Metin Türkay and Erdal Aydın (Eds.)
Proceedings of the 31st European Symposium on Computer Aided Process Engineering
June 6th to 9th, 2021, Istanbul, Turkey.

Multi-scenario Design Optimization using ADMM of a Thermal Energy Storage System

Sandeep Prakash,[a] Dinesh Krishnamoorthy,[a] Johannes Jäschke[a*]

[a]*Norwegian University of Science and Technology, Trondheim 7050, Norway*

johannes.jaschke@ntnu.no

Abstract

Using nonlinear models to represent multi-scenario design optimization problems can lead to very large NLPs that can become intractable to be solved centrally due to the available memory in the computing device used. In this paper, we consider a simple but general approach for partitioning the large problem into smaller NLPs by adding consensus constraints. A distributed algorithm is then developed by applying the Alternating Direction Method of Multipliers (ADMM) to solve the partition problems separately in a distributed manner and overcome the memory limitations. The approach is demonstrated using a simple case study and compared against the solution obtained by solving the problem centrally.

Keywords: ADMM, thermal energy storage, optimal design.

1. Introduction

Thermal energy storage (TES) systems help manage the asynchronous behavior between supply and demand for thermal energy that occur in many industrial processes. They can store excess energy during off-peak periods and discharge it during peak demand, thus reducing reliance on external utilities and decrease operating costs. During design stages, the profiles for the supply and demand for energy under which the system is expected to operate in are uncertain and need to be accounted for. The uncertainty can be represented by a set of discrete scenarios $\mathcal{S} := \{1,\ldots,S\}$ with cost weights ω_s to represent the likelihood of scenario *s* being realized. Stochastic programming approaches can be used to cast the optimal design problem as a two-stage optimization problem where the design variables w^{des} are first-stage (here and now) decisions while the operations variables w_s^{oper} are second stage (wait and see) decisions for each scenario (Thombre et al., 2020). The dynamics of the system in each scenario can be discretized into N equally spaced sampling intervals represented by the set $\mathcal{K} := \{0,\ldots,N-1\}$ and the optimal design problem cast as an NLP in the centralized form as (1).

$$\min_{w^{des},w^{oper}} \quad \phi^{capex}\left(w^{des}\right) + \sum_{s\in\mathcal{S}} \omega_s \left[\sum_{k\in\mathcal{K}} l\left(x_{k,s},u_{k,s}\right)\right] \tag{1a}$$

$$s.t \quad g\left(x_{k,s},u_{k,s},\theta_{k,s},w^{des}\right) \le 0 \qquad \forall k\in\mathcal{K},\forall s\in\mathcal{S} \tag{1b}$$

$$x_{k+1,s} = h\left(x_{k,s},u_{k,s},\theta_{k,s},w^{des}\right) \qquad \forall k\in\mathcal{K},\forall s\in\mathcal{S} \tag{1c}$$

$$x_{0,s} = x_0 \qquad \forall s\in\mathcal{S} \tag{1d}$$

The vectors *x*, *u* and θ represent the differential states, control inputs applied to the plant during operations, and time-varying parameters respectively. Subscript $(.)_{k,s}$ represents the s^{th} scenario at timestep *k* and the initial condition for all scenarios is x_0. The operations variable for scenario *s* is then built by stacking the vectors in each timestep k as, $\left(w_s^{oper}\right)^T := \left[\left(x_{0,s}\right)^T,\ldots,\left(x_{N,s}\right)^T,\left(u_{0,s}\right)^T,\ldots,\left(u_{N-1,s}\right)^T\right]$. Functions $\phi^{capex}(.)$ and $l(.)$ represent the capital costs and operating costs. The function $g(.)$ represents inequality constraints and $h(.)$ is used to represent the dynamics of the system. When considering many scenarios and longer time horizons, solving the optimal design problem as formulated in (1) can become computationally intractable due to the limited memory available in the computing device used. In this paper, we explore the option of dividing this problem into smaller partition problems which can be solved separately by multiple smaller machines to handle the memory limitations. The partition problems can be solved iteratively with a coordination step in between to reach a solution to the central problem.

2. Methodology

2.1. Reformulating the design problem as a general form consensus optimization problem

We can divide the two-stage dynamic optimization problem into P partitions denoted by the set $\mathcal{P} := \{1,\ldots,P\}$ in a very flexible manner. For example, besides considering each scenario as a partition, we could also form partitions within the dynamic optimization problem itself or even bundle together similar parts from multiple scenarios. Each partition *p* is a separate optimization problem with its own local variables for design (w_p^{des}) and operations (w_p^{oper}) and can be solved individually. They must be coordinated to reach a solution of the central problem. An illustration of a design problem with 2 scenarios divided into 6 partitions is shown in Figure 1.

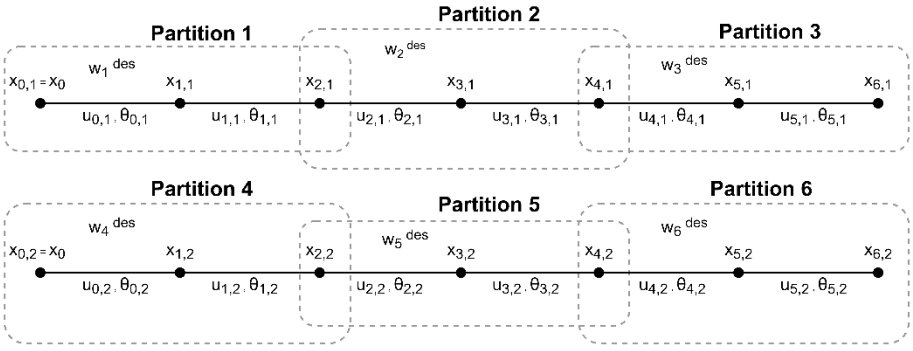

Figure 1: Illustration of dividing the optimal design problem with 2 scenarios into 6 partitions.

The solutions of the individual partitions put together is a solution to the centralized problem (1) when,

- All the partitions are in consensus of the value of their local design variables
- Adjacent partitions are in consensus of the value of the differential state variable shared between them (at the edge of the partitions)

We introduce a global copy of all the variables that must reach consensus into a vector ν and the consensus requirements then imposed as an additional constraint in each partition *p*. We can thus write the optimal design problem as a sum of smaller partition problems that are linked with consensus constraints as,

$$\min_{w_p,\nu} \quad \sum_{p\in\mathcal{P}} \Phi_p^{capex}\left(w_p^{des}\right)+\Phi_p^{oper}\left(w_p^{oper}\right) \tag{2a}$$

$$s.t \quad g_p\left(w_p^{oper},\theta_p,w_p^{des}\right)\le 0 \qquad \forall p\in\mathcal{P} \tag{2b}$$

$$h_p\left(w_p^{oper},\theta_p,w_p^{des}\right)=0 \qquad \forall p\in\mathcal{P} \tag{2c}$$

$$w_p^{des}-\nu^{des}=0 \qquad \forall p\in\mathcal{P} \tag{2d}$$

$$A_p w_p^{oper}-B_p\nu^{oper}=0 \qquad \forall p\in\mathcal{P} \tag{2e}$$

$$w_p^T=\left[\left(w_p^{des}\right)^T,\left(w_p^{oper}\right)^T\right] \qquad \forall p\in\mathcal{P} \tag{2f}$$

A_p and B_p are selection matrices used to link a subset of the local variables of partition *p* to the corresponding sections in the global copy ν. The objective function terms in the partition problems are chosen appropriately to add up to the original objective in (1).

2.2. Applying ADMM to get a distributed algorithm

The individual partition problems in (2) are not trivially separable due to the presence of constraints that enforce the consensus condition. We can solve this problem in a distributed approach using ADMM as described below. The augmented Lagrangian (AL) function of (2) can be formed by taking the constraints (2d) and (2e) to the objective as,

$$L_\rho\left(w_p,\nu,\lambda_p\right)=\sum_{p\in\mathcal{P}}\left[\begin{array}{l}\Phi^{capex}\left(w_p^{des}\right)+\Phi_p^{oper}+\left(\lambda_p^{des}\right)^T\left(w_p^{des}-\nu^{des}\right)\\+\dfrac{\rho^{des}}{2}\left\|w_p^{des}-\nu^{des}\right\|_2^2+\left(\lambda_p^{oper}\right)^T\left(A_p w_p^{oper}-B_p\nu^{oper}\right)\\+\dfrac{\rho^{des}}{2}\left\|A_p w_p^{oper}-B_p\nu^{oper}\right\|_2^2\end{array}\right]$$

where λ_p are the lagrange multipliers associated with the consensus constraints and ρ is a vector of chosen penalty parameters. The AL function is additively separable except for the quadratic penalty terms. The ADMM algorithm involves solving the partition problems while keeping ν and λ fixed and then updating them by keeping the local variables w_p fixed in an alternating fashion until convergence. The i^{th} iteration of the ADMM algorithm thus takes the form

$$\left(w_p\right)^{i+1} = \arg\min_{w_p\in\mathcal{W}_p}\mathcal{L}_\rho\left(w_p,\nu^i,\lambda_p^i\right) \qquad \forall p\in\mathcal{P} \tag{3a}$$

$$\left(\nu\right)^{i+1} = \arg\min_{\nu}\mathcal{L}_\rho\left(w_p^{i+1},\nu,\lambda_p^i\right) \tag{3b}$$

$$\left(\lambda_p^{des}\right)^{i+1} = \left(\lambda_p^{des}\right)^i+\rho^{des}\left(\left(w_p^{des}\right)^{i+1}-\left(\nu^{des}\right)^{i+1}\right) \qquad \forall p\in\mathcal{P} \tag{3c}$$

$$\left(\lambda_p^{oper}\right)^{i+1} = \left(\lambda_p^{oper}\right)^i+\rho^{oper}\left(A_p\left(w_p^{oper}\right)^{i+1}-B_p\left(\nu^{oper}\right)^{i+1}\right) \qquad \forall p\in\mathcal{P} \tag{3d}$$

Step (3a) involves solving an optimization problem for each partition separately while the global variable is kept constant. Thus, each partition problem can be solved in separate

machines and in parallel. Step (3b) is the minimization of the AL function while the local variables in partitions are kept constant. This step for consensus optimization problems reduces to finding the minimum of a quadratic function and can be shown to be the averaging operator (Rodriguez et al., 2018) defined as,

$$v^{i+1}(j) = \frac{1}{|\mathcal{P}_j|} \sum_{p \in \mathcal{P}} w_p^{i+1}(j)$$

where $\mathcal{P}_j \subseteq \mathcal{P}$ denotes the set of partitions connected to the j^{th} element of the global variable. Steps (3c) and (3d) are updates to the lagrange multipliers of consensus constraints and can be carried out in each partition separately. The termination criteria for ADMM iterations are that primal residual (r^{i+1}) and dual residual (s^{i+1}) be reasonably close to zero as explained in Boyd et al. (2010), which for our case is

$$r_p^{i+1} = \begin{bmatrix} \left(w_p^{des}\right)^{i+1} - \left(v^{des}\right)^{i+1} \\ A_p \left(w_p^{oper}\right)^{i+1} - B_p \left(v^{oper}\right)^{i+1} \end{bmatrix} \quad s_p^{i+1} = \begin{bmatrix} \left(\lambda_p^{des}\right)^{i+1} - \left(\lambda_p^{des}\right)^{i} \\ \left(\lambda_p^{oper}\right)^{i+1} - \left(\lambda_p^{oper}\right)^{i} \end{bmatrix} \qquad \forall p \in \mathcal{P}$$

3. The TES Design problem

We use the flowsheet in Figure 2 to represent the heating section of a district heating network where water is used in a closed-loop to satisfy the heating requirement in an area.

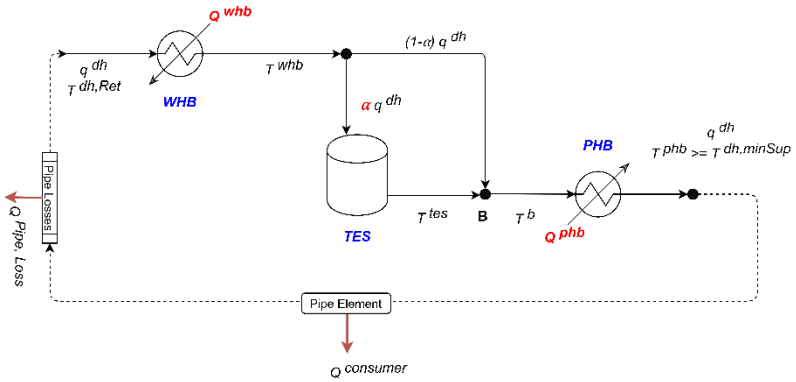

Figure 2: Simplified flowsheet of a district heating system.

The volumetric flow rate of water in the loop is represented as q^{dh} which is determined by the number of consumers and their heating demands at any time. The temperature of water returned is $T^{dh,\mathrm{Ret}}$ and determined by the heat losses in the system and correlated to ambient weather conditions. The operational objective of the heating system is to heat this water to a temperature T^{phb} which must be above a contractually specified temperature $T^{dh,\min \mathrm{Ret}}$. A cheap source of heat is available in the form of a process stream from an industrial process which can exchange heat with the district heating system using the heat exchanger WHB. Any additional heating required in the district heating side is satisfied by using the peak heat boiler PHB. There is a temporal mismatch in the supply of heat from the process stream and the demand for heating. To better manage this

mismatch and decrease the reliance on external utilities, a thermal energy storage system in the form of a simple buffer tank is being considered. This simple TES system can charge/ discharge by raising/ lowering the temperature of the tank by manipulating the flow split α .

Mass and energy balance equations can be written out for the system to model the dynamics of the system, details of which can be found in Prakash (2020). The differential states are $x^T = \left[T^{whb}, T^{tes}, T^{phb}\right]$ and the control inputs $u^T = \left[\alpha, Q^{phb}, Q^{dc}\right]$. We consider the uncertain parameters at each timestep k as $\theta_k^T = \left[q_k^{dh}, T_k^{dh,Ret}, Q^{whb}\right]$ to build the representative scenarios. The design problem is then to find the optimal volume of the TES tank (V^{tes}) that must be installed, given the uncertain profiles of future supply and demand of thermal energy.

4. Results and Discussions

We present a simulation study to demonstrate the distributed approach and compare it against solving the design problem as a single NLP centrally. Two equally likely scenarios with N = 60 discretizations to represent future operations. The profiles for Q^{whb} has a step change at timesteps k = 20 and k = 40, while all other parameters are held constant. Quadratic functions are used to represent capital cost $\phi^{capex} = 0.001\left(V^{tes}\right)^2$ and the operating cost $l = \left(Q^{phb}\right)^2$. The design variables and operations variables were all scaled to be within zero and one while formulating the optimization problem. Scaling facilitated selecting the AL penalty parameter ρ to tune the convergence speed in our distributed approach more easily. The penalty parameters used are $\rho^{des} = 0.001$ and $\rho^{oper} = 0.1$ to roughly balance the magnitudes of the objective and penalization terms.

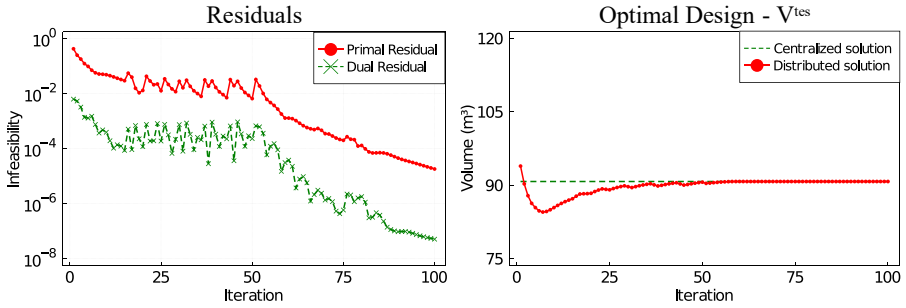

Figure 3: Results from solving the design problem centrally vs distributed approach.

In Figure 3, we can see that the primal and dual residuals become sufficiently small in the ADMM algorithm, indicating convergence in our distributed approach. The optimal design variables can be seen to converge to the same solution as found by solving the problem centrally. An important point to note is that this behavior is not guaranteed by ADMM in the case of nonconvex problems. When applied to nonconvex problems, ADMM need not converge and even when it does converge, it need not converge to an optimal point and must be hence considered just as another local optimization method (Boyd et al., 2010). Although the convergence guarantees for ADMM in the case of

complex nonconvex NLPs are poorly understood, it has been shown to perform satisfactorily in practice (Rodriguez et al., 2018). Three snapshots of the optimal differential state trajectory T^{tes} in the partitions of scenario 1 are shown in Figure 4.

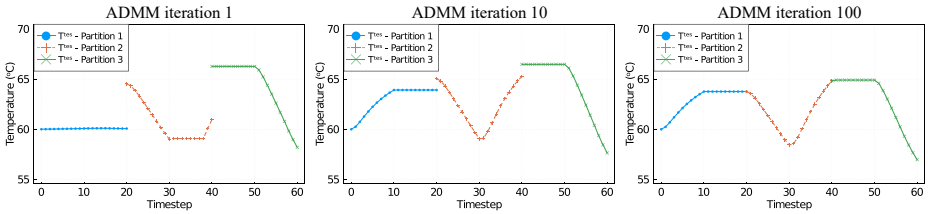

Figure 4: Snapshots at ADMM iterations (i = 1,10, 100) in the partition problems.

An interesting observation in our approach is that all partitions apart from the leftmost partition has the initial condition of the dynamic optimization problem as a variable. Thus in Figure 4, we can see that during the initial iterations (left subplot), the optimal solution is to initialize the TES tank at a high temperature. The penalization terms added are then updated to close the gap between the partitions and achieve consensus (middle and right subplots). In this aspect our approach shares similarities to the multiple shooting (MS) approach in dynamic optimization. The key difference is that MS is solved centrally and the state continuity (consensus constraint) is enforced explicitly as an equality constraint by the solver. In ADMM, we solve it in a distributed way where the consensus constraint is relaxed by forming the AL. It is then enforced implicitly by minimizing the AL.

5. Conclusion

In this paper, we presented the optimal design of a simple TES system under uncertainty as a two-stage nonlinear dynamic optimization problem. Due to limitations in memory of solving the problem centrally in a single machine, an approach for forming smaller partition problems in a general fashion was shown. The ADMM algorithm was applied to coordinate between the subproblems which could be solved separately and in parallel. A simple simulation exercise was used to demonstrate the approach.

References

S. Boyd, N. Parikh, E. Chu, B. Paleto, J. Eckstein, 2010, Distributed optimization and statistical learning via the alternating direction method of multipliers, Foundations and Trends in Machine Learning, 3 (1), 1-122.

S. Prakash, 2020, Distributed optimization using ADMM for Optimal Design of Thermal Energy Storage systems, Master's Thesis, Norwegian University of Science and Technology, Norway.

J. S. Rodriguez, B. Nicholson, C. Laird, V. M. Zavala, 2018, Benchmarking ADMM in nonconvex NLPs, Computers and Chemical Engineering, 119, 315-325.

M. Thombre, S. Prakash, B. R. Knudsen, J. Jäschke, 2020, Optimizing the Capacity of Thermal Energy Storage in Industrial Clusters, 30th European Symposium on Computer Aided Process Engineering. Computer Aided Chemical Engineering, 48, 1459–1464.