**Reporting and postprocessing in MESSAGEix**

Reporting is the process of looking into the results, after solving a scenario in GAMS. There are different ways to look into the output of a MESSAGEix scenario. Using each method depends on the objective and the extent of details to be examined.

**1. Quick look into the results (through ixmp)**

One can use ixmp built-in functions to see the results of a scenario. In this case, you need to open a python session and load the scenario that has a solution to check and visualize some output results. Some useful hints:

***1.1. Load a scenario and check if it has solution.***

Loading platform and your desirable scenario:

import ixmp

import message\_ix

mp = ixmp.Platform()

scen = message\_ix.Scenario(‘your\_model’, ‘your\_scenario’)

Check if solution exists. If not, you won’t be able to do the reporting.

scen.has\_solution()

***1.2. Checking the output of one or a group of technologies***

The output of a scenario can be seen through calling scenario.var() function.

For example, activity of one technology in all regions:

scen.var(‘ACT’, {‘technology’: ‘solar\_pv\_ppl’})

For example, activity of one technology in one region and one year:

scen.var(‘ACT’, {‘technology’: ‘solar\_pv\_ppl’, ‘node\_loc’: ‘Westeros’, ‘year\_act’: 2020})

This can be extended to see the activity, capacity, emissions, etc. of different technologies in different years and regions.

**2. MESSAGEix reporting feature (new reporting)**

MESSAGEix has a sophisticated and structured way of reporting for pulling the output and processing that. This feature is well known in the group as “new reporting”. This is based on a set of python functionalities that are represented as a standard software package, included when installing MESSAGEix. New reporting can be used for any MESSAGEix scenario, i.e., global, national, Westeros, etc.

For more information, please refer to the documentation:

<https://docs.messageix.org/en/stable/reporting.html>

There is also one tutorial showing how to work with the MESSAGEix reporting tool:

<https://github.com/iiasa/message_ix/blob/master/tutorial/westeros/westeros_report.ipynb>

**3. Old reporting of the MESSAGEix Global model**

There is a set of python scripts that are used internally in the group to report the output of the global model. This package, which is known in the group as “old reporting” has been developed over time by Oliver, and it has the following features:

- it generates a comprehensive output report of a global MESSAGEix scenario. The output file is a large Excel file.

- it is not flexible to be used for different models (as it has a lot of hard-coded data related to our global model)

- it has not been widely tested so far. So, some issues are observed in difficult parts of the reporting, e.g., in fuel blending, CCS capacities, etc.

- it is not version controlled, and it’s not optimized for performance. It takes 10 min to run the whole reporting for a global model.

- it can be customized to only report specific parts.

- it is not possible to report the model output at sub-annual time slices with the current format of the package.

**3.1. How to get the old reporting package?**

Old reporting is a set of python, text, and Excel files that are saved under IIASA’s message\_data repository. This repo is internal and not public access is available. Please follow the steps below to be able to run the old reporting.

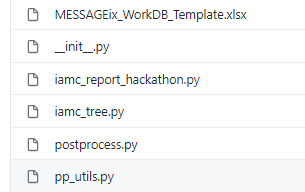
a. Fork the following branch:

<https://github.com/behnam-zakeri/message_data/tree/reporting>

b. After forking and cloning the repo, a number of folders and files will be transferred to your machine. You can find the reporting files under this path (on your machine):

https://.../message\_data /message\_data/tools/post\_processing

You should be able to see these files:



c. The main reporting function is *iamc\_report\_hackathon.py* which is interlinked with other files and modules in this folder.

**3.2. How to use the old reporting package?**

The old reporting can be run either from a command-line or from another python script. For running from another python script please see the example in the file *run\_reporting\_old.py.* The procedure is as follows:

1. **Output folder**: Define a path in your machine, that you want to save the output of reporting. The output of reporting is a large Excel file, so you can decide where you want to save these files in your machine. This folder for the Excel files does not need to be in the same message\_data repository path above. So, you can make an “output” folder for example like:

C:/.../my documents /output

2. Then, run the reporting interface (*run\_reporting\_old.py*), which will generate the output Excel file, saved in your output folder. There are some instructions in the run\_reporting\_old file as well.