FIXME

Release v0.1.0

Remi Gau

Jun 02, 2023

CONTENT

This repository was created using the cookiecutter template. There may be some unused files and folders left over from the template.

CHAPTER

ONE

INVERSE STREAMFLOW ROUTING

Inverse streamflow routing (ISR) uses a flow direction map and time series of discharge measurements at points along the river to estimate runoff throughout the river basin.

1.1 Contents

Scripts: Workflows for ISR Functions: Main and secondary functions for performing ISR and evaluating the results.

1.2 References

- Pan, M., & Wood, E. F. (2013). Inverse streamflow routing. Hydrology and Earth System Sciences, 17(11), 4577–4588. https://doi.org/10.5194/hess-17-4577-2013
- Fisher, C. K., Pan, M., & Wood, E. F. (2020). Spatiotemporal assimilation-interpolation of discharge records through inverse streamflow routing. Hydrology and Earth System Sciences, 24(1), 293–305. https://doi.org/10.5194/hess-24-293-2020
- Yang, Y., Lin, P., Fisher, C. K., Turmon, M., Hobbs, J., Emery, C. M., … Pan, M. (2019). Enhancing SWOT discharge assimilation through spatiotemporal correlations. Remote Sensing of Environment, 234(September), 111450. https://doi.org/10.1016/j.rse.2019.111450

Fisher et al., 2021, HESS

CHAPTER

TWO

FUNCTION DESCRIPTION

List of functions in the `src` folder.

`src.`**`my_fibonacci`**`(`*varargin*`)`

Returns vector of n iterations of the Fibonacci sequence.

USAGE:

```
results = my_fibonacci(nb_iterations)
```

Parameters
`foo` (`positive integer`) – Optional argument. Number of iterations to run. Default = 5;

Returns

- **results**
  (array) (1 x nb_iterations + 2)

Example:

```
results = my_fibonacci(5);
```

2.1 Utilities

`src.utils.`**`is_octave`**`()`

Returns true if the environment is Octave.

USAGE:

```
retval = is_octave()
```

`src.utils.`**`root_dir`**`()`

Returns fullpath the root of the repository.

USAGE:

```
retval = root_dir()
```

Returns

- **root_dir**
  (path)

`src.utils.`**`get_version`**`()`

Reads the version number of the pipeline from the txt file in the root of the repository.

USAGE:

```
version_number = get_version()
```

Returns

- **version_number**
  (string) Use semantic versioning format (like v0.1.0)

CHAPTER

THREE

DOCUMENTATION STYLES

You can choose different ways of documenting the help section of your code.

Those are adapted from their equivalent in python.

Those functions can be found here

—

src.**count_line_google_style_help**(*file*, *line*)

Counts the number of times a line occurs. Case-sensitive. White space padding are ignored.

USAGE:

```
num_instances = count_line_google_style_help(file, line)
```

Arguments:

file (cellstr): content of file to scan

line (char): the line to count

Returns:

num_instances (int): the number of times the line occurs.

—

src.**count_line_numpy_style_help**(*file*, *line*)

Counts the number of times a line occurs. Case-sensitive. White space padding are ignored.

USAGE:

```
num_instances = count_line_google_style_help(file, line)
```

Parameters

file: cellstr

content of file to scan

line: char

the line to count

Returns

num_instances: int

the number of times the line occurs.

—

src.**count_line_rst_style_help**(*file*, *line*)

Counts the number of times a line occurs. Case-sensitive. White space padding are ignored.

USAGE:

```
num_instances = count_line_google_style_help(file, line)
```

Parameters

- **file** – content of file to scan
- **line** – the line to count

Returns

- **num_instances**
  (int) the number of times the line occurs.

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

MATLAB MODULE INDEX