Increasing Energy Efficiency in the Water Sector (IEE)
Renewable Energy in the Water Sector (REW)

<u>Manual</u>

Open Source Energy Water Model Jordan (*enwajo*)

Background

As part of the Renewable Energy in the Water Sector (REW) project, the new open source energy water model – *enwajo* – has been developed for and with the Jordanian water and energy sector. This model aims to support the integrated planning among both sectors and to understand the challenges of each other. With this approach a unique and useful tool is provided to strengthen the exchange and to make use of synergies over time. Doing so *enwajo* addresses one key challenge in the Jordanian Water and Energy sector.
The following manual provides a short overview about how to install *enwajo* and how to use it to create benefits for the water and energy sector. The first version of the model was developed to assess the effect of a pumped hydro storage on the Jordanian energy system. Due to the open source approach, adaptations for different assessments are possible.

Developed by:

Europa-Universität Flensburg
Centre of Sustainable Energy Systems (ZNES)

September 2020

Table of Contents

Introduction

Why an open source approach has been selected?

To enable a common understanding of the following manual, let us understand what is to gain and what we are doing by using an open source energy system modeling tool.

1. Using an energy system model can **improve the understanding of the system** set up itself. Especially the included result plots, which are automatically generated, can show you e.g. the energy mix specific for every hour of the year.
2. Create a **cost-optimized** potential **future energy system** using backcasting rather than forecasting. One does not have to follow expected trends and fixed contracts but can analyze freely various electricity mixes and adjust to flexible developments.
3. The power of **open source** models is extraordinary. First and foremost (!) open source does not mean your results or input data are publicly shared and available. This is not the case. It simply means, the source code itself is openly available and can be inspected, used and altered by everyone. This leads to the following positive benefits:
    1. Enables any user to change the input data, constraints and any other aspect of the model. This increases transparency and reproducibility of results.
    2. Low cost of ownership, as no licensing costs are required.
    3. Flexible adjustment to new given circumstances within the system.
    4. Increased transparency can increase acceptance of proposed models.

Who should use the new model and how to apply *enwajo*?

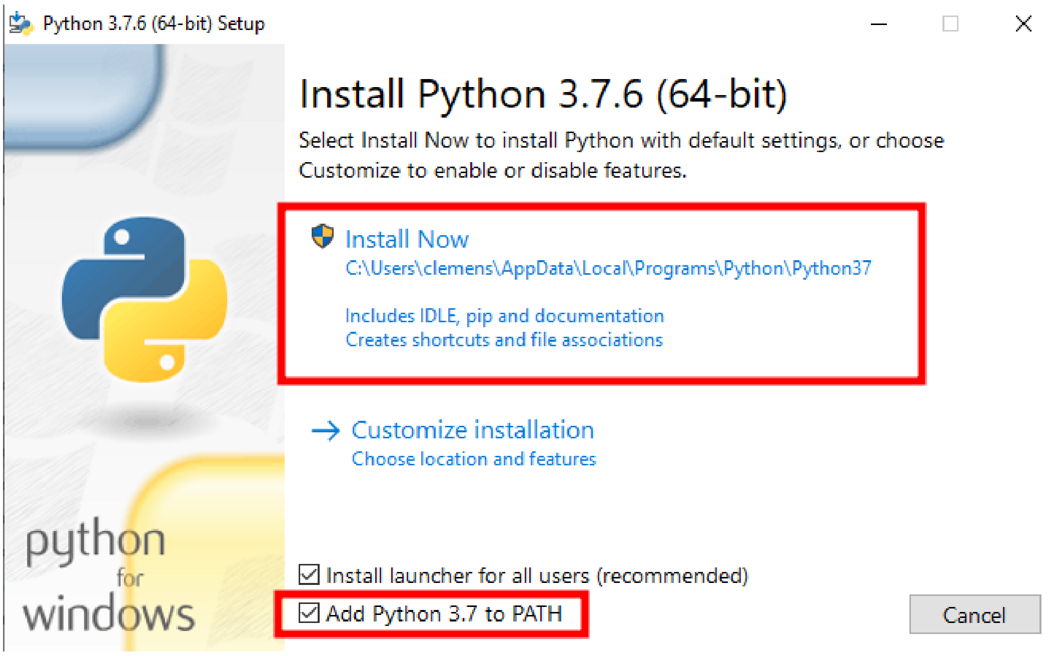
Energy system modelling can be beneficial and is accessible for anyone in the energy or water or any other relevant sector. Or, to be honest, anyone who is interested. Using an open source tool can improve the understanding between sectors, as well as the application of different forms of energy (e.g. renewable or fossil in combination with storage) and the system itself. It can be a facilitator to make use of synergies in both sectors and therefore provides huge potential for planners and decision makers in the Jordanian water and energy sector.

The new energy water model – *enwajo* – is easy to install and to run. Following the step-by-step manual below might seem intimidating at first but is really simple and easy to follow without extensive modeling or programming knowledge.

enwajo – Installation Guide

Install Python

1. Uninstall Anaconda (if it was previously installed).
2. Uninstall Python and Python Launcher (if another version from 3.7.6 was previously installed).
3. Download and install Python 3.7.6 from:
   https://www.python.org/ftp/python/3.7.6/python-3.7.6-amd64.exe .
   If the link does not work, you can find the file at:
   https://www.python.org/downloads/release/python-376/ .
   1. Make sure to enable option "Add Python 3.7 to PATH".
   2. Use the standard installation option.



   If you do not have Admin privileges, uncheck the option “Install launcher for all users”.
   3. At the end of the wizard, use the option to "Disable path length limit", if available.
   4. To finish the installation, please reboot your machine.

Install the new energy water model *enwajo*

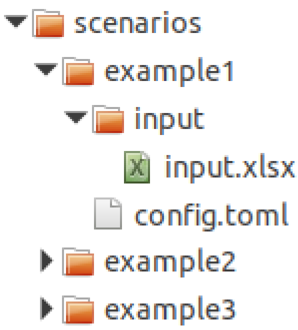
1. Download Source Code as a zip file from: https://github.com/znes/enwajo/releases .
2. Unzip it to a location of your choosing.
3. Install the necessary packages with double clicking on install.py .
4. To test the installation:
    1. Run the model with double clicking on run.py
    2. Follow the instructions in the console output
    3. When prompted choose a scenario folder, e.g. scenarios/example1

If you have any problems on the way, create an issue at:
https://github.com/znes/enwajo/issues .

Usage of *enwajo*

enwajo was designed to simulate the cost optimized operation of an energy system. The methodology is described in the mathematical model description in the docs folder. Additionally, you can see all relevant constraints in the file **model.py**. Here you can modify and add constraints as well, if you want to adapt the model for other purposes.

The energy system to analyze is defined with scenario parameters. A new scenario always consists of a separate folder with a config.toml file and an input subfolder:



The **config.toml** file is just a plain text file in a specific format (toml) and holds general parameters like:

- the amount of timesteps (t_start, t_end) to use for the simulation,
- colors for the plots ([colors} section),
- if you want to model with minimum load constraints for the power plants (pmin=true),
- if you want to include part load efficiencies (eta_partial=true).

It can be opened and edited with a normal text editor like **Notepad**. Here is a small example about how you can use the **config.toml** file:

> <u>Example: config.toml</u>
> Because the runtime of the model depends on the size of variables you can start with only a small amount of timesteps for the development of a scenario. E.g. with only the first 24 timesteps which would represent the first day of the year:
>
> t_start = 0
> # end timestep (8760 for a whole year)
> t_end = 24
>
> On that basis you can find a convenient setting of scenario assumptions like installed power plants and total demand of electricity. The runtime for a model run will be very short. Then you can simulate a whole year in hourly resolution. You just have to change the following in the config.toml file:
>
> t_start = 0
> # end timestep (8760 for a whole year)
> t_end = 8760

The **input.xlsx** file – which is the main file for you to easy enter key scenario parameters according to your needs – holds all necessary scenario assumptions. It is separated into different worksheets:

The **worksheet conventional** holds all necessary parameter for conventional power plants. Every carrier assigned has to be specified in the carrier worksheet as well. p_max and p_min have only to be specified if you want to model with minimum load constraints (see example2 below). a and b have only to be specified if you want to model with part load efficiencies (see example3 below).

column	unit or range	description
unit		name of power plant
p_nom	MW	nominal electrical power
eta	>= 0, <=1	efficiency
vom	$/MWh	variable operational and maintenance cost
p_max	>= 0, <=1	max
p_min	>= 0, <=1	
carrier		refers to carrier worksheet
a		for part load efficiencies, see docs/model-description.pdf
b		for part load efficiencies, see docs/model-description.pdf
tech	st, gt, cc, de	type of technology (steam turbine, gas turbine, combined cycle, diesel engine)

The **worksheet renewable** contains all parameter for fluctuating power sources. In addition to a conventional power plant, a scaled profile (normalized with the nominal power) has to be added in the **worksheet profiles**. This profile restrict the maximum power output for every hour.

column	unit or range	description
unit		name of power plant
p_nom	MW	nominal electrical power
carrier		refers to carrier worksheet
profile		refers to profiles worksheet
vom	$/MWh	variable operational and maintenance cost

The **worksheet demand** contains the annual electricity demand in MWh. It refers to a scaled demand-profile (normalized with the annual demand) in the **worksheet profiles**.

The **worksheet storage** holds all parameter for storage units (e.g. pumped hydro storage, batteries).

column	unit or range	description
units		name of storage unit
p_nom_in	MW	nominal electrical input
p_nom_out	MW	nominal electrical output
e_nom	MWh	nominal storage capacity
eta_in	>=0, <=1	charging efficiency
eta_out	>=0, <=1	discharging efficiency
loss	>=0, <=1	losses as share of filling level per hour
e_init	>=0, <=1	initial storage level as share of nominal storage capacity
vom	$/MWh	variable operational and maintenance cost

The **worksheet carrier** contains information on the energy carrier being used as fuel/source for conventional and renewable power plants.

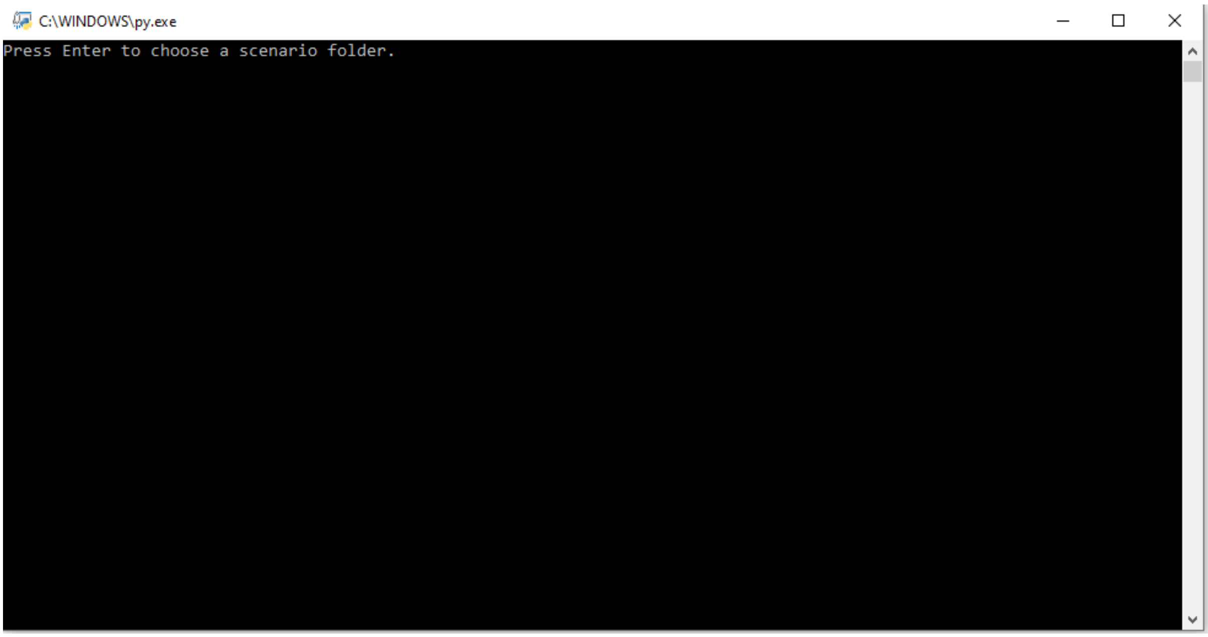
column	unit or range	description
name		name of carrier
cost	$/MWh	fuel cost for carrier (per MWh of used carrier)
emission_factor	t/MWh	CO2 emission factor (per MWh of used carrier)

You can either edit an already existing scenario or create a new scenario by copying and already existing scenario folder. In the examples section below you can find relevant screenshots.

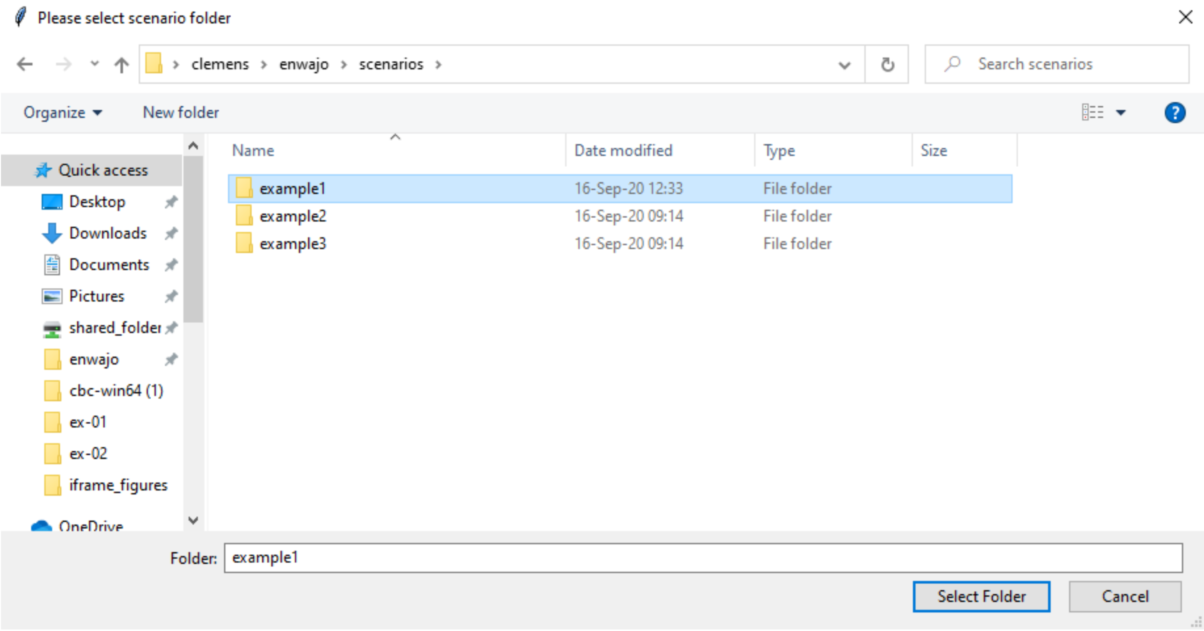
To run the model, just double click on **run.py** and follow the instructions in the console output:

Name	Date modified	Type	Size
__pycache__	16-Sep-20 11:55	File folder	
docs	16-Sep-20 09:14	File folder	
scenarios	16-Sep-20 09:14	File folder	
cbc.exe	10-Aug-20 06:37	Application	4,286 KB
coin-license-for-cbc.txt	10-Aug-20 06:37	Text Document	12 KB
install.py	16-Sep-20 11:54	Python File	1 KB
model.py	16-Sep-20 11:46	Python File	13 KB
plotting.py	16-Sep-20 11:51	Python File	4 KB
README.md	16-Sep-20 12:19	MD File	2 KB
requirements.txt	16-Sep-20 09:14	Text Document	2 KB
run.py	16-Sep-20 11:55	Python File	1 KB

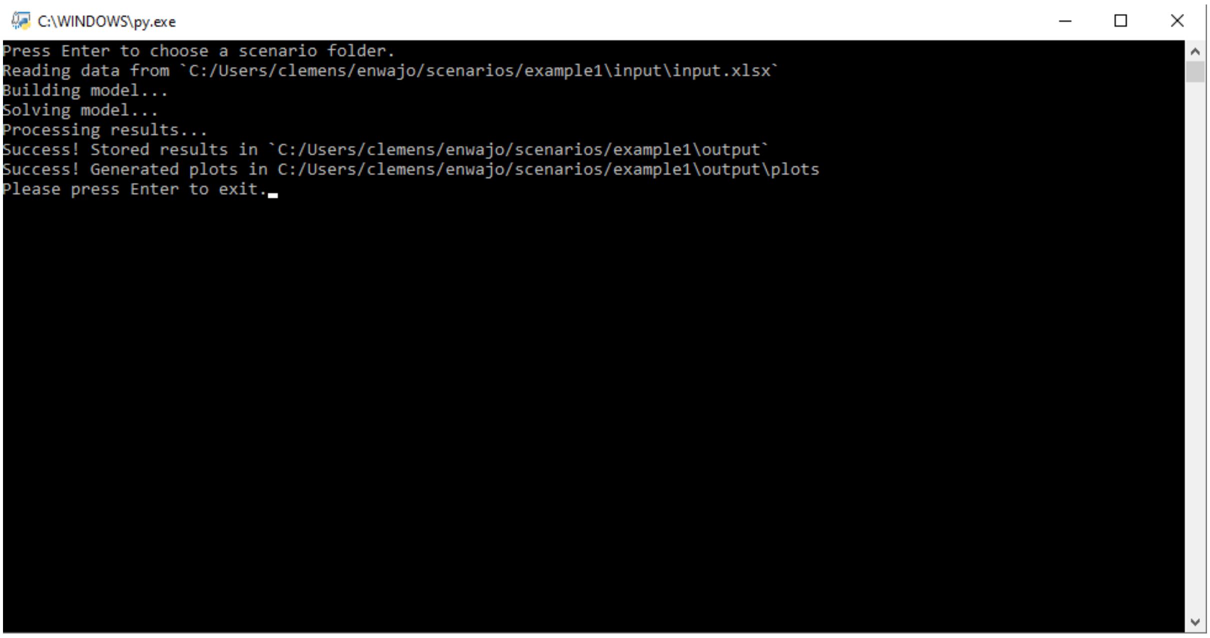
A new window will open which displays the console output:



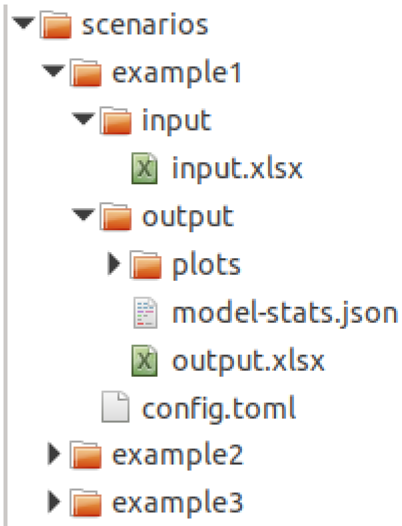
When prompted choose the respective scenario folder, you want to model.



After selecting a scenario folder, the respective scenario is being simulated.



When the model is finished the results will be saved in a subfolder called output in the scenario folder:



All relevant data is stored in the file output.xlsx and you can find some standard plots in the same folder.

Examples

Included Examples

There are three example scenarios included with the model, from which you can learn. They only differ slightly to illustrate the flexibility of the model itself when it comes to depth of detail.

example1 is a more or less realistic and simple scenario of the Jordanian energy system.

scenarios/example1/input/input.xlsx:

	A	B	C	D	E	F	G	H	I	J
1	unit	p_nom	eta	vom	p_max	p_min	carrier	a	b	tech
2	Aqaba1	121	0.5	0.509			gas			st
3	Aqaba2	121	0.5	0.509			gas			st
4	Risha	30	0.5	0.764			gas			gt
5	Rehab	25	0.5	2.43			gas			gt
6	RehabCC	270	0.5	2.43			gas			cc
7	Samra1	308	0.5	0.109			gas			cc
8	Samra2	300	0.5	0.109			gas			cc
9	Amman-East	424	0.5	0.071			gas			cc
10	Qatranah	420	0.5	0.357			gas			cc
11	Samra3	420	0.5	0.109			gas			cc
12	IPP3	15	0.5	11.86			gas			de
13	IPP4	15	0.5	10.16			gas			de
14	Samra4	210	0.5	0.109			gas			cc
15	Zarqa-ACWA	485	0.5	0.198			gas			cc
16										

conventional | renewable | profiles | demand | storage | carrier

scenarios/example1/conf.toml:

config.toml - Notepad

File Edit Format View Help

```
[constraints]
# binary constraints for minimal load of of units
# NOTE: you need to set `p_min` for the units
pmin = false
# efficiency is based on function with partial load losses
# requires p_min = true AND a,b coefficients for part load efficiecy modelling
eta_partial = false
```

example2 adds a more in detail representation of the power plants with introducing minimum load constraints. Observe the differences in the input.xlsx and conf.toml files as compared to example1.

scenarios/example2/input/input.xlsx:

	A	B	C	D	E	F	G	H	I	J
1	unit	p_nom	eta	vom	p_max	p_min	carrier	a	b	tech
2	Aqaba1	121	0.5	0.509	1	0.45	gas			st
3	Aqaba2	121	0.5	0.509	1	0.45	gas			st
4	Risha	30	0.5	0.764	1	0.33	gas			gt
5	Rehab	25	0.5	2.43	1	0.40	gas			gt
6	RehabCC	270	0.5	2.43	1	0.78	gas			cc
7	Samra1	308	0.5	0.109	1	0.71	gas			cc
8	Samra2	300	0.5	0.109	1	0.73	gas			cc
9	Amman-East	424	0.5	0.071	1	0.66	gas			cc
10	Qatranah	420	0.5	0.357	1	0.67	gas			cc
11	Samra3	420	0.5	0.109	1	0.81	gas			cc
12	IPP3	15	0.5	11.86	1	1.00	gas			de
13	IPP4	15	0.5	10.16	1	1.00	gas			de
14	Samra4	210	0.5	0.109	1	0.57	gas			cc
15	Zarqa-ACWA	485	0.5	0.198	1	0.50	gas			cc
16										

conventional | renewable | profiles | demand | storage | carrier

scenarios/example2/conf.toml:

config.toml - Notepad

File Edit Format View Help

```
[constraints]
# binary constraints for minimal load of of units
# NOTE: you need to set `p_min` for the units
pmin = true
# efficiency is based on function with partial load losses
# requires p_min = true AND a,b coefficients for part load efficiecy modelling
eta_partial = false
```

example3 adds even more detail with introducing part load efficiencies of the power plants. You can find the description on how the part load efficiencies are modeled in the mathematical model description in the docs folder.

The best way to understand the structure is to open the necessary files yourself. Have a look at the differences between the three scenarios in input.xlsx and config.toml. Run the examples and compare the results (e.g. output/plots/hourly-dispatch.html).

Support

If you need help, you can contact the trained expert in your institution. For contact details refer to:

GIZ Jordan, Thomas Fink, thomas.fink@giz.de

If you encounter any problems, find a bug or have a question, feel free to create an issue at: https://github.com/znes/enwajo/issues .