plan4res installation

version 1.0 may 6, 2024

1 Quick Install in a nutshell

Windows specific

See **https://gitlab.com/cerl/plan4res/p4r-env#windows**.

Requirements: Windows 7 Pro 64bit SP1 or higher and PowerShell 3.0 or higher / CPU must support hardware virtualization. (virtualization features may need to be enabled in the BIOS)

- Install Git for Windows (use default settings) https://git-for-windows.github.io/
- Install VirtualBox and Extension Pack https://www.virtualbox.org/wiki/Downloads
- Install Vagrant https://www.vagrantup.com/downloads.html
- (Optional) Install Vagrant Manager http://vagrantmanager.com/downloads/
- Run *Git Bash*.
- (if behind proxy): export http_proxy = <proxy address>:<port>
  export https_proxy = ${http_proxy}
  vagrant plugin install vagrant-proxyconf
- vagrant up # Start the Virtual Machine. First execution requires downloading the image..

For all operating systems

Create empty directory (for example *install_dir)* where you want to install plan4res. It should not contain special characters or whitespaces!

We recommend using CPLEX as the solver. You must have a CPLEX linux installer available, such as cplex_studio2211.linux_x86_64.bin, referenced as *cplex_studioXXXXXX.bin* below.

1. `cd install_dir`
2. `git config submodule.recurse true`
3. `git clone https://github.com/openENTRANCE/plan4res.git`
4. `git clone --recursive https://gitlab.com/cerl/plan4res/p4r-env.git`
1. `cd p4r-env`
2. `bin/p4r` *# downloads the image and executes the environment*
3. `exit`
4. `bin/p4r add-on stopt`

Important note for Windows: *install_dir\p4r-env\scripts\add-ons* must be edited by replacing the 3 instances of `make -j$(getconf _NPROCESSORS_ONLN)` with `make -j1`.

5. `bin/p4r add-on sms++ getdev`
6. `bin/p4r add-on sms++ compile CPLEX=cplex_studioxxx.bin`
7. `bin/p4r add-on sms++ install`
8. `cd scripts`
9. `mkdir python`
10. `cd python`
11. `git clone https://github.com/openENTRANCE/openentrance.git`
12. `git clone https://github.com/openENTRANCE/plan4res-scripts.git`
13. `cd ../../..`
14. `mv plan4res/Launch* p4r-env/scripts/`
15. `mv plan4res/BSPar-Investment.txt  p4r-env/config/`
16. `mv plan4res/sddp_greedy_investment.txt p4r-env/config/`
17. `mv plan4res/sddp_solver.txt p4r-env/config/`
18. `mv plan4res/uc_solverconfig.txt p4r-env/config/`
19. `mv plan4res/sddp_greedy.txt p4r-env/config/`

You are now ready to run the model!

Chapter 2 provides an in-depth description of the plan4res environment (p4r-env). Chapter 3 provides the procedure for installing p4r-env and chapter 4 provides the procedure for installing the necessary software within p4r-env.

2 Description of the plan4res environment

The installation procedure is laid out in section 3Install p4r-env.

The p4r-env environment is a container (see CERL / Plan4Res / p4r-env · GitLab) that allows for a simple installation and use of all plan4res solvers and tools. Note that all components of plan4res can also run without this environment but it requires to install all dependencies manually, which can prove time-consuming and tricky.

The p4r-env environment can be installed on Linux, Windows, Mac and even CRAY machines. Once the environment is installed, the functional behavior will be the same on all kinds of systems. Computation time will depend on the computer's performance.
p4r-env is a singularity container, which means that it requires singularity to be installed.

Instructions to install are summarized in section 4 but you may look at detailed instructions here https://gitlab.com/cerl/plan4res/p4r-env/-/tree/master#p4r-environment-installation-execute-once
Windows has some specific prerequisites, summarized at the beginning of section 1, which are detailed here https://gitlab.com/cerl/plan4res/p4r-env/-/tree/master#prerequisite

The other software installed in the environment to run plan4res are:

- **StOpt** (see Stochastic Control / StOpt · GitLab): a stochastic optimization library used for solving the Seasonal Storage Valuation (SSV) problem. See installation procedure in section 4.1.
- **SMS++** (see SMS++ / The SMS++ Project · GitLab): modelling and optimization library including the 3 main solvers of plan4res in the OM4A toolbox. See installation procedure in section 4.2.
- (Optional) **A preprocessing tool** `p4r-env/scripts/python/plan4res-scripts/CreateInputPlan4res.py`: this tool allows to create plan4res dataset out of long-term pluriannual energy system scenarios (eg from GENeSYS-MOD).
- **A formatting tool**, `p4r-env/scripts/python/plan4res-scripts/format.py`, (mandatory apart if you are creating the NetCDF input data files in the format required by SMS++ on your own); this tool creates input data file with the format required by SMS++ taking as inputs user friendly CSV and XLSX files.
- (Optional) **Postprocessing tool**: this tool transforms the files created by the solver (SMS++) into more user-friendly files and creates some synthetic results.
- (Optional) **IAMC format transformation**: this tool transforms the results from the postprocessing tool into files within the IAMC format, which can then be uploaded to the Scenario Explorer hosted at https://data.ece.iiasa.ac.at/openmod4africa.
- (Optional) **Visualization tool**: creates graphs. For OpenMod4Africa, the toolbox will include specific visualization tools that will work on the model output in the IAMC format, so you may skip this step.

See [D4.1][1] for more details on the plan4res model.

p4r-env structure

At the end of the installation process, the p4r-env repository contains the following sub-repositories:

- `bin/`: contains all software executables. Added to the user PATH environment variable.
- `config/`: contains configuration files:
  - p4r-env configuration files
  - sms++ configuration files
- `data/`: this repository will store all data and results. This repository contains 4 sub-repositories:
  - `cache/`: for data cached from external sources.
  - `local/`: for storing user's data and case study specific settings and config files (SMS++ configuration files and settings for python scripts).
  - `scratch/`: used as working directory of tools and can be used for volatile data during program runs.
  - `staging/`: for storing data that need to be uploaded to external repositories.
- `scripts/`: contains the main workflow scripts (Launch*), which are executing different tasks repeatedly or sequentially, as well as the following sub-repositories:
  - `scripts/add-ons/` contains the scripts for installing stopt and sms++.
  - `scripts/python/` contains all python codes:
    - `scripts/python/plan4res-scripts/` contains the preprocessing, formatting, postprocessing python scripts as well as their configuration files settings*.yaml.
    - `scripts/python/openentrance/` contains the nomenclature python code.
- `site/`: used by p4r-env users to install their own local software

For conducting a case study, a user will have to create a repository for the case study within data/local (eg MyCaseStudy). All python configuration files (settings*.yaml from `scripts/python/plan4res-scripts/` as well as all SMS++ configuration files from `config/` shall be copied by the user in `data/local/MyCaseStudy` as those configuration files may be case study dependent.

Optional: using p4r-env shell

To launch the plan4res environment, go in the `p4r-env` repository and type `bin/p4r` (on Windows, use Git Bash to do so). This will start a shell in the plan4res containerized environment such as:

```
[P4R-ENV] /home/Plan4Res/P4Rtest/p4r-env >
```

In classical uses of plan4res, you will not need to work from the shell as scripts for running all different workflows are available which include launching the environment and all software.

[1] openMod4Africa *D4.1 Catalogue of models and their functionalities* for a description of the software and models.

3 Install p4r-env

3.1 Windows install

The procedure is available at **https://gitlab.com/cerl/plan4res/p4r-env#windows**. It is reproduced below with some more information.

Installation requires Windows 7 Pro 64bit SP1 or higher and PowerShell 3.0 or higher. Furthermore, the CPU must support hardware virtualization. On many systems, the hardware virtualization features first need to be enabled in the BIOS.

3.1.1 Required packages Installation (execute once)

- Install Git for Windows (use default settings) https://git-for-windows.github.io/
- Install VirtualBox and Extension Pack https://www.virtualbox.org/wiki/Downloads
- Install Vagrant https://www.vagrantup.com/downloads.html
- (Optional) Install Vagrant Manager http://vagrantmanager.com/downloads/

The goal of Vagrant and VirtualBox is to emulate a UNIX system on the Windows computer. Vagrant Manager provides a GUI to facilitate the management of the VM.

3.1.2 Installation

- `Run Git Bash.`

If working behind a proxy, define the following environment variables:

- `export http_proxy = <proxy address>:<port>`
- `export https_proxy = ${http_proxy}`

Then enter the following commands:

- `git clone --recursive` https://gitlab.com/cerl/plan4res/p4r-env
- `cd p4r-env`
- `git config submodule.recurse true`
- `vagrant plugin install vagrant-proxyconf`
- `vagrant up` # Starts the Virtual Machine. First execution requires downloading the image. See NOTE below for more details.

Proceed with the generic installation procedure below.

3.2 Linux install

1. Create a directory (eg *`install_dir`*)
2. Download p4r-env:
   `git clone --recursive` https://gitlab.com/cerl/plan4res/p4r-env.git
   This will create the directory *`install_dir /p4r-env`*
3. `cd p4r-env`
4. `bin/p4r`
   This will download the container

3.3 Updating the environment

To update the environment, use the following commands:

1. `cd p4r-env`
2. `bin/p4r`

⚠ bin/p4r downloads an updated Singularity Image File (SIF) , whose size is very big which can take a long time.

In case you wish to avoid updating (and thus downloading a very big file) each time you run p4r-env, you can edit the config file: p4r-env/config/plan4res.conf by replacing `P4R_SINGULARITY_IMAGE_PRESERVE=0` with `P4R_SINGULARITY_IMAGE_PRESERVE=1` (or uncommenting line `P4R_SINGULARITY_IMAGE_PRESERVE=1`).

4 Install software in p4r-env

In order to run plan4res you need to install:

- StOpt
- SMS++ which will create the 3 executables for the CEM, SSV and UC models
- The python scripts:
    - Formatting tool
    - Preprocessing tool
    - Postprocessing tool
    - Visualization tool
    - IAMC writing tool

4.1 StOpt

StOpt must be installed before SMS++

1. If you had previously installed (even if it failed), remove StOpt and sms++:
   ```
   bin/p4r add-on stopt uninstall
   bin/p4r add-on sms++ unistall
   ```
2. Install StOpt:
   ```
   bin/p4r add-on stopt
   ```

4.2 SMS++

SMS++ uses its own solving libraries as well as external libraries for solving linear problems. It can use both CPLEX, GUROBI, SCIP, HIGHS, and COIN-OR.

Instructions for installation of these software can be found here https://gitlab.com/smspp/smspp-project/-/wikis/Installing-SMS++#sms

As an example, in case you are using CPLEX for solving optimization problems, you need to copy your CPLEX installer file (cplex_studioxxx.bin) in the p4r_env directory and ensure your CPLEX license is valid.

3. We recommend using the development version of SMS++:
   ```
   bin/p4r add-on sms++ getdev
   bin/p4r add-on sms++ compile CPLEX=cplex_studioxxx.bin
   ```
   (if using cplex)
   ```
   bin/p4r add-on sms++ install
   ```

See SMS++ / The SMS++ Project · GitLab for more information.

4.3 Python scripts

The python scripts need to be installed by the user in `p4r-env/scripts/python` (after creation of this repository)

```
cd p4r-env/scripts
mkdir python
cd python
```

- **openENTRANCE nomenclature definitions:**

  This python package contains definitions of variables and regions, it is used by the other python scripts.

  ```
  git clone https://github.com/openENTRANCE/openentrance.git
  ```

- **Python scripts for preprocessing, postprocessing, formatting, visualization and IAMC writing**

  ```
  git clone https://github.com/openENTRANCE/plan4res-scripts.git
  ```

  This command will create the repository `/scripts/python/plan4res-scripts` which contains python scripts as well as their settings files: settings*.yaml, which should not be changed in this repository but will serve as templates. Those settings files settings*.yaml should be copied to each case study specific repository, eg. `/data/local/MyCaseStudy/settings/`.

4.4 Get documentation, configuration files and launching scripts

- **General workflow scripts**

  These are the scripts Launch*, which are part of the https://github.com/openENTRANCE/plan4res.git repository. These scripts are meant to be in `p4r-env/scripts`

  - `LaunchSSV` proceeds to the following steps:
    - Create netcdf files
    - Run sddp_solver (computes Bellman values for seasonal storages)
  - `LaunchSIM` proceeds to the following steps:
    - Create netcdf files
    - Run sddp_solver (compute simulations)
    - Post-treat results, write IAMC files and create graphs
  - `LaunchCEM` proceeds to the following steps:
    - Create netcdf files
    - Run investment_solver (compute best investment and run simulations)
    - Post-treat results, write IAMC files and create graphs
  - In case the users wish to conduct the workflow step-by-step, additional scripts are available:
    - `LaunchCREATE_*` creates the csv plan4res dataset
    - `LaunchFORMAT_*` creates the netcdf SMS++ input files
  - Full workflow scripts allow to run the different steps:
    - `LaunchSSVandSIM` proceeds to the following steps:
      - Create plan4res csv input files

        - Create SMS++ netcdf input files for SSV
        - Run sddp_solver (computes Bellman values)
        - Creates SMS++ netcdf input files for simulation
        - Runs sddp_solver in simulation mode
        - Post-treat results, write IAMC and create graph
    - `LaunchSSVandCEM` proceeds to the following steps:
        - Create plan4res csv input files
        - Create SMS++ netcdf input files for SSV
        - Run sddp_solver (computes Bellman values)
        - Creates SMS++ netcdf input files for capacity expansion
        - Runs investment_solver
        - Post-Treat results, write IAMC and create graphs

- **SMS++ configuration files** are also in the https://github.com/openENTRANCE/plan4res.git repository. They need to be moved to `p4r-env/config`. These are are template files that should not be modified: as for settings*.yaml, they should be copied to each case study specific repository, eg. `/data/local/MyCaseStudy/settings/`.