Folder Organization

In the following description of the project's folder, the names "($VARIABLE)" refer to:

- PLANT ∈ {COCHE ; MOUTI ; MTRIG ; POMBL ; RANDE ; TIGNE} → Name of the hydropower plant
- ACTIVE_TURBINES ∈ N → Number of active turbines
- STOCHASTICITY ∈ {no ; robust ; free} → Type of stochasticity on the duration of the maintenances
- SEASON ∈ {winter ; summer} → Season of the scenario
- NUMBER_OF_SCENARIOS ∈ {2 ; 1} → Number of scenarios
- APPROXIMATION_METHOD ∈ {convex_hull/20_hyperplanes, ...} → Name of the approximation method

PROJECT

- data
  - maintenances
    - maintenance_tasks.json
    - ($STOCHASTICITY)_stochasticity.jl
  - production_functions
    - ($APPROXIMATION_METHOD)
      - parameters.jl
      - ($PLANT)_($ACTIVE_TURBINES).json
  - scenarios
    - ($SEASON)
      - ($NUMBER_OF_SCENARIOS)_scenarios
        - parameters.jl
        - general_parameters.json
        - inflow_scenarios.json
        - price_scenarios.json
- GMSP.jl
- complete_test_case_Z.jl
- M L
- P l o t _ a p p r o x i m a t i o n

Maintenances

The JSON file "maintenance_tasks.json" is a dictionary containing the following keys:

- M → List of the names of the maintenance tasks
  ↪ M = ["MT1", "MT2", ...]
- T_m → Dictionary linking each $m \in$ M to the list of the time periods when m can begin
  ↪ T_m["MT1"] = [1, 2, ...]
- M_i → Dictionary linking each $i \in$ I to the list of the maintenances associated to i
  ↪ M_i["TIGNE"] = ["MT1", "MT2", ...]
- C_m_t → Dictionary linking each $m \in$ M to the list of the cost of m if m begins at $t \in$ T
  ↪ C_m_t["MT1"] = [36, 36, ...]
  ↪ The prices are in € and the lists are of the same size as T
- D_m → Dictionary linking each $m \in$ M to its expected duration
  ↪ D_m["MT1"] = 5
  ↪ The durations are in days
- E_m → Dictionary linking each $m \in$ M to the earliest time period when m can begin
  ↪ E_m["MT1"] = 1
- L_m → Dictionary linking each $m \in$ M to the latest time period when m can begin
  ↪ L_m["MT1"] = 6

The Julia files "($STOCHASTICITY)_stochasticity.jl" define the values of the stochastic parameters:

- Alpha_1 → Value of the parameter α_1 in the considered type of stochasticity
- Alpha_2 → Value of the parameter α_2 in the considered type of stochasticity
- Nu → Value of the parameter ν in the considered type of stochasticity
- Gamma_1 → Value of the parameter γ_1 in the considered type of stochasticity
- Gamma_2 → Value of the parameter γ_2 in the considered type of stochasticity

Note that because of some changes, old names of stochasticity are used. The correspondance is as follows:

- no → Deterministic case
- robust → Pessimistic stochasticity
- free → Optimistic stochasticity

PRODUCTION FUNCTIONS

For each plant $i \in$ I and each number of active turbines $k \in$ K$_i$ allowed in the model for the hydropower plant *i*, there is a JSON file named "*i*_*k*.json" containing the data relative to the hydropower function of *i* when *k* turbines are active for the given approximation method.

The approximation methods available in this project are the following:

- convex_hull/10_hyperplanes → Convex Hull approximation with 10 hyperplanes
- convex_hull/20_hyperplanes → Convex Hull approximation with 20 hyperplanes
- convex_hull/30_hyperplanes → Convex Hull approximation with 30 hyperplanes
- piecewise_convex_hull → Piecewise Convex Hull Approximation (Extended Model)
- piecewise_convex_hull_bis → Piecewise Convex Hull Approximation (Compact Model)
- picewise_linear → Piecewise Linear Approximation of Two-Variables Functions
- polynomial → Polynomial Approximation

For each approximation method, a Julia script "parameters.jl" is designed to retrieve the data from those JSON files and make them usable in another script.

The description of the JSON files is explained directly in those Julia scripts.

For example, in the "parameters.jl" file of the approximation method "convex_hull/10_hyperplanes", we can read the following line in the declaration of the variables:

```
# Definition of the coefficients of u in the different hyperplanes [MWh*s/(m^3/day)] (cf. β_h^u)
Beta_u_i_k_h = Dict{String, Dict{Int64, Array{Float64, 1}}}(i => Dict{Int64, Array{Float64, 1}}() for i in I);
```

In the part of the script which parses the JSON files, we can then read:

```
Beta_u_i_k_h[i][k] = hyperplanes_equations["Beta_u"]
```

This means that the key "Beta_u" in the JSON file "*i*_*k*.json" contains a list of the values of the coefficients of *u* in the different hyperplanes.

SCENARIOS

For a given scenario, a Julia script "parameters.jl" is designed to retrieve the data from the associated JSON files and make them usable in another script.

In the same way than for the approximation methods, everything is explained in the "parameters.jl" file.

For example, in the "parameters.jl" file of the scenario "summer/2_scenarios", we can read the following line in the declaration of the variables:

```
# Definition of the probabilities of realization of the different scenarios (cf. ϕω)
Phi_omega = convert(Dict{String, Float64}, JSON.Parser.parse(open(string(FOLDER, "general_parameters.json"), "r"))["Phi"]);
```

This means that the key "Phi" in the JSON file "general_parameters.json" contains a list of the values of the probabilities of realization of the different scenarios.

GENERAL PARAMETERS

This Julia script is there to initialize all the general parameters of the model regarding the hydropower plants and the maintenances (i.e.) the parameters that do not change from a scenario to another and from an approximation method to another. Everything is explained in details in the file.

SOLVER

The Julia script "GMSP.jl" contains the function "solve_model" which requires the fol- lowing arguments:

- APPROXIMATION_METHOD::String → Name of the approximation to use to solve the model
- SEASON::String → Season considered in this model
- NB_SCENARIOS::Int64 → Number of scenarios to use in this model
- STOCHASTICITY::String → Type of stochasticity to use to model the uncertainty on the duration of the maintenances
- SCHEDULE_PATH::String → Optional argument − Give the path to a JSON file which contains a maintenance planning to fix this schedule in the model. The solver will then solve the unit-commitment problem with the approximation APPROXIMATION_METHOD for this given schedule. It has to be of the form ".../file_with_schedule.json" and it must contain the values to set up the variables "y_m_t", "r_i_t" and "z_i_t_k" in the model.

The results of a run are saved in the following file:

"results/($SEASON)/($NB_SCENARIOS)_scenarios/($STOCHASTICITY)_stochasticity/($APPROXIMATION_METHOD)_5T.json"

ML

Two machine learning methods, XGBoost and LSTM, are applied to two datasets: one for Norway and one for China. The Norwegian dataset includes three variables—rainfall, inflows, and environmental factors—while the Chinese dataset contains only inflows. The `_org.csv` files contain the original data, while the other files are used for training the model. The problem is solved using Python.