**Before You Start**

Before launching the workflow, create a new folder inside the repository root (this is NOT required for the provided case study).

Inside this new folder, create a sub-folder for all remote-sensing images you will process and place your JPG or TIF scenes there.

Next, copy statistic\_template.csv from the repository root to your working directory and fill it in as described below.

Finally, create a folder for the final results.

These three steps are MANDATORY; the code will fail otherwise. Everything else is generated automatically.

File naming  
All images must be named YYYYMMDD, where the date reflects the acquisition day. Images must share the same format (JPG OR TIF) and have identical dimensions.

statistic\_template.csv  
This file lists every image pair you intend to process.

Column 1 – Date of the older image (YYYYMMDD)  
Column 2 – Date of the newer image (YYYYMMDD)  
Column 3 – Temporal baseline in days (you must calculate this manually)  
Column 4 – Reserved for the code; fill with 0  
Column 5 – Reserved for the code; fill with 0

Columns 4 and 5 must still be present and the row count must match the number of pairs.  
Refer to Example\statistic.csv for a clean example.

In the Example\ bundle the three required parts are:

* Example\jpg – image folder
* Example\statistic.csv – pair list
* Example\result – output folder

Names can be freely changed for your own examples.

Optional: intermediate files  
To keep the directory tidy, create an additional folder for intermediate products, e.g. Example\processing. This step is optional; any writable folder (including the case root) will work.

**Cross-correlation (NCC)**

Running the code takes three short steps:

1. Activate the conda environment that satisfies requirement.txt (or the one created through NCC\_strain.cls).
2. Change the working directory to the repository root (where NCC\_core.py lives):  
   cd Yours\NCC\_glacier
3. Execute  
   python NCC\_core.py --config User\_config\_ncc.txt

When the console reports completion, results appear in Example\result.

If the script exits immediately  
Clear the cached state:  
python Clear\_statistic.py –sta Example\statistic –file Example\processing  
Then re-run the cross-correlation command. This behaviour is intentional: unfinished jobs are protected from being overwritten. After every successful run, execute Clear\_statistic.py if you intend to re-process the same pairs.

Adapting to your own project  
Edit User\_config\_ncc.txt – the configuration file that defines every variable used by NCC.

Masks for irregular areas  
For non-rectangular regions, supply an additional mask: a CSV file containing a 2-D boolean array. True marks pixels to be processed; False ignores them.

Output units  
Velocities are in m day⁻¹. Sign convention: positive eastward, positive northward.

## Cross-correlation – Configuration File (User\_config\_ncc.txt)

ker\_size – Side length (px) of the square correlation kernel. REQUIRED.  
win\_size – Side length (px) of the square search window. REQUIRED.  
stride – Step size in pixels; set to 1 if omitted.  
startxy – Pixel coordinates (Matlab convention) of the upper-left corner of the extraction window. REQUIRED.  
lengthxy – Width and height (px) of the extraction window. REQUIRED.  
basis\_fold – Path to the folder that will store intermediate files. REQUIRED.  
sta\_name – Path + filename (without extension) of the pair list (statistic.csv). REQUIRED.  
pict\_file – Path to the folder that contains the images. REQUIRED.  
target\_file – Path to the folder where velocity results will be written. REQUIRED.  
maskurf – Optional mask file path + filename (no extension). Defaults to None; must match lengthxy if used.

**Along-flow Strain-rate Calculation**

We provide a routine to compute the average strain rate along an arbitrary line, following the method described in the paper (Not published yet).

Input requirements  
If you used the NCC workflow above, no extra steps are necessary.  
For external velocity fields:

1. Convert to CSV format.
2. Provide both orthogonal horizontal components.
3. Name the files exactly  
   vx\_YYYYMMDD(older)-YYYYMMDD(newer).csv  
   vy\_YYYYMMDD(older)-YYYYMMDD(newer).csv

Quick test  
In the provided case study, just run  
python Line\_gradient.py –config User\_config\_lg.txt  
Results are written to Example/statistic\_gra.csv.

Output format (5 columns)  
1 – Older image date  
2 – Newer image date  
3 – Temporal baseline (matches your original list)  
4 – Gradient from vx  
5 – Gradient from vy

Adapting to your project  
Edit User\_config\_lg.txt.

## Strain-rate Line – Configuration File (User\_config\_lg.txt)

stride – Same meaning as in NCC; default 1.  
velocity\_fold – Folder containing velocity results (usually target\_file from NCC). REQUIRED.  
sta\_name – Identical to the key in the NCC config. REQUIRED.  
target\_output – Path + filename for the output CSV. REQUIRED.  
xrange – Start and end x-coordinates of the line. REQUIRED.  
yrange – Start and end y-coordinates of the line. REQUIRED.  
GRAtype – Gradient method: [‘np’, ‘sobel’, ‘nye’]. Default np.

## Strain-rate Fitting

Following the approach in paper (Not published yet), the computed strain rates can be fitted to retrieve two parameters:

a – empirical parameter linked to velocity-resolution error  
dv – mean true strain rate of the chosen region

Prerequisite  
Prepare a CSV file of strain-rate data (recommended output from the previous step).

Running the fit  
In the example case:  
python Fitting\_gradient.py –config User\_config\_fit.txt  
The console prints two numbers separated by a space: a dv.

Adapting to your project  
Edit User\_config\_fit.txt.

## Strain-rate Fitting – Configuration File (User\_config\_fit.txt)

d – Velocity resolution in pixels. REQUIRED.  
is\_given\_a – Boolean. If True, the next value (a\_given) is used and a is NOT fitted. REQUIRED.  
a\_given – Prescribed value of a (used only if is\_given\_a is True). REQUIRED.  
is\_given\_dv – Boolean. If True, the next value (dv) is used and dv is NOT fitted. REQUIRED.  
dv – Prescribed value of dv (used only if is\_given\_dv is True). REQUIRED.  
gra\_sta – Path + filename of the strain-rate CSV produced by Line\_gradient.py (target\_output). REQUIRED.  
gra\_sta\_row – Zero-based column index (Python convention) of the strain-rate values to fit. REQUIRED.  
If you used Line\_gradient.py, choose column 3 (vx) or 4 (vy).