ismn Documentation

Release unknown

TU Wien

May 03, 2021

CONTENTS

Readers for the data from the International Soil Moisture Database (ISMN).

CHAPTER

ONE

CITATION

If you use the software in a publication then please cite it using the Zenodo DOI. Be aware that this badge links to the latest package version.

Please select your specific version at https://doi.org/10.5281/zenodo.855308 to get the DOI of that version. You should normally always use the DOI for the specific version of your record in citations. This is to ensure that other researchers can access the exact research artefact you used for reproducibility.

You can find additional information regarding DOI versioning at http://help.zenodo.org/#versioning

CHAPTER

TWO

INSTALLATION

This package should be installable through pip:

```
pip install ismn
```

The cartopy-package needs to be installed manually by using the following command:

```
conda install -c conda-forge cartopy
```

2.1 Example installation script

The following script will install miniconda and setup the environment on a UNIX like system. Miniconda will be installed into `$HOME/miniconda`.

```
wget https://repo.continuum.io/miniconda/Miniconda-latest-Linux-x86_64.sh -O␣
↪miniconda.sh
bash miniconda.sh -b -p $HOME/miniconda
export PATH="$HOME/miniconda/bin:$PATH"
git clone git@github.com:TUW-GEO/ismn.git ismn
cd ismn
conda env create -f environment.yml
source activate ismn
pip install -r test-requirements.txt
```

This script adds `$HOME/miniconda/bin` temporarily to the `PATH` to do this permanently add `export PATH="$HOME/miniconda/bin:$PATH"` to your `.bashrc` or `.zshrc`

The second to last line in the example activates the `ismn` environment.

After that you should be able to run:

```
python setup.py test
```

to run the test suite.

CHAPTER

THREE

DESCRIPTION

ISMN data can be downloaded for free after registration from the ISMN Website

In case of the ISMN, two different formats are provided:

- Variables stored in separate files (CEOP formatted)

  this format is supported 100% and should work with all examples

- Variables stored in separate files (Header+values)

  this format is supported 100% and should work with all examples

If you downloaded ISMN data in one of the supported formats in the past it can be that station names are not recognized correctly because they contained the '_' character which is supposed to be the separator. If you experience problems because of this please download new data from the ISMN since this issue should be fixed.

3.1 Landcover Classification

The ISMN data comes with information about landcover classification from the ESA CCI land cover project (years 2000, 2005 and 2010) as well as from in-situ measurements. To use ESA CCI land cover variables for filtering the data in the get_dataset_ids function, set the keyword parameters (landcover_2000, landcover_2005 or landcover_2010) to the corresponding integer values (e.g. 10) in the list below. To get a list of possible values for filtering by in-situ values (keyword parameter: "landcover_insitu"), call the get_landcover_types method of your ISMN_Interface object and set landcover='landcover_insitu'.

- 10: Cropland, rainfed
- 11: Cropland, rainfed / Herbaceous cover
- 12: Cropland, rainfed / Tree or shrub cover,
- 20: Cropland, irrigated or post-flooding,
- 30: Mosaic cropland (>50%) / natural vegetation (tree, shrub, herbaceous,
- 40: Mosaic natural vegetation (tree, shrub, herbaceous cover) (>50%) / cropland (<50%),
- 50: Tree cover, broadleaved, evergreen, Closed to open (>15%),
- 60: Tree cover, broadleaved, deciduous, Closed to open (>15%),
- 61: Tree cover, broadleaved, deciduous, Closed (>40%),
- 62: Tree cover, broadleaved, deciduous, Open (15-40%),
- 70: Tree cover, needleleaved, evergreen, closed to open (>15%),
- 71: Tree cover, needleleaved, evergreen, closed (>40%),

- 72: Tree cover, needleleaved, evergreen, open (15-40%),
- 80: Tree cover, needleleaved, deciduous, closed to open (>15%),
- 81: Tree cover, needleleaved, deciduous, closed (>40%),
- 82: Tree cover, needleleaved, deciduous, open (15-40%),
- 90: Tree cover, mixed leaf type (broadleaved and needleleaved),
- 100: Mosaic tree and shrub (>50%) / herbaceous cover (<50%),
- 110: Mosaic herbaceous cover (>50%) / tree and shrub (<50%),
- 120: Shrubland,
- 121: Shrubland / Evergreen Shrubland,
- 122: Shrubland / Deciduous Shrubland,
- 130: Grassland,
- 140: Lichens and mosses,
- 150: Sparse vegetation (tree, shrub, herbaceous cover) (<15%),
- 152: Sparse vegetation (tree, shrub, herbaceous cover) (<15%) / Sparse shrub (<15%),
- 153: Sparse vegetation (tree, shrub, herbaceous cover) (<15%) / Sparse herbaceous cover (<15%),
- 160: Tree cover, flooded, fresh or brakish water,
- 170: Tree cover, flooded, saline water,
- 180: Shrub or herbaceous cover, flooded, fresh/saline/brakish water,
- 190: Urban areas,
- 200: Bare areas,
- 201: Consolidated bare areas,
- 202: Unconsolidated bare areas,
- 210: Water,
- 220: Permanent snow and ice,

3.2 Climate Classification

The ISMN data comes with information about climate classification from the Koeppen-Geiger Climate Classification (2007) as well as in-situ measurements. To use Koeppen-Geiger variable for filtering the data in the get_dataset_ids function, set the keyword parameter "climate" to the corresponding keys (e.g. 'Af') in the list below. To get a list of possible values for filtering by in-situ values (keyword parameter: "climate_insitu"), call the get_climate_types method of your ISMN_Interface object and set climate='climate_insitu'.

- Af: Tropical Rainforest
- Am: Tropical Monsoon
- As: Tropical Savanna Dry
- Aw: Tropical Savanna Wet
- BWk: Arid Desert Cold
- BWh: Arid Desert Hot

- BWn: Arid Desert With Frequent Fog
- BSk: Arid Steppe Cold
- BSh: Arid Steppe Hot
- BSn: Arid Steppe With Frequent Fog
- Csa: Temperate Dry Hot Summer
- Csb: Temperate Dry Warm Summer
- Csc: Temperate Dry Cold Summer
- Cwa: Temperate Dry Winter, Hot Summer
- Cwb: Temperate Dry Winter, Warm Summer
- Cwc: Temperate Dry Winter, Cold Summer
- Cfa: Temperate Without Dry Season, Hot Summer
- Cfb: Temperate Without Dry Season, Warm Summer
- Cfc: Temperate Without Dry Season, Cold Summer
- Dsa: Cold Dry Summer, Hot Summer
- Dsb: Cold Dry Summer, Warm Summer
- Dsc: Cold Dry Summer, Cold Summer
- Dsd: Cold Dry Summer, Very Cold Winter
- Dwa: Cold Dry Winter, Hot Summer
- Dwb: Cold Dry Winter, Warm Summer
- Dwc: Cold Dry Winter, Cold Summer
- Dwd: Cold Dry Winter, Very Cold Winter
- Dfa: Cold Dry Without Dry Season, Hot Summer
- Dfb: Cold Dry Without Dry Season, Warm Summer
- Dfc: Cold Dry Without Dry Season, Cold Summer
- Dfd: Cold Dry Without Dry Season, Very Cold Winter
- ET: Polar Tundra
- EF: Polar Eternal Winter
- W: Water

3.3 Documentation

https://ismn.readthedocs.io

CHAPTER

FOUR

CONTRIBUTE

We are happy if you want to contribute. Please raise an issue explaining what is missing or if you find a bug. We will also gladly accept pull requests against our master branch for new features or bug fixes.

4.1 Development setup

For Development we also recommend a `conda` environment. You can create one including test dependencies and debugger by running `conda env create -f environment.yml`. This will create a new `ismn` environment which you can activate by using `source activate ismn`.

4.2 Guidelines

If you want to contribute please follow these steps:

- Fork the ismn repository to your account
- Clone the repository
- make a new feature branch from the ismn master branch
- Add your feature
- Please include tests for your contributions in one of the test directories. We use py.test so a simple function called test_my_feature is enough
- submit a pull request to our master branch

CHAPTER

FIVE

NOTE

This project has been set up using PyScaffold 2.5.7. For details and usage information on PyScaffold see http://pyscaffold.readthedocs.org/.

CHAPTER

SIX

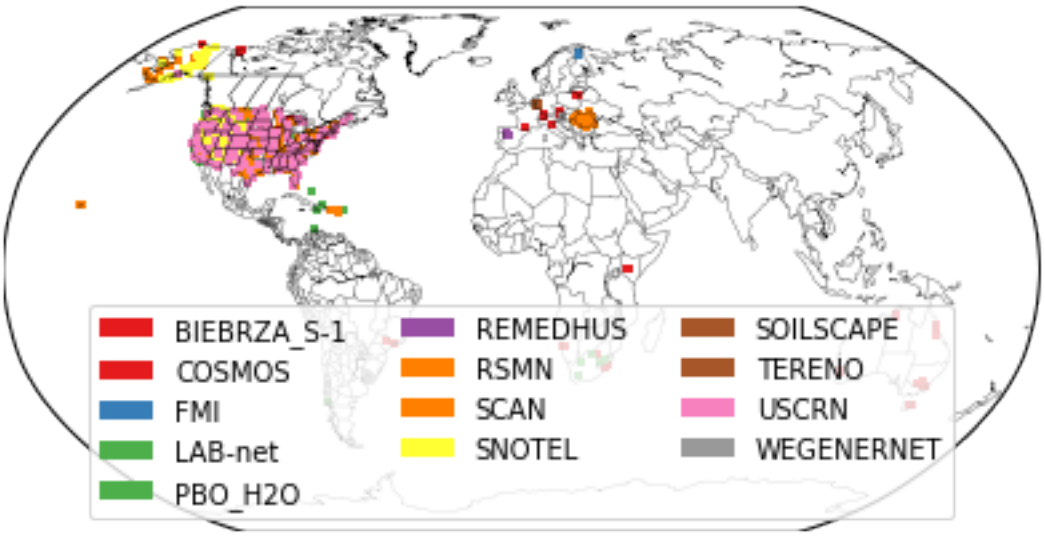
READING AND PLOTTING DATA FROM THE ISMN

This example program chooses a random Network and Station and plots the first variable,depth,sensor combination.

```
from ismn.interface import ISMN_Interface
import matplotlib.pyplot as plt
import random
%matplotlib inline
```

```
path_to_ismn_data = "/data/CGLS/SQE_2016/cgls-validationreports/ISMN/raw/data_
↪20160101_20161231/"
```

```
#initialize interface, this can take up to a few minutes the first
#time, since all metadata has to be collected

ISMN_reader = ISMN_Interface(path_to_ismn_data)

#plot available station on a map
fig, ax = ISMN_reader.plot_station_locations()
plt.show()
```



Next we explore the available networks and stations and select a random network and station to plot data from

```
networks = ISMN_reader.list_networks()
print "Available Networks:"
print networks
```

```
Available Networks:
['BIEBRZA_S-1' 'COSMOS' 'FMI' 'LAB-net' 'PBO_H2O' 'REMEDHUS' 'RSMN' 'SCAN'
 'SNOTEL' 'SOILSCAPE' 'TERENO' 'USCRN' 'WEGENERNET']
```

```
network = random.choice(networks)
stations = ISMN_reader.list_stations(network = network)
print "Available Stations in Network %s"%network
print stations
```

```
Available Stations in Network SNOTEL
['AGUA_CANYON' 'ANCHOR_RIVER_DIVIDE' 'ANNIE_SPRINGS' 'ARAPAHO_RIDGE'
 'ATIGUN_PASS' 'ATLANTA_SUMMIT' 'BAKER_BUTTE_SMT' 'BALDY' 'BANNER_SUMMIT'
 'BEAR_CREEK' 'BEAR_RIVER_RS' 'BEAVER_DAMS' 'BEAVER_DIVIDE' 'BEAVER_PASS'
 'BEN_LOMOND_PEAK' 'BEN_LOMOND_TRAIL' 'BERRY_CREEK' 'BERTHOUD_SUMMIT'
 'BIG_BEND' 'BIG_CREEK_SUM' 'BIG_FLAT' 'BIG_GOOSE' 'BIG_MEADOW'
 'BIG_SANDY_OPENING' 'BILLIE_CREEK_DIVIDE' 'BIRD_CREEK' 'BLACKHALL_MTN'
 'BLACKS_FORK_JCT' 'BLACKTAIL_MTN' 'BLACK_BEAR' 'BLACK_FLAT-U.M._CK'
 'BLACK_PINE' 'BLUE_LAKES' 'BOGUS_BASIN' 'BONE_SPRINGS_DIV' 'BOURNE'
 'BOX_CREEK' 'BRIAN_HEAD' 'BRIGHTON' 'BRISTLECONE_TRAIL' 'BROWN_DUCK'
 'BROWN_TOP' 'BRUMLEY' 'BUCKBOARD_FLAT' 'BUCKINGHORSE' 'BUCKSKIN_JOE'
 'BUCKSKIN_LOWER' 'BUCK_FLAT' 'BUCK_PASTURE' 'BUG_LAKE' 'BURNSIDE_LAKE'
 'BURNT_MOUNTAIN' 'BURRO_MOUNTAIN' 'BURTS_MILLER_RANCH' 'BUTTE'
 'Baker_Butte' 'Bar_M' 'Bevans_Cabin' 'Black_Mesa' 'CAMP_JACKSON'
 'CARSON_PASS' 'CASCADE_#2' 'CASCADE_MOUNTAIN' 'CASTLE_CREEK'
 'CASTLE_VALLEY' 'CAVE_MOUNTAIN' 'CAYUSE_PASS' 'CHALK_CREEK_#1'
 'CHALK_CREEK_#2' 'CHEMULT_ALTERNATE' 'CHEPETA' 'CHOCOLATE_GULCH'
 'CINNABAR_PARK' 'CLACKAMAS_LAKE' 'CLAYTON_SPRINGS' 'CLEAR_CREEK_#1'
 'CLEAR_CREEK_#2' 'CLEAR_LAKE' 'CLOVER_MEADOW' 'COCHETOPA_PASS' 'COLDFOOT'
 'CORRAL_CANYON' 'CRAB_CREEK' 'CRATER_MEADOWS' 'CROW_CREEK' 'CSS_LAB'
 'CULEBRA_#2' 'CURRANT_CREEK' 'Chalender' 'Columbia_Basin'
 'Copper_Mountain' 'Corduroy_Flat' 'Corral' 'DANIELS-STRAWBERRY'
 'DIAMOND_PEAK' 'DILLS_CAMP' 'DISASTER_PEAK' 'DIVIDE' 'DONKEY_RESERVOIR'
 'DORSEY_BASIN' 'DRAW_CREEK' 'DRY_BREAD_POND' 'DRY_FORK' 'DRY_LAKE'
 'Defiance_Mines' 'Dollarhide_Summit' 'Dry_Creek' 'EAGLE_SUMMIT'
 'EAST_RIM_DIVIDE' 'EAST_WILLOW_CREEK' 'EBBETTS_PASS' 'ECHO_PEAK'
 'EF_BLACKS_FORK_GS' 'ELK_PEAK' 'EXIT_GLACIER' 'Elk_Cabin' 'FALLEN_LEAF'
 'FARMINGTON' 'FARMINGTON_LOWER' 'FARNSWORTH_LAKE' 'FAWN_CREEK'
 'FISH_LAKE_UTAH' 'FIVE_POINTS_LAKE' 'FLATTOP_MTN.' 'FORESTDALE_CREEK'
 'FORT_VALLEY' 'FRANKLIN_BASIN' 'FREMONT_PASS' 'Fry_Canyon' 'GALENA_SUMMIT'
 'GARDEN_CITY_SUMMIT' 'GARDNER_PEAK' 'GBRC_HQ' 'GBRC_MEADOWS'
 'GEORGE_CREEK' 'GIVEOUT' 'GOBBLERS_KNOB' 'GOLCONDA' 'GOLD_AXE_CAMP'
 'GOOSEBERRY_R.S.' 'GOOSEBERRY_R.S._UP' 'GRAND_TARGHEE' 'GRANITE_CRK'
 'GRANITE_PEAK' 'GREEN_MOUNTAIN' 'GRIZZLY_PEAK' 'GROUSE_CAMP' 'GUTZ_PEAK'
 'Gallegos_Peak' 'Granite_Creek' 'Gunsight_Pass' 'HAGANS_MEADOW'
 'HAPPY_JACK' 'HARDSCRABBLE' 'HARRIS_FLAT' 'HARTS_PASS' 'HAYDEN_FORK'
 'HEAVENLY_VALLEY' 'HEWINTA' 'HICKERSON_PARK' 'HIGH_RIDGE' 'HILTS_CREEK'
 'HOLE-IN-MOUNTAIN' 'HOLE-IN-ROCK' 'HOLLAND_MEADOWS' 'HOOSIER_PASS'
 'HORSE_MEADOW' 'HORSE_RIDGE' 'HYNDMAN' 'Hawley_Lake' 'Hobble_Creek'
 'Hopewell' 'Huntington_Horse' 'IMNAVIAT_CREEK' 'INDEPENDENCE_CAMP'
 'INDEPENDENCE_CREEK' 'INDEPENDENCE_LAKE' 'INDIAN_CANYON' 'INDIAN_ROCK'
 'JACKSON_PEAK' 'JACKS_PEAK' 'JACKWHACKER_GULCH' 'JACK_CREEK_UPPER'
 'JONES_CORRAL' 'Jakes_Creek' 'KALAMAZOO' 'KELLEY_R.S.' 'KELLY_STATION'
```

(continues on next page)

(continued from previous page)

```
'KENAI_MOOSE_PENS' 'KILFOIL_CREEK' 'KIMBERLY_MINE' 'KINGS_CABIN'
'KLONDIKE_NARROWS' 'KOLOB' 'LAKEFORK_#1' 'LAKEFORK_#3' 'LAKEVIEW_RIDGE'
'LAMANCE_CREEK' 'LAMOILLE_#3' 'LAPRELE_CREEK' 'LARSEN_CREEK'
'LASAL_MOUNTAIN' 'LASAL_MOUNTAIN-LOWER' 'LAUREL_DRAW' 'LEAVITT_LAKE'
'LEAVITT_MEADOWS' 'LEE_CANYON' 'LEWIS_LAKE_DIVIDE' 'LEWIS_PEAK'
'LICK_CREEK' 'LIGHTNING_RIDGE' 'LILY_LAKE' 'LILY_POND' 'LITTLE_BEAR'
'LITTLE_CHENA_RIDGE' 'LITTLE_GOOSE' 'LITTLE_GRASSY' 'LITTLE_SNAKE_RIVER'
'LIZARD_HEAD_PASS' 'LOBDELL_LAKE' 'LONE_CONE' 'LONG_DRAW_RESV' 'LONG_FLAT'
'LONG_VALLEY' 'LONG_VALLEY_JCT' 'LOOKOUT' 'LOOKOUT_PEAK' 'LOST_CREEK_RESV'
'LOST_DOG' 'LOST_HORSE' 'LOUIS_MEADOW' 'LYNX_PASS' 'Lakefork_Basin'
'Little_Valley' 'Lonesome_Beaver' 'MADISON_BUTTE' 'MAGIC_MOUNTAIN'
'MAMMOTH-COTTONWOOD' 'MANY_GLACIER' 'MARLETTE_LAKE' 'MEDANO_PASS'
'MERCHANT_VALLEY' 'MF_Nooksack' 'MICA_CREEK' 'MICHIGAN_CREEK'
'MIDDLE_FORK_CAMP' 'MIDWAY_VALLEY' 'MILL-D_NORTH' 'MILLER_WOODS'
'MINING_FORK' 'MONAHAN_FLAT' 'MONITOR_PASS' 'MONTE_CRISTO'
'MONUMENT_CREEK' 'MOORE_CREEK_BRIDGE' 'MORMON_MTN_SUMMIT' 'MOSBY_MTN.'
'MOSCOW_MOUNTAIN' 'MOSES_MTN' 'MOSQUITO_RIDGE' 'MOSS_SPRINGS'
'MOUNT_LOCKHART' 'MT._HOWARD' 'MT._RYAN' 'MT_Baldy' 'MT_ROSE_SKI_AREA'
'MUD_FLAT' 'MUNSON_RIDGE' 'MYRTLE_CREEK' 'Marten_Ridge' 'McNeil_River_SGS'
'Med_Bow' 'Merritt_Mountain' 'Midas' 'Mormon_Mountain' 'Mt_Pennell'
'NAVAJO_WHISKEY_CK' 'NEVADA_RIDGE' 'NUKA_GLACIER' 'OAK_CREEK' 'PALO'
'PARADISE' 'PARK_CONE' 'PARK_CREEK_RIDGE' 'PARK_RESERVOIR'
'PARLEYS_SUMMIT' 'PARRISH_CREEK' 'PAYSON_R.S.' 'PHANTOM_VALLEY'
'PICKLE_KEG' 'PIERCE_R.S.' 'PINE_CREEK' 'POCKET_CREEK' 'POISON_FLAT'
'POLE_CREEK_R.S.' 'PORPHYRY_CREEK' 'PORT_GRAHAM' 'PRUDHOE_BAY'
'Panguitch_Lake_RS' 'Pole_Canyon' 'QUARTZ_MOUNTAIN' 'QUARTZ_PEAK'
'Quemazon' 'RAGGED_MOUNTAIN' 'RAINBOW_CANYON' 'RAINY_PASS'
'RED_PINE_RIDGE' 'RED_RIVER_PASS_#2' 'REYNOLDS_CREEK'
'ROCKY_BASIN-SETTLEME' 'ROCKY_POINT' 'ROCK_CREEK' 'ROCK_SPRINGS'
'ROUGH_AND_TUMBLE' 'RUBICON_#2' 'Redden_Mine_Lwr' 'Rees_Flat'
'Rio_Santa_Barbara' 'SAGE_CREEK_BASIN' 'SALMON_MEADOWS' 'SALT_CREEK_FALLS'
'SALT_RIVER_SUMMIT' 'SASSE_RIDGE' 'SAVAGE_PASS' 'SCHNEIDER_MEADOWS'
'SCHOFIELD_PASS' 'SEELEY_CREEK' 'SENTINEL_BUTTE' 'SEVENTYSIX_CREEK'
'SHANGHI_SUMMIT' 'SHARKSTOOTH' 'SHEEP_MTN.' 'SHUREE' 'SIERRA_BLANCA'
'SILVER_CREEK' 'SILVIES' 'SLEEPING_WOMAN' 'SLUMGULLION' 'SMILEY_MOUNTAIN'
'SMITH_and_MOREHOUSE' 'SNAKE_RIVER_STATION' 'SNOWBIRD' 'SNOW_MOUNTAIN'
'SOLDIER_PARK' 'SOMSEN_RANCH' 'SONORA_PASS' 'SOURDOUGH_GULCH' 'SOUTH_MTN.'
'SPIRIT_LK' 'SPRATT_CREEK' 'SPUR_PARK' 'SQUAW_SPRINGS' 'SQUAW_VALLEY_G.C.'
'STEEL_CREEK_PARK' 'STRAWBERRY_DIVIDE' 'SUCKER_CREEK' 'SUMMIT_CREEK'
'SUMMIT_LK' 'SUMMIT_MEADOW' 'SUMMIT_RANCH' 'SUSITNA_VALLEY_HIGH'
'SWEDE_PEAK' 'Santa_Fe' 'Sawtooth' 'Senorita_Divide_#2' 'Sherwin'
'Silver_Creek_Nv' 'Snowstorm_Mtn' 'Stag_Mountain' 'State_Line'
'Sunflower_Flat' 'Suu_Ranch' 'TAHOE_CITY_CROSS' 'TAOS_POWDERHORN'
'TAYLOR_BUTTE' 'TAYLOR_CANYON' 'TEMPLE_FORK' 'THAYNES_CANYON' 'TIMBERLINE'
'TIMPANOGOS_DIVIDE' 'TIPTON' 'TOE_JAM' 'TOGWOTEE_PASS' 'TOKOSITNA_VALLEY'
'TONY_GROVE_LAKE' 'TONY_GROVE_RS' 'TOUCHET' 'TOWNSEND_CREEK' 'TRIAL_LAKE'
'TROUGH' 'TROUT_CREEK' 'TRUCKEE_#2' 'Takka_Wiiya' 'Tent_Mtn_Lower'
'Thistle_Flat' 'Thumb_Divide' 'Tres_Ritos' 'UPPER_NOME_CREEK'
'UPPER_RIO_GRANDE' 'UPPER_SAN_JUAN' 'UPPER_TAYLOR' 'UPPER_TSAINA_RIVER'
'USU_DOC_DANIEL' 'Upper_Joes_Valley' 'VACARRO_SPRING' 'VAN_WYCK'
'VERNON_CREEK' 'VIRGINIA_LAKES_RIDGE' 'Vacas_Locas' 'WARD_CREEK_#3'
'WARD_MOUNTAIN' 'WATERHOLE' 'WEBSTER_FLAT' 'WESNER_SPRINGS'
'WEST_YELLOWSTONE' 'WHEELER_PEAK' 'WHISKEY_CK' 'WHITE_HORSE_LAKE'
'WHITE_MILL' 'WHITE_RIVER_#1' 'WIDTSOE_#3' 'WILDHORSE_DIVIDE' 'WILD_BASIN'
'WILSON_CREEK' 'WINDY_PEAK' 'WOLF_CREEK_SUMMIT' 'White_River_Nv'
'Wrigley_Creek' 'Yankee_Reservoir' 'ZIRKEL']
```

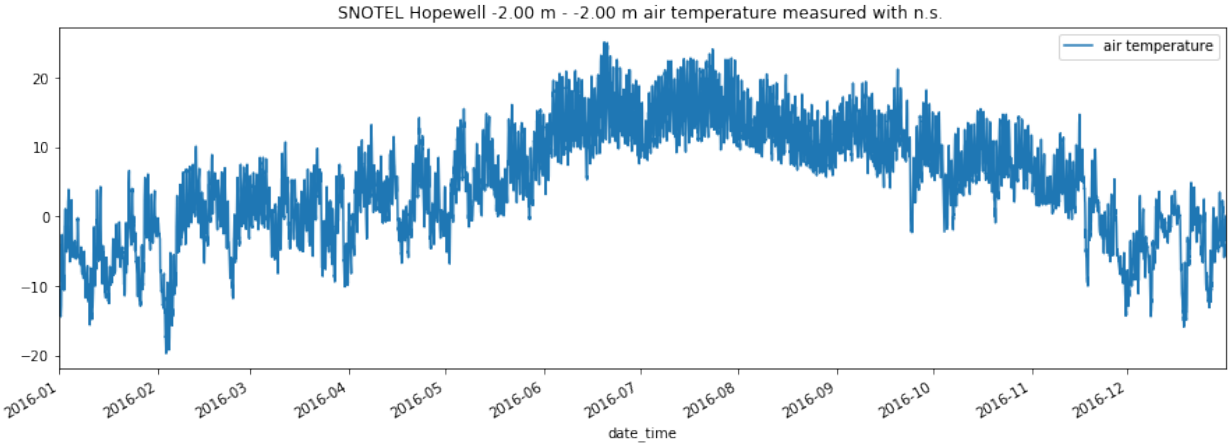
```
station = random.choice(stations)
station_obj = ISMN_reader.get_station(station)
print "Available Variables at Station %s"%station
#get the variables that this station measures
variables = station_obj.get_variables()
print variables
```

```
Available Variables at Station Hopewell
['air temperature' 'snow depth' 'snow water equivalent' 'soil moisture'
 'soil temperature']
```

```
#to make sure the selected variable is not measured
#by different sensors at the same depths
#we also select the first depth and the first sensor
#even if there is only one
depths_from,depths_to = station_obj.get_depths(variables[0])

sensors = station_obj.get_sensors(variables[0],depths_from[0],depths_to[0])

#read the data of the variable, depth, sensor combination
time_series = station_obj.read_variable(variables[0],depth_from=depths_from[0],depth_
↪to=depths_to[0],sensor=sensors[0])

#print information about the selected time series
print "Selected time series is:"
print time_series
```

```
Selected time series is:
SNOTEL Hopewell -2.00 m - -2.00 m air temperature measured with n.s.
```

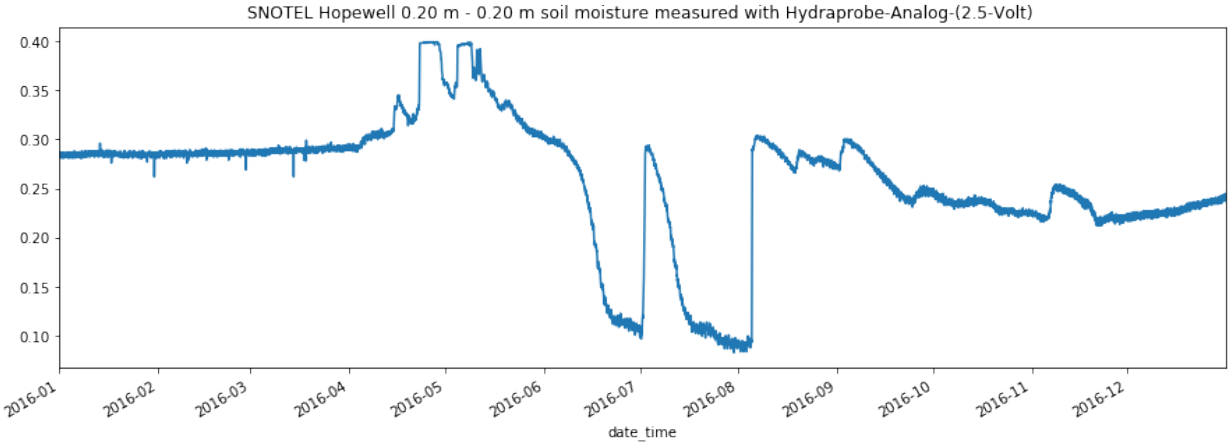
```
#plot the data
time_series.plot()
plt.legend()
plt.show()
```




```
#we also want to see soil moisture
sm_depht_from,sm_depht_to = station_obj.get_depths('soil moisture')
print sm_depht_from,sm_depht_to
```

```
[ 0.2   0.51  0.05] [ 0.2   0.51  0.05]
```

```
#read sm data measured in first layer 0.2-0.2m
sm = station_obj.read_variable('soil moisture',depth_from=0.2,depth_to=0.2)
sm.plot()
plt.show()
```




```
# the data attribute is a pandas.DataFrame
time_series.data
```

Selection of ISMN stations by landcover or climate classification:

```
# Return all available landcover classifications (ESA CCI landcover 2000) for the␣
↪variable soil moisture.
# To use ESA CCI landcover data from the year 2005 or 2010 set landcover parameter to
↪'landcover_2005' and
# 'landcover_2010', respectively.
lc_2000 = ISMN_reader.get_landcover_types(variable='soil moisture', landcover=
↪'landcover_2000')

# return all available landcover classifications (ESA CCI landcover 2005) for the␣
↪variable soil moisture
# (depths from 0 to 0.1m)
lc_2005 = ISMN_reader.get_landcover_types(variable='soil moisture', landcover=
↪'landcover_2005'
                                          min_depth=0, max_depth=0.1)
# return all available landcover classifications (ESA CCI landcover 2010) for the␣
↪variable soil moisture
# (depths from 0.1 to 0.5m)
lc_2010 = ISMN_reader.get_landcover_types(variable='soil moisture', landcover=
↪'landcover_2010'
                                          min_depth=0.1, max_depth=0.5)
# return all available landcover classifications (in situ) for the variable soil␣
↪moisture
lc_insitu = ISMN_reader.get_landcover_types(variable='soil moisture', landcover=
↪'landcover_insitu')

# return all available climate classifications (Koeppen-Geiger 2007) for the variable␣
↪soil moisture
clim = ISMN_reader.get_climate_types(variable='soil moisture', climate='climate')
```

(continues on next page)

(continued from previous page)

```
# return all available climate classifications (in situ) for the variable soil␣
↪moisture
clim_insitu = ISMN_reader.get_climate_types(variable='soil moisture', climate=
↪'climate_insitu')


# print all landcover classes covered by the ESA CCI landcover classification
ISMN_reader.print_landcover_dict()
# print all climate classes covered by the Koeppen-Geiger classification
ISMN_reader.print_climate_dict()


# Select ISMN stations where soil moisture at depths from 0 to 0.1m is available and␣
↪the landcover
# classification is equal to 130 (Grassland). In this example the ESA CCI landcover␣
↪classification
# for the year 2010 (landcover_2010) is used.
ids1 = ISMN_reader.get_dataset_ids(variable='soil moisture', min_depth=0, max_depth=0.
↪1, landcover_2010=130)
# read time series from first element in the returned list
ts_1 = ISMN_reader.read_ts(ids1[0])

# Select ISMN stations where soil moisture at depths from 0 to 0.1m is available, the␣
↪landcover
# class (year 2005) is equal to 130 (Grassland) and the climate class is equal to Csa␣
↪(Temperate
# Dry Hot Summer)
ids2 = ISMN_reader.get_dataset_ids(variable='soil moisture', min_depth=0, max_depth=1,
                                   landcover_2005=130, climate='Csa')
# read time series from first element in the returned list
ts_2 = ISMN_reader.read_ts(ids2[0])
```

CHAPTER

SEVEN

CONTENTS

7.1 License

7.2 Contributors

- Christoph Paulik <christoph.paulik@geo.tuwien.ac.at>
- Irene Himmelbauer <irene.himmelbauer@geo.tuwien.ac.at>
- Luca Zappa <luca.zappa@geo.tuwien.ac.at>
- Philip Buttinger <philip.buttinger@geo.tuwien.ac.at>

7.3 ismn

7.3.1 ismn package

Submodules

ismn.interface module

ismn.metadata_collector module

ismn.readers module

Module contents

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

i

INDEX