WandaToolbox

Release 0.1.0

Michiel Tukker, Sam van der Zwan

Jul 12, 2022

CONTENTS

CHAPTER

ONE

WANDA TOOLBOX

Toolbox (python scripts) for Wanda modellers. This toolbox includes several tools and utilities that can help with Wanda modelling, running simulations and analyzing and visualizing simulation results. This module is closely related to PyWanda, the Python bindings for the Wanda API.

Wanda is an advanced water hammer software, designed for engineers by engineers. The Python API bindings can be used to create Wanda models, run simulations, and extract the simulation results. The Wanda Toolbox module includes additional scripts to go beyond the basic data retrieval of pywanda.

1.1 Installation

Run the following to install this package:

```
pip install WandaToolbox
```

Also see: https://pypi.org/project/wandatoolbox/

1.2 Usage

Tutorial and reference documentation is provided at wandatoolbox.readthedocs.io. A PDF version of the manual is available here. And the source code is always available at https://github.com/MichielTukker/WandaToolbox.

1.3 Support

No official support! For questions/improvements/comments, contact Deltares or Wanda support desk?

CHAPTER

TWO

CONTENTS

2.1 Changelog

2.1.1 Changelog

All notable changes to this project will be documented in this file.

The format is based on Keep a Changelog, and this project adheres to Semantic Versioning.

[## [Unreleased]] - 2021-9-20

Added

- Added Dashboard example @samvanderzwan.
- Added test for wandaplot_table()
- sphinx documentation
- Documentation pages

Changed

- Set github actions for publishing packages automatically
- changed unit-test to use mocked pywanda objects
- modified github workflow
- updated some required packages

[0.0.4] - 2020-03-11

Added

- Added unit tests, fixing test names.
- Added automatic builds via github actions.

Changed

- fixed linting errors.

[0.0.3] - 2020-03-08

Added

- Added unit tests, fixing test names.
- Added automatic builds via github actions.

Changed

- fixed linting errors.

2.2 Contributing

We'd love to accept your patches and contributions to this project. There are just a few small guidelines you need to follow.

2.2.1 Guidelines

1. Write your patch
2. Add a test case to your patch
3. Make sure that all tests run properly
4. Send your patch as a PR

2.2.2 Installation

You can install the WandaToolbox package as follows:

```
pip install wandatoolbox
```

Note that not all dependencies are installed by default, check requirements.txt

2.2.3 Development installation and testing

For development you can use the 'editable' installation:

```
pip install -r requirements.txt
pip install -e .
pytest
```

2.2.4 Building the package

Building the backage is done as follows:

```
pip install setuptools wheel
python setup.py sdist bdist_wheel
```

2.2.5 Creating a release

We use bump2version to update the version numbers. Bump2version will also create a tag and commit the changes. This can then be pushed using

```
cd <root of project>
pip install bump2version
bump2version <major|minor|patch>
git push origin <branch> --tags
```

The Github Actions have been configured to run the tests and flake8 linting and publish to test.Pypi on every push. If a commit is also tagged it will also publish to Pypi.

2.2.6 Development roadmap (Todo)

functionality for first official release:

- [x] syschar plot implementation
- [x] image plot implementation
- [x] text plot implementation
- [x] table plot implementation
- [x] Monte-carlo scripts
- [ ] unit testing
- [ ] tox testing
- [ ] Goal seek algorithm
- [ ] parameter script read input, run,read output
- [ ] calibration routine (goalseek?) for scalar, multiple scalar, timeseries
- [ ] calibration for multiple parameters?
- [ ] Simple optimization (1 parameter, 1 output?)
- [ ] Valve characteristic creation and data import in Wanda
- [ ] Pump characteristic creation and data import in Wanda
- [ ] Epanet scripts and skeletonizer functions
- [ ] Any other future developments ??

2.3 Plotting figures

2.3.1 Usage

WandaToolbox supports plotting various objects to a formatted PDF, which can be included as an appendix in your report. WandaToolbox.wanda_plot supports time and location series where data is exported directly from Wanda models. It also supports adding tables, images or text blocks on pages, and supports generating system characteristics for a range of flow rates and discharge points.

Code example:

```
from wandatoolbox.wanda_plot import PlotSyschar, PlotText, PlotTable, PlotImage, plot
import matplotlib.pyplot as plt
from matplotlib.backends.backend_pdf import PdfPages
import pandas as pd
import pywanda as pw

model = pw.WandaModel(r'c:\Wandamodel.wdi', 'c:\Wanda 4.6\Bin\\')
img = plt.imread('WandaToolbox\data\DELTARES_ENABLING_CMYK.png')
df = pd.read_excel(r'example_data\syschar_test.xlsx', header=0, index_col=0)
scenario_names = ["Current min", "Current max", "Future min", "Future max"]

with PdfPages(f'Document.pdf') as pdf:
    subplots_table = [
        PlotTable(df, ['description', "Current min", "Current max", "Future min",
↪"Future max"]),
        PlotImage(img), PlotText("Yada yada yada"),
        PlotSyschar("BOUNDQ B1", 105.0, "Supplier #1", df, 'Wanda_name',
        scenario_names, 3, "Industry description", 'Discharge (m3/day)', 'Head (m)')
    ]
    plot(model, subplots_table,
        'Main title',
        f'Subtitle 1',
        'Subtitle 2',
        'Subtitle 3',
        'Subtitle 4',
        f'Figure number: 1',
        company_image=plt.imread('WandaToolbox\data\DELTARES_ENABLING_CMYK.png'),
        fontsize=10)
    pdf.savefig()
    plt.close()
```

2.4 Parameterscript

2.5 Misc. functions and tools

2.6 analysis tools

2.6.1 Monte-carlo analysis

Generic usage of the monte-carlo class:

```
from wandatoolbox.analysis.monte_carlo import MonteCarloInputProperty, MonteCarloOutputProperty, WandaMonteCarlo
import pywanda as pw
import os

def main():
    wandacase_fullpath = os.path.join(os.getcwd(), "Sewage_transient.wdi")
    wanda_bin_directory = r'c:\Program Files (x86)\Deltares\Wanda 4.6\Bin\\'
    model = pw.WandaModel(wandacase_fullpath, wanda_bin_directory)
    parameters = [MonteCarloInputProperty(" PIPES", "Wall roughness", 2.5 / 1000, 0.5 / 1000, "normal", True)]
    outputs = [MonteCarloOutputProperty(" PIPES", "Pressure", keyword=True, extreme="MIN"),
               MonteCarloOutputProperty(" PIPES", "Pressure", keyword=True, extreme="MAX")]
    analysis = WandaMonteCarlo(model, parameters, outputs, nruns=25, n_workers=2)
    analysis.run()
    analysis.plot_results(filename_prefix="test", width=1000, height=800)
    analysis.cleanup()


if __name__ == "__main__":
    main()  # This main() method is essential due to the way Python's multiprocessing module works
```

The monte carlo toolset will run multiple simulation in parallel, greatly reducing the simulation time.

2.7 Callibration

2.8 Optimization

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search