s

Description of the Case Study for Gradual Increase

Demonstration of the Results

Vadim Omelčenko, PhD

vadim224@yahoo.com

Optimal balancing of wind parks with virtual power plants, November 2021

Agenda

1. Main results
2. Main Inputs, Functions, and Modules
3. Description of Optimization problems
4. Toy problems – simulated assets parameters
5. Backup

Advantage of usage of GI

Good feasible solutions yielded by GI before the presolving of the entire problem is finished

The entire pool consists of 35 Assets. Within GI a sub-pool of 9 Assets performed the task. The rest of assets was optimized in isolation: optimized against prices. The concatenations of all of these schedules yields a feasible solution of the entire problem. This solution is obtained in 15 seconds. Without GI it took about 200 seconds to obtain better objective value.

MIPgap

Usage of GI enabled to reduce MIPgap 2-3 times.

Demonstration of GI on small problems

The entire computation time is around 5 minutes.

Two problems (case studies) are solved

Optimization types

- Maximization of the cumulative profits
  Coupling constraint is an inequality:

$$A_1 x_1 + \cdots + A_N x_N \geq Task$$

- Minimization of costs.
  Coupling constraint:

$$A_1 x_1 + \cdots + A_N x_N = Task$$

Comparisons

1. MIPgap
2. Objective values
3. Upper bound

Input, Functions, and Output Data

Inputs

Two pickle files that contain the assets' parameters, objective value coefficient and the right-hand side of coupling constraints.

Functions

Optimization of a portfolio

1. GI_Scratch . . . Calculation from scratch. But this function generates warm starts.
2. GI_WS . . . Optimization with warm start.

Optimization of single assets

1. ModelSinglePP_Biogas Based on Equations from the peper
2. ModelSinglePP_BiogasMTRX

Computation Time

The entire computation time is around 5 minutes.

Input, Functions, and Output Data

Outputs

- First Case Study: MIPgap yielded by GI: 0.0591%; MIPgap Brute Force: 0.1098%
- Second Case Study: MIPgap yielded by GI: 0.0166%; MIPgap Brute Force: 0.1036%

Main pieces of Code

The content from the zip file has to be stored in the same directory. The programs CLASSIC_METHOD_GI and CLASSIC_METHOD_GI.py_Eq.py take the inputs from the pickle files TestGI.pkl and OptiInput.pkl, and they use functions defined in libraries gradual_increase_lib and gradual_increase_lib_Eq.

Modules

There are two libraries: gradual_increase_lib and gradual_increase_libEq which contain functions for optimization and refer to the first and the second second case study, respectively.

Constants and variables

Constants

- $\mathrm{Frc}_t^{(\mathrm{DA})}$ ... The vector of consecutive day-ahead prices
- T ... The length of $\mathrm{Frc}_t^{(\mathrm{DA})}$ or in other words the horizon
- $\mathrm{C}^{\mathrm{v}}_{\mathrm{ki}}(\mathrm{C}^{\mathrm{v}}_{\mathrm{ki}})$ ... The switch-on (switch-off) costs of Turbine i of Asset k
- $Pmax_{k,i}(Pmin_{k,i})$ ... The maximum (minimum) power produced of Turbine i of Asset k
- F_k ... The constant gas inflow into the storage

Comparisons

1. $p_{t,k,i}$ ... The power produced at time t by Turbine i of Asset k
2. $u_{t,k,i}, (u, v, w, 3bin)$ ... The binary variable that indicates that Turbine i of Asset k is on. v and w are decisions to switch on or switch off, respectively
3. $L_{t,k}$ ... The storage level of the storage facility of Asset k at time t

Maximization of the Revenues of the Portfolio of Assets

The optimization problem is formulated as follows:

$$\mathrm{maximize} \sum_{t=1}^{T} \sum_{k=1}^{\#\mathrm{A}} \sum_{i=1}^{\#\mathrm{T}(k)} \left(\mathrm{Frc}_t^{(\mathrm{DA})} p_{t,k,i}^{\mathrm{DA}} - \mathrm{C}_{ki}^{v} v_{t,k,i} - \mathrm{C}_{ki}^{w} w_{t,k,i} \right)$$

s.t. Isolated constraints associated with each asset k

$$\sum_{t=1}^{T} \sum_{k=1}^{\#\mathrm{A}} \sum_{i=1}^{\#\mathrm{T}(k)} p_{t,k,i}^{(\mathrm{DA})} \geq Task.$$

Maximization of the Revenues of the Portfolio of Assets

$$\text{maximize} \sum_{t=1}^{T} \sum_{k=1}^{\#\mathrm{A}} \sum_{i=1}^{\#\mathrm{T}(k)} \left(\mathrm{Frc}_t^{(\mathrm{DA})} p_{t,k,i} - \mathrm{C}_{ki}^{v} v_{t,k,i} - \mathrm{C}_{ki}^{w} w_{t,k,i} \right)$$

$$\text{s.t.}\, u_{t,k,i} - u_{t-1,k,i} = v_{t,k,i} - w_{t,k,i}, \quad u \text{ is binary}$$

$$Pmax_{k,i} u_{t,k,i} - p_{t,k,i} \geq 0, \quad p_{t,k,i} - Pmin_{k,i} u_{t,k,i} \geq 0,$$

$$L_{t,k} = L_{t-1,k} + F_k - \sum_{i=1}^{\#\mathrm{T(k)}} p_{t,k,i}, \quad L_{t,k} \in [0, \mathrm{C}],$$

Timing constraints: min up- and downtime (See Backup)

$$\sum_{t=1}^{T} \sum_{k=1}^{\#\mathrm{A}} \sum_{i=1}^{\#\mathrm{T}(k)} p_{t,k,i} \geq Task.$$

Minimization of the Costs of the Portfolio of Assets

$$\text{minimize} \sum_{t=1}^{T} \sum_{k=1}^{\#\mathrm{A}} \sum_{i=1}^{\#\mathrm{T}(k)} (\mathrm{C}^{v}_{ki} v_{t,k,i} + \mathrm{C}^{w}_{ki} w_{t,k,i})$$

$$\text{s.t.}\, u_{t,k,i} - u_{t-1,k,i} = v_{t,k,i} - w_{t,k,i}, \quad u \text{ is binary}$$

$$Pmax_{k,i} u_{t,k,i} - p_{t,k,i} \geq 0, \quad p_{t,k,i} - Pmin_{k,i} u_{t,k,i} \geq 0,$$

$$L_{t,k} = L_{t-1,k} + F_k - \sum_{i=1}^{\#\mathrm{T(k)}} p_{t,k,i}, \quad L_{t,k} \in [0, \mathrm{C}],$$

Timing constraints: min up- and downtime (See Backup)

$$\sum_{t=1}^{T} \sum_{k=1}^{\#\mathrm{A}} \sum_{i=1}^{\#\mathrm{T}(k)} p_{t,k,i} = Task.$$

The assets' parameters are simulated

- The portfolios of assets are represented as dictionaries in Python.
- The number of turbines, the Pmin, Pmax, the capacity of the storage and the constant flow of the fuel into the storage can be simulated using random number generators in numpy.
- The price vector is a random series of historical day-ahead prices on the German market. The right-hand side was simulated. In the input files, there are five right-hand sides of the coupling constraints, and one scenario of prices.

Backup – Timing Constraints

If at time t a turbine i of asset k is on than it has to be on for at least $UT^{(k,i)}$ periods which is exressed as follows:

$$\sum_{j=t-UT^{(k,i)}+1}^{t} v_{j,k,i} \leq u_{t,k,i} \quad \forall t, k, i.$$

If at time t a turbine i of asset k is off than it has to be off for at least $DT^{(k,i)}$ periods which is exressed as follows:

$$\sum_{j=t-DT^{(k,i)}+1}^{t} w_{j,k,i} \leq 1 - u_{t,k,i} \quad \forall t, k, i$$

Practical Implementation

1. In the practical implementation, the following variable is used:

$$Smp_t = \sum_{k=1}^{\#\mathrm{A}} \sum_{i=1}^{\#\mathrm{T}(k)} p_{t,k,i}, \quad \forall t \in \{1, T\}.$$

2. The output of optimization for each Asset k contains the following variables:

$u_i, v_i, w_i, p_i, Smp, Storage, Spill$. The *Spill* is penalized