\boldsymbol{q}_z, \boldsymbol{x}_z} \sum_t \Big[g_{z,t}(p_{z,t}) + h_z(\eta_{z,t}) + \sum_{e \in z^+} l_{e,t}(f_{ez,t}) \Big] + \psi_z(x_{z,T}) \\ & \text{s.t.} \quad Bq_{z,t} + A_z \boldsymbol{f}_{z,t} - \sum_{e \in z^-} \phi_{ez,t} = d_{z,t}, \quad \forall t \\ & x_{z,t+1} = x_{zt} - u_{zt} + i_{zt}, \quad \forall t \\ & q_{z,t} \in \mathcal{P}_{z,t}, \ f_{ez,t} \in F_{e,t}, \ \phi_{ez,t} \in F_{e,t}, \quad \forall e, t \end{aligned}$$

Finally, we can put problem (5) in the following separable form :

$$\min_{\boldsymbol{q}, \boldsymbol{f}, \boldsymbol{\phi}} \sum_z \boldsymbol{Q}_z(\boldsymbol{f}_z, \boldsymbol{\phi}_z) \quad (\boldsymbol{f}, \boldsymbol{\phi}) \in \mathcal{A}$$

Each $\boldsymbol{Q}_z$ is a convex function defined on a polyhedron with a maximal monotone subdifferential operator, allowing the application of PDA as seen below.

3.1 Proximal decomposition algorithm (PDA)

The Proximal Decomposition algorithm is a generalized version of Spingarn's Partial Inverse method ([19]) analyzed by Mahey et al [15]. It is specially tailored to minimize separable convex functions on a coupling subspace.

The key ingredient is that we need a pair of primal and dual variables lying respectively in subspace $\mathcal{A}$ and its orthogonal subspace $\mathcal{A}^\perp$. We denote the dual variables by (u, v) so that we have the following dual relations :

$$\boldsymbol{X} = \left[\begin{array}{c} \boldsymbol{f} \\ \boldsymbol{\phi} \end{array}\right] = \left[\begin{array}{c} \mathbb{I} \\ \mathbb{I} \end{array}\right] \alpha = E\alpha, \text{for some} \quad \alpha \in \mathbb{R}^{n(n-1)} \tag{10}$$

$$W = \left[\begin{array}{c} \boldsymbol{v} \\ \boldsymbol{w} \end{array}\right] = \left[\begin{array}{c} \mathbb{I} \\ -\mathbb{I} \end{array}\right] \beta = D\beta, \text{for some} \quad \beta \in \mathbb{R}^{n(n-1)} \tag{11}$$

Algorithm 1 Proximal decomposition algorithm

Step 1. Zonal subproblems: Compute $(\boldsymbol{q}_z^{k+1}, \boldsymbol{f}_z^{k+1/2}, \boldsymbol{\phi}_z^{k+1/2})$ as solution of

$$\min_{(\boldsymbol{q}_z, \boldsymbol{f}_z, \boldsymbol{\phi}_z)} \boldsymbol{Q}_z(\boldsymbol{q}_z, \boldsymbol{f}_z) + \frac{1}{2\lambda}\|\boldsymbol{f}_z - \boldsymbol{f}_z^k - \lambda \boldsymbol{v}_z^k\|^2 + \frac{1}{2\lambda}\|\boldsymbol{\phi}_z - \boldsymbol{\phi}_z^k - \lambda \boldsymbol{w}_z^k\|^2$$

Step 2. Dual update:

$$\begin{aligned} \boldsymbol{v}_z^{k+1/2} &= \frac{1}{\lambda}(\boldsymbol{f}_z^{k+1/2} + \lambda \boldsymbol{v}_z^k - \boldsymbol{f}_z^k) \\ \boldsymbol{w}_z^{k+1/2} &= \frac{1}{\lambda}(\boldsymbol{\phi}_z^{k+1/2} + \lambda \boldsymbol{w}_z^k - \boldsymbol{\phi}_z^k) \end{aligned}$$

Step 3. Projection step: Compute $\boldsymbol{X}^{k+1} = (\boldsymbol{f}^{k+1}, \boldsymbol{\phi}^{k+1})$ and $\boldsymbol{W}^{k+1} = (\boldsymbol{v}^{k+1}, \boldsymbol{w}^{k+1})$ as follows.

$$\begin{aligned} \boldsymbol{f}_{ez}^{k+1} = \boldsymbol{\phi}_{ez'}^{k+1} &= \frac{1}{2}(\boldsymbol{f}_{ez}^{k+1/2} + \boldsymbol{\phi}_{ez'}^{k+1/2}) \\ \boldsymbol{v}_{ez}^{k+1} = -\boldsymbol{w}_{ez'}^{k+1} &= \boldsymbol{v}_{ez}^{k+1/2} - \frac{1}{2}(\boldsymbol{v}_{ez}^{k+1/2} + \boldsymbol{w}_{ez'}^{k+1/2}) \end{aligned}$$

The method alternates proximal steps on the primal variables (the optimal production of each zone), the convex subproblem with equilibrium guaranteed by each zone independently, and projection steps on each subspace. The cost function of the zonal subproblems take the following form at iterations k

$$\min_{(\boldsymbol{q}_z, \boldsymbol{f}_z, \boldsymbol{\phi}_z)} \boldsymbol{Q}_z(\boldsymbol{q}_z, \boldsymbol{f}_z) + \frac{1}{2\lambda} \parallel \boldsymbol{X}_z - \boldsymbol{X}_z^k - \lambda \boldsymbol{W}_z^k \parallel^2$$

which yields the intermediate solution $(\boldsymbol{q}_z^{k+1}, \boldsymbol{X}_z^{k+1/2})$. The dual update are such that $\boldsymbol{X}^{k+1/2} + \lambda \boldsymbol{W}^{k+1/2} = \boldsymbol{X}^k + \lambda \boldsymbol{W}^k$ so that $(\boldsymbol{X}^{k+1/2}, \boldsymbol{W}^{k+1/2})$ is the projection of $(\boldsymbol{X}^k, \boldsymbol{W}^k)$ onto the graph of the separable maximal monotone operator $\partial[\boldsymbol{Q}_1 \cdots \boldsymbol{Q}_n]$.

The different steps of the proximal algorithm are presented in Algorithm 1. In Step 2, we use the same ordering of the variables $\boldsymbol{X}$ and $\boldsymbol{W}$ ($\boldsymbol{f}_{ez}$ at the same column index as $\boldsymbol{\phi}_{ez'}$, and $\boldsymbol{w}_{ez}$ at the same column index as $\boldsymbol{v}_{ez'}$).

Observe that $\boldsymbol{X}^k \in \mathcal{A}$ and $\boldsymbol{W}^k \in \mathcal{A}^\perp, \forall k$, so that $\boldsymbol{f}^k$ is a feasible flow for graph G. On the other hand, $\boldsymbol{f}^{k+1/2}$ is not feasible but it will converge asymptotically.

It is well-known that numerical enhancements are necessary to take full profit of the numerical stability and linear rate of convergence of Algorithm 1, mainly :

- Inexact proximal steps;
- Adjustable scaling parameter λ^k;
- Additional relaxation parameter extending the Douglas-Rachford splitting.

3.2 Alternating direction method of multiplier (ADMM)

Alternating direction method of multiplier (also known as Uzawa block relaxation method) has been used in various domains as an operator-splitting method: nonlinear mechanics

(e.g., [14, 9, 10]), image processing (e.g., [11, 4]).

Algorithm 2 Alternating direction method of multipliers

Step 1. Zonal production subproblems: Compute $(\boldsymbol{q}_z^{k+1})$ as solution of

$$\begin{aligned}\min_{(\boldsymbol{q}_z\in\mathcal{P}_z)} \quad & \sum_{t=0}^{T-1}\Big[g_{zt}(p_{z,t})+h_z(\eta_{z,t})+y_{z,t}^k\big[p_{z,t}+u_{z,t}+\eta_{z,t}+\sum_{e\in z^+}f_{e,t}^k-\sum_{e\in z^-}f_{e,t}-d_{z,t}\big]\\ & +\frac{1}{2\lambda}\big[p_{z,t}+u_{z,t}+\eta_{z,t}+\sum_{e\in z^+}f_{e,t}^k-\sum_{e\in z^-}f_{e,t}^k-d_{z,t}\big]^2\Big]+\psi_z(x_{z,T})\\ \text{s.t.} \quad & x_{z,t+1}=x_{zt}-u_{zt}+i_{zt}, \quad \forall t\end{aligned}$$

Flow subproblem: Compute $\boldsymbol{f}_t^{k+1}$ as solution of

$$\begin{aligned}\min_{\boldsymbol{f}_t\in\boldsymbol{F}_t} \quad & \sum_e l_{e,t}(f_{e,t})+(\boldsymbol{y}_t^k)^\top(\boldsymbol{p}_t^{k+1}+\boldsymbol{u}_t^{k+1}+\boldsymbol{\eta}_t^{k+1}+A\boldsymbol{f}_t-\boldsymbol{d}_t)+\\ & \frac{1}{2\lambda}\|\boldsymbol{p}_t^{k+1}+\boldsymbol{u}_t^{k+1}+\boldsymbol{\eta}_t^{k+1}+A\boldsymbol{f}_t-\boldsymbol{d}_t\|^2\end{aligned}$$

Step 2. Multiplier update

$$\boldsymbol{y}_t^{k+1} \;=\; \boldsymbol{y}_t^k+\frac{1}{\lambda}(B\boldsymbol{q}_t^{k+1}+A\boldsymbol{f}_t^{k+1}-\boldsymbol{d}_t),\forall t$$

Back to model (5), we now apply directly ADMM, building the corresponding Augmented Lagrangian, using dual multipliers $\boldsymbol{y}_t$ with components $y_{z,t}$ associated with the flow equations (6) :

$$\mathcal{L}_\lambda(\boldsymbol{q},\boldsymbol{f},\boldsymbol{y})=\sum_{z\in Z}G_z(\boldsymbol{q}_z)+\sum_{e\in E}L_e(\boldsymbol{f}_e)+\sum_t\left[\boldsymbol{y}_t^\top(B\boldsymbol{q}_t+A\boldsymbol{f}_t-\boldsymbol{d}_t)+\frac{1}{2\lambda}\|B\boldsymbol{q}_t+A\boldsymbol{f}_t-\boldsymbol{d}_t\|^2\right],$$

where λ is the penalty parameter. Starting with $(\boldsymbol{q}^0,\boldsymbol{f}^0,\boldsymbol{y}^0)$, ADMM alternates minimization steps on the Augmented Lagrangian w.r.t. primal production variables $\boldsymbol{q}$ which decompose into zonal subproblems, and to primal flow variables $\boldsymbol{f}$ which decompose into single stage static subproblems :

$$\boldsymbol{q}^{k+1} \;=\; \arg\min_{\boldsymbol{q}\in\mathcal{P}}\mathcal{L}_\lambda(\boldsymbol{q},\boldsymbol{f}^k,\boldsymbol{y}^k) \tag{12}$$

$$\boldsymbol{f}^{k+1} \;=\; \arg\min_{\boldsymbol{f}\in\boldsymbol{F}}\mathcal{L}_\lambda(\boldsymbol{q}^{k+1},\boldsymbol{f},\boldsymbol{y}^k) \tag{13}$$

$$\boldsymbol{y}_t^{k+1} \;=\; \boldsymbol{y}_t^k+\frac{1}{\lambda}(B\boldsymbol{q}_t^{k+1}+A\boldsymbol{f}_t^{k+1}-\boldsymbol{d}_t), \quad \forall t. \tag{14}$$

where the set $\mathcal{P}$ represents constraints (7) for every zone z and period t with corresponding bounds, and the set $\boldsymbol{F}$ represents the bound constraints on the flow variables.

Observe that the production subproblems (12) decompose in n zonal subproblems which are quadratic dynamic programs with $4(T-1)$ variables. On the other hand, the network flow subproblems are quadratic minimizations with simple bounds which decompose in $T-1$ static subproblems with m variables.

The alternating direction method of multiplier algorithm for problem (5) is outlined in Algorithm 2.

As already observed, similar zonal subproblems and static flow subproblems are solved in Algorithms 1 and 2. The main differences lie in the fact that zonal and flow subproblems are solved in parallel in Algorithm 1 whereas they are solved sequentially in Algorithm 2 (like Jacobi's and Gauss-Seidel's algorithms in iterative methods. Additionally, projection steps on the coupling subspaces are performed in Algorithm 1 to define feasible subsequences.

4 Numerical experiments

We compare, in this section, the decomposition algorithms outlined in the previous sections. The computations have been carried out on a Dell Precision T3500 computer, equipped with Intel Xeon 2.67GHz processor with 12GB RAM, using Matlab version 7. We use the MATLAB function `quadprog` to solve the (quadratic programming) zonal problems. `quadprog` implements an interior-point algorithm designed for large-scale convex quadratic programs exploiting sparsity. All algorithms are stopped if the gradient of the Lagrangian function and the relative residual of the production/demand equilibrium constraint

$$\|B\boldsymbol{q}^k + A\boldsymbol{f}^k - d\| \leq \varepsilon$$

where $\varepsilon = 10^{-4}$ is a tolerance defined by the user.

We will first compare both algorithms with the direct application of `quadprog` to the centralized model on a dense network.

4.1 Dense network

We consider a complete transportation graph with $m = (n-1)n$. We generate randomly the additional uncontrolled water input $i_{z,t}$ and the demand d_{zt} as autocorrelated processes. We also generate randomly, U_z^{max}, $x_{z,0}$, $x_{z,t}^{max}$, P_{zt}^j and c_{zt}^j. We set

- interzonal transfer cost $c_e^{inter} = 1, \forall e$;
- failure cost $c^{fail} = 10^3$;
- final state cost $c_z^{final} = 10^6, \forall z$;
- arc capacity $\kappa_{et} = 5, \forall t, e$.

After some tests, we set $\lambda = 10^{-5}$ in Algorithm 1-2 (observe that update formulae for the scaling parameter have been proposed in the literature, see [12] for instance, but where not used here).

Table 1 shows he comparative performance of Algorithm 1-2 with MATLAB function `quadprog` for different instances.

`quadprog` is faster but we should observe that significant speedup could have been obtained by solving subproblems in parallel in the splitting algorithms. One can notice

Problem data			Algorithm 1		Algorithm 2		`quadprog`	
n	N	N_c/M_c	iter	CPU (Sec.)	iter	CPU (Sec.)	iter	CPU (Sec.)
4	524	240/40	2	0.662	2	0.467	11	0.053
8	1688	1120/80	2	1.911	2	0.805	14	0.251
16	5936	4800/160	3	5.131	2	2.417	13	0.743
32	22112	19840/320	3	17.445	2	24.98	13	3.034
64	85184	80640/640	5	137.783	15	> 1000	13	19.841

Table 1: Centralized GQP Vs Algorithms 1, 2 on dense networks: N number of unknowns; N_c number of coupling unknowns; M_c number of coupling constraints.

that both algorithms are almost equivalent in terms of the number of iterations required for convergence. For small problems ($n = 4, 8, 16$), Algorithm 2 is up to twice as fast as Algorithm 1. For the largest problem (i.e. $n = 64$), Algorithm 1 stops after 137.783 seconds while Algorithm 2 dose not stop after 1000 seconds. This is due to the large number of coupling variables since Algorithm 2 has to solve a global network subproblem (Step 2.)

4.2 Sparse network

We consider a sparse transportation graph $G = (Z, E)$ with $|Z| = n$ and E such that each node has no more than 4 connections. We generate data randomly as in the previous subsection. For c_e^{inter}, c^{fail}, c_z^{final}, κ_{et} and λ, we use the same values as in the previous subsection.

Table 2 shows the performance of our decomposition algorithms on a sparse transportation network. One can again notice that both algorithms are almost comparable in terms of the number of iterations. Due to the sparsity pattern of the network (no more than 4 connections for each node), Algorithm 2 is faster as noticed in Section 4.1. MATLAB function `quadprog` is again faster for these instances, but higher speedup is expected with a parallel implementation of our decomposition Algorithm 1-2.

Problem data			Algorithm 1		Algorithm 2		`quadprog`	
n	N	N_c/M_c	iter	CPU (Sec.)	iter	CPU (Sec.)	iter	CPU (Sec.)
4	444	160/40	2	0.610	2	0.485	14	0.078
8	968	400/80	3	1.386	2	0.698	16	0.114
16	2096	960/160	3	2.667	2	1.102	15	0.250
32	4352	2080/320	3	5.093	2	1.978	15	0.523
64	9024	4480/640	4	13.549	2	4.063	15	1.156

Table 2: Performances of Algorithms 1-2 on a sparse transportation network: N number of unknowns; N_c number of coupling unknowns; M_c number of coupling constraints.

4.3 A realistic network

The concrete energy planning problem considered here is a one-year simulation of a standard year in a far future (say in 2025). The granularity of the dynamic model is one day, thus ignoring intra-days fluctuations. The objective is to simulate import-export marginal prices between the zones (and not to solve the detailed operation plan) for a set of adapted scenarios.

We mean by 'realistic' here a network with 8 to 20 zones representing the European countries but the data used to test the decomposition algorithms do not correspond to real-life data but to randomly generated scenarios of daily variations of demand and water input for each country during one year simulations.

For illustration purpose, we consider an energy network consisting of 8 zones: Belgium (BE), Spain (ES), France (FR), Germany (GE), Italy (IT), Portugal (PT), Switzerland (SW), United Kingdom (UK). The network is depicted in Figure 2 with the arcs capacity. The hydroelectric production data are given in Table 3 while Table 4 summarize the thermic production data. The other data are

- interzonal transfer cost $c_e^{inter} = 1$ (euro/MWh);
- failure cost $c^{fail} = 10^3$ (euro/MWh);
- final state cost $c_z^{final} = 10^6$ (euro/MWh)

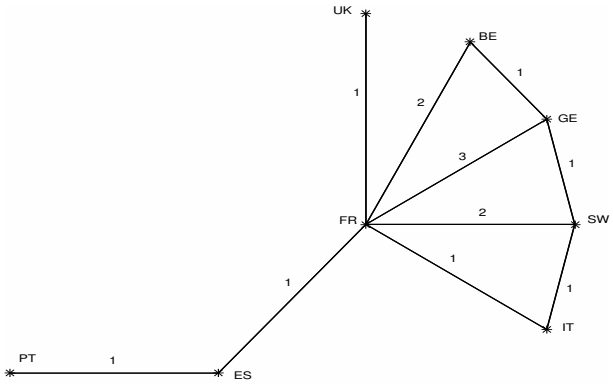

Figure 2: Energy network with arc capacity (in GW)

z	BE	ES	FR	GE	IT	PT	SW	UK
$U_z^{\max}$ (GW)	0	9	12	3	10.5	1.5	12	1.5
$x_{z,0}$ (TWh)	0	6	7.5	3	3	1.5	7.5	1.5
$X_{zt}^{\min}$ (TWh)	0	0	0	0	0	0	0	0
$X_{zt}^{\max}$ (TWh)	0	8	10	4	4	2	10	2

Table 3: Hydroelectric maximal production $U_z^{\max}$, initial level ($x_{z,0}$), minimal and maximal levels

z	BE	ES	FR	GE	IT	PT	SW	UK
a_z	0.5714	0.5714	0.2135	0.1786	0.5714	2.2857	2.2857	0
b_z	493.7143	493.7143	132.0177	351.4286	493.7143	918.8571	918.8571	768.3840

Table 4: Thermal generation data $g_z(p_{zt}) = \frac{1}{2}a_z p_{zt}^2 + b_z p_{zt}$

As in the previous subsections, the additional uncontrolled (water) input $i_{z,t}$ and the demand $d_{z,t}$ are randomly generated as autocorrelated processes. For example, the annual variation of energy consumption is modelled as the following sinusoid :

$$\mu_{z,t} = \alpha_z + \beta_z \cos(\frac{2\pi t}{T}), t = 1, \ldots, T-1$$

Then the zonal demand is generated recursively by independent centered random sampling $(\xi_{z,t}, t = 1, \ldots, T-1)$:

$$\begin{aligned} d_{z,0} &= [\mu_{z,0} + \sigma\xi_{z,0}]\delta_h \\ d_{z,t} &= [\mu_{z,t} + e^{-\omega}(d_{z,t-1} - \mu_{z,t-1}) + \sigma\xi_{z,t}]\delta_h \end{aligned}$$

where σ, ω are adhoc parameters and $\delta_h = 24$.

Similarly, water inflows are generated with sinusoidal fluctuations, but the latter are shifted in phase to model the fact that consumption peaks appear at the beginning of the year and water inflows will be higher during spring.

We set $T = 365$ (one year). Consequently, the size of the original problem is 35048. In Algorithm 1, the size of zonal subproblems varies from 3286 to 6936. In Algorithm 2 the size of zonal production subproblems is 2556 while the size of flow subproblems is 7300 (20 for each time step).

Table 5 shows the performances of the algorithms with respect to the parameter λ. We can notice that Algorithm 1 is more sensitive to the choice of the parameter λ. Overall, since the network is sparse, Algorithm 2 is more efficient.

λ		10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}
Algorithm 1	Iter	11	11	12	22	19
	CPU (sec.)	150.072	201.670	218.316	398.440	330.119
Algorithm 2	Iter	13	13	13	13	13
	CPU(sec.)	167.877	154.525	169.173	174.140	147.035

Table 5: Number of iterations Versus λ for Algorithm 1-2

Figures 3-10 show the annual production for each zone. One can notice that

- BE and UK are self sufficient;
- FR, GE and PT import energy;
- ES, IT and SW export energy.

Due to the final storage (high) cost, the hydroelectric generation is privileged as expected.

in Figure 10-11, two distinct scenarios are applied applied to illustrate (on a given zone, here UK) that the zones can change their status of exporting or importing.

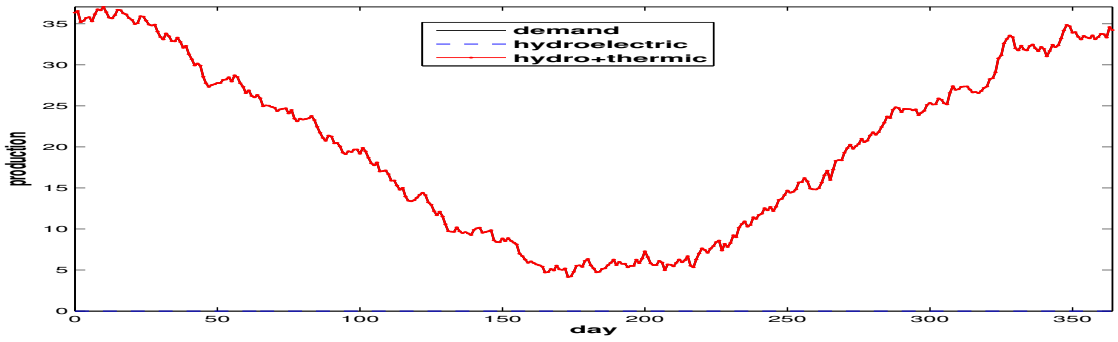

Figure 3: Computed hydroelectric and thermic production for Belgium

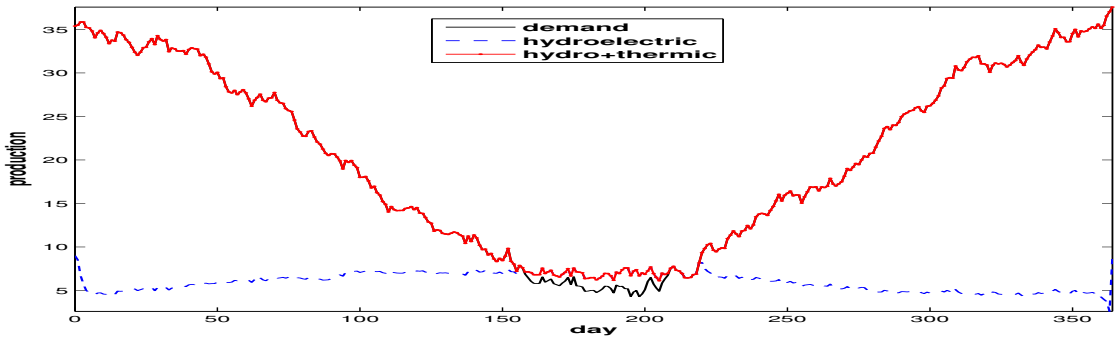

Figure 4: Computed hydroelectric and thermal production for Spain

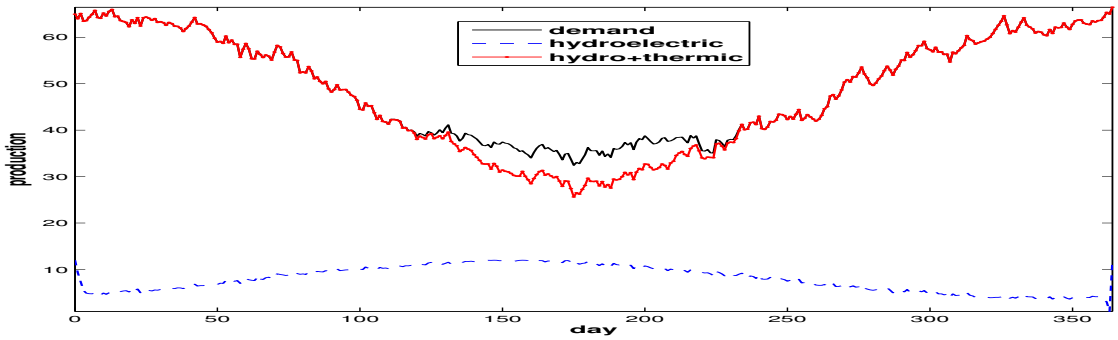

Figure 5: Computed hydroelectric and thermal production for France

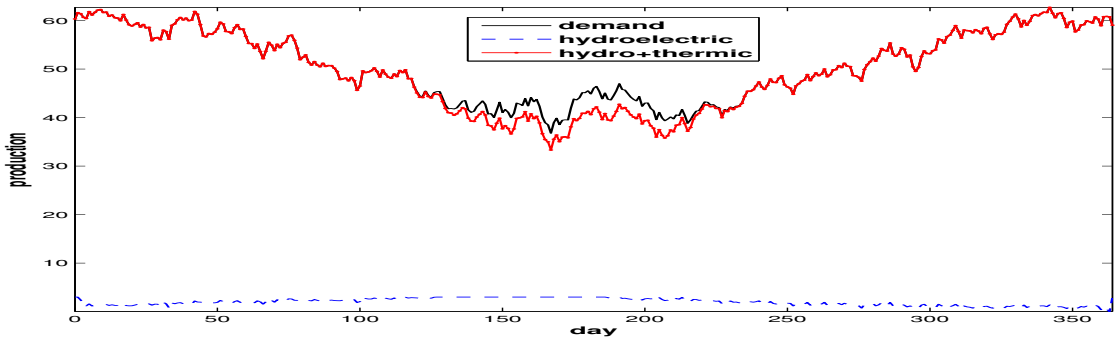

Figure 6: Computed hydroelectric and thermal production for Germany

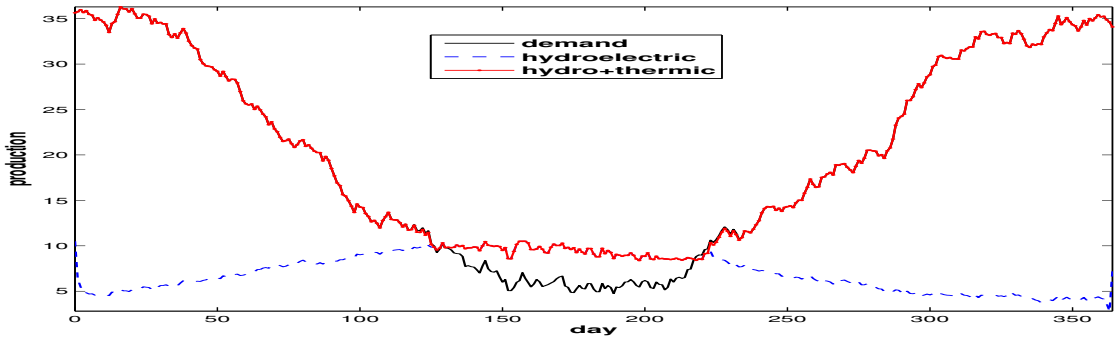

Figure 7: Computed hydroelectric and thermal production for Italy

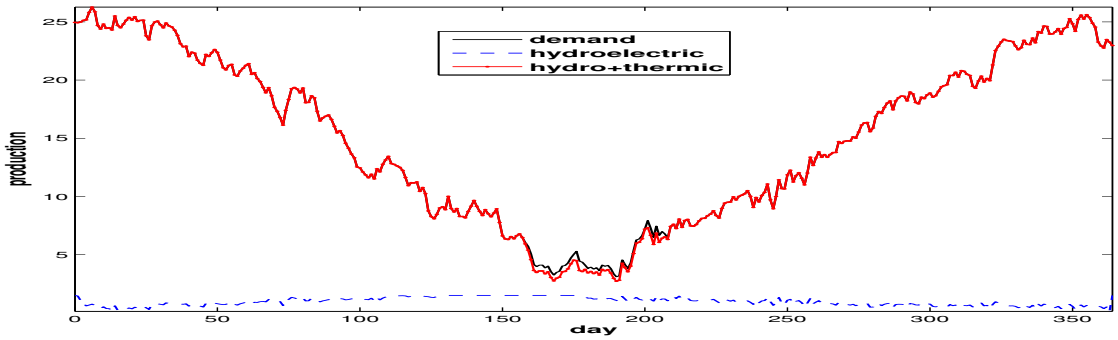

Figure 8: Computed hydroelectric and thermal production for Portugal

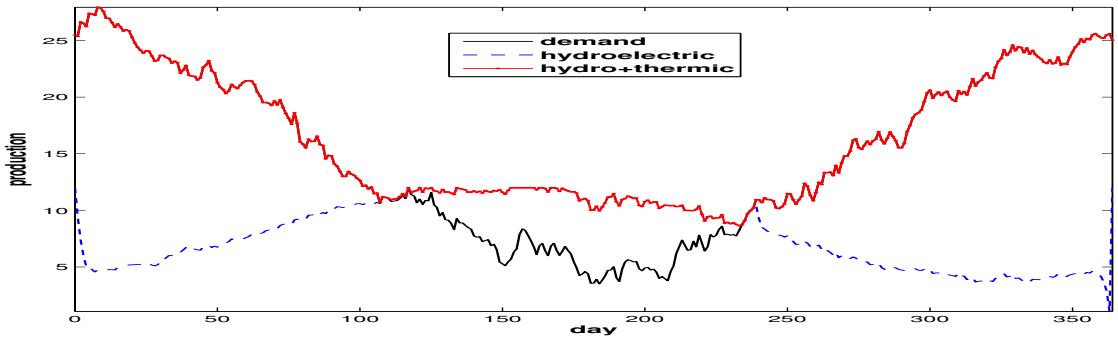

Figure 9: Computed hydroelectric and thermal production for Switzerland

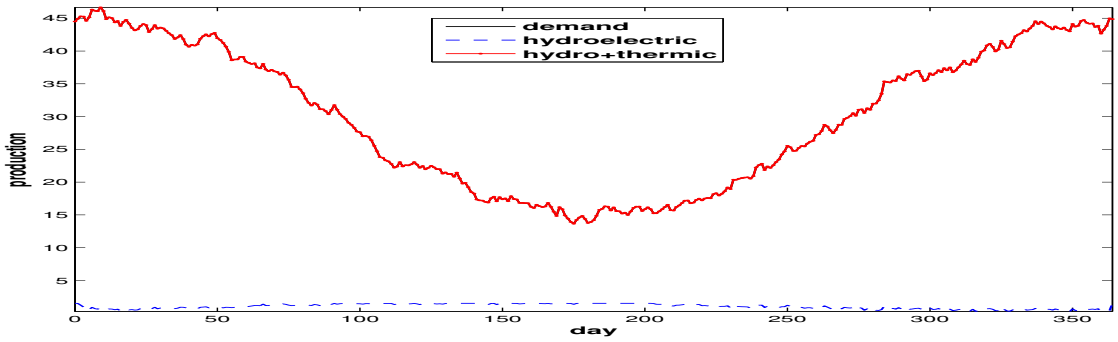

Figure 10: Computed hydroelectric and thermal production for UK

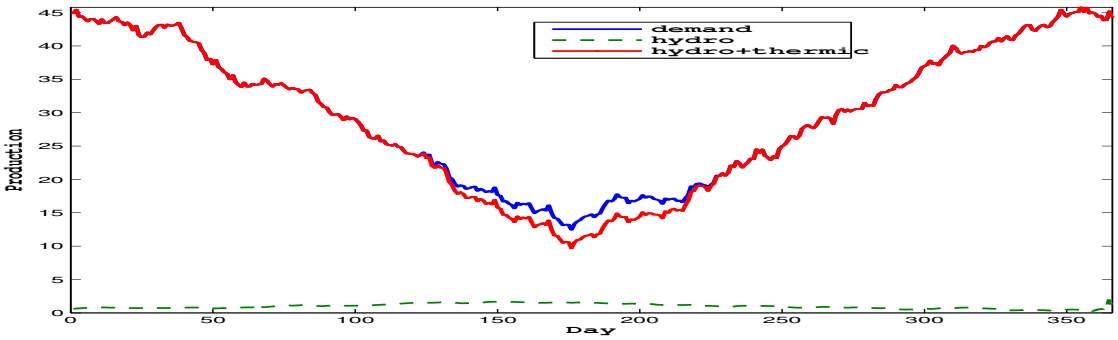

Figure 11: Computed hydroelectric and thermal production for UK, scenario 2

5 Conclusion

We have applied two different splitting method to a dynamic long-term multizonal energy planning problem. Both methods lead to zonal subproblems which adjust the production levels of each zone in each period. In the first method, PDA, the subproblems include all ingoing and outgoing arcs of the zone by the use of the arc flow copies. In the second method, ADMM, zonal subproblems are decoupled from the network subproblems, the latter being decomposed by periods. In both cases, convergence is guaranteed by the convexity assumptions on the local cost functions and the performance of the resulting algorithms depends heavily on the choice of the scaling parameter λ.

Further study is underway to enhance the model. Indeed, the demand d_z and the additional water inflows i_z are random processes. To simulate the behavior of the production network and to estimate the interzonal prices of electricity in a far future, we need to define a stochastic long-term planning model where the zonal demands and the water inflows are random processes, and the cost function is the total expected cost on the whole horizon (see [5] for details about the stochastic model). This stochastic problem could be solved by Stochastic Dual Dynamic Programming techniques (see [17]) but it will be limited by the huge dimension of the problem even with a small number of scenarios. We have thus shown in this paper that an appropriate decomposition techniques like the Proximal Decomposition method is able to decentralize the corresponding computation by solving local dynamic programs of relatively small dimensions.

Acknowledgements

The authors acknowledge the financial support of PGMO program (Programme Gaspard Monge pour l'Optimisation). The authors are very grateful to the anonymous referees for their helpful and valuable suggestions and remarks, which greatly improved the earlier version of this paper.

References

[1] H.H. Bauschke and P.L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer V., 2011.

[2] D.P. Bertsekas. *Dynamic Programming and Optimal Control, 2nd Edition, vol. 1-2*. Athena Scientific, 2000.

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning with the alternating direction method of multipliers. In M. Jordan, editor, *Foundations and Trends in Machine Learning*, volume 3, pages 1–122. 2011.

[4] P.L. Combettes and J.C. Pesquet. Proximal splitting methods in signal processing. In H.H. Bauschke, R.S. Burachik, P.L. Combettes, V. Elser, D.R. Luke, and H. Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer V., 2011.

[5] A. Dallagi and A. Lenoir. A stochastic zonal decomposition algorithm for network energy management problems. Research Report, EDF, 2013.

[6] J. Eckstein and D. P. Bertsekas. On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math. Programming*, 55:293–318, 1992.

[7] M. Fukushima. Application of the alternating direction method of multipliers to separable convex programming. *Comput. Optimization and Appl.*, 1:93–112, 1992.

[8] D. Gabay. Applications of the method of multipliers to variational inequalities. Studies in Mathematics and its Applications 15, 1983.

[9] R. Glowinski and P. Le Tallec. *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. SIAM Philadelphia, 1989.

[10] J. Koko. Uzawa block relaxation for the unilateral contact problem. *J. Comput. Appl. Math.*, 235:2343–2356, 2011.

[11] J. Koko. and S. Jehan-Besson. An augmented lagrangian method for TV_g+L^1-norm minimization. *J. Math. Imaging Vis.*, 38:182–196, 2010.

[12] A. Lenoir and P. Mahey. Monotone operator splitting methods and decomposition of convex programs. *RAIRO*, 51:17–41, 2017.

[13] P.L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16:964–979, 1979.

[14] M. Fortin and R. Glowinski. *Augmented Lagrangian Methods: Application to the Numerical Solution of Boundary-Value Problems*. North-Holland, Amsterdam, 1983.

[15] P. Mahey, S. Oualibouch, and D.T. Pham. Proximal decomposition on the graph of a maximal monotone operator. *SIAM J. Optimization*, 5:454–466, 1995.

[16] A. Ouorou, P. Mahey, and J.P. Vial. A survey of algorithms for convex multicommodity flow problems. *Management Science*, 46:126–147, 2000.

[17] M.V.F. Pereira and L.M.V.G. Pinto. Multistage stochastic optimization applied to energy planning. *Math. Programming*, 52:359–375, 1991.

[18] R. T. Rockafellar and R. J-B Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16:119–147, 1991.

[19] J.E. Spingarn. Applications of the method of partial inverses to convex programming:decomposition. *Mathematical Programming*, 32:199–223, 1985.