IIASA

ENGAGE
FEASIBILITY OF
CLIMATE PATHWAYS

The Python package *pyam* for analysis, validation & visualization of integrated-assessment and energy-systems scenarios

License Apache 2.0 | python 3.7 \| 3.8 \| 3.9 | chat Slack | mail groups.io

code style black | pytest passing | docs passing | codecov 95%

DOI 10.5281/zenodo.1470400 | ORE 10.12688/openreseurope.13633.2

Repository hosted on

GitHub

Community supported by

Groups.io slack

Documentation hosted by

Read the Docs

Daniel Huppmann, Philip Hackstock

ENGAGE Capacity Building Workshop on Good Practices in Modeling
February 1, 2022

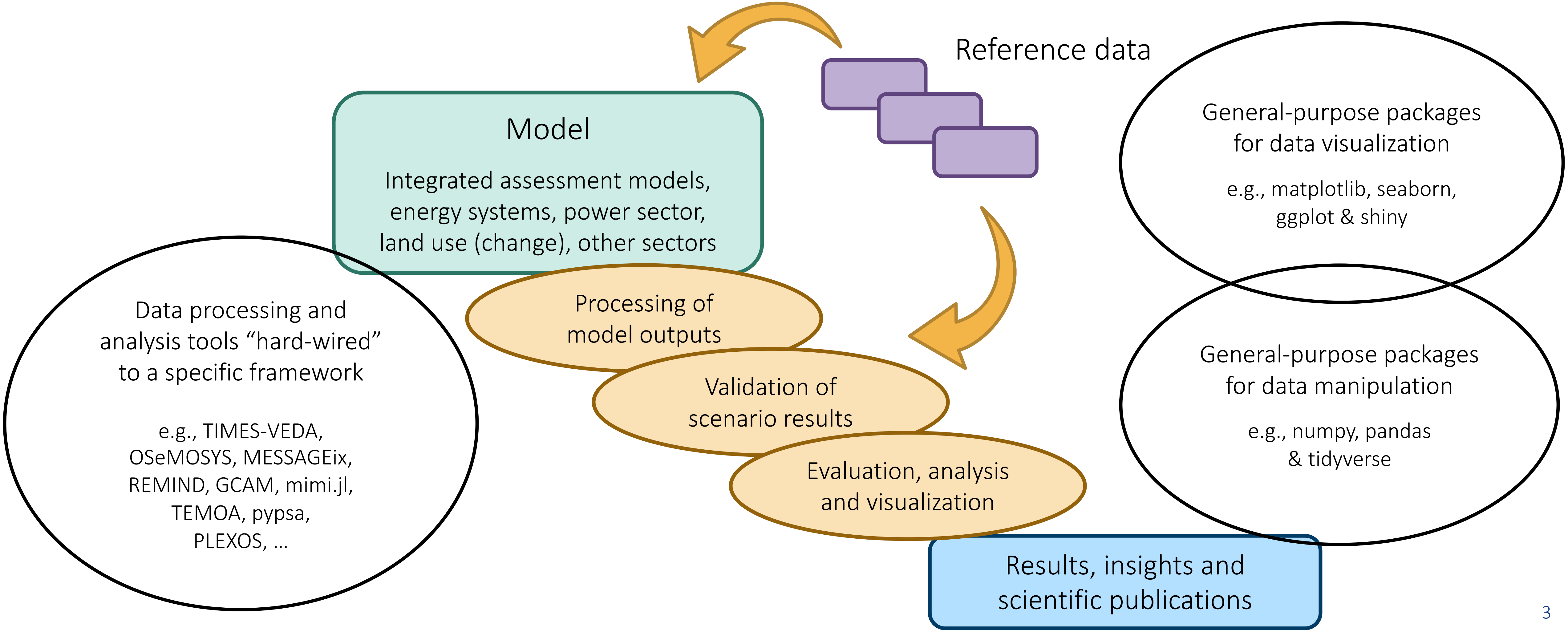
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 821471 (ENGAGE).

Part 1

Introduction & motivation

Introduction: From model results to scenario analysis

There are many solutions and tools for scenario analysis & data visualization, but most are tools either "hard-wired" to a modelling framework or general purpose



Reference data

Model

Integrated assessment models, energy systems, power sector, land use (change), other sectors

Processing of model outputs

Validation of scenario results

Evaluation, analysis and visualization

Results, insights and scientific publications

Data processing and analysis tools "hard-wired" to a specific framework

e.g., TIMES-VEDA, OSeMOSYS, MESSAGEix, REMIND, GCAM, mimi.jl, TEMOA, pypsa, PLEXOS, …

General-purpose packages for data visualization

e.g., matplotlib, seaborn, ggplot & shiny

General-purpose packages for data manipulation

e.g., numpy, pandas & tidyverse

Better-practices for scripts for scenario analysis and data visualization

Many modelling frameworks adopt “best-practice of collaborative development“, but scripts for scenario analysis are often written in an ad-hoc fashion

- A common approach to scenario analysis & data visualization
  - ⇒ Write a few lines of code for a simple feature – a few features – and a little bit more …
- Caveats of this incremental approach (not always, but way too often)
  - ⇒ *copy-paste* of large snippets of code from one project to the next
  - ⇒ No version management for the analysis scripts
  - ⇒ Insufficient documentation of code
  - ⇒ No testing, no *continuous-integration*-strategy
- Why is this a problem for open & reproducible science?
  - ⇒ Limited reproducibility or transparency of the results
  - ⇒ Risk of errors or bugs in existing features during further development
  - ⇒ Risk of errors or bugs due to dependency updates

Our vision: a community Python toolbox for energy & climate research

The pyam package offers a suite of model-independent methods to streamline the processing, analysis & visualization of scenario results

- Design principles:
  - ⇒ Harmonized data structure and formats
  - ⇒ Model-independent standardized methods for scenario analysis & visualization
  - ⇒ Modular package architecture and simple integration in other packages & workflows

- Advantages for modellers and analysts
  - ⇒ Standardized interface following the ***pandas*** & ***matplotlib*** packages
  - ⇒ Comprehensive documentation, tutorials, email list, Slack workspace, …
  - ⇒ High-performance implementation as `pandas.Series` statt `pandas.DataFrame`
  - ⇒ Increased transparency & better intelligibility through shorter analysis scripts
  - ⇒ Higher reliability thanks to a well-testing package with a *continuous-integration*-strategy

Part 2

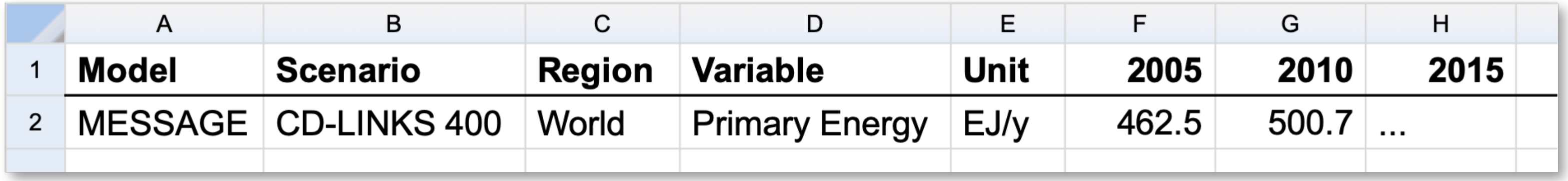
The pyam package

Supported data models and file formats

The package supports various formats & types of timeseries data and is currently used by more than a dozen modelling teams

Supported timeseries data formats:

The *pyam* package was initially developed to work with the *IAMC template,* a tabular format for yearly timeseries data



	A	B	C	D	E	F	G	H
1	**Model**	**Scenario**	**Region**	**Variable**	**Unit**	**2005**	**2010**	**2015**
2	MESSAGE	CD-LINKS 400	World	Primary Energy	EJ/y	462.5	500.7	...

But the package also supports sub-annual time resolution

⇒ Continuous-time formats (e.g., hourly timeseries data)

⇒ Representative sub-annual timeslices (e.g., “winter-night”)

Compatible i/o and file formats:

⇒ Full integration with the *pandas* data analysis package

⇒ Tabular data (`xlsx`, `csv`) & “frictionless” datapackage format

The *pyam* package for integrated assessment & macro-energy modelling

A community package for scenario processing, analysis & visualization following best practice of collaborative scientific software development

python™

Use cases and features

⇒ Data processing Data i/o & file format conversion, aggregation, downscaling, unit conversion, …

⇒ Validation Checks for completeness of data, internal/external consistency, numerical plausibility …

⇒ Analysis & visualization Categorization and statistics of scenario ensembles, plotting library, …

D. Huppmann, M. Gidden, et al. (2021). *Open Research Europe* **1**:74. doi: 10.12688/openreseurope.13633.2

License | Apache 2.0 python | 3.7 | 3.8 | 3.9 chat | Slack mail | groups.io

code style | black pytest | passing docs | passing codecov | 95%

DOI | 10.5281/zenodo.1470400 ORE | 10.12688/openreseurope.13633.2

Repository hosted on GitHub

Community supported by Groups.io slack

Documentation hosted by Read the Docs

#pyam_iamc

pyam-iamc.readthedocs.io

Developing a community for a community package

We made an effort to make the pyam package usable for modellers & analysts with a wide range of experience levels and scientific backgrounds

- Simple installation

  ⇒ Available via the common Python managers *pypi* and *conda*

- Open-access manuscript & comprehensive documentation

  ⇒ Several tutorials and full-fledged API documentation

- For novice users or moderate-interest users:

  ⇒ An email list for announcements of new releases and questions

- For users interested in frequent updates, tips-and-tricks and more interaction

  ⇒ A Slack workspace with a **#helpdesk** channel

- For expert users and anyone interested in contributing

  ⇒ The GitHub repo for collaborative scientific software development like issues and pull requests, continuous-integration workflows, release management, etc.

Part 3

A live demo & interactive discussion

See the open-source notebook at
https://github.com/danielhuppmann/ENGAGE-pyam-tutorial

Thank you very much for your attention!

Dr. Daniel Huppmann

Research Scholar – Energy, Climate, and Environment Program (ECE)

International Institute for Applied Systems Analysis (IIASA)
Schlossplatz 1, A-2361 Laxenburg, Austria

huppmann@iiasa.ac.at

@daniel_huppmann

www.iiasa.ac.at/staff/huppmann