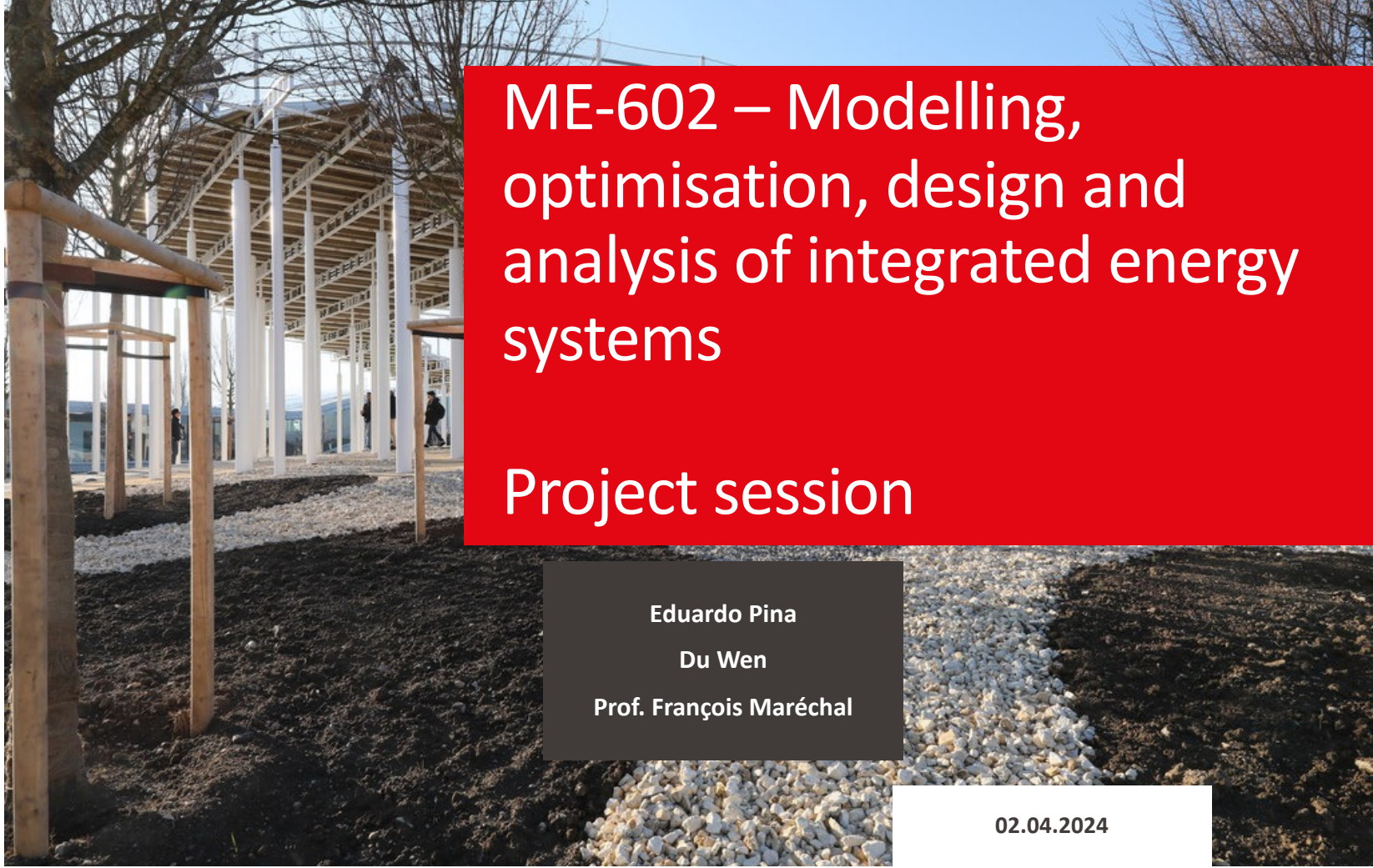


ME-602 – Modelling, optimisation, design and analysis of integrated energy systems

Project session

Eduardo Pina

Du Wen

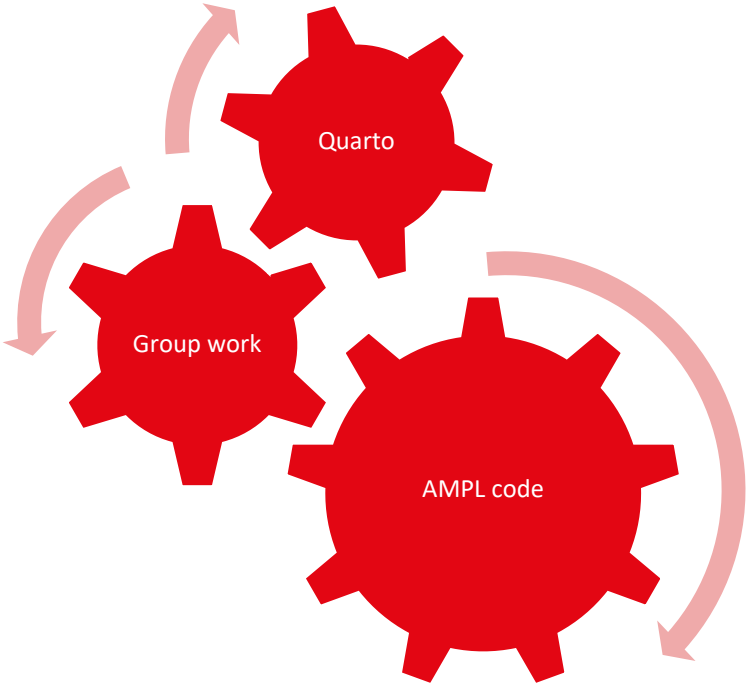
Prof. François Maréchal

02.04.2024

École polytechnique fédérale de Lausanne

General instructions

- All files are on Moodle.
- Project session: 13:30–17:30.
- **Mattmark room**.



Project

- The course is divided in **4 main tasks**:

Day 1. Non-Linear Programming (**NLP**) problem on cider bottling production line; integration with heat recovery options.

Day 2. Linear Programming (**LP**) problem on combined heat and water integration.

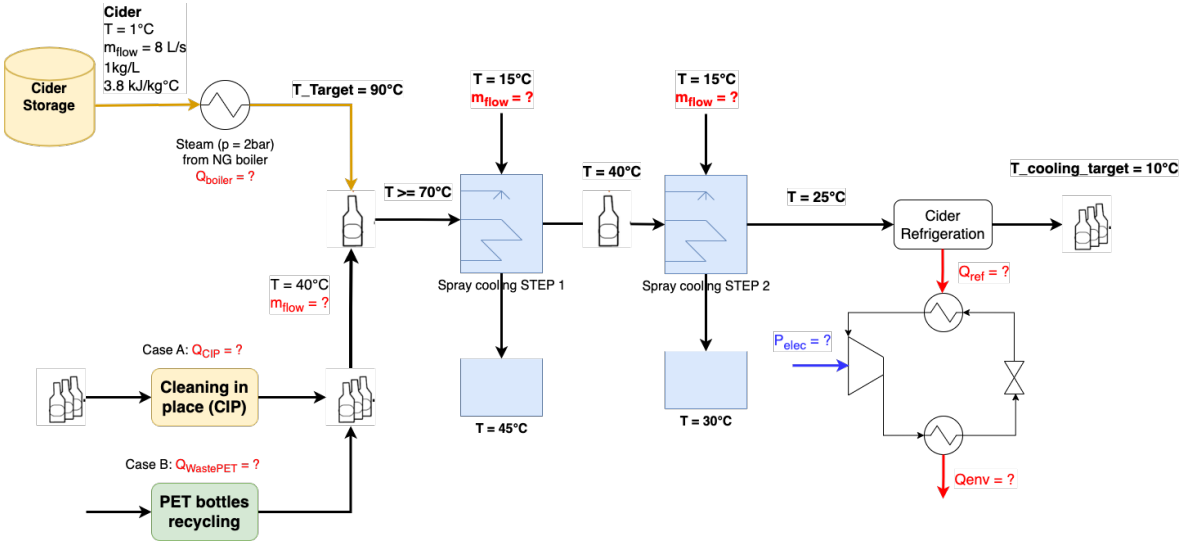
Day 3. Life Cycle Assessment (**LCA**) of energy systems (PET and glass bottles).

Day 4. System integration including renewable energy technologies and multi-objective optimization approach based on Mixed-Integer Linear Programming (**MILP**).

- Each task corresponds to 1 workday of the material covered in the lectures.
- Report delivery **31st May, 2024** (23:59 CET).

Project

- **NLP** – Non-linear problem formulation.

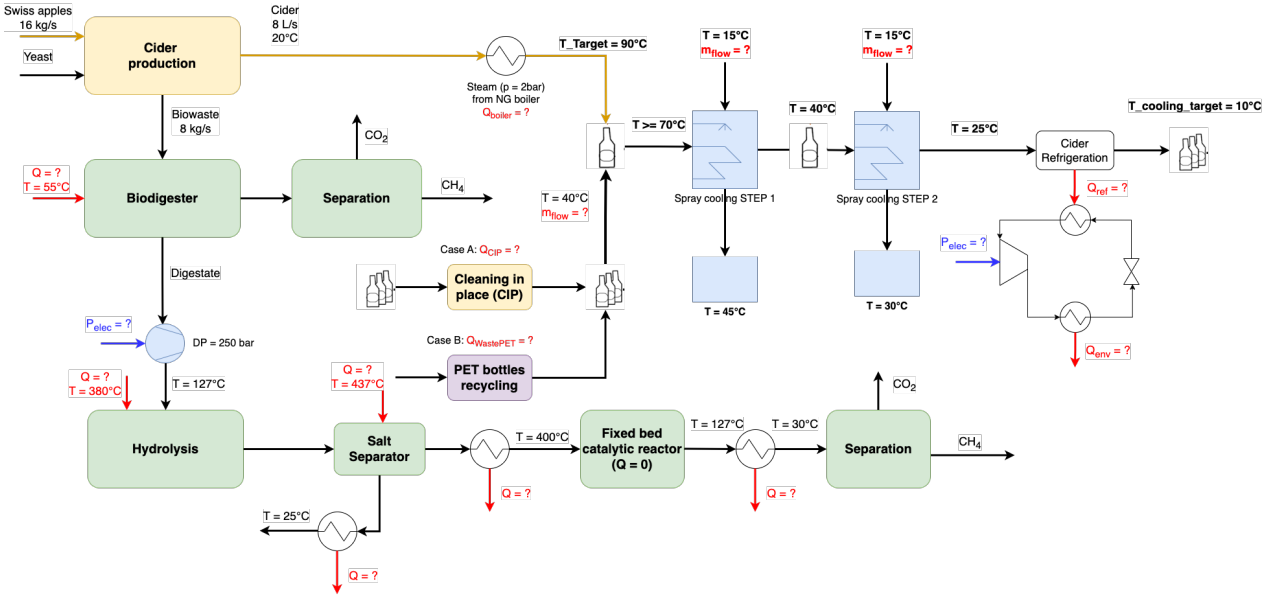


Existing bottling process

- Calculate current bill.
- Suggest heat exchanger integration.
- Make sure target temperatures are reached.
- Do we really need 2 spray coolers?
- Follow Q1–Q7.

Project

- **NLP** – Non-linear problem formulation



Bottling process with biodigester

- Complete mass and energy values.
- Calculate the new optimized case.
- Suggest heat exchanger integration.
- Pay-back time of investment.
- Follow Q8-Q11.

A Mathematical Programming Language (AMPL) code

- Only a **.mod** file: Example of Day 1: **juice17.mod**.

Parameters

```
# Heat capacities
param cp_juice          := 3.8;
param cp_water          := 4.18;
param cp_empty_bottles  := 0.75;
param cp_bottles        := 2.6;

# Flow rates
param m1                := 8;
param m_empty_bottles   := 0.37;
param bps               := m1/0.5;
param m_bottles         := bps*m_empty_bottles + m1;

# Temperatures
param T1                := 1;
param Ttarget           := 90;
param Tempty_bottles    := 40;
param Twater            := 15;
param Trefrigeration    := 10;
param DTmin             := 2;
param Tsteam            := 120;
```

Variables

```
#######################################################
## VARIABLES
#######################################################

var ex1 >= 1;           # example variable
var ex2 >= 3 <= 50;     # example variable
```

Constraints

```
#######################################################
## CONSTRAINTS
#######################################################


#########################################
# Example constraint

subject to example1:
ex1 = NGprice*ex2 + 3;
```

Objective and solve instruction

```
minimize Obj: ex1;


solve;
```

Printing options

```
#####################################################################
## DISPLAY COMMANDS
#####################################################################
printf '\n\n';
printf '-----------------------------------------------------\n';
printf 'RESULTS\n';
printf '-----------------------------------------------------\n';
printf 'Objective function value          :\t %0.4f $\n', ex1;
printf '-----------------------------------------------------\n';
printf 'Test succeeded\n';
printf '-----------------------------------------------------\n';
```

A Mathematical Programming Language (AMPL) code

- Different files: Example of Day 2.

- A **.mod** file

```
param juice_mass;        # juice mass flow [kg/s]

# calculated parameters

param bottle_mass_flow  := juice_mass/bottle_cap*bottle_weight; # mass flow of bottles
param bottle_cold_dh    := bottle_mass_flow * bottle_cp * abs(bottle_cold_tin - bottle_cold_tout);
param bottle_hot_dh     := bottle_mass_flow * bottle_cp * abs(bottle_hot_tin - bottle_hot_tout);
param heat_load_initial := ; # heat load of the initial rinsing and soda bath (what you have calcu
param heat_load_final   := ; # heat load of the final rinsing (what you have calculated, although
```

Calculated Parameters

```
# ----------
# sets

set SOURCES default {};
set SINKS default {};
set UNITS := SOURCES union SINKS;
set UNITS_MASS := SOURCES inter SINKS;
set CONTAMINANT default {};

# ----------
# parameters based on sets

param units_cont_max_in {u in SINKS, c in CONTAMINANT} default 0;
param units_cont_max_out {u in SOURCES, c in CONTAMINANT} default 0;
param units_mass_load {u in UNITS_MASS, c in CONTAMINANT} default 0;
param t_u {u in UNITS diff UNITS_MASS} default 15;

# minimum and maximum size of units
param units_fmin {u in UNITS} :=
    if (u in UNITS_MASS) then
        max {c in CONTAMINANT} units_mass_load[u,c]*1000/units_cont_max_out[u,c]
    else
        0;

param units_fmax {u in UNITS} :=
    if (u in UNITS_MASS) then

    else
        100;
```

Sets and parameters belonging to sets

```
##########################################################
## VARIABLES
##########################################################

var ex1 >= 1;           # example variable
var ex2 >= 3 <= 50;     # example variable
```

Variables

```
##########################################################
## CONSTRAINTS
##########################################################

#########################################
# Example constraint

subject to example1:
ex1 - MGprice*ex2 = 3;
```

Constraints

- A **.dat** file

```
set SOURCES     := u_final u_middle u_initial u_fresh u_steam;
set SINKS       := u_final u_middle u_initial u_waste;
set CONTAMINANT := dirt;

param units_cont_max_in :=
    u_final     dirt        20
    u_middle    dirt        150
    u_initial   dirt        550
    u_waste     dirt        1000;

param units_cont_max_out :=
    #final rinsing#     dirt        60;

param units_mass_load :=
    #final rinsing#     dirt        0.26

# steam at 2 bar
# TEMPERATURE OF THE UNITS
param t_u :=
    u_fresh_water 15;
# INPUT PARAMETERS REQUIRED
param juice_mass        := 19.0793; # juice mass flow [kg/s]
param bottle_weight     := 0.37;    # bottle weight
param bottle_cap        := 0.5;     # bottle capacity [l/bottle = kg/bottle]
param bottle_cp         := 0.75;    # heat capacity [kJ/kgK]
param bottle_cold_tin   := 15;      # inlet temperature of cold dirty bottle [C]
param bottle_cold_tout  := 70;      # outlet temperature of cold dirty bottle [C]
param bottle_cold_dt    := 7;       # delta T approach
param bottle_hot_tin    := 70;      # inlet temperature of hot clean bottle
param bottle_hot_tout   := 40;      # outlet temperature of hot clean bottle
param bottle_hot_dt     := 2;       # delta T approach
param eff_he_bottle_in  := 7;       # assuming an efficiency in pre-rinsing for heat
param eff_he_bottle_out := 7;       # the bottles are soaked in water, higher effici
param price_water       := 0.01;    # price of water CHF/kg
param price_steam       := 0.06;    # price of steam CHF/kWh
param op_time           := 4350;    # operating hours per year
param water_dt          := 7;       # heat recovery approach temperature for water
```

- A **.run** file

```
model cip_model.mod;
data  cip_data.in;

option solver 'minos';
#option solver 'snopt';
#option minos_options  'timing = 1';
#option snopt_options  'timing = 1 meminc = 1 iterations = 1000000 feas_tol = 0.0001 ';

option display_1col 10000;
option display_eps 0.0001;
option presolve_eps 1e-4;
option omit_zero_rows 1;
option omit_zero_cols 1;
```

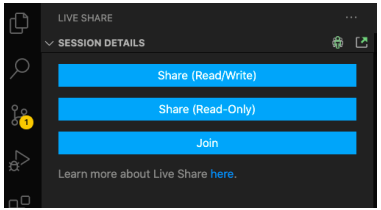
Load model and data

Define solver and solver options

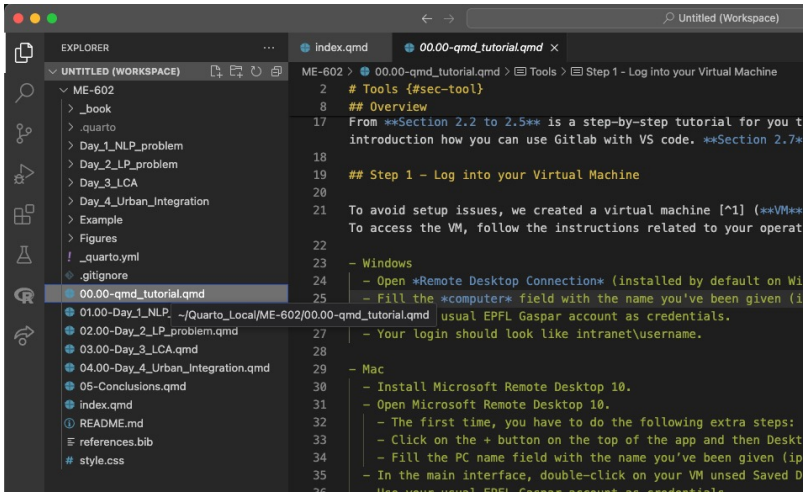
Printing commands

Tools

Visual Studio Code



00.00-qmd_tutorial.qmd -> Report template



- Report template on Moodle and Gitlab.
- Collaborative work.
- Track changes easily.
- IDE with AMPL interface (install the extension).
- Collaborative code editing possible.

Students distribution

- 2 groups

Group A	Group B
• Jaïr Campfens • Soline Corre • Sanjay Venkatachalam • Arthur Waeber	• Goel Abhishek • Vibhu Baibhav • Manuel Cintas • Sai Ravi