FuzzyCorr

Release 0.0.1

Beatriz Negreiros

Nov 09, 2020

CONTENTS

This repository contains the work developed for a Master Thesis on fuzzy map comparison methods to evaluate the performance of hydro-morphodynamic numerical models. Please read the License terms for code usage and re-distribution.

Sediment transport and hydraulic processes can be reproduced with numerical models such as SSIIMM, Hydro_AS_2D, TELEMAC and many more. The accuracy of numerical models is assessed through comparing the simulated and the observed datasets, which constitutes a model validation. With the purpose of analyzing simulated and observed bed elevation change, two methods of comparison can be applied:

1. Comparison via statistical methods such as RMSE (Root Mean Squared Error) or visual human comparison. However, local measures of similarity (or a similarity) like the RMSE are very sensible to uncertainty of location and amount, thus indicating low agreement even when overall patterns were adequately simulated.
2. Visual comparison captures global similarity, which is one of the reasons why modelers often use it for model validation. Humans are capable of finding patterns without deliberately trying, and therefore, this type of comparison provides substantial advantages over local similarity measures. Nevertheless, more research has to be done to implement automated validation tools that emulate human thinking. This is necessary because human comparison is not transparent, prone to subjective interpretations, time consuming, and hardly reproducible.

In this context, the concept of fuzzy set theory has capacities to consider similarity of spatial pattern analogous to human thinking. For instance, fuzziness of location introduces a tolerance regarding spatial uncertainty in the results of hydro-morphodynamic models. To this end, fuzzy logic enables an objective validation of such models by overcoming uncertainties in the model structure, parameters and input data.

The algorithms provided with `fuzzycorr` address the necessity in evaluating (or validating) model performance through the use of fuzzy map comparison. Future developments aim to go beyond a one-way validation towards a two-way communication between the validation algorithms and the models. The two-way communication represents a feedback loop that will eventually enable an automated calibration of numerical hydro-morphodynamic models.

CHAPTER

ONE

INSTALL DEPENDENCIES

The necessary modules for running this repo are specified in `environment.yml`. To install all packages in the environment:

- Navigate (`cd`) with the Anaconda Prompt through your directories to the `.yml` file
- Type `conda env create -f environment.yml`
- Active the new environment with `conda activate env-fuzzycorr`

CHAPTER

TWO

USAGE

The best way to learn the usage is by examples. In the directory `examples`, the usage of the modules are demonstrated in a case study. Inside the folder `salzach_case`, the results from a hydro-morphodynamic numerical simulation (i.e., simulated bed elevation change, deltaZ) are located in `raw_data`. For more details on the hydro-morphodynamic numerical refer to Beckers et al. (2020).

- `prepro_salzach.py`: example of the usage of the class `PreProFuzzy` of the module `prepro.py`, where vector data is interpolated and rasterized.
- `classification_salzach.py`: example of the usage of the class `PreProCategorization` of the module `prepro.py`.
- `fuzzycomparison_salzach.py`: example of the usage of the class `FuzzyComparison` of the module `fuzzycomp.py`, which creates a correlation (similarity) measure between simulated and observed datasets.
- `plot_salzach.py`, `plot_class_rasters.py` and `performance_salzach`: example of the usage of the module `plotter.py`.
- `random_map`: example of generating a raster following a uniform random distribution, which uses the module `prepro.py`.

CHAPTER

THREE

STRUCTURE

This package contains the following modules, which were designed in *Python 3.6*:

- `prepro.py`: Includes functions for reading, normalizing and rasterizing vector data. These are preprocessing steps for fuzzy map comparison (module fuzzycomp).
- `fuzzycomp.py`: Provides routines for fuzzy map comparison in continuous valued rasters. The reader is referred to Hagen(2006) for more details (more to come).
- `plotter.py`: Visualization routines for output and input rasters.
- The package documentation is located in the folder `docs`.

class prepro.**PreProCategorization**(*raster*)
Structured for ... (UNCLEAR)

Parameters **raster** – string, path of the raster to be categorized

categorize_raster(*class_bins*, *map_out*, *save_ascii=True*)
Classifies the raster according to the classification bins

Parameters

- **map_out** – path of the project directory
- **class_bins** – list of floats
- **save_ascii** – bool

Returns saves the classified raster in the chosen directory

nb_classes(*n_classes*)
Generates class bins based on the Natural Breaks method

Parameters **n_classes** – integer, number of classes

Returns list of optimized bins

class prepro.**PreProFuzzy**(*df*, *attribute*, *crs*, *nodatavalue*, *res=None*, *ulc=(numpy.nan, numpy.nan)*, *lrc=(numpy.nan, numpy.nan)*)
Parent pre-processing structure for the comparison of numeric maps

Parameters

- **pd** – pandas dataframe, can be obtained by reading the textfile as pandas dataframe
- **attribute** – string, name of the attribute to burn in the raster (ex.: deltaZ, Z)
- **crs** – string, coordinate reference system
- **nodatavalue** – float, value to indicate nodata cells
- **res** – float, resolution of the cell (cell size), is the same for x and y

- **ulc** – tuple of floats, upper left corner coordinate, optional
- **lrc** – tuple of floats, lower right corner coordinate, optional

array2raster(*array*, *raster_file*, *save_ascii=True*)
Saves a raster using interpolation

Parameters

- **raster_file** – string, path to save the rasterfile
- **save_ascii** – boolean, true to save also an ascii raster

Returns saves the raster with the selected filename

create_polygon(*shape_polygon*, *alpha=numpy.nan*)
Creates a polygon surrounding a cloud of shapepoints

Parameters

- **shape_polygon** – string, path to save the shapefile
- **alpha** – float, excentricity of the alphashape (polygon) to be created

Returns saves the polygon (*.shp) with the selected filename

norm_array(*method='linear'*)
Normalizes the raw data in equally distanced points depending on the selected resolution

Returns interpolated and normalized array with selected resolution

---

Hint: Read more at https://github.com/rosskush/skspatial

---

plain_raster(*shapefile*, *raster_file*, *res*)
Converts a shapefile(.shp) to a GeoTIFF raster without normalizing

Parameters

- **shapefile** – string, filename with path of the input shapefile (*.shp)
- **raster_file** – string, filename with path of the output raster (*.tif)
- **res** – float, resolution of the cell

Returns saves the raster in the default directory

points_to_grid()
Creates a grid of new points in the target resolution

Returns array of size nrow, ncol

Hints: Read more at http://chris35wills.github.io/gridding_data/

random_raster(*raster_file*, *save_ascii=True*, ***kwargs*)
Creates a raster of randomly generated values

Keyword Arguments **minmax** – tuple of floats, (zmin, zmax) min and max ranges for random values

Returns array of random values within a range of the same size and chape as the original

prepro.**clip_raster**(*polygon*, *in_raster*, *out_raster*)
Clips a raster based on the given polygon

Parameters

- **polygon** – string, file with path of the polygon (*.shp)
- **in_raster** – string, file and path of the input raster (*.tif)
- **in_raster** – string, file and path of the output raster (*.tif)

Returns no output, saves the raster (*.tif) with the selected filename

class fuzzycomp.**FuzzyComparison**(*rasterA*, *rasterB*, *neigh=4*, *halving_distance=2*)

Performing fuzzy map comparison :param rasterA: string, path of the raster to be compared with rasterB :param rasterB: string, path of the raster to be compared with rasterA :param neigh: integer, neighborhood being considered (number of cells from the central cell), default is 4 :param halving_distance: integer, distance (in cells) to which the membership decays to its half, default is 2

fuzzy_numerical(*comparison_name*, *save_dir*, *map_of_comparison=True*)

Compares a pair of raster maps using fuzzy numerical spatial comparison

Parameters

- **save_dir** – string, directory where to save the results
- **comparison_name** – string, name of the comparison
- **map_of_comparison** – boolean, create map of comparison in the project directory if True

Returns Global Fuzzy Similarity and comparison map

fuzzy_rmse(*comparison_name*, *save_dir*, *map_of_comparison=True*)

Compares a pair of raster maps using fuzzy root mean square error as spatial comparison

Parameters

- **comparison_name** – string, name of the comparison
- **save_dir** – string, directory where to save the results of the map comparison
- **map_of_comparison** – boolean, if True it creates map of of local squared errors (in the project directory)

Returns global fuzzy RMSE and comparison map

neighbours(*array*, *x*, *y*)

Captures the neighbours and their memberships :param array: array A or B :param x: int, cell in x :param y: int, cell in y :return: np.array (float) membership of the neighbours (without mask), np.array (float) neighbours' cells (without mask)

save_comparison_raster(*array_local_measures*, *dir*, *file_name*)

Create map of comparison

save_results(*measure*, *dir*, *name*)

Saves a results file

fuzzycomp.**f_similarity**(*centrall_cell*, *neighbours*)

Calculates the similarity function for each pair of values (fuzzy numerical method)

Parameters

- **centrall_cell** – float, cell under analysis in map A
- **neighbours** – np.array of floats, neighbours in map B

Returns np.array of floats, each similarity between each of two cells

fuzzycomp.**jaccard**(*a*, *b*)

Creates a . . .

Parameters

- **a** (*float*) –
- **b** (*float*) –

Returns jac

Return type float

fuzzycomp.**squared_error**(*centrall_cell*, *neighbours*)
Calculates the error measure fuzzy rmse

Parameters

- **centrall_cell** – float, cell under analysis in map A
- **neighbours** – np.array of floats, neighbours in map B

Returns np.array of floats, each similarity between each of two cells

class plotter.**RasterDataPlotter**(*path*)
Class of raster for plotting

Parameters **path** – string, path of the raster to be plotted

make_hist(*legendx*, *legendy*, *fontsize*, *output_file*, *figsize*, *set_ylim=None*, *set_xlim=None*)
Creates a histogram of numerical raster

Parameters

- **legendx** – string, legend of the x axis of he histogram
- **legendy** – string, legend of the y axis of he histogram
- **fontsize** – integer, size of the font
- **output_file** – string, path for the output file
- **figsize** – tuple of integers, size of the width x height of the figure
- **set_ylim** – float, set the maximum limit of the y axis
- **set_ylim** – float, set the maximum limit of the x axis

Returns saves the figure of the histogram

plot_categorical_raster(*output_file*, *labels*, *cmap*, *box=True*)
Creates a figure of a categorical raster

Parameters

- **output_file** – path, file path of the figure
- **labels** – list of strings, labels (i.e., titles)for the categories
- **cmap** – string, colormap to plot the raster
- **box** – boolean, if False it sets off the frame of the picture

Returns saves the figure of the raster

plot_categorical_w_window(*output_file*, *labels*, *cmap*, *xy*, *width*, *height*, *box=True*)
Creates a figure of a categorical raster with a zoomed window

Parameters

- **output_file** – path, file path of the figure
- **labels** – list of strings, labels (i.e., titles)for the categories

- **cmap** – string, colormap to plot the raster
- **xy** – tuple (x,y), origin of the zoomed window, the upper left corner
- **width** – integer, width (number of cells) of the zoomed window
- **height** – integer, height (number of cells) of the zoomed window

Returns saves the figure of the raster

plot_continuous_raster(*output_file*, *cmap*, *vmax=numpy.nan*, *vmin=numpy.nan*, *box=True*)

Creates a figure of a continuous valued raster

Parameters

- **output_file** – path, file path of the figure
- **cmap** – string, colormap to plot the raster
- **vmax** – float, optional, value maximum of the scale, this value is used in the normalization of the colormap
- **vmin** – float, optional, value minimum of the scale, this value is used in the normalization of the colormap
- **box** – boolean, if False it sets off the frame of the picture

Returns saves the figure of the raster

plot_continuous_w_window(*output_file*, *xy*, *width*, *height*, *bounds*, *cmap=None*, *list_colors=None*)

Create a figure of a raster with a zoomed window :param output_file: path, file path of the figure :param xy: tuple (x,y), origin of the zoomed window, the upper left corner :param width: integer, width (number of cells) of the zoomed window :param height: integer, height (number of cells) of the zoomed window :param bounds: list of float, limits for each color of the colormap :param cmap: string, optional, colormap to plot the raster :param list_colors: list of colors (str), optional, as alternative to using a colormap :returns: saves the figure of the raster

plotter.**read_raster**(*path*)

Opens a raster

Parameters **path** (*str*) – directory and name of a raster

Returns a numpy array of the raster

Return type ndarray

3.1 References

- Ross Kushnereit
- Chris Wills

CHAPTER

FOUR

DISCLAIMER AND LICENSE

4.1 Disclaimer (general)

No warranty is expressed or implied regarding the usefulness or completeness of the information provided for *fuzzycorr* and its documentation. References to commercial products do not imply endorsement by the Authors of *fuzzycorr*. The concepts, materials, and methods used in the codes and described in the docs are for informational purposes only. The Authors have made substantial effort to ensure the accuracy of the code and the docs and the Authors shall not be held liable, nor their employers or funding sponsors, for calculations and/or decisions made on the basis of application of *fuzzycorr*. The information is provided "as is" and anyone who chooses to use the information is responsible for her or his own choices as to what to do with the code, docs, and data and the individual is responsible for the results that follow from their decisions.

4.2 BSD 3-Clause License

PYTHON MODULE INDEX

f

p

INDEX