Phydrus Documentation

Release 0.2.0

R.A. Collenteur, G. Brunetti, M. Vremec

Mar 30, 2021

CONTENTS

1 Getting started with Phydrus **3**
- 1.1 Installing and Updating Phydrus . . . 3
- 1.2 Compiling the Fortran Source Code . . . 4
- 1.3 Modeling Approach in Phydrus . . . 5

2 Examples **7**

3 Developers Section **9**

4 API-docs **11**
- 4.1 Model . . . 11
- 4.2 phydrus.read . . . 25
- 4.3 phydrus.profile . . . 28
- 4.4 Plots . . . 30
- 4.5 phydrus.utils . . . 32

5 Release Notes **35**

Python Module Index **37**

Index **39**

This package provides a Python implementation of the HYDRUS-1D unsaturated zone model developed by Šimůnek, J., M. Th. van Genuchten, and M. Šejna. More information on the HYDRUS-1D model is available here. This software is licenced under the GNU GENERAL PUBLIC LICENSE found here. The Phydrus code is developed by R.A. Collenteur, G. Brunetti and M. Vremec. With Phydrus, a HYDRUS-1D model can be created, calibrated and visualized through Python scripts, making it easy to adjust the model and providing a 100% reproducible workflow of your modeling process.

CHAPTER

ONE

GETTING STARTED WITH PHYDRUS

In this section you can find guidance on how to get started with Phydrus. Before being able to run a Phydrus model from Python Console you need to 1) install the Phydrus package, and 2) obtain a compiled version of the Hydrus-1D Fortran code. After you have succesfully completed these two steps, you can start creating your first Phydrus model. The final section will help to get you started by explaining the modeling approach applied in Phydrus.

1.1 Installing and Updating Phydrus

1.1.1 Installing Python

To install Phydrus, a working version of Python 3.7 or higher has to be installed. We recommend using the Anaconda Distribution of Python. This Python distribution includes most of the python package dependencies and the Jupyter Lab software to run the notebooks. Moreover, it includes the Graphical User Interface (GUI) Spyder to start scripting in Python. However, you are free to install any Python distribution you want.

1.1.2 Installing the Phydrus package

The latest stable version of the Phydrus package is available from the Pypi package index.

```
>>> pip install phydrus
```

To install in developer mode, clone the GitHub repository and use the following syntax:

```
>>> pip install -e .
```

1.1.3 Updating the Phydrus package

If you have already installed Phydrus, it is possible to update Phydrus easily. To update, open a Windows command screen or a Mac terminal and type:

```
>>> pip install phydrus --upgrade
```

1.1.4 Dependencies

Phydrus depends on a number of Python packages, which are all automatically installed when using the pip install manager. The following packages are necessary for the installation of Phydrus:

```
matplotlib >= 3.1
pandas >= 1.1
numpy >= 1.16
```

1.2 Compiling the Fortran Source Code

After installing the Phydrus package, you need to obtain an executable for the Hydrus-1D model. There after two options to obtain this executable: **1)** download a pre-compiled version, or **2)** compiled the fortan code locally. Below both approaches are described. Compiling the Fortran source code can be a bit more challenging depending on your environment. Please carefully read the instructions below and read through the dedicated GitHub Discussions category on this topic before opening a new Discussion.

1.2.1 Compiling the source code

The recommended option is to compile the adapted Hydrus-1D Fortran77 files for your own environment and computer. The following steps should be taken:

1. Download the Phydrus-optimized fortran code from this dedicated repository.

2. Open a Windown command line or Linux / MacOS Terminal and move into the *source* folder (e.g., *cd path/to/directory*).

3. and use the following syntax in your terminal or windows command line to compile the source code:

```
>>> make
```

4. This should create a Windows or Unix Executable that can be used to run the HYDRUS-1D simulation. In the Python code, you have to reference to the location of the executable, so you can store it anywhere you want.

Troubleshooting

Depending on your operation system (e.g., Windows/MacOS/Linux) you may need to install additional tools to compile:

- The compile the source code for MacOS/Linux, Gfortran needs to be installed. Instructions can be found here: https://gcc.gnu.org/wiki/GFortranBinaries.
- Please let us know when you find other requirements to add to this list!

1.2.2 Using pre-compiled executables

It is in principle possible to use pre-compiled versions of Hydrus-1D. Note that the Phydrus software has been developed based on the the Hydrus-1D 4.08 Fortran Code and other versions may not be suppported.

If you have the Graphical User Interface for Hydrus-1D for Windows installed (which can be obtained from https://www.pc-progress.com/), you may directly point to that exectuable. You can also download the pre-compiled versions from the source code repository. However, it can not be guaranteed that these executables work for your personal computing environment.

1.3 Modeling Approach in Phydrus

CHAPTER

TWO

EXAMPLES

CHAPTER

THREE

DEVELOPERS SECTION

CHAPTER

FOUR

API-DOCS

This section contains the Documentation of the Application Programming Interface (API) of Phydrus. The information in this section is automatically created from the documentation strings in original Python code. In the left-hand menu you will find the different categories of the API documentation.

model.Model	Basic Phydrus model container.
read	The read module contains methods that can be used to read in- and output files.
profile	The profile module contains utility functions to help create the profile DataFrame.
plot.Plots	Class that contains all the methods to plot a Phydrus Model.
utils	The utils module contains utility funtions for Phydrus.

4.1 Model

class phydrus.model.**Model**(*exe_name*, *ws_name*, *name='model'*, *description=None*, *length_unit='cm'*, *time_unit='days'*, *mass_units='mmol'*, *print_screen=False*)

Basic Phydrus model container.

Parameters

- **exe_name** (*str*) – String with the path to the Hydrus-1D executable.
- **ws_name** (*str*) – String with the workspace name. Folder is created if it does not exist.
- **name** (*str, optional*) – String with the name of the model.
- **description** (*str, optional*) – String with the description of the model.
- **length_unit** (*str, optional*) – length units to use in the simulation. Options are "mm", "cm", and "m". Defaults to "cm".
- **time_unit** (*str, optional*) – time unit to use in the simulation, options are "seconds", "minutes", "hours", "days, "years". Defaults to "days".
- **mass_units** (*str, optional*) – Mass units to use in the simulation, Options are "mmol". Defaults to "mmol". Only used when transport process is added.
- **print_screen** (*bool, optional*) – Print the results to the screen during code execution.

Examples

```
>>> import phydrus as ps
>>> ws = "example"
>>> exe = os.path.join(os.getcwd(), "hydrus")
```

```
>>> ml = ps.Model(exe_name=exe, ws_name=ws, mass_units="mmol",
>>>               time_unit="min", length_unit="cm")
```

4.1.1 Attributes

n_layers
n_materials
n_solutes

4.1.2 Methods

__init__	Initialize self.
add_atmospheric_bc	Method to add the atmospheric boundary condition to the model.
add_drains	Method to add a drain to the model.
add_heat_transport	Method to add heat transport to the model.
add_material	Method to add a material to the model.
add_obs_nodes	Method to add observation points.
add_profile	Method to add the soil profile to the model.
add_root_growth	Method to add root growth to the model.
add_root_uptake	Method to add rootwater update modeule to the model.
add_solute	Method to add a solute to the model.
add_solute_transport	Method to add solute transport to the model.
add_time_info	Method to produce time information.
add_waterflow	Method to add a water_flow module to the model.
get_empty_heat_df	Get an empty DataFrame to fill in the heat parameters.
get_empty_material_df	Get an empty DataFrame with the soil parameters as columns.
get_empty_solute_df	Get an empty DataFrame with the solute parameters as columns.
read_alevel	Method to read the A_LEVEL.OUT output file.
read_balance	Method to read the BALANCE.OUT output file.
read_i_check	Method to read the I_CHECK.OUT output file.
read_nod_inf	Method to read the NOD_INF.OUT output file.
read_obs_node	Method to read the OBS_NODE.OUT output file.
read_profile	Method to read the PROFILE.OUT output file.
read_run_inf	Method to read the RUN_INF.OUT output file.
read_solutes	Method to read the SOLUTE.OUT output file.
read_tlevel	Method to read the T_LEVEL.OUT output file.

continues on next page

Table 3 – continued from previous page

set_executable	Method to set the path to the Hydrus-1D executable.
simulate	Method to call the Hydrus-1D executable.
write_atmosphere	Method to write the ATMOSPH.IN file
write_input	Method to write the input files for the HYDRUS-1D simulation.
write_profile	Method to write the PROFILE.DAT file.
write_selector	Write the SELECTOR.IN file.

phydrus.model.Model.__init__

Model.**__init__**(*exe_name*, *ws_name*, *name='model'*, *description=None*, *length_unit='cm'*, *time_unit='days'*, *mass_units='mmol'*, *print_screen=False*)

Initialize self. See help(type(self)) for accurate signature.

phydrus.model.Model.add_atmospheric_bc

Model.**add_atmospheric_bc**(*atmosphere*, *ldailyvar=False*, *lsinusvar=False*, *llai=False*, *rextinct=0.463*, *hcrits=1e+30*, *tatm=0*, *prec=0*, *rsoil=0*, *rroot=0*, *hcrita=100000.0*, *rb=0*, *hb=0*, *ht=0*, *ttop=0*, *tbot=0*, *ampl=0*)

Method to add the atmospheric boundary condition to the model.

Parameters

- **atmosphere** (*pandas.DataFrame*) – Pandas DataFrame with at least the following columns: tAtm, Prec, rSoil, rRoot, hCritA, rB, hB, hT, tTop, tBot, and Ampl.
- **ldailyvar** (*bool, optional*) – True if HYDRUS-1D is to generate daily variations in evaporation and transpiration. False otherwise.
- **lsinusvar** (*bool, optional*) – True if HYDRUS-1D is to generate sinusoidal variations in precipitation. False otherwise.
- **llai** (*bool, optional*) – Boolean indicating that potential evapotranspiration is to be divided into potential evaporation and potential transpiration using eq. (2.75) of the manual.
- **rextinct** (*float, optional*) – A constant for the radiation extinction by the canopy (rExtinct=0.463) [-]. only used when lLai is True.
- **hcrits** (*float, optional*) – Maximum allowed pressure head at the soil surface [L]. Default is 1e+30.
- **tatm** (*float, optional*) – Time for which the i-th data record is provided [T].
- **prec** (*float, optional*) – Precipitation rate [LT-1] (in absolute value).
- **rsoil** (*float, optional*) – Potential evaporation rate [LT-1] (in absolute value). rSoil(i) is interpreted as KodTop when a time variable Dirichlet or Neumann boundary condition is specified.
- **rroot** (*float, optional*) – Potential transpiration rate [LT-1] (in absolute value).
- **hcrita** (*float, optional*) – Absolute value of the minimum allowed pressure head at the soil surface [L].
- **rb** (*float, optional*) – Bottom flux [LT-1] (set equal to 0 if KodBot is positive, or if one of the logical variables qGWLF, FreeD or SeepF is .true.).

- **hb** (*float, optional*) – Groundwater level [L], or any other prescribed pressure head boundary condition as indicated by a positive value of KodBot (set equal to 0 if KodBot is negative, or if one of the logical variables qGWLF, FreeD or SeepF is .true.).
- **ht** (*float, optional*) – Prescribed pressure head [L] at the surface (set equal to 0 if KodBot is negative).
- **ttop** (*float, optional*) – Soil surface temperature [oC] (omit if both lTemp and lChem are equal to .false.).
- **tbot** (*float, optional*) – Soil temperature at the bottom of the soil profile [oC] (omit if both lTemp and lChem are equal to .false., set equal to zero if kBotT=0).
- **ampl** (*float, optional*) – Temperature amplitude at the soil surface [K] (omit if both lTemp and lChem are equal to .false.).

Notes

The index of the atmosphere DataFrame should be a RangeIndex with integers.

phydrus.model.Model.add_drains

Model.**add_drains**()
Method to add a drain to the model.

phydrus.model.Model.add_heat_transport

Model.**add_heat_transport**(*parameters*, *ampl*, *top_bc*, *top_temp*, *bot_bc*, *bot_temp*, *tperiod=1*, *icampbell=1*, *snowmelt=0.43*)
Method to add heat transport to the model.

Parameters

- **parameters** (*pandas.DataFrame*) – DataFrame describing the heat transport material properties. An empty DataFrame may be obtained using ml.get_empty_heat_df().
- **ampl** (*float, optional*) – Temperature amplitude at the soil surface [K].
- **top_bc** (*int, optional*) – Code which specifies the type of upper boundary condition: 1 = Dirichlet boundary condition, -1 = Cauchy boundary condition.
- **top_temp** (*float, optional*) – Temperature of the upper boundary, or temperature of the incoming fluid [degree Celsius].
- **bot_bc** (*int, optional*) – Code which specifies the type of lower boundary condition: 1 = Dirichlet boundary condition, 0 = continuous temperature profile, zero gradient, -1 = Cauchy boundary condition.
- **bot_temp** (*float, optional*) – Temperature of lower boundary, or temperature of the incoming fluid [degree Celsius].
- **tperiod** (*float, optional*) – Time interval for completion of one temperature cycle (usually 1 day, default) [T].
- **icampbell** (*int, optional*) – Set equal to 1 if Campbell [1985] formula is to be used to calculate the thermal conductivity (Default). Set equal to 0, when Chung and Horton [1987] formula is to be used.

- **snowmelt** (*float, optional*) – Amount of snow that will melt during one day for each degree Celsius (e.g., 0.43cm default).

Notes

This method provides all the information necessary for "Block E - Heat transport information".

See also:

`phydrus.Model.get_empty_heat_df`

phydrus.model.Model.add_material

`Model.`**`add_material`**(*material*)

Method to add a material to the model.

Parameters **material** (*pandas.DataFrame*) – Pandas DataFrame with the parameter names as columns and the values for each material as one row. The index for each is the reference number for each material and must be unique. The number of columns depends on the water flow model that has been chosen.

Examples

```
>>> m = pd.DataFrame({1: [0.08, 0.3421, 0.03, 5, 1, -0.5]}, index=[1],
                     columns=["thr", "ths", "Alfa", "n" "Ks", "l"])
>>> ml.add_material(m)
```

See also:

`phydrus.Model.get_empty_material_df`

phydrus.model.Model.add_obs_nodes

`Model.`**`add_obs_nodes`**(*depths*)

Method to add observation points.

Parameters **depths** (*list of ints*) – List of floats denoting the depth of the nodes. The depth is defined in the same length unit as selected in ml.model function. The function defines the closest node to the desired depth.

phydrus.model.Model.add_profile

`Model.`**`add_profile`**(*profile*)

Method to add the soil profile to the model.

phydrus.model.Model.add_root_growth

Model.**add_root_growth**(*irootin=0*, *ngrowth=None*, *tgrowth=None*, *rootdepth=None*, *irfak=None*, *trmin=None*, *trmed=None*, *trmax=None*, *xrmin=None*, *xrmed=None*, *xrmax=None*, *trperiod=None*)

Method to add root growth to the model.

Parameters

- **irootin** (*int*) – 0 = (default) The root depth is specified together with other time-variable boundary condition, such as atmospheric fluxes. 1 = the root depth is given in a table 2 = the root depth is calculated using the growth function.
- **ngrowth** (*int*) – Number of data points in the root depth table. Only used when irootin = 1.
- **tgrowth** (*float, optional*) – Days. Only used when irootin = 1.
- **rootdepth** (*list of float, optional*) – Rooting depth [L]. List has a length of ngrowth. Only used when irootin = 1.
- **irfak** (*int, optional*) – Method to calculate the root growth factor, r. Only used when irootin = 2. 0= the root growth factor is calculated from given data [xRMed, tRMed]. 1 = the root growth factor is calculated based on the assumption that 50% of the rooting depth, (xRMax+xRMin)/2., is reached at the midpoint of the growing season, (tRMin+tRHarv)/2.
- **trmin** (*float, optional*) – Initial time of the root growth period [T]. Only used when irootin = 2.
- **trmed** (*float, optional*) – Time of known rooting depth (set equal to zero if iRFak=1) [T]. Only used when irootin = 2.
- **trmax** (*float, optional*) – Time at the end of the root water uptake period [T]. Only used when irootin = 2.
- **xrmin** (*float, optional*) – Initial value of the rooting depth at the beginning of the growth period (recommended value = 1 cm) [L]. Only used when irootin = 2.
- **xrmed** (*float, optional*) – Value of known rooting depth (set equal to zero if iRFak=1) [L]. Only used when irootin = 2.
- **xrmax** (*float, optional*) – Maximum rooting depth, which may be reached at infinite time [L]. Only used when irootin = 2.
- **trperiod** (*float, optional*) – Time period at which the growth function repeats itself. Only used when irootin = 2.

phydrus.model.Model.add_root_uptake

Model.**add_root_uptake**(*model=0*, *crootmax=0*, *omegac=0.5*, *p0=- 10*, *p2h=- 200*, *p2l=- 800*, *p3=- 8000*, *r2h=0.5*, *r2l=0.1*, *poptm=None*, *p50=- 800*, *pexp=3*, *lsolred=False*)

Method to add rootwater update modeule to the model.

Parameters

- **model** (*int, optional*) – Type of root water uptake stress response function. 0 = Feddes et al. [1978]. 1 = S-shaped, van Genuchten [1987]
- **crootmax** (*float, optional*) – Maximum allowed concentration in the root solute uptake term for the first solute [ML-3]. When the nodal concentration is lower than cRootMax, all solute is taken up. When the nodal concentration is higher than cRootMax, additional solute stays behind.

- **omegac** (*float, optional*) – Maximum allowed concentration in the root solute uptake term for the last solute [ML-3].
- **p0** (*float, optional*) – Only used if model=0. Value of the pressure head, h1, below which roots start to extract water from the soil.
- **p2h** (*float, optional*) – Only used if model=0. Value of the limiting pressure head, h3, below which the roots cannot extract water at the maximum rate (assuming a potential transpiration rate of r2H).
- **p2l** (*float, optional*) – Only used if model=0. As above, but for a potential transpiration rate of r2L.
- **p3** (*float, optional*) – Only used if model=0. Value of the pressure head, h4, below which root water uptake ceases (usually equal to the wilting point).
- **r2h** (*float, optional*) – Only used if model=0. Potential transpiration rate [LT-1] (currently set at 0.5 cm/day).
- **r2l** (*float, optional*) – Only used if model=0. Potential transpiration rate [LT-1] (currently set at 0.1 cm/day).
- **poptm** (*list, optional*) – Value of the pressure head, h2, below which roots start to extract water at the maximum possible rate. The length of poptm should equal the No. of materials.
- **p50** (*float, optioal*) – Only used if model=1. Value of the pressure head, h50, at which the root water uptake is reduced by 50%. Default is -800cm.
- **pexp** (*float, optional*) – Only used if model=1. Exponent, p, in the S-shaped root water uptake stress response function. Default value is 3.
- **lsolred** (*bool, optional*) – Set to True if root water uptake is reduced due to salinity. NOT IMPLEMENTED YET.

phydrus.model.Model.add_solute

Model.**add_solute**(*data*, *difw=0*, *difg=0*, *top_conc=0*, *bot_conc=0*)

Method to add a solute to the model.

Parameters

- **data** (*pandas.DataFrame*) – Pandas DataFrame with the material properties. Get an empty DataFrame for this ml.get_empty_solute_df().
- **difw** (*float, optional*) – Ionic or molecular diffusion coefficient in free water, Dw [L2T-1].
- **difg** (*float, optional*) – Ionic or molecular diffusion coefficient in gas phase, Dg [L2T-1].
- **top_conc** (*float, optional*) – Concentration of the upper boundary, or concentration of the incoming fluid [ML-3].
- **bot_conc** (*float, optional*) – Concentration of the lower boundary, or concentration of the incoming fluid [ML-3].

See also:

phydrus.Model.get_empty_solute_df, phydrus.Model.add_solute_transport

phydrus.model.Model.add_solute_transport

Model.**add_solute_transport**(*model=0*, *epsi=0.5*, *lupw=False*, *lartd=False*, *ltdep=False*, *ctola=0*, *ctolr=0*, *maxit=0*, *pecr=2*, *ltort=True*, *lwatdep=False*, *top_bc=- 1*, *bot_bc=0*, *dsurf=None*, *catm=None*, *tpulse=1*)

Method to add solute transport to the model.

Parameters

- **model** (*int, optional*) – Code describing type of nonequilibrium for solute transport: 0 = equilibrium solute transport (Default) 1 = one-site sorption model (chemical nonequilibrium) 2 = two-site sorption model (chemical nonequilibrium) 3 = two kinetic sorption sites model (attachment/detachment; chemical nonequilibrium). This model is often used for particle (viruses, colloids, bacteria) transport. 4 = two kinetic sorption sites model (attachment/detachment) (chemical nonequilibrium). Attachment coefficients are calculated using filtration theory. 5 = dual-porosity model (mobile-immobile regions; physical nonequilibrium). 6 = dual-porosity model (mobile-immobile regions) with two-site sorption in the mobile zone (physical and chemical nonequilibrium). 7 = dual-permeability model (physical nonequilibrium). 8 = dual-permeability model with either an immobile region in the matrix domain (physical nonequilibrium) or with two-site sorption in both domains (physical and chemical nonequilibrium).
- **epsi** (*float, optional*) – Temporal weighing coefficient. 0.0 for an explicit scheme (Default). 0.5 for a Crank-Nicholson implicit scheme. =1.0 for a fully implicit scheme.
- **lupw** (*bool, optional*) – True if upstream weighing formulation is to be used. False if the original Galerkin formulation is to be used.
- **lartd** (*bool, optional*) – True if artificial dispersion is to be added in order to fulfill the stability criterion PeCr (see Section 8.4.4), else False.
- **ltdep** (*bool, optional*) – True if at least one transport or reaction coefficient (ChPar) is temperature dependent, else False. If ltdep=True, then all values of ChPar(i,M) should be specified at a reference temperature Tr=20 degrees celsius.
- **ctola** (*float, optional*) – Absolute concentration tolerance [ML-3], the value is dependent on the units used (set equal to zero if nonlinear adsorption is not considered).
- **ctolr** (*float, optional*) – Relative concentration tolerance [-] (set equal to zero if nonlinear adsorption is not considered).
- **maxit** (*int, optional*) – Maximum number of iterations allowed during any time step for solute transport - usually 20 (set equal to zero if nonlinear adsorption is not considered).
- **pecr** (*float optional*) – Stability criteria. Set to zero when lUpW=True.
- **ltort** (*bool, optional*) – True if the tortuosity factor [Millington and Quirk, 1961] is to be used. False if the tortuosity factor is assumed to be equal to one.
- **lwatdep** (*bool, optional*) – True if at least one degradation coefficient (ChPar) is water content dependent.
- **top_bc** (*int, optional*) – Code which specifies the type of upper boundary condition 1 = Dirichlet boundary condition, -1 = Cauchy boundary condition. -2 = a special type of boundary condition for volatile solutes as described by equation (3.46).
- **bot_bc** (*int, optional*) – Code which specifies the type of lower boundary condition: 1 = Dirichlet boundary condition, 0 = continuous concentration profile, -1 = Cauchy boundary condition.
- **dsurf** (*float, optional*) – Thickness of the stagnant boundary layer, d [L]. Only when kTopCh=-2.

- **catm** (*float, optional*) – Concentration above the stagnant boundary layer, g_atm [ML-3]. Only when kTopCh=-2.
- **tpulse** (*float, optional*) – Time duration of the concentration pulse [T].

See also:

`phydrus.Model.add_solute`

phydrus.model.Model.add_time_info

`Model.`**add_time_info**(*tinit=0*, *tmax=1*, *dt=0.01*, *dtmin=1e-05*, *dtmax=5*, *print_times=False*, *printinit=None*, *printmax=None*, *dtprint=None*, *nsteps=None*, *from_atmo=False*, *print_array=None*)

Method to produce time information.

Parameters

- **tinit** (*int, optional*) – Initial time of the simulation [T].
- **tmax** (*int, optional*) – Final time of the simulation [T].
- **print_times** (*boolean, optional*) – Set to True. if information of pressure head, water contents, temperatures, and concentrations in observation nodes, and the water and solute fluxes is to be printed at a constant time interval of 1 time unit.
- **printinit** (*int, optional*) – First specified print-time [T].
- **printmax** (*int, optional*) – Last specified print-time [T].
- **dtprint** (*int, optional*) – Specified time increment for print times [T].
- **nsteps** (*str, optional*) – Number of required time steps between the first specified print-time (printinit) and the final specified print-time (printmax)".
- **from_atmo** (*boolean, optional.*) – Set to True if time information is determined from the atmospheric boundary condition input data.
- **dt** (*float, optional*) – Initial time increment [T].
- **dtmin** (*float, optional*) – Minimum permitted time increment [T].
- **dtmax** (*float, optional*) – Maximum permitted time increment [T].
- **print_array** (*array of float, optional*) – Array of specified print-times.

phydrus.model.Model.add_waterflow

`Model.`**add_waterflow**(*model=0*, *maxit=10*, *tolth=0.001*, *tolh=1*, *ha=1e-06*, *hb=10000.0*, *linitw=False*, *top_bc=0*, *bot_bc=0*, *hseep=0*, *rtop=None*, *rbot=None*, *rroot=None*, *gw_level=None*, *aqh=None*, *bqh=None*, *hysteresis=0*, *ikappa=- 1*)

Method to add a water_flow module to the model.

Parameters

- **model** (*int, optional*) – Soil hydraulic properties model: 0 = van Genuchten"s [1980] model with 6 parameters. 1 = modified van Genuchten"s model with 10 parameters [Vogel and Cislerova, 1988]. 2 = Brooks and Corey"s [1964] model with 6 parameters. 3 = van Genuchten"s [1980] model with air-entry value of -2 cm and with 6 parameters. 4 = Kosugi's [1996] model with 6 parameters. 5 = dual-porosity model of Durner [1994] with 9 parameters. 6 = dual-porosity system with transfer proportional to the effective saturation (9 parameters). 7 = dual-porosity system with transfer proportional to the pressure head (11

parameters). 9 = dual-permeability system with transfer proportional to the pressure head (17 parameters) !!! model>3 options are not available with the major ion chemistry. module.

- **maxit** (*int, optional*) – Maximum number of iterations allowed during any time step.
- **tolth** (*float, optional*) – Absolute water content tolerance for nodes in the unsaturated part of the flow region [-]. TolTh represents the maximum desired absolute change in the value of the water content, , between two successive iterations during a particular time step.
- **tolh** (*float, optional*) – Absolute pressure head tolerance for nodes in the saturated part of the flow region [L] (its recommended value is 0.1 cm). TolH represents the maximum desired absolute change in the value of the pressure head, h, between two successive iterations during a particular time step.
- **ha** (*float, optional*) – Absolute value of the upper limit [L] of the pressure head interval below which a table of hydraulic properties will be generated internally for each material.
- **hb** (*float, optional*) – Absolute value of the lower limit [L] of the pressure head interval for which a table of hydraulic properties will be generated internally for each material.
- **linitw** (*bool, optional*) – Set to True if the initial condition is given in terms of the water content. Set to False if given in terms of pressure head.
- **top_bc** (*int, optional*) – Upper Boundary Condition: 0 = Constant Pressure Head. 1 = Constant Flux. 2 = Atmospheric Boundary Condition with Surface Layer. 3 = Atmospheric Boundary Condition with Surface Run Off. 4 = Variable Pressure Head. 5 = Variable Pressure Head/Flux.
- **bot_bc** (*int, optional*) – Lower Boundary Condition: 0 = Constant Pressure Head. 1 = Constant Flux. 2 = Variable Pressure Head. 3 = Variable Flux. 4 = Free Drainage. 5 = Deep Drainage. 6 = Seepage Face. 7 = Horizontal Drains.
- **hseep** (*float, optional*) – Pressure head (i.e., 0) that initiates flow over the seepage face bottom boundary.
- **rtop** (*float, optional*) – Prescribed top flux [LT-1] (in case of a Dirichlet BC set this variable equal to zero).
- **rbot** (*float, optional*) – Prescribed bottom flux [LT-1] (in case of a Dirichlet BC set this variable equal to zero).
- **rroot** (*float, optional*) – Prescribed potential transpiration rate [LT-1] (if no transpiration occurs or if transpiration is variable in time set this variable equal to zero).
- **gw_level** (*float, optional*) – Reference position of the groundwater table (e.g., the x-coordinate of the soil surface).
- **aqh** (*float, optional*) – Value of the parameter Aqh [LT-1] in the q(GWL)-relationship.
- **bqh** (*float, optional*) – Value of the parameter Bqh [L-1] in the q(GWL)-relationship.
- **hysteresis** (*int, optional*) – Hysteresis in the soil hydraulic properties: 0 = No hysteresis. 1 = Hysteresis in the retention curve only. 2 = Hysteresis in both the retention and hydraulic conductivity functions. 3 = Hysteresis using Robert Lenhard's model [Lenhard et al., 1991; Lenhard and Parker, 1992]. (Not available with major ion chemistry module.)

- **ikappa** (*int, optional*) – Set to -1 if the initial condition is to be calculated from the main drying branch. Set to 1 if the initial condition is to be calculated from the main wetting branch.

phydrus.model.Model.get_empty_heat_df

Model.**get_empty_heat_df**()
Get an empty DataFrame to fill in the heat parameters.

phydrus.model.Model.get_empty_material_df

Model.**get_empty_material_df**(*n=1*)
Get an empty DataFrame with the soil parameters as columns.

Parameters **n** (*int, optional*) – Number of materials to add.

Returns Pandas DataFrame with the soil parameters as columns.

Return type pandas.DataFrame

Examples

```
>>> m = ml.get_empty_material_df(n=2)
>>> m.loc[1:2] = [[0.08, 0.3421, 0.03, 5, 1, -0.5],
>>>               [0.08, 0.3421, 0.03, 5, 0.1, -0.5]]
>>> ml.add_material(m)
```

phydrus.model.Model.get_empty_solute_df

Model.**get_empty_solute_df**()
Get an empty DataFrame with the solute parameters as columns.

phydrus.model.Model.read_alevel

Model.**read_alevel**(*fname='A_LEVEL.OUT'*, *usecols=None*)
Method to read the A_LEVEL.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the t_level out file. default is "A_LEVEL.OUT".
- **usecols** (*list of str optional*) – List with the names of the columns to import.

Returns **data** – Pandas with the a_level data

Return type pandas.DataFrame

phydrus.model.Model.read_balance

Model.**read_balance**(*fname='BALANCE.OUT'*, *usecols=None*)
Method to read the BALANCE.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the run_inf out file. default is "BALANCE.OUT".
- **usecols** (*list of str optional*) – List with the names of the columns to import. By default: ['Area','W-volume','In-flow','h Mean','Top Flux', 'Bot Flux', 'WatBalT','WatBalR'].

Returns **data** – Pandas with the balance data

Return type pandas.DataFrame

phydrus.model.Model.read_i_check

Model.**read_i_check**(*fname='I_CHECK.OUT'*)
Method to read the I_CHECK.OUT output file.

Parameters **path** (*str, optional*) – String with the name of the I_Check out file. default is "I_Check.OUT".

Returns **data** – Dictionary with the node as a key and a Pandas DataFrame as a value.

Return type dict

phydrus.model.Model.read_nod_inf

Model.**read_nod_inf**(*fname='NOD_INF.OUT'*, *times=None*)
Method to read the NOD_INF.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the NOD_INF out file. default is "NOD_INF.OUT".
- **times** (*int, optional*) – Create a DataFrame with nodal values of the pressure head, the water content, the solution and sorbed concentrations, and temperature, etc, at the time "times". default is None.

Returns **data** – Dictionary with the time as a key and a Pandas DataFrame as a value.

Return type dict

phydrus.model.Model.read_obs_node

Model.**read_obs_node**(*fname='OBS_NODE.OUT'*, *nodes=None*, *conc=False*, *cols=None*)
Method to read the OBS_NODE.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the OBS_NODE out file. default is "OBS_NODE.OUT".
- **nodes** (*list of ints, optional*) – nodes to imoport

- **conc** (*boolean, optional*) –
- **cols** (*list of strs, optional*) – List of strings with the columns to read.

Returns **data** – Dictionary with the node as a key and a Pandas.DataFrame as a value.

Return type dict

phydrus.model.Model.read_profile

Model.**read_profile**(*fname='PROFILE.OUT'*)

Method to read the PROFILE.OUT output file.

Parameters **path** (*str, optional*) – String with the name of the profile out file. default is "PROFILE.OUT".

Returns **data** – Pandas with the profile data

Return type pandas.DataFrame

phydrus.model.Model.read_run_inf

Model.**read_run_inf**(*fname='RUN_INF.OUT'*, *usecols=None*)

Method to read the RUN_INF.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the run_inf out file. default is "RUN_INF.OUT".
- **usecols** (*list of str optional*) – List with the names of the columns to import. Default is all columns.

Returns **data** – Pandas with the run_inf data

Return type pandas.DataFrame

phydrus.model.Model.read_solutes

Model.**read_solutes**(*fname='SOLUTE{}.OUT'*, *solute=1*)

Method to read the SOLUTE.OUT output file.

Parameters **path** (*str, optional*) – String with the name of the solute out file. default is "SOLUTE1.OUT".

Returns **data** – Pandas with the a_level data

Return type pandas.DataFrame

phydrus.model.Model.read_tlevel

Model.**read_tlevel**(*fname='T_LEVEL.OUT'*, *usecols=None*)
Method to read the T_LEVEL.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the t_level out file. default is "T_LEVEL.OUT".
- **usecols** (*list of str optional*) – List with the names of the columns to import. By default only the real fluxes are imported and not the cumulative fluxes. Options are: "rTop", "rRoot", "vTop", "vRoot", "vBot", "sum(rTop)", "sum(rRoot)", "sum(vTop)", "sum(vRoot)", "sum(vBot)", "hTop", "hRoot", "hBot", "RunOff", "sum(RunOff)", "Volume", "sum(Infil)", "sum(Evap)", "TLevel", "Cum(WTrans)", "SnowLayer".

Returns **data** – Pandas with the t_level data

Return type pandas.DataFrame

phydrus.model.Model.set_executable

Model.**set_executable**(*exe_name*)
Method to set the path to the Hydrus-1D executable.

Parameters **exe_name** (*str*) – String with the path to the Hydrus-1D executable.

Examples

```
>>> exe = os.path.join(os.getcwd(), 'hydrus.exe')
>>> ml.set_executable(exe)
```

Notes

This method may also be used to re-set the path to the executable.

phydrus.model.Model.simulate

Model.**simulate**()
Method to call the Hydrus-1D executable.

phydrus.model.Model.write_atmosphere

Model.**write_atmosphere**(*fname='ATMOSPH.IN'*)
Method to write the ATMOSPH.IN file

Parameters **fname** (*str, optional*) – String with the filename. Written to the workspace folder ('ws').

phydrus.model.Model.write_input

Model.**write_input**()
Method to write the input files for the HYDRUS-1D simulation.

phydrus.model.Model.write_profile

Model.**write_profile**(*fname='PROFILE.DAT'*)
Method to write the PROFILE.DAT file.

Parameters **fname** (*str, optional*) – String with the filename. Written to the workspace folder ('ws').

phydrus.model.Model.write_selector

Model.**write_selector**(*fname='SELECTOR.IN'*)
Write the SELECTOR.IN file.

Parameters **fname** (*str, optional*) – String with the filename. Written to the workspace folder ('ws').

4.2 phydrus.read

The read module contains methods that can be used to read in- and output files. The methods can be used stand-alone but are also available from the Model object. All the methods return the data as Pandas DataFrames.

Examples

```
>>> import phydrus as ps
>>> ps.read.read_obs_node()
```

or

```
>>> ml.read_obs_node()
```

Functions

read_alevel	Method to read the A_LEVEL.OUT output file.
read_balance	Method to read the BALANCE.OUT output file.
read_i_check	Method to read the I_CHECK.OUT output file.
read_nod_inf	Method to read the NOD_INF.OUT output file.
read_obs_node	Method to read the OBS_NODE.OUT output file.
read_profile	Method to read the PROFILE.OUT output file.
read_run_inf	Method to read the RUN_INF.OUT output file.
read_solute	Method to read the SOLUTE.OUT output file.
read_tlevel	Method to read the T_LEVEL.OUT output file.

4.2.1 phydrus.read.read_alevel

phydrus.read.**read_alevel**(*path='A_LEVEL.OUT'*, *usecols=None*)
Method to read the A_LEVEL.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the t_level out file. default is "A_LEVEL.OUT".
- **usecols** (*list of str optional*) – List with the names of the columns to import.

Returns **data** – Pandas with the a_level data

Return type pandas.DataFrame

4.2.2 phydrus.read.read_balance

phydrus.read.**read_balance**(*path='BALANCE.OUT'*, *usecols=None*)
Method to read the BALANCE.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the run_inf out file. default is "BALANCE.OUT".
- **usecols** (*list of str optional*) – List with the names of the columns to import. By default: ['Area','W-volume','In-flow','h Mean','Top Flux', 'Bot Flux', 'WatBalT','WatBalR'].

Returns **data** – Pandas with the balance data

Return type pandas.DataFrame

4.2.3 phydrus.read.read_i_check

phydrus.read.**read_i_check**(*path='I_CHECK.OUT'*)
Method to read the I_CHECK.OUT output file.

Parameters **path** (*str, optional*) – String with the name of the I_Check out file. default is "I_Check.OUT".

Returns **data** – Dictionary with the node as a key and a Pandas DataFrame as a value.

Return type dict

4.2.4 phydrus.read.read_nod_inf

phydrus.read.**read_nod_inf**(*path='NOD_INF.OUT'*, *times=None*)
Method to read the NOD_INF.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the NOD_INF out file. default is "NOD_INF.OUT".
- **times** (*int, optional*) – Create a DataFrame with nodal values of the pressure head, the water content, the solution and sorbed concentrations, and temperature, etc, at the time "times". default is None.

Returns **data** – Dictionary with the time as a key and a Pandas DataFrame as a value.

Return type dict

4.2.5 phydrus.read.read_obs_node

phydrus.read.**read_obs_node**(*path='OBS_NODE.OUT'*, *nodes=None*, *conc=False*, *cols=None*)
Method to read the OBS_NODE.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the OBS_NODE out file. default is "OBS_NODE.OUT".
- **nodes** (*list of ints, optional*) – nodes to imoport
- **conc** (*boolean, optional*) –
- **cols** (*list of strs, optional*) – List of strings with the columns to read.

Returns **data** – Dictionary with the node as a key and a Pandas.DataFrame as a value.

Return type dict

4.2.6 phydrus.read.read_profile

phydrus.read.**read_profile**(*path='PROFILE.OUT'*)
Method to read the PROFILE.OUT output file.

Parameters **path** (*str, optional*) – String with the name of the profile out file. default is "PROFILE.OUT".

Returns **data** – Pandas with the profile data

Return type pandas.DataFrame

4.2.7 phydrus.read.read_run_inf

phydrus.read.**read_run_inf**(*path='RUN_INF.OUT'*, *usecols=None*)
Method to read the RUN_INF.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the run_inf out file. default is "RUN_INF.OUT".
- **usecols** (*list of str optional*) – List with the names of the columns to import. Default is all columns.

Returns **data** – Pandas with the run_inf data

Return type pandas.DataFrame

4.2.8 phydrus.read.read_solute

phydrus.read.**read_solute**(*path='SOLUTE1.OUT'*)
Method to read the SOLUTE.OUT output file.

Parameters **path** (*str, optional*) – String with the name of the solute out file. default is "SOLUTE1.OUT".

Returns **data** – Pandas with the a_level data

Return type pandas.DataFrame

4.2.9 phydrus.read.read_tlevel

phydrus.read.**read_tlevel**(*path='T_LEVEL.OUT'*, *usecols=None*)
Method to read the T_LEVEL.OUT output file.

Parameters

- **path** (*str, optional*) – String with the name of the t_level out file. default is "T_LEVEL.OUT".
- **usecols** (*list of str optional*) – List with the names of the columns to import. By default only the real fluxes are imported and not the cumulative fluxes. Options are: "rTop", "rRoot", "vTop", "vRoot", "vBot", "sum(rTop)", "sum(rRoot)", "sum(vTop)", "sum(vRoot)", "sum(vBot)", "hTop", "hRoot", "hBot", "RunOff", "sum(RunOff)", "Volume", "sum(Infil)", "sum(Evap)", "TLevel", "Cum(WTrans)", "SnowLayer".

Returns **data** – Pandas with the t_level data

Return type pandas.DataFrame

4.3 phydrus.profile

The profile module contains utility functions to help create the profile DataFrame.

Functions

create_profile	Method to create a DataFrame describing the soil profile.
profile_from_file	Method to create a profile DataFrame from a profile.dat file

4.3.1 phydrus.profile.create_profile

phydrus.profile.**create_profile**(*top=0*, *bot=- 1*, *dx=0.1*, *h=0*, *lay=1*, *mat=1*, *beta=0*, *ah=1.0*, *ak=1.0*, *ath=1.0*, *temp=20.0*, *conc=None*, *sconc=None*)
Method to create a DataFrame describing the soil profile.

Parameters

- **top** (*float, optional*) – Top of the soil column.
- **bot** (*float or list of float, optional*) – Bottom of the soil column. If a list

is provided, multiple layers are created and other arguments need to be of the same length (e.g. mat).

- **dx** (*float, optional*) – Size of each grid cell. Default 0.1 meter.
- **h** (*float, optional*) – Initial values of the pressure head.
- **lay** (*int or list of int, optional*) – Subregion number (for mass balance calculations).
- **mat** (*int or list of int, optional*) – Material number (for heterogeneity).
- **beta** (*float or list of float, optional*) – Value of the water uptake distribution, b(x) [L-1], in the soil root zone at node n. Set Beta(n) equal to zero if node n lies outside the root zone.
- **ah** (*float or list of float, optional*) – Scaling factor for the pressure head (Axz in profile.dat).
- **ak** (*float or list of float, optional*) – Scaling factor the hydraulic conductivity (Bxz in profile.dat).
- **ath** (*float or list of float, optional*) – Scaling factor the the water content (Dxz in profile.dat).
- **temp** (*float, optional*) – Initial value of the temperature at node n [oC] (do not specify if both lTemp or lChem are equal to .false.; if lTemp=.false. and lChem=.true. then set equal to 0 or any other initial value to be used later for temperature dependent water flow and solute transport).
- **conc** (*float, optional*) –
- **sconc** (*float, optional*) –

4.3.2 phydrus.profile.profile_from_file

phydrus.profile.**profile_from_file**(*fname='PROFILE.DAT'*, *ws=None*)

Method to create a profile DataFrame from a profile.dat file

Parameters

- **fname** (*str, optional*) – String with the path to the PROFILE.DAT file.
- **ws** (*str, optional*) – String with the path to the workspace.

Returns **data** – Pandas DataFrame with the soil profile data.

Return type pandas.DataFrame

Examples

```
>>> profile = ps.create_profile(h=0.342)
```

4.4 Plots

class phydrus.plot.**Plots**(*ml*)

Class that contains all the methods to plot a Phydrus Model.

Parameters **ml** (*phydrus.Model*) – Phydrus Model Instance to connect the methods to the model.

Examples

```
>>> ml.plots.profile()
```

4.4.1 Methods

__init__	Initialize self.
obs_points	Method to plot the pressure heads, water contents and water fluxes.
profile	Method to plot the soil profile.
profile_information	Method to plot the soil profile information.
soil_properties	Method to plot the soil hydraulic properties.
water_flow	Method to plot the water flow information.

phydrus.plot.Plots.__init__

Plots.**__init__**(*ml*)

Initialize self. See help(type(self)) for accurate signature.

phydrus.plot.Plots.obs_points

Plots.**obs_points**(*data='h'*, *figsize=(4, 3)*, ***kwargs*)

Method to plot the pressure heads, water contents and water fluxes.

Parameters

- **data** (*str, optional*) – String with the variable of the variable to plot. You can choose between: "h", "theta", "Temp", "Conc".
- **figsize** (*tuple, optional*) –

Returns **axes**

Return type matplotlib axes instance

phydrus.plot.Plots.profile

Plots.**profile**(*figsize=(3, 6)*, *title=None*, *cmap='YlOrBr'*, *color_by='Ks'*, *show_grid=True*, ***kwargs*)
Method to plot the soil profile.

Parameters

- **figsize** (*tuple, optional*) – Tuple with the size of the figure in inches.
- **title** (*str, optional*) – String with the title of the Figure.
- **cmap** (*str, optional*) – String with a named Matplotlib colormap.
- **color_by** (*str, optional*) – Column from the material properties sed to color the materials. Default is "Ks".
- **show_grid** (*bool, optional*) – Show the grid in the plot. Default is True.

Returns ax

Return type matplotlib axes instance

phydrus.plot.Plots.profile_information

Plots.**profile_information**(*data='Pressure Head'*, *times=None*, *legend=True*, *figsize=(5, 3)*, ***kwargs*)
Method to plot the soil profile information.

Parameters

- **data** (*str, optional*) – String with the variable of the profile information to plot. You can choose between: "Pressure Head", "Water Content", "Hydraulic Conductivity","Hydraulic Capacity", "Water Flux", "Root Uptake", "Temperature". Default is "Pressure Head".
- **times** (*list of int*) – List of integers of the time step to plot.
- **figsize** (*tuple, optional*) –
- **legend** (*boolean, optional*) –

Returns ax

Return type matplotlib axes instance

phydrus.plot.Plots.soil_properties

Plots.**soil_properties**(*data='Water Content'*, *figsize=(6, 3)*, ***kwargs*)
Method to plot the soil hydraulic properties.

Parameters

- **data** (*str, optional*) – String with the variable of the water flow information to plot. You can choose between: "Water Content", "Pressure head", "log Pressure head", "Hydraulic Capacity", "Hydraulic Conductivity", "log Hydraulic Conductivity", "Effective Water Content". Default is "Water Content".
- **figsize** (*tuple, optional*) –

Returns axes

Return type matplotlib axes instance

phydrus.plot.Plots.water_flow

Plots.**water_flow**(*data='Potential Surface Flux'*, *figsize=(6, 3)*, ***kwargs*)
Method to plot the water flow information.

Parameters

- **data** (*str, optional*) – String with the variable of the water flow information to plot. You can choose between: "Potential Surface Flux", "Potential Root Water Uptake", "Actual Surface Flux", "Actual Root Water Uptake", "Bottom Flux", "Pressure head at the soil surface", "Mean value of the pressure head over the region", "Pressure head at the Bottom of the soil profile", "Surface runoff", "Volume of water in the entire flow domain". Default is "Potential Surface Flux".
- **figsize** (*tuple, optional*) –

Returns ax

Return type matplotlib axes instance

4.5 phydrus.utils

The utils module contains utility funtions for Phydrus.

Functions

add_file_handlers	Method to add file handlers in the logger of Phydrus.
remove_console_handler	Method to remove the console handler to the logger of Phydrus.
remove_file_handlers	Method to remove any file handlers in the logger of Phydrus.
set_console_handler	Method to add a console handler to the logger of Phydrus.
set_log_level	Set the log-level for which to log Phydrus messages.
show_versions	Method to print the version of dependencies.

4.5.1 phydrus.utils.add_file_handlers

phydrus.utils.**add_file_handlers**(*logger=None*, *filenames=('info.log', 'errors.log')*, *levels=(20, 40)*, *maxBytes=10485760*, *backupCount=20*, *encoding='utf8'*, *datefmt='%d %H:%M'*, *fmt='%(asctime)s-%(name)s-%(levelname)s-%(message)s'*)
Method to add file handlers in the logger of Phydrus.

Parameters **logger** (*logging.Logger*) – A Logger-instance. Use ps.logger to initialise the Logging instance that handles all logging throughout Phydrus, including all sub modules and packages.

4.5.2 phydrus.utils.remove_console_handler

phydrus.utils.**remove_console_handler**(*logger=None*)
Method to remove the console handler to the logger of Phydrus.

Parameters **logger** (*logging.Logger*) – A Logger-instance. Use ps.logger to initialise the Logging instance that handles all logging throughout Phydrus, including all sub modules and packages.

4.5.3 phydrus.utils.remove_file_handlers

phydrus.utils.**remove_file_handlers**(*logger=None*)
Method to remove any file handlers in the logger of Phydrus.

Parameters **logger** (*logging.Logger*) – A Logger-instance. Use ps.logger to initialise the Logging instance that handles all logging throughout Phydrus, including all sub modules and packages.

4.5.4 phydrus.utils.set_console_handler

phydrus.utils.**set_console_handler**(*logger=None*, *level=20*, *fmt='%(levelname)s: %(message)s'*)
Method to add a console handler to the logger of Phydrus.

Parameters **logger** (*logging.Logger*) – A Logger-instance. Use ps.logger to initialise the Logging instance that handles all logging throughout Phydrus, including all sub modules and packages.

4.5.5 phydrus.utils.set_log_level

phydrus.utils.**set_log_level**(*level*)
Set the log-level for which to log Phydrus messages.

Parameters **level** (*str*) – String with the level to log messages to the screen for. Options are: "INFO", "WARNING", and "ERROR".

Examples

```
>>> import phydrus as ps
>>> ps.set_log_level("ERROR")
```

4.5.6 phydrus.utils.show_versions

phydrus.utils.**show_versions**()
Method to print the version of dependencies.

Examples

```
>>> import phydrus as ps
>>> ps.show_versions()
```

Python version: 3.8.2 (default, Mar 25 2020, 11:22:43) [Clang 4.0.1 (tags/RELEASE_401/final)] Numpy version: 1.19.2 Pandas version: 1.2.1 Phydrus version: 0.1.0 Matplotlib version: 3.3.2

CHAPTER

FIVE

RELEASE NOTES

PYTHON MODULE INDEX

p

phydrus.profile, 28
phydrus.read, 25
phydrus.utils, 32

INDEX

Symbols

__init__() (*phydrus.model.Model method*), 13
__init__() (*phydrus.plot.Plots method*), 30

A

add_atmospheric_bc() (*phydrus.model.Model method*), 13
add_drains() (*phydrus.model.Model method*), 14
add_file_handlers() (*in module phydrus.utils*), 32
add_heat_transport() (*phydrus.model.Model method*), 14
add_material() (*phydrus.model.Model method*), 15
add_obs_nodes() (*phydrus.model.Model method*), 15
add_profile() (*phydrus.model.Model method*), 15
add_root_growth() (*phydrus.model.Model method*), 16
add_root_uptake() (*phydrus.model.Model method*), 16
add_solute() (*phydrus.model.Model method*), 17
add_solute_transport() (*phydrus.model.Model method*), 18
add_time_info() (*phydrus.model.Model method*), 19
add_waterflow() (*phydrus.model.Model method*), 19

C

create_profile() (*in module phydrus.profile*), 28

G

get_empty_heat_df() (*phydrus.model.Model method*), 21
get_empty_material_df() (*phydrus.model.Model method*), 21
get_empty_solute_df() (*phydrus.model.Model method*), 21

M

Model (*class in phydrus.model*), 11
module
    phydrus.profile, 28
    phydrus.read, 25
    phydrus.utils, 32

O

obs_points() (*phydrus.plot.Plots method*), 30

P

phydrus.profile
    module, 28
phydrus.read
    module, 25
phydrus.utils
    module, 32
Plots (*class in phydrus.plot*), 30
profile() (*phydrus.plot.Plots method*), 31
profile_from_file() (*in module phydrus.profile*), 29
profile_information() (*phydrus.plot.Plots method*), 31

R

read_alevel() (*in module phydrus.read*), 26
read_alevel() (*phydrus.model.Model method*), 21
read_balance() (*in module phydrus.read*), 26
read_balance() (*phydrus.model.Model method*), 22
read_i_check() (*in module phydrus.read*), 26
read_i_check() (*phydrus.model.Model method*), 22
read_nod_inf() (*in module phydrus.read*), 26
read_nod_inf() (*phydrus.model.Model method*), 22
read_obs_node() (*in module phydrus.read*), 27
read_obs_node() (*phydrus.model.Model method*), 22
read_profile() (*in module phydrus.read*), 27
read_profile() (*phydrus.model.Model method*), 23
read_run_inf() (*in module phydrus.read*), 27
read_run_inf() (*phydrus.model.Model method*), 23
read_solute() (*in module phydrus.read*), 28
read_solutes() (*phydrus.model.Model method*), 23
read_tlevel() (*in module phydrus.read*), 28
read_tlevel() (*phydrus.model.Model method*), 24

remove_console_handler() (*in module phydrus.utils*), 33
remove_file_handlers() (*in module phydrus.utils*), 33

S

set_console_handler() (*in module phydrus.utils*), 33
set_executable() (*phydrus.model.Model method*), 24
set_log_level() (*in module phydrus.utils*), 33
show_versions() (*in module phydrus.utils*), 33
simulate() (*phydrus.model.Model method*), 24
soil_properties() (*phydrus.plot.Plots method*), 31

W

water_flow() (*phydrus.plot.Plots method*), 32
write_atmosphere() (*phydrus.model.Model method*), 24
write_input() (*phydrus.model.Model method*), 25
write_profile() (*phydrus.model.Model method*), 25
write_selector() (*phydrus.model.Model method*), 25