A quick-start to calculating catchmentwide erosion rates

Copy this jupyter notebook together with the folder `test_data` to any location on your computer.
Note that a folder `plots` will be created in your working directory for graphical output.

In [1]:
```
# Install the riversand package if you haven't done so,
# or upgrade to the latest version:
#%pip install riversand
#%pip install --upgrade riversand
```

Workflow

The basic workflow consists of two steps: (1) Import and validate all input data; (2) calculate the catchmentwide erosion rates.

(1) Import and validate input data:

Raster data

You can add three types of raster datasets (geotiffs):

- `'elevation'` is a digital elevation model in meters above sea level
- `'shielding'` is a shielding factor between 0 and 1 (e.g. topographic shielding calculated with the TopoToolbox toposhielding function)
- `'quartz'` is a binary raster with 1 indicating quartz-bearing and 0 indicating quartz-free lithologies

All raster data must have the same projection and resolution. Projection must be equal area (e.g. UTM), geographic coordinate reference systems (lat/long) are not permitted. An elevation raster is mandatory.

Sample data

Sample and nuclide information can be added manually or imported from a spreadsheet. The requirements are those for the online calculator (see [http://stoneage.hzdr.de/docs/documentation.html#input_format (http://stoneage.hzdr.de/docs/documentation.html#input_format)](http://stoneage.hzdr.de/docs/documentation.html#input_format)).

Entries that are recognized (processed) by the calculator are `name` , `press_flag` , `thickness` , `density` , `shielding` , `erate` , `year` , `nuclide` , `mineral` , `N` , `delN` and `standardization` ; any additional information is ignored. A detailed description of input sample data is given at the end of this notebook.

Catchment shapefile

Catchments are imported from a polygon shapefile with one or several catchment outlines. The shapefile must have the same projection as the raster data. If the shapefile has more than one catchment polygon, the attribute table must include a field with unique catchment names that are used to match catchments to samples.

Input data validation

Use the function `.validate()` to verify projection and resolution of geospatial data and to ensure that the sample data is valid.

(2) Calculate catchmentwide erosion rates:

There are two functions `.process_single_catchment()` and `.process_multi_catchment()` to calculate erosion rates for single-catchment and multi-catchment datasets, respectively. In addition, the function `.catchment_stats()` calculates some basic catchment statistics such as area, mean elevation, relief, etc.

Example 1: Single catchment

In [2]:
```
import pandas as pd
import numpy as np

import riversand

# Create a new 'Riversand' object and specify the folder with the input data:
rv = riversand.Riversand("test_data") # subfolder in the working directory

# Add raster data:
rv.add_raster('dem_utm_35m.tif', dtype='elevation')
rv.add_raster('toposhielding_35m.tif', dtype='shielding') # optional
#rv.add_raster('quartz_35m.tif', dtype='quartz') # optional
#rv.quartz = None # remove the quartz raster

# Add sample data manually; see multi-catchment example for how to upload data from
test = {'name'    : 'Ph-1',    # sample name
        'N'       : 1.2e6,     # nuclide concentration, at/g
        'delN'    : 3.6e4,     # uncertainty, at/g
        'nuclide': 'Be-10',   # nuclide 'Be-10' or 'Al-26'
        'density': 2.6,       # density, g/cm3
       }
rv.add_samples(test) # add the sample to the project

# Add catchment shapefile with a single catchment polygon:
rv.add_catchments('test_single_catchment.shp')
```

The the previous cell may result in error messages, e.g. if a file does not exist or cannot be read.

Type `rv` to summarize the data that has been imported (or `rv.samples`, `rv.catchments`, `rv.elevation`, `rv.shielding`, `rv.quartz` for the individual data sets). In addition, the function `riversand.plot_raster()` can be used to quickly display the raster data.

(Use `?riversand.plot_raster` for help.)

In [3]:
```
#riversand.plot_raster(rv) # defaults to elevation raster
#riversand.plot_raster(rv, dtype='shielding') # specify which raster to plot
#riversand.plot_raster(rv, dtype='shielding', fname="my_image.jpg") # save image to
```

In [4]:
```
rv.samples # display sample data; note that default values exist for data that are
```

Out[4]:

	name	press_flag	thickness	density	shielding	erate	year	nuclide	mineral	N	delN	standardizatio
0	Ph-1	std	0	2.6	1.0	0	2010	Be-10	quartz	1200000	36000	07KNSTI

The function `.validate()` verifies if all geospatial datasets have matching projections and resolutions. It also sets several parameters such as `.epsg`, `.crs` and `.res` that reflect the projection and resolution of the geospatial datasets.

In [5]:
```
# Validate the dataset:
rv.validate()
```

```
Raster data valid
Sample data valid
Catchment data valid
```

Processing

Continue only if the dataset is validated (function `.validate()`).

Use the function `.process_single_catchment()` if the shapefile contains only a single polygon; sample or catchment names are ignored.

The following parameters need to be specified:

- `bins` : bin size for elevation statistics in metres, e.g. `bins=100`
- `scaling` : scaling method as implemented in the online calculator: `'St'` , `'Lm'` or `'LSDn'`
- `shielding` : method for shielding correction:
    - `'topo'` : compute from shielding raster; raises an error if no shielding raster is defined.
    - `'sample'` : use value from sample dataset; raises an error if no shielding is defined for the sample.
    - `numeric` : use a constant value between 0 and 1.

Optional parameters:

- `plot` : `'jpg'` or `'png'` to save plots with an automatically generated name or `'show'` to display the plots.
- `unit` : unit of erosion rates for display; `riversand.params.units` shows all valid options.

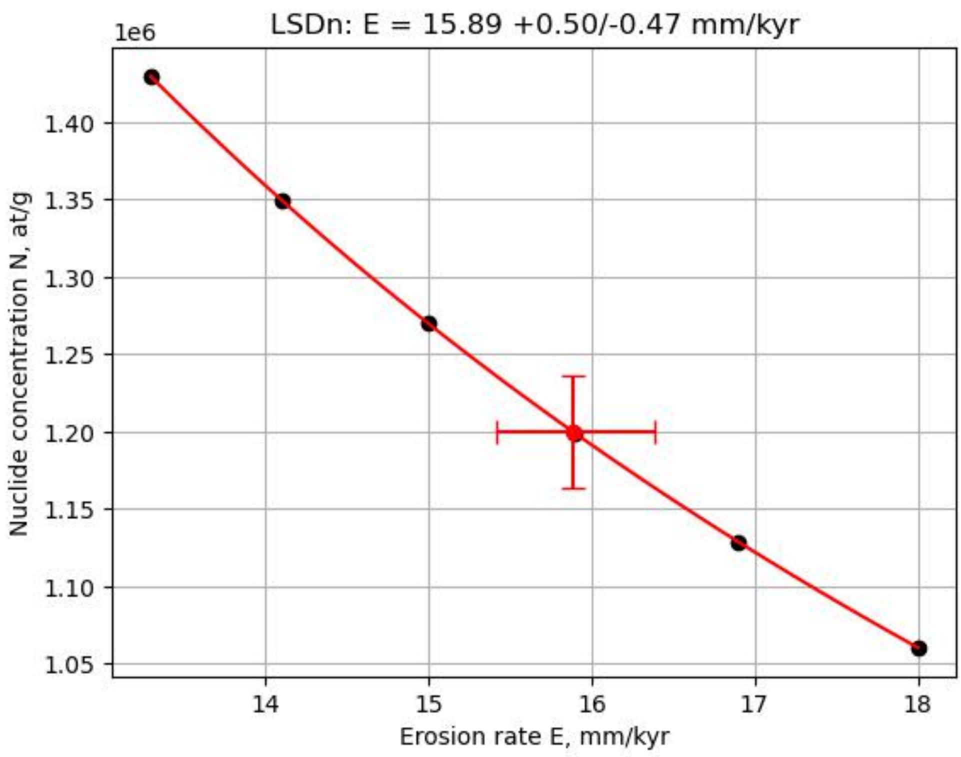
Plots are saved in a folder `plots` in the current working directory. Existing files with identical names will be overwritten.

In [6]:
```
#riversand.params.units
```

In [7]:
```
# Process a single catchment:
results1 = rv.process_single_catchment(bins=100, # bin size in meters
                                       scaling='LSDn', # 'St', 'Lm', 'LSDn'
                                       shielding='topo', # 'topo', 'sample' or nume
                                       plot='jpg', # 'jpg' or 'png'
                                       unit='mm/kyr') # units for plotting
```

```
Processing single catchment
Bin size : 100 m
Scaling method : LSDn
Topographic shielding from topo data
Saving plots as .jpg in 'plots'

Processing finished.
```



If the parameter `plot` is set a **figure** is saved as `./plots/0_Ph-1_LSDn.jpg` :

- The black points are six initial erosion rates E sent to the online calculator and the nuclide concentrations $N(E)$ predicted by the online calculator.
- The red curve is a polynomial fit $N(E)$.
- The red point and vertical error bar are the input nuclide concentration and uncertainty of the sample $N \pm delN$.
- The horizontal error bar is the uncertainty on the erosion rate resulting from the analytical uncertainty $delN$; it depends on the shape of the function $N(E)$ and is therefore asymmetric.

Note that the units for this plot are defined by the parameter `unit` .

Results table:

- `E` , `delE-` , `delE+` : erosion rate and uncertainty in cm/yr
- `NRMSE` : normalised root mean squared error
- `Tavg` : averaging time scale in yr

Use `.to_excel()` to save the results to Excel or OpenOffice or `.to_csv()` for comma-separated values. This may require additional packages such as `openpyxl` or `odfpy` .

In [8]:
```
results1 # display results (cm/yr)
```

Out[8]:

	name	scaling	nuclide	qtz	E	delE-	delE+	NRMSE	Tavg	error
0	Ph-1	LSDn	Be-10 quartz	100	0.001589	0.000047	0.00005	0.000016	38735	

In [9]:
```
results1.to_excel('Example1_LSDn.xlsx') # save data to spreadsheet
```

Calculate some **catchment statistics** for this catchment. Note that in this example the catchment name (`.cid`) has not been specified (see below).

In [10]:
```
stats1 = rv.catchment_stats()
stats1 # display catchment statistics
```

```
Processing  finished.
```

Out[10]:

	name	centr_lat	centr_long	mean_elev	stdev_elev	relief	area	mean_sf	qtz_pc
0		45.59974	7.34807	2453.7	561.7	3382.0	257.6	0.92309	NaN

Example 2: Same catchment, different settings

Repeat the analysis with a **correction for quartz-free lithologies** and with a **constant shielding factor** of 0.95. The previous plots will be overwritten.

In [11]:
```
# Add a quartz raster dataset:
rv.add_raster('quartz_35m.tif', dtype='quartz')
rv.validate()
```

```
Raster data valid
Sample data valid
Catchment data valid
```

In [12]:
```python
# Process with a constant shielding factor 'shielding=0.95':
results2 = rv.process_single_catchment(bins=100, # bin size in meters
                                       scaling='LSDn', # 'St', 'Lm' or 'LSDn'
                                       shielding=0.95, # 'topo', 'sample' or numeri
                                       plot='jpg', # 'jpg' or 'png'
                                       unit='mm/kyr') # units for plotting
results2 # display results (cm/yr)
```

```
Processing single catchment
Bin size : 100 m
Scaling method : LSDn
Topographic shielding : 0.95
Correcting for quartz-free lithologies
Saving plots as .jpg in 'plots'

Processing finished.
```

Out[12]:

	name	scaling	nuclide	qtz	E	delE-	delE+	NRMSE	Tavg	error
0	Ph-1	LSDn	Be-10 quartz	38.1	0.001348	0.00004	0.000042	0.000114	45635	

Example 3: Multi-catchment dataset

To process a shapefile with more than one catchment polygon, import sample data from a spreasdsheet. The spreadsheet must have a column `name` with sample names that are used to match samples to catchment polygons. The shapefile must have an attribute field with the sample names. Use the function `.set_cid()` to specify, which attribute field of the shapefile to use.

In [13]:
```python
# Create a new Riversand object:
rv = riversand.Riversand("test_data")

# Add raster data:\n",
rv.add_raster('dem_utm_35m.tif', dtype='elevation')
rv.add_raster('toposhielding_35m.tif', dtype='shielding') # optional
rv.add_raster('quartz_35m.tif', dtype='quartz') # optional

# Add sample data from a spreadsheet:
rv.add_samples('test_samples.ods') # .xlsx, .ods, .csv

# Add catchment shapefile with one or several catchment polygons:
rv.add_catchments('test_multi_catchment.shp')
# Set the catchment identifier:
rv.set_cid('name') # mandatory for shapefiles with more than one catchment

rv.validate()
```

```
Using default values for missing columns:
   press_flag, thickness, erate, year, mineral, standardization

Raster data valid
Sample data valid
Catchment data valid

Valid catchments / samples:
   Found 5 match(es)
```

In [14]:
```
rv.samples
```

Out[14]:

	name	press_flag	thickness	density	shielding	erate	year	nuclide	mineral	N	delN	standardization
0	DB01	std	0	2.7	0.92	0	2010	Be-10	quartz	12900	700	07KNSTD
1	DB02	std	0	2.7	0.94	0	2010	Be-10	quartz	10800	700	07KNSTD
2	DB03	std	0	2.7	0.94	0	2010	Be-10	quartz	23500	1400	07KNSTD
3	DB04	std	0	2.7	0.94	0	2010	Be-10	quartz	22000	1100	07KNSTD
4	DB05	std	0	2.7	0.95	0	2010	Be-10	quartz	20500	1000	07KNSTD
5	DB06	std	0	2.7	0.95	0	2010	Be-10	quartz	15400	800	07KNSTD
6	DB07	std	0	2.7	0.95	0	2010	Be-10	quartz	22500	2600	07KNSTD
7	DB08	std	0	2.7	0.96	0	2010	Be-10	quartz	48500	2100	07KNSTD
8	DB12	std	0	2.7	0.95	0	2010	Be-10	quartz	12600	800	07KNSTD
9	DB17	std	0	2.7	0.95	0	2010	Be-10	quartz	27100	1300	07KNSTD

For a multi-catchment dataset the function `.validate()` displays how many catchments with matching sample data have been found. The valid catchment names are stored in `.valid_catchments` (also function `.get_valid_catchments()`).

If the result is not what you expect the following functions / variables help trouble-shooting:

- `.catchments.get_names()` : show all catchment names; note that non-unique names are invalid
- `.samples` or `.samples.name` : display all sample names
- `.catchments.cid` : display the attribute field used for catchment names
- `.catchments.attrs` : display all available attribute fields

In [15]:
```
#rv.get_valid_catchments() # get names of all valid catchments
rv.valid_catchments # display names of valid catchments
#rv.samples # display sample data
#rv.catchments.cid # display catchment identifier
#rv.catchments.attrs # display all attribute fields of the shapefile
```

Out[15]:
```
['DB02', 'DB03', 'DB04', 'DB05', 'DB17']
```

Multi-catchment processing

The function `.process_multi_catchment()` is equivalent to the single-catchment processing but iterates over all samples and computes erosion rates for matching catchment-sample pairs.

In [16]:
```
# Process a multi-catchment dataset:
results = rv.process_multi_catchment(bins=100, # bin size in meters
                                     scaling='LSDn', # 'St', 'Lm' or 'LSDn'
                                     shielding='topo', # 'topo', 'sample' or numeri
                                     plot='jpg', # 'jpg' or 'png'
                                     unit='mm/kyr') # units for plotting
```

```
Processing multi-catchment dataset
Bin size : 100 m
Scaling method : LSDn
Topographic shielding from topo data
Correcting for quartz-free lithologies
Saving plots as .jpg in 'plots'


 0 DB01 : no catchment polygon
 1 DB02 : catchment out of bounds
 2 DB03 : 789.6+/-51.1 mm/kyr
 3 DB04 : 829.8+/-44.7 mm/kyr
 4 DB05 : 763.9+/-39.9 mm/kyr
 5 DB06 : no catchment polygon
 6 DB07 : no catchment polygon
 7 DB08 : no catchment polygon
 8 DB12 : no catchment polygon
 9 DB17 : 679.3+/-34.6 mm/kyr

Processing finished.
```

In [17]:
```
results # display results (cm/yr)
```

Out[17]:

	name	scaling	nuclide	qtz	E	delE-	delE+	NRMSE	Tavg	error
0	DB01	LSDn	Be-10 quartz	100	NaN	NaN	NaN	NaN	NaN	no catchment polygon
1	DB02	LSDn	Be-10 quartz	100	NaN	NaN	NaN	NaN	NaN	catchment out of bounds
2	DB03	LSDn	Be-10 quartz	55.4	0.07896	0.004502	0.005113	0.000159	750	
3	DB04	LSDn	Be-10 quartz	68.5	0.082983	0.004026	0.004468	0.000224	714	
4	DB05	LSDn	Be-10 quartz	38.1	0.076386	0.003608	0.003987	0.000363	775	
5	DB06	LSDn	Be-10 quartz	100	NaN	NaN	NaN	NaN	NaN	no catchment polygon
6	DB07	LSDn	Be-10 quartz	100	NaN	NaN	NaN	NaN	NaN	no catchment polygon
7	DB08	LSDn	Be-10 quartz	100	NaN	NaN	NaN	NaN	NaN	no catchment polygon
8	DB12	LSDn	Be-10 quartz	100	NaN	NaN	NaN	NaN	NaN	no catchment polygon
9	DB17	LSDn	Be-10 quartz	91.6	0.06793	0.003154	0.00346	0.000014	872	

Results table:

- `E`, `delE+` and `delE-` are the catchmentwide erosion rate and (asymmetric) uncertainty in cm/yr.
- `NRMSE` is the normalized root mean squared error; a warning is issued for samples with NRMSE>1e-3 indicating a poor fit of the polynomial function.
- `Tavg` is the averaging time scale in yr.
- The column `qtz` shows the quartz-bearing area of the catchment in percent (100% if no quartz raster was specified).
- `error` indicates errors or warnings that may have occurred during the calculation.

The table has the same ordering of the samples as the input table `rv.samples` for easy merging of the spreadsheets.

Drop empty rows with `results = results[results['E'].notna()]` if desired.

Catchment statistics (function `.catchment_stats()`) are calculated for all catchments regardless of duplicate catchment names or missing sample data, but if the shapefile has more than one catchment polygon the catchment identifier (`cid`) must be set.

In [18]:
```
results.to_excel('Example3_LSDn.xlsx') # save data to spreadsheet
```

In [19]:
```
# same table without the empty rows
results = results[results['E'].notna()]
results
```

Out[19]:

	name	scaling	nuclide	qtz	E	delE-	delE+	NRMSE	Tavg	error
2	DB03	LSDn	Be-10 quartz	55.4	0.07896	0.004502	0.005113	0.000159	750	
3	DB04	LSDn	Be-10 quartz	68.5	0.082983	0.004026	0.004468	0.000224	714	
4	DB05	LSDn	Be-10 quartz	38.1	0.076386	0.003608	0.003987	0.000363	775	
9	DB17	LSDn	Be-10 quartz	91.6	0.06793	0.003154	0.00346	0.000014	872	

In [20]:
```
stats3 = rv.catchment_stats()
stats3 = stats3[stats3['mean_elev'].notna()] # remove empty rows
stats3 # display catchment statistics
```

```
Processing DB03, DB04, DB02, DB05, DB19, DB17, DB12, DB12 finished.
```

Out[20]:

	name	centr_lat	centr_long	mean_elev	stdev_elev	relief	area	mean_sf	qtz_pc	error
1	DB03	45.55541	7.19667	2424.8	575.5	2911.0	81.9	0.92011	55.4	NaN
2	DB04	45.56224	7.15175	2397.4	590.7	2961.0	192.4	0.92023	68.5	NaN
3	DB05	45.64024	7.30834	2175.7	584.4	2818.0	98.1	0.9179	38.1	NaN
6	DB17	45.60752	7.05417	2417.7	529.1	2985.0	145.0	0.93132	91.6	NaN

In [21]:
```
# re-calculate catchment statistics without the lithology correction
rv.quartz = None # remove the 'quartz' raster from the imports
stats4 = rv.catchment_stats()
stats4 = stats4[stats4['mean_elev'].notna()] # remove empty rows
stats4 # display catchment statistics
```

```
Processing DB03, DB04, DB02, DB05, DB19, DB17, DB12, DB12 finished.
```

Out[21]:

	name	centr_lat	centr_long	mean_elev	stdev_elev	relief	area	mean_sf	qtz_pc	error
1	DB03	45.55922	7.20885	2509.0	560.6	3307.0	147.7	0.92084	NaN	NaN
2	DB04	45.5647	7.16756	2447.2	571.3	3357.0	280.7	0.91892	NaN	NaN
3	DB05	45.59974	7.34807	2453.7	561.7	3382.0	257.6	0.92309	NaN	NaN
6	DB17	45.60825	7.05543	2426.5	519.8	2985.0	158.3	0.93202	NaN	NaN

Sample data

(see also http://stoneage.hzdr.de/docs/documentation.html#input_format (http://stoneage.hzdr.de/docs/documentation.html#input_format))

Columns that are recognized (processed) by the calculator are `name`, `press_flag`, `thickness`, `density`, `shielding`, `erate`, `year`, `nuclide`, `mineral`, `N`, `delN` and `standardization`. Additional columns are ignored.

Mandatory columns are:

- `name` : Sample name consisting of letters, numbers and hyphens; avoid names that may be misinterpreted as numbers: use 'A2' instead of '2'.
- `N` and `delN` : Nuclide concentration and uncertainty in atoms/grams quartz.

Optional columns are:

- `press_flag` : Atmospheric pressure model, 'std' or 'ant'; the default is 'std'.
- `density` : Subtrate density in g/cm^3; the default is 2.65 g/cm^3.
- `year` : Year of sampling; the default is 2010.
- `nuclide` : 'Be-10' or 'Al-26'; the default is 'Be-10'.
- `shielding` : A catchmentwide shielding factor; these values are ignored if shielding is calculated from a raster dataset, the default is 1.
- `standardization` : see http://stoneage.hzdr.de/docs/documentation.html#standardization (http://stoneage.hzdr.de/docs/documentation.html#standardization). If `standardization` is specified the `nuclide` must be specified as well.

Default values (see `riversand.params.default_values`) are used if a column is missing. Columns `thickness`, `erate` and `mineral` (only valid value: 'quartz') are irrelevant for the calculation of catchmentwide erosion rates, but if they are present they must contain valid values.

The calculator assumes default **standardizations** of '07KNSTD' for Be-10 and 'KNSTD' for Al-26 data. The input data will be re-standardized if a different standardization implemented in the online calculator is specified in the sample data (see `riversand.params.Be_stds` and `riversand.params.Al_stds`). However, it is recommended that if the samples have been measured against a different standard you re-standardize the input data before calculating catchmentwide erosion rates, e.g. using the following correction factors: http://hess.ess.washington.edu/math/docs/al_be_v22/AlBe_standardization_table.pdf (http://hess.ess.washington.edu/math/docs/al_be_v22/AlBe_standardization_table.pdf)

The `name` is used to match samples to catchment polygons. If several samples were measured from the same location, or if both Al-26 and Be-10 were measured, the data must be in separate rows with the same sample name (additional information may be stored in a custom separate colum).

Version of the online calculator

In [22]:
```
# Get the version of the **online calculator** used for the calculation:
riversand.get_version()
```

Out[22]:
```
{'wrapper': '3.0',
 'validate': 'validate_v3_input.m - 3.0',
 'erates': '3.0',
 'muons': '3.1',
 'consts': '2022-12-03'}
```

In [23]:
```
# Get the version of your **riversand** installation:
riversand.__version__
```

Out[23]:
```
'1.2.2'
```

In [ ]:
```
```