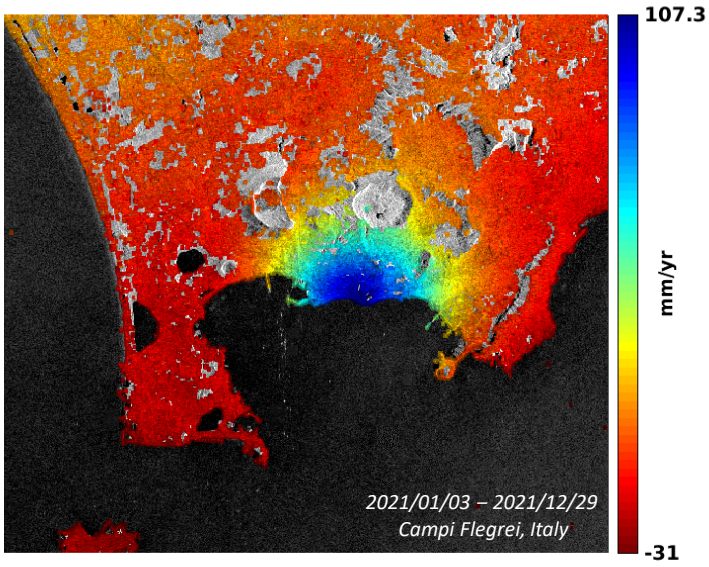
EZ-InSAR Manual

Version 2.0.2 Beta



Alexis Hrysiewicz, Xiaowen Wang, Holohan Eoghan

UCD School of Earth Sciences, University College Dublin

25 April 2022

Table of contents

Part I: Introduction

1.1 The dependencies……3

1.2 The main features……3

1.3 Contributors……6

1.4 Acknowledgement……7

Part II: Installation and configuration

2.1 Setup Python environment……8

2.2 Install SAR processing packages and EZ-InSAR……10

2.3 Test the installation and additional notes……12

Part III: Processing demo with Sentinel-1 data

3.1 Preparations……14

3.2 Download SAR images……16

3.3 Generate SLC or interferogram stack……19

3.4 InSAR time series analysis using StaMPS……24

3.5 InSAR time series analysis using MintPy……29

Part IV: Processing with the other SAR sensor data

4.1 For Strimap data……36

4.2 For Spotlight……36

Part V: Change history

Part VI: Bibliography

Part I

Introduction

EZ-InSAR (formerly named as MIESAR for Matlab Interface for Easy InSAR) is a toolbox written in MATLAB to conduct Interferometric Synthetic Aperture Radar (InSAR) data processing using the open-source software (ISCE+StaMPS/MintPy) within a user-friendly Graphic-User-Interface (GUI).

Nowadays, InSAR has been widely used in measuring ground surface deformation induced by either by tectonic processes (e.g., earthquakes, volcanoes) or anthroponomic activities (e.g., mining, ground water pumping). Many SAR data processing software exist, but few of them have a GUI for manipulating the processing from data input to result visualization except for the commercial ones, such as ***SARscape*** (*Pasquali et al., 2012*) and ***SARPROZ*** (*Perissin et al., 2011*). Also, most of the open source InSAR code are implemented in a UNIX environment, meaning that the operator not only needs expertise in the theoretical background of InSAR, but also the UNIX system itself.

The original purpose of the development of EZ-InSAR is creating a user-friendly SAR data processing chain for ones who are not familiar with InSAR but also interested in producing and analyzing ground displacement products by themselves. The spirit of EZ-InSAR is thus minimizing the work of user in downloading, parametrizing, and processing SAR data and visualizing the results.

EZ-InSAR is also a contribution to the Platform for Atlantic Geohazard Risk Management (AGEO) project, which is funded by Interreg Atlantic Area Programme through the *European* Regional Development Fund (*Ortega Rodriguez et al., 2021*). AGEO aims launching several Citizens' Observatory pilots on geohazards according to regional priorities. The project also aims engaging with local communities to actively participate in risk preparedness and monitoring, and incorporate local capacities into risk management systems.

EZ-InSAR is currently developed on a Linux platform. The toolbox can generate SAR interferograms using ISCE and conduct displacement time series analysis with either Persistent Scatters (PS) or Small-Baselines (SBAS) approaches using StaMPS and MintPy. EZ-InSAR can save the results in several popular formats such as GeotTIFF, NetCDF, KMZ, and ROIPAC, which can be then imported into the other platforms such as QGIS, GMT, and Google Earth for visualization and post-analysis.

EZ-InSAR is still under development by the *Nature Hazard Research* group at School of Earth Sciences, University College Dublin (UCD). In the future, an independent visual toolbox will be incorporated into EZ-InSAR, allowing users to do advanced post-analysis such as displacement displaying, filtering, time-series analyzing, and modelling. The toolbox now supports the processing of data from several SAR sensors, including the Sentinel-1A/B, TerraSAR-X, PAZ-SAR and COSMO-SkyMed.

1.1 The dependencies

Currently, EZ-InSAR is developed within the commercial software MATLAB. Accordingly, some toolboxes of MATLAB are needed to run the toolbox (see *Installation & Configuration in* **Part II**). In the near feature, we will release the Python-based version that are totally free of use.

EZ-InSAR incorporates several the most popular open source InSAR processors to perform SAR interferometric and displacement time series analysis. These processors are:

- **ISCE** - Interferometric synthetic aperture radar Scientific Computing Environment (ISCE)

  ISCE is an InSAR processing software developed by NASA's Jet Propulsion Laboratory (JPL) and it is currently the best free available software of its kind. For Sentinel-1 TOPS data, ISCE provides a set of applications (e.g., topsApp.py and stackSentinel.py) to easily control the data processing.

- **StaMPS** - Stanford Method for Persistent Scatterers (StaMPS)

  StaMPS is a software package that can implement PS or SBAS methods to extract ground displacements from time series of synthetic aperture radar (SAR) acquisitions. The original version was developed at Stanford University but subsequent development has taken place at the University of Iceland, Delft University of Technology and the University of Leeds.

- **MintPy** - The Miami INsar Time-series software in PYthon (MintPy)

  MintPy is an open-source package for InSAR time series analysis. It can read the stack of interferograms (coregistered and unwrapped) and produces three-dimensional (2D in space and 1D in time) ground surface displacement in line-of-sight direction. It includes a routine time series analysis application (i.e., smallbaselineApp.py) and some independent toolbox.

Some additional dependencies are required to run the above InSAR processors or using the enhanced functions of the toolbox. For example, users may need the **TRAIN** package to correct for tropospheric errors in SAR interferograms when using StaMPS, while you may need the **PyAPS** package to do the similar work when using MintPy. The detailed descriptions on installation and configuration of the SAR processors and their dependencies will be shown in **Part II**.

1.2 The main features

Figure 1.1 shows a snapshot of EZ-InSAR. Basically, the interface contains three independent modules shown as the "**Data Preparation Module**", "**ISCE InSAR Processing Module**", and "**InSAR Time Series Analysis Module**". The "EZ-InSAR Paths" button allows the user to define the work path for processing the data. The StaMPS and MintPy processors can be activated by clicking the corresponding tables in the "**InSAR Time Series**

Analysis Module" module, respectively. Also, there are a **progress bar** showing the running progress of each step and an **information box** showing the useful tip during data processing at the bottom of the interface.

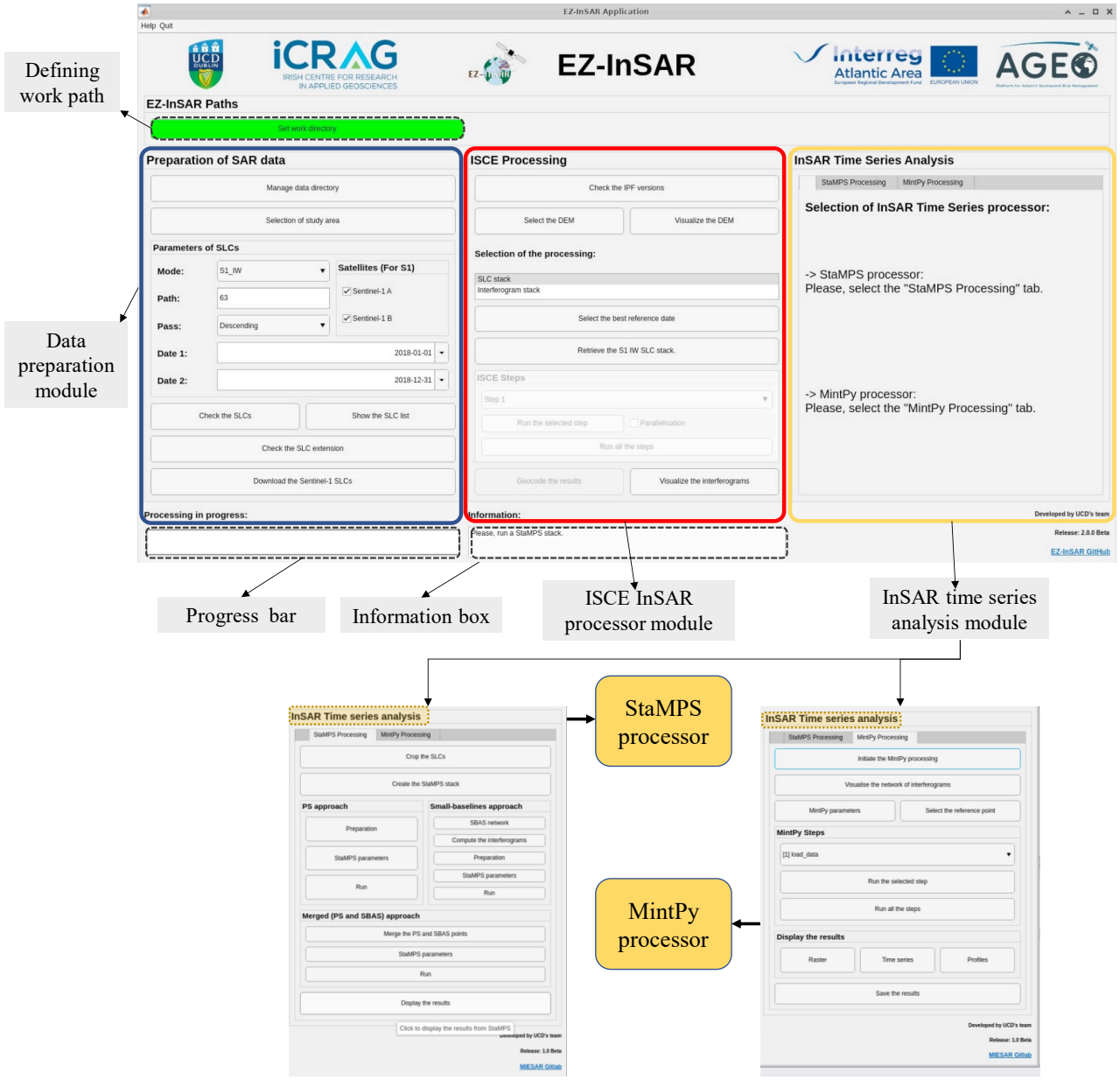


Figure 1.1 The interface of EZ-InSAR.

Figure 1.2 shows an overview of the data processing flow chart of **EZ-InSAR**. We can define that the InSAR analysis generally have the following seven steps: (1) Select a region where we want to compute the ground displacements; (2) Download the SAR images in SLC format; (2)Download a DEM file covering the study region; (3)Coregistration of the SLC stack; (4) The computation of interferogram stack; (5) The computation of displacement time series using PS or SBAS approaches; (6) Visualize the results such as displacement velocity and time series. EZ-InSAR finishes a processing task by running the above seven steps one-by-one.

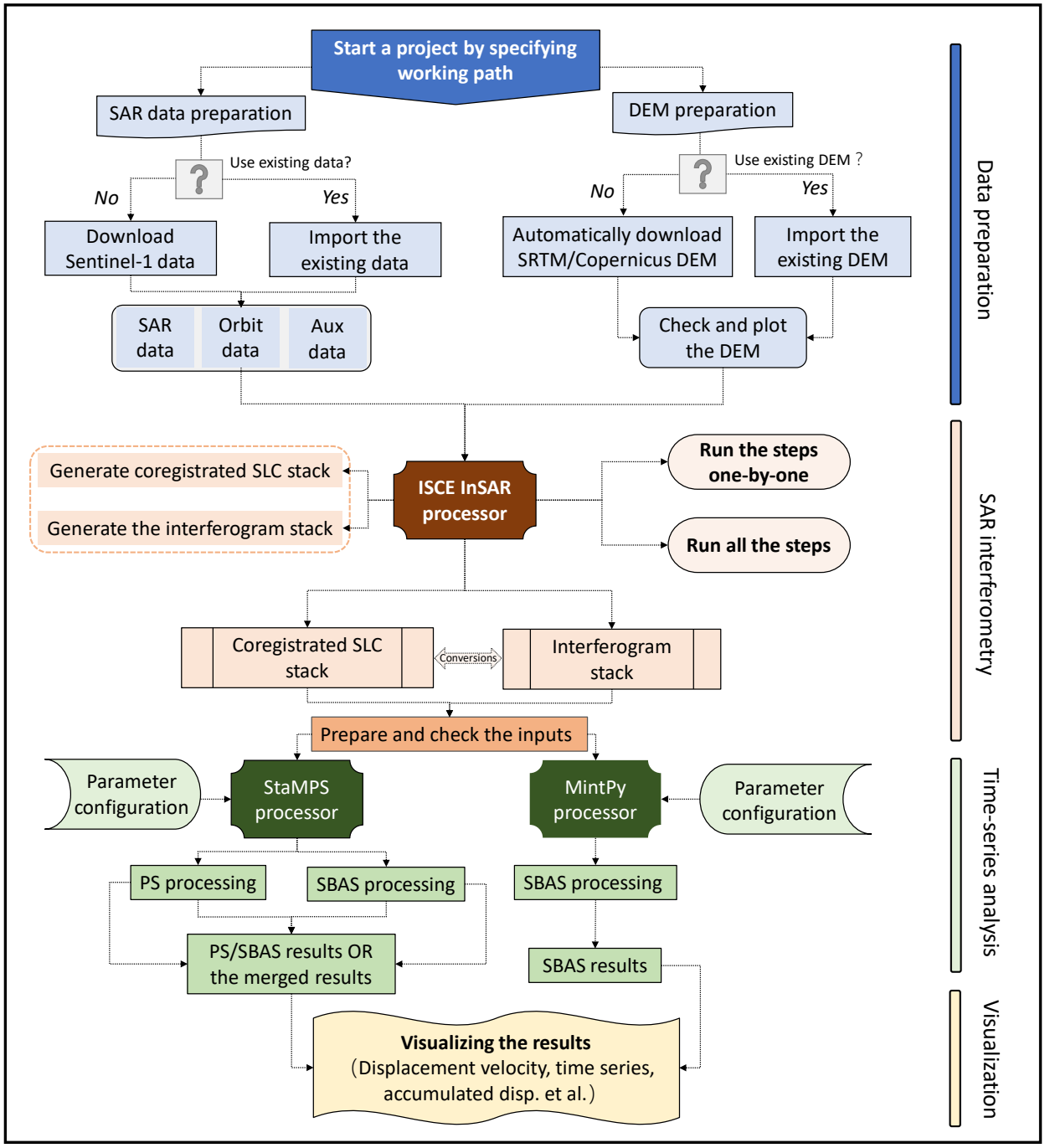


Figure 1.2 The flow chart of SAR data processing in EZ-InSAR.

During the data processing, **EZ-InSAR** calls back the MATLAB scripts, and the scripts provided by the embed InSAR processors. Note that ISCE is run by calling the Python and Bash scripts, while StaMPS is written in MATLAB. Except for the first step in which one has to import a KML file defining the geographic boundary of the study area, users can finish the processing automatically by clicking the designed buttons. Compared to the other free open-source InSAR processing software packages, **Figure 1.3** summarizes the unique features of EZ-InSAR.

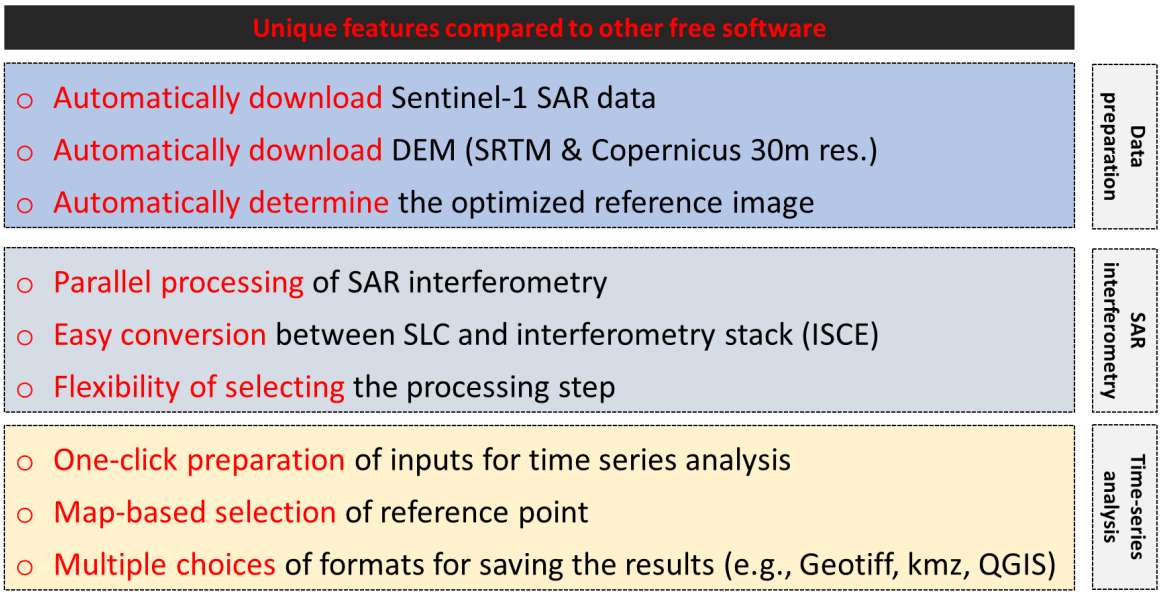


Figure 1.3 The unique features of EZ-InSAR.

In the following part (**Part II**), we will show how to install and configure the modules needed for running EZ-InSAR. A processing demo with Sentinel-1 SAR images will be demonstrated in detail in **Part III,** with an instruction on processing images from the other SAR sensors in **Part IV. Part V** will summarize the release history of **EZ-InSAR** and **Part V** will give a few useful reference resources for assisting the SAR processing.

1.3 Contributors

EZ-InSAR is developed and maintained by the *Nature Hazard Research* group lead by ***Eoghan Holohan*** at School of Earth Sciences, University College Dublin (UCD). Users can download **EZ-InSAR** through the project repository at GitHub: https://github.com/alexisInSAR/EZ-InSAR

The people developing and documenting the toolbox are acknowledged below:

- *Alexis Hrysiewicz,* School of Earth Sciences, UCD
- *Xiaowen Wang*, School of Earth Sciences, UCD & Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University (China).

When using this software please cite the reference below:

Hrysiewicz, A., Wang, X. & Holohan, E.P. EZ-InSAR: An easy-to-use open-source toolbox for mapping ground surface deformation using satellite interferometric synthetic aperture radar. Earth Sci Inform (2023). https://doi.org/10.1007/s12145-023-00973-1

In addition, cite dependencies used such as ISCE (*Rosen et al., 2012*), StaMPS (*Hopper et al., 2004, 2008*), TRAIN (*Bekaert et al., 2015a, 2015b*), MintPy (*Zhang et al., 2019*) depending on your workflow.

1.4 Acknowledgement

We acknowledge that the open-source InSAR processing software and code used by **EZ-InSAR** are cited properly. **EZ-InSAR** is distributed for free under the GNU General Public License (https://www.gnu.org/licenses/).

Part II

Installation and configuration

[Ubuntu 20.04], updated on 03 August, 2022

2.1. Setup Python environment

2.1.1 Check and install *conda* distribution

It is recommended to use *conda* to install the python environment and the prerequisite packages. Either the *Anconda* or *Miniconda* distributions are applicable, while we suggest using *Miniconda* since it is a small, bootstrap version of Anaconda.

- Check whether the *conda* distribution has already been installed. If you have already installed *conda*, go to the **Step 2.1.2.**

```
conda -V
```

- Install *Miniconda* if no conda distribution is detected. By default, it will be installed in the `$HOME` directory.

```
mkdir -p ~/mconda; cd ~/mconda
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
bash Miniconda3-latest-Linux-x86_64.sh -b -p ~/mconda/miniconda3
~/mconda/miniconda3/bin/conda init bash
```

- Close and restart the shell for changes to take effect.

2.1.2 Create virtual Python environment: InSARenv

Here we create a virtual environment "*InSARenv*", and all the packages supporting the EZ-InSAR toolbox will be installed in this Python environment.

```
# A. Install the basic tools and utils

sudo apt install gcc g++ gawk tcsh build-essential make git
conda config --add channels conda-forge
conda config --set channel_priority strict
conda install wget git tree mamba --yes

# B. Create InSAREnv & activate it

conda create --name InSAREnv --yes
conda activate InSAREnv
```

2.1.3 MATLAB

MATLAB is used to run EZ-InSAR, thus you need a licensed MATLAB installed on your computer. Also, the InSAR post-processing with StaMPS will also use MATLAB, and the successful running of StaMPS need some specific MATLAB toolbox to be pre-installed.

- Check MATLAB has been installed on your OS, and can be accessed in a terminal.

```
#A. Check whether MATLAB has been installed and be launched

which matlab

#B. If the echo of "which matlab" is null, then install MATLAB first, and then
specify the MATLAB PATH below, and create a soft-link it to the directory
'/usr/bin'

matlab_install_path="ADD-THE-MATLAB-PATH-HERE"
sudo ln -s $matlab_install_path/bin/matlab /usr/bin

#C. Then start MATLAB in a terminal
matlab
```

- Type `ver` in MATLAB and check whether the toolboxes below are available. If some of them are not installed, just click "Add-Ons" ---> "Get Add-Ons" in the MATLAB interface menu to fix the installation.

```
>> ver
----------------------------------------
Curve Fitting Toolbox
Financial Toolbox
Image Processing Toolbox
Mapping Toolbox
Optimization Toolbox
Parallel Computing Toolbox
Signal Processing Toolbox
Statistics and Machine Learning Toolbox
```

2.2 Install InSAR processing packages and EZ-InSAR

This part will first show the "Preparations" you have to do, including the download of the required software or packages. Then, the installations of the three InSAR processors "ISCE", "MintPy", and "StaMPS" will be illustrated. Finally, it will give a instruction on the configuration of EZ-InSAR.

2.2.1 Preparations

```
# A. Define the path where you want the packages to be installed (Default:Your
HOME path-"$HOME")

EZINSAR_HOME="$HOME"                    #Set the install path

tools_insar=$EZINSAR_HOME/tools_insar
tool_DIR=$tools_insar/proc_insar
sudo mkdir -p $tool_DIR

# B. Download the source code (EZ-InSAR, ISCE, MintPy, StaMPS, TRAIN)
## Download EZ-InSAR source file from github, and put the unzipped EZ-InSAR
into the "EZINSAR" directory.

sudo git clone https://github.com/alexisInSAR/EZ-InSAR.git $tools_insar/EZINSAR

## Download the InSAR processor codes
sudo git clone https://github.com/isce-framework/isce2.git $tool_DIR/isce2
sudo git clone https://github.com/insarlab/MintPy.git      $tool_DIR/MintPy
sudo git clone https://github.com/insarwxw/StaMPS.git      $tool_DIR/StaMPS
sudo git clone https://github.com/dbekaert/TRAIN.git
$tool_DIR/StaMPS/TRAIN

# C. Edit the configuration file
## 1) - Copy the configure template file "config_InSARenv.template" from the
"EZINSAR/EZINSAR_BIN/docs/" directory into "$tools_insar";
##    - Check and replace the PATH variable "$EZINSAR_HOME" in
"config_InSARenv.rc" (Line #3).
##    - Check and replace the Path varialbe $APS_toolbox in TRAIN
##    - The other variables do not need to be modified if you strictly follow
this install instruciton.

sudo cp $tools_insar/EZInSAR/EZINSAR_BIN/docs/config_InSARenv.template
$tools_insar/config_InSARenv.rc

sudo sed -i "/EZINSAR_HOME=/c\EZINSAR_HOME=$EZINSAR_HOME"
 $tools_insar/config_InSARenv.rc
sudo sed -i "/APS_toolbox=/c\APS_toolbox=$tool_DIR/StaMPS/TRAIN"
$tool_DIR/StaMPS/TRAIN/APS_CONFIG.sh

## 2) Add the following lines in your "$HOME/.bashrc" file.
##    Note you have to change the variable "$EZINSAR_HOME" if it is installed
in a differnt PATH (e.g., /usr/local).

# EZ-InSAR & InSARenv
EZINSAR_HOME="$HOME"
```

```
export tools_insar="$EZINSAR_HOME/tools_insar"
alias load_insar='conda activate InSARenv; source
$tools_insar/config_InSARenv.rc'

# **IMPORTANT**: Run `load_insar` in the terminal to load the "InSARenv"
environmental and PATH variables before running EZ-InSAR each time.
```

2.2.2 Install ISCE

ISCE-2 is now available on the `conda-forge` channel. Thus, one could install it by simply running:

```
conda install -c conda-forge isce2
```

2.2.3 Install MintPy

MintPy is written purely in Python. So, the use of MintPy just needs the installation of dependent python packages and then setup the environment variables properly.

[Note]: You can change the PATH variables for MintPy in `config_InSARenv.templte`.

```
# A. Install the mintpy requirements
cd $tool_DIR
conda install -c conda-forge --file MintPy/requirements.txt

# B. Install dependencies not available from conda
sudo ln -s ${CONDA_PREFIX}/bin/cython ${CONDA_PREFIX}/bin/cython3
$CONDA_PREFIX/bin/pip install scalene      # CPU, GPU and memory profiler
$CONDA_PREFIX/bin/pip install ipynb        # import functions from ipynb files
```

2.2.4 Install StaMPS

- Check the *gcc* version of your OS first because the `src` code in StaMPS only supports gcc-7.
- However, the default *gcc/g++* in Ubuntu 20.04 is gcc-9 or above. If in this case, you need to install the gcc-7 first.
- Follow the **instruciton** here to set the proper gcc/g++ versions. If the error such as "E: Package 'gcc-7' has no installation candidate" appear, please refer to the solution **here**.

```
### check which version of gcc
gcc -v
#$ gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)

sudo apt install software-properties-common
sudo add-apt-repository ppa:ubuntu-toolchain-r/test
sudo apt install -y gcc-7 g++-7
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-7 7
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-7 7
sudo update-alternatives --config gcc
sudo update-alternatives --config g++
```

- Compile the `src` files and install the required dependencies

```
# A. Compile the files in the "src" directory in StaMPS

cd $tool_DIR/StaMPS/src
sudo make
sudo make install

# B. Install "snaphu" & "triangle"
# After the installation, run `which snaphu` && `which triangle` in the
terminal to check their paths.
# If the echo paths are not "/usr/bin", then modify the varialbes "SNAPHU_BIN"
and "TRIANGLE_BIN" in the "config_InSARenv.rc" to the correct values.

sudo apt install snaphu
sudo apt install triangle-bin
```

2.2.5 Install EZ-InSAR

- Install the dependencies of some python script required by EZ-InSAR to your InSARenv.

```
conda install fiona geopandas rasterio
```

- EZ-InSAR uses "aws" to download the NASADEM or Copernicus DEM. Using the following commands to install it.

```
cd $tool_DIR
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"
sudo unzip awscliv2.zip
sudo ./aws/install
```

- Set your ASF account and credentials for downloading the Sentinel-1 SAR data in "pathinformation.txt" in EZINSAR_BIN.

```
ASFID   account-name
ASFPWD  password
```

2.3 Test the installation and additional notes

2.3.1 Test the installation

- Open a new terminal, and type the commands below to check the PATH environment

```
source $HOME/.bashrc
load_insar  # warm up conda environment
echo $PATH
echo $TOOL_DIR $DATA_DIR $WEATHER_DIR
```

- Run the following to test the installation:

```
###
topsApp.py -h            # test ISCE-2
smallbaselineApp.py -h   # test MintPy
solid_earth_tides.py -h  # test PySolid
tropo_pyaps3.py -h       # test PyAPS
mt_prep_isce             # test StaMPS
```

2.3.2 Some important notes for running MintPy

(1) Notes on account setup for ERA5

MintPy has the option of using ERA5 to correct for tropospheric delay. ERA5 data set is redistributed over the Copernicus Climate Data Store (CDS). Registration is required for the data access and downloading.

- Create a new account on the CDS website if you don't own a user account yet.
- Create a file named `.cdsapirc` in your `$HOME` directory and add the following two lines:

```
url: https://cds.climate.copernicus.eu/api/v2
key: 12345:abcdefghij-134-abcdefgadf-82391b9d3f
```

> where *12345* is your personal user ID (UID), *the part behind the colon* is your personal API key. More details can be found here. **Make sure** that you accept the data license in the Terms of use on ECMWF website.

- Test the account setup for ERA5 by running:

```
git clone https://github.com/insarlab/PyAPS.git --depth 1
python PyAPS/tests/test_dload.py
```

`WEATHER_DIR`: Optionally, if you defined an environment variable named `WEATHER_DIR` to contain the path to a directory, MintPy applications will download the GAM files into the indicated directory. You can change `WEATHER_DIR` in the configure file "config_InSARenv.rc". Also, the MintPy application will look for the GAM files in the directory before downloading a new one to prevent downloading multiple copies if you work with different dataset that cover the same date/time.

(2) Notes on dask for parallel processing

MintPy uses dask for the parallel processing at some of the steps. It is recommended setting the `temporary-directory` in Dask configuration file, e.g. `~/dask/dask.yaml`, by adding the following line, to avoid potential workspace lock issue. Check more details on parallel processing with Dask here.

```
temporary-directory: /tmp  # Directory for local disk like /tmp, /scratch, or
/local

## If you are sharing the same machine with others, use the following instead
to avoid permission issues with others.
# temporary-directory: /tmp/{replace_this_with_your_user_name}  # Directory for
local disk like /tmp, /scratch, or /local
```

==========**END**==========

Part III

Processing demo with Sentinel-1 data

In this part, a demo of running EZ-InSAR will be presented step-by-step. The demo uses the Campi Flegrei volcano in Italy as a test region. 31 Sentinel-1A SAR images along the descending path (Path Number: 22) acquired between January 03 and December 27, 2021 over the volcano are used to show how **EZ-InSAR** works. This demo is run on a Linux OS (***Ubuntu 20.04.4***) with a MATLAB licensed at a version of ***2021b***.

The contents of this demo include the preparation of SAR data (**Section 3.1**), generating SAR interferograms using the ISCE processor (**Section 3.2**), and doing displacement time series analysis using StaMPS (**Section 3.3**) and MintPy (**Section 3.4**). Note that this tutorial will not cover the detailed explanations of the parameters setting for running ISCE, StaMPS, and MintPy, although **EZ-InSAR** allows the users to modify these parameters in an interactive way. Users should refer to the manuals or training documents of these InSAR processors to set proper values for their own processing demands (see the references in **Part V**).

The text in different font obeys the following rules:

> **Button**: The designed button in EZ-INSAR GUI for running one function.
>
> *Linux Command*: The Linux command mentioned during the data processing.
>
> ①-⑨: The steps of running the task.
>
> Folder: The directory mentioned in the working path.
>
> **Option**: The options to be selected in data processing.
>
> Tips! Some explanations on the executed step or the generated results.

3. 1 Preparations

Users need to prepare a KML file defining the study region before the data processing. **EZ-InSAR** will use this KML file as a constraint to search the available Sentinel-1 SAR data such as Alaska Satellite Facility (ASF) Data Search Vertex (*https://search.asf.alaska.edu/#/*).

① Open Google Earth, plot a polygon over the study area and then save the polygon as a KML file.

② Open the ASF Data Search platform, search the Sentinel-1 SAR data over the study area and determine the **Path number** and **Flight Direction** of the SAR data to be processed (**Figure 3.1**).

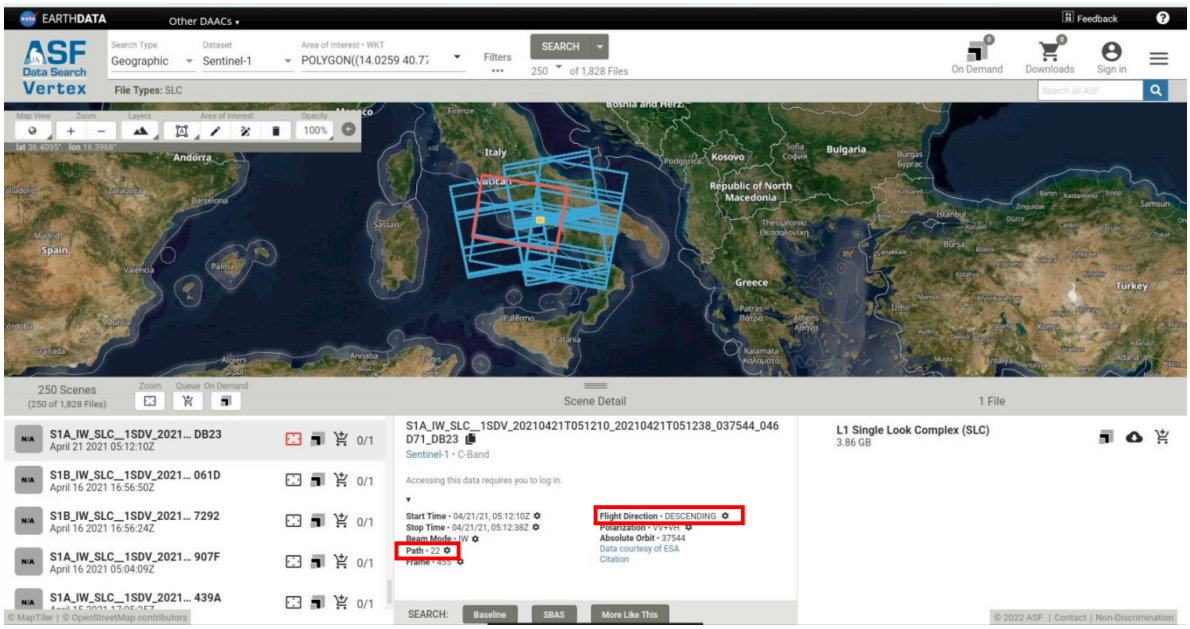


Figure 3.1 Sentinel-1 data search results at ASF Vertex.

③ Open a terminal and run "*load_insar*" to activate the "InSAREnv" Python environment first (see **Part II**); then type "*matlab*" in the terminal to start MATLAB (**Figure 3.2**).

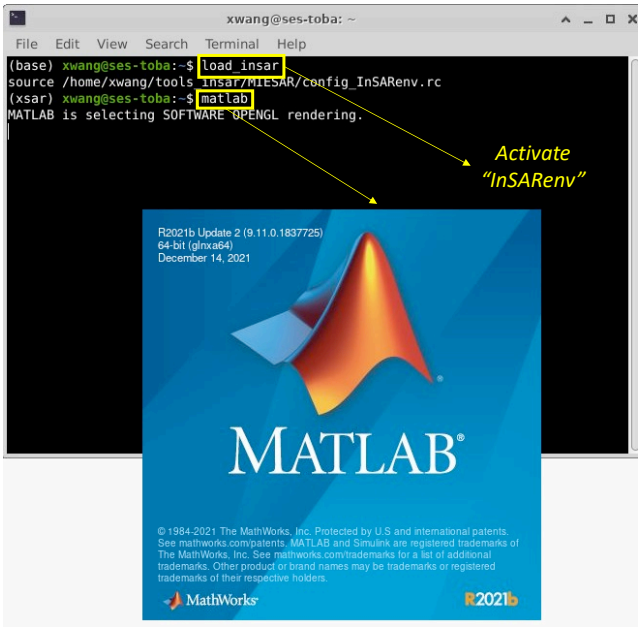


Figure 3.2 Activate "InSAREnv" and start MATLAB in a terminal.

3. 2 Download SAR images

① In MATLAB, and type "**EZ_InSAR**" to lunch the application interface. You can see the button "**Set Work directory**" in red color.

Figure 3.3 The workflow of download SAR images.

②Click "**Set work directory**": Define the work path where you want to run the data processing. The full interface will appear. **Figure 3.3** shows the overview of the buttons used for preparing the Sentinel-1 SAR data.

Tips!

This step will check whether the work path already contains the "slc", "dem", "orbits", and "aux" folders. These folders will be automatically created if they are not existing. You can also manually create these folders, while note that the folder names should strictly follow the definitions above. You can also put the SLC zip files or precise orbit POE files into the corresponding directories manually.

③ Click "**Manage SLC's directory**": Open a dialog showing the paths of the "slc", "dem", and "aux" folders. The text of the paths will be in green color if the folders are successfully detected. Click "**Validation**" to finish the checking of the paths (**Figure 3.4A**).

④ Click "**Selection of study region**": Select the "*.kml" file covering your study region that you have prepared in **Section 3.1**. A map showing your study region will then come up following this step (**Figure 3.4B & C**).

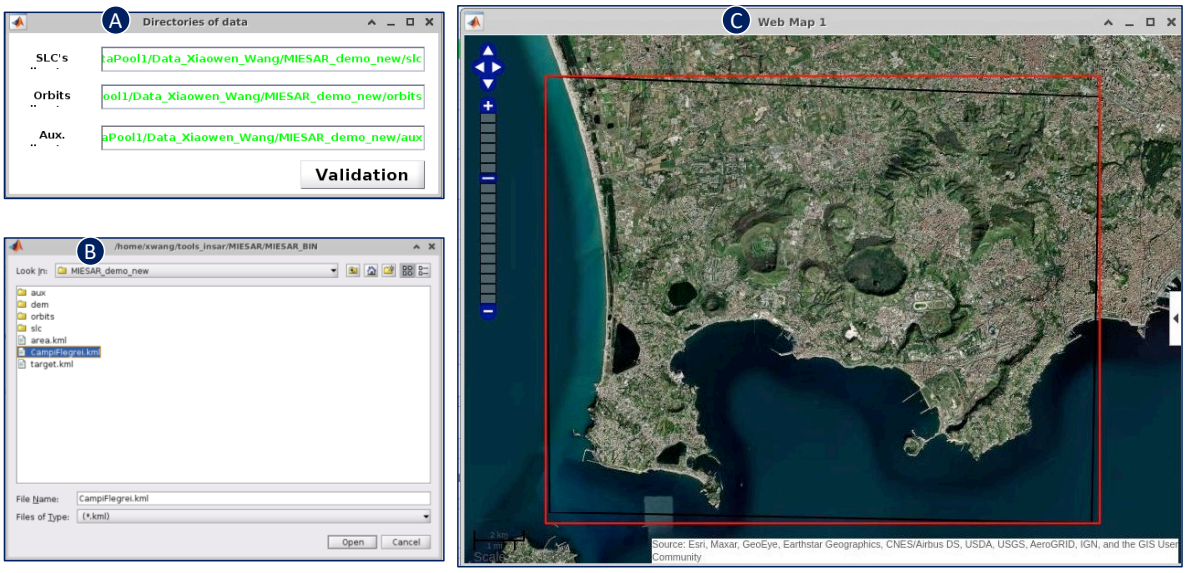


Figure 3.4 Validation of the data paths and import of the KML file defining the study region.

⑤ In the "**Parameters of SLC**" panel: Input the parameters for the "**Mode (IW)**", "**Path** (Path number)", "**Pass** (Ascending or Descending)", "**Date 1** (Start date), **Date 2** (End date)" of the Sentinel-1 SAR data, and chose the **Satellite** (Sentinel-1A, Sentinel-1B). Some of these parameters should be checked by referring to the preparation step of **Section 3.1**.

⑥ Click "**Check the SLCs**": **EZ-InSAR** will inquire the available SLC images covering the study area.

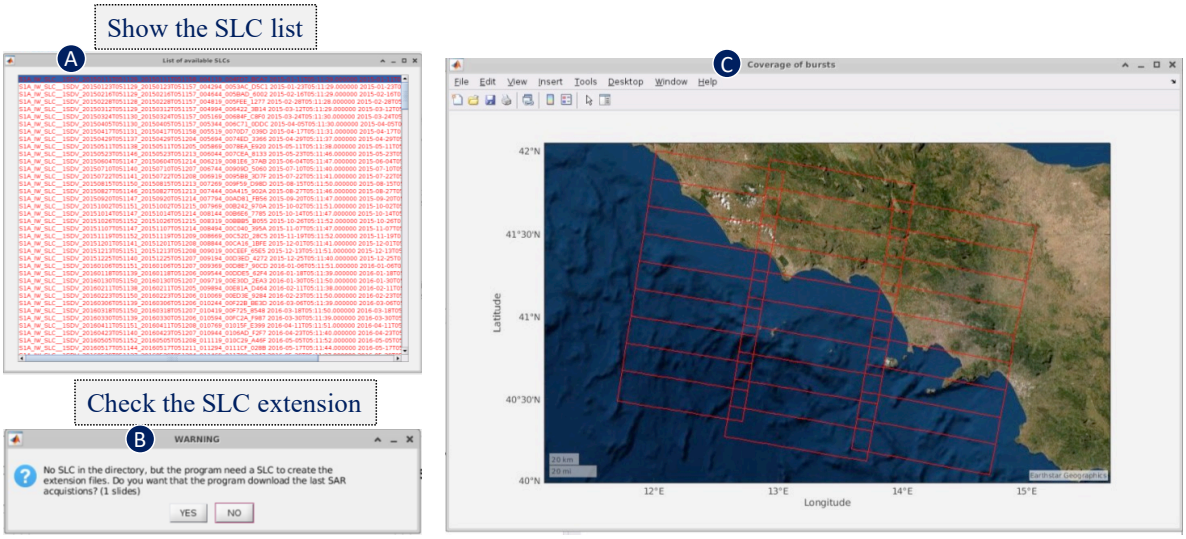


Figure 3.5 Check the available SLC images (A) and the extension coverage (B & C).

⑦ Click "**Show the SLC list**": Open a window showing the available SLC list. If the SLC "*zip" files are already existing in the "slc" folder, the data list will be shown with green color (**Figure 3.5A**).

⑧ Click "**Check the SLC extension**": If you click this button at the first time, a question dialog will show and ask which data scene would you like to select to show its geographical extension (**Figure 3.5B**). By default, **EZ-InSAR** will select the last scene as input. The parameter files of the selected scene will be extracted to calculated the footprints of each burst, which will be shown and overlapped on an optical satellite base map (**Figure 3.5C**).

⑨Click "**Download the SLCs**": Open the "Sentinel-1 IW Downloader" (**Figure 3.6**). You can use the buttons "**Select all**", "**Deselect all**" to manipulate the image selection. The **Sequential selection** option allows you to select the images at a specified time interval (with a 6-day interval). After the selection, you can click "**Save selection**" to create "*savedselectionSLC.mat*" file in the work path, and then use "**Load selection**" if you want to re-select or change the data selection later. Click "**Download**" to conduct the data downloading from the ASF Vertex website. Please make sure you have created an account at ASF and filled the "**user-name**" and "**password**" in the "*pathinformation.txt*" file in the "EZINSAR _BIN" folder.

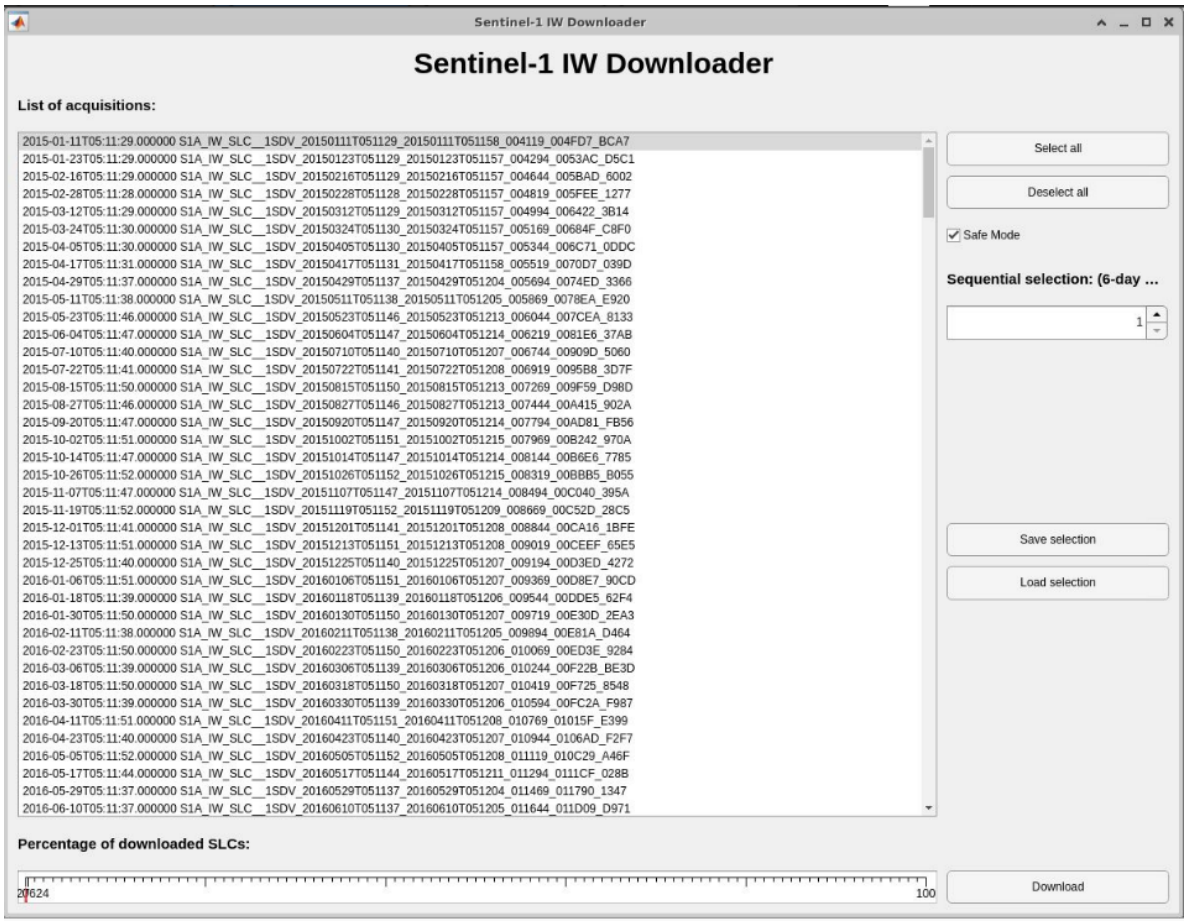


Figure 3.6 The toolbox for selecting and downloading Sentinel-1 SAR data.

Tips!

After the above steps from ①-⑨, the "slc" directory in the work path will contain the downloaded Sentinel-1 SLC images (*.zip). The work path will also include the saved files "*SLC.list*", "*area.kml*", "*traget.kml*", "*parmsSLC.mat*". These files will be detected and loaded if you re-run the above steps.

3.3 Generate SLC or Interferogram stack using ISCE

The workflow of generating SLC or Interferogram stack with ISCE includes 8 steps, as shown in **Figure 3.7**. Step 1-3 will prepare the auxiliary and DEM files used for InSAR processing; Step 4-5 will prepare the configuration and task running files for generating either "SLC stack" or "Interferogram stack"; Step 6-7 will perform SAR data processing using ISCE for the selected task.

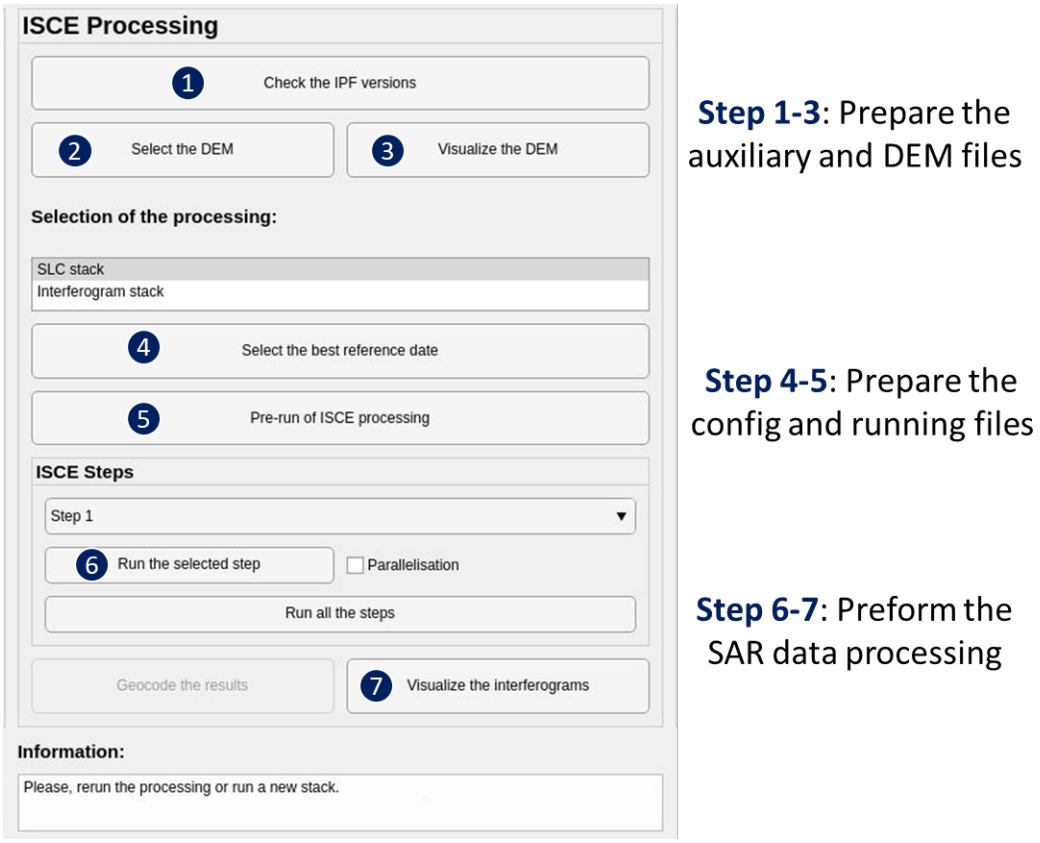


Figure 3.7 The workflow for generating SLC or Interferogram stack using ISCE.

① Click "**Check the IPF versions**": Check whether the auxiliary files are required for the downloaded SLCs. Sentinel-1 auxiliary files will be used to do Elevation Antenna Pattern correction to the interferometric phase generated with image pairs that have Instrument Processing Facility (IPF) version crossing v2.3.6. If the situation is true, **EZ-InSAR** will download the auxiliary files automatically into the "aux" folder. A figure showing the version of IPF for all the SLCs will appear after the checking process (**Figure 3.8**).

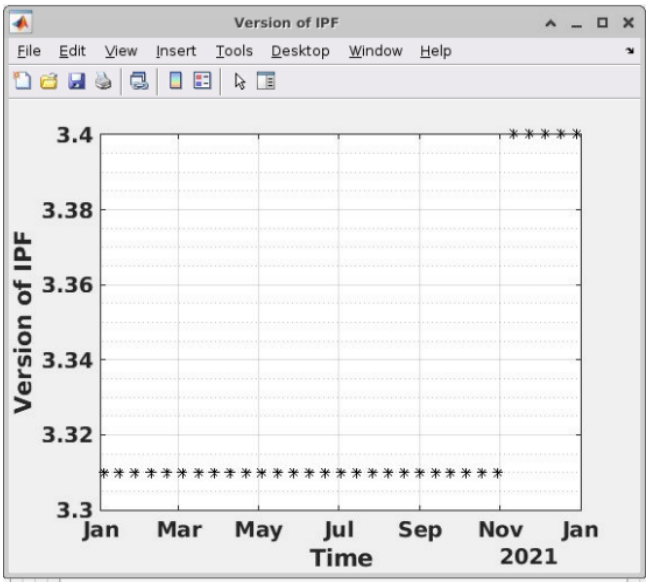


Figure 3.8 The plot of the IPF values for the Sentinel-1 SAR data.

② Click "**Select the DEM**": **EZ-InSAR** will direct you to prepare a DEM over the study region to support ISCE InSAR processing (**Figure 3.9A**). Click "**Localhosted DEM**" to use the existing dem already prepared on your local disk. Click "**Download the DEM**" to download either the **NASARDEM** or **COPERNICUS 1arc** (30 meter) resolution DEM (**Figure 3.9B**). You can also put your own DEM files (*.tif) in Geotiff format in the "dem/dem_tiff folder". Then click "**Download DEM**" ---> "**Existing Geotiff DEM**" to generate the dem file.

③ Click "**Visualize the DEM**": Open the shaded relief map of the DEM data (**Figure 3.9C**).

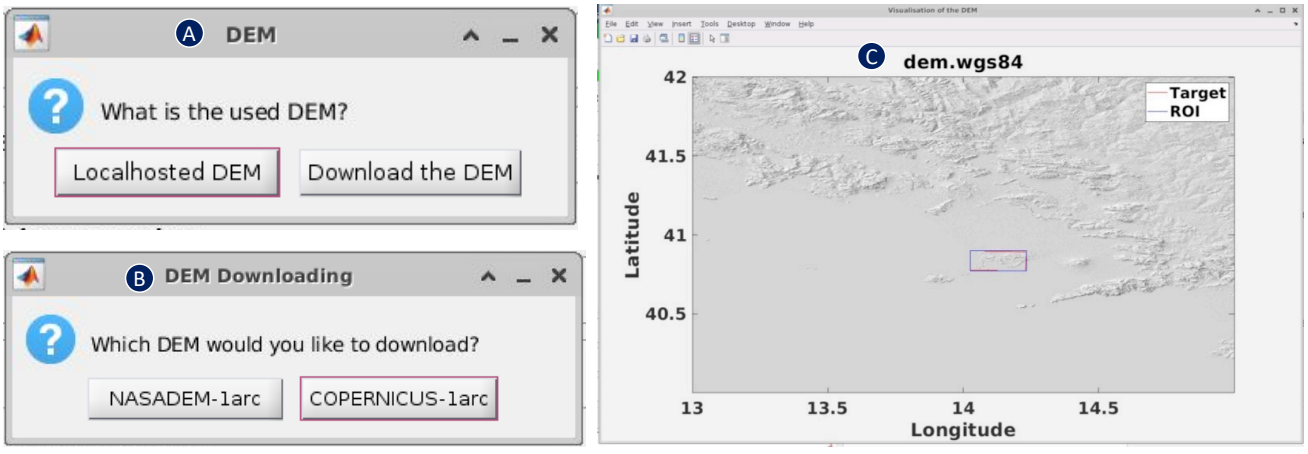


Figure 3.9 Prepare the DEM for InSAR processing using ISCE.

Tips!

After the step 3, users have to choose which task (task "**SLC stack**" or "**Interferogram stack**") they would like to conduct in the following steps with ISCE. EZ-INSAR provides a function "**Select the best reference date**" to help users determine the optimal date as the reference date. This is done by calculating the perpendicular and temporal baselines of image pairs within the single reference network, and the image in the network graphic center is determined as the optimal one.

④ Click "**Select the best reference date**": In the dialog window, input "POD" for the **Orbit MODE** option if you want to use the precise orbit data, and input a date in "*yyyymmdd*" format for the **Potential Reference date** option. By default, you can leave both of these two options as default ("*None*"). Click "**OK**", a figure showing the network will appear with the best reference data annotated with red color (**Figure 3.10**).

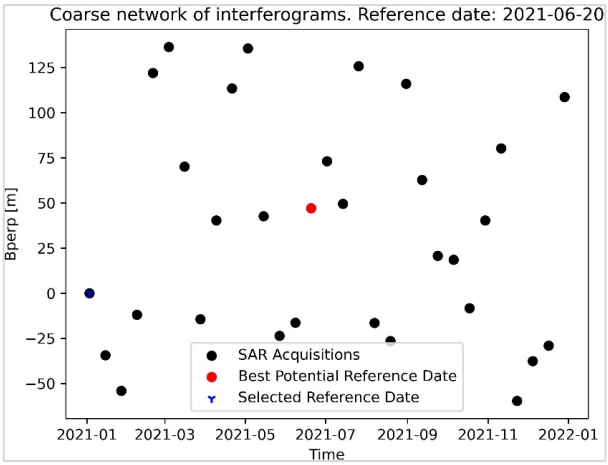


Figure 3.10 The perpendicular baseline network of the SAR images.

Tips!

- The "**SLC stack**" mode will only generate a stack of coregistrated SLCs and will not do interferometric processing, which is suitable for ones would like to time series analysis using StaMPS because it will invoke the ISCE applications doing interferometric processing for the selected image pairs.
- The "**Interferogram stack**" mode will not only do SLC coregistration but also generate SAR interferograms, filter and unwrap the interferograms. The products generated with this mode can be directly imported into MintPy to do displacement time series analysis.
- One can run the ISCE processing under "**SLC stack**" mode first, and then using the button "**Continue to generate interferogram stack**" to complete the interferometry processing (**Figure 3.1·A**). EZ-INSAR will generate "merged_stamps" and "merged_mintpy" respectively in the work directory. Users can then do the post analysis either using StaMPS or MintPy. Also, if you run ISCE processing by first selecting the "**inteferogram stack**", we provide a "**Retrieve the SLC stack**" button to generate a "merged_stamps" directory that contains the slc stack supporting the post-analysis using StaMPS (**Figure 3.1·B**).

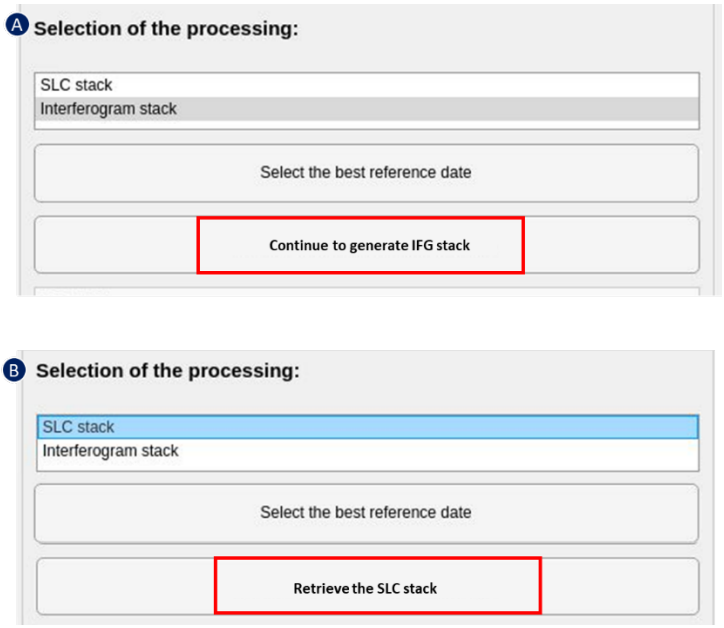


Figure 3.11 Conversions between the interferogram (IFG) stack and SLC stack.

⑤ Click "**Pre-run of ISCE processing**": Select the processing task, and then input the required parameters for lunching the stack processing task using ISCE (**Figure 3.12**). After this step, a set of configure files and shell scripts will be generated in the directories termed "configs" and "run_files", respectively, in the work path.

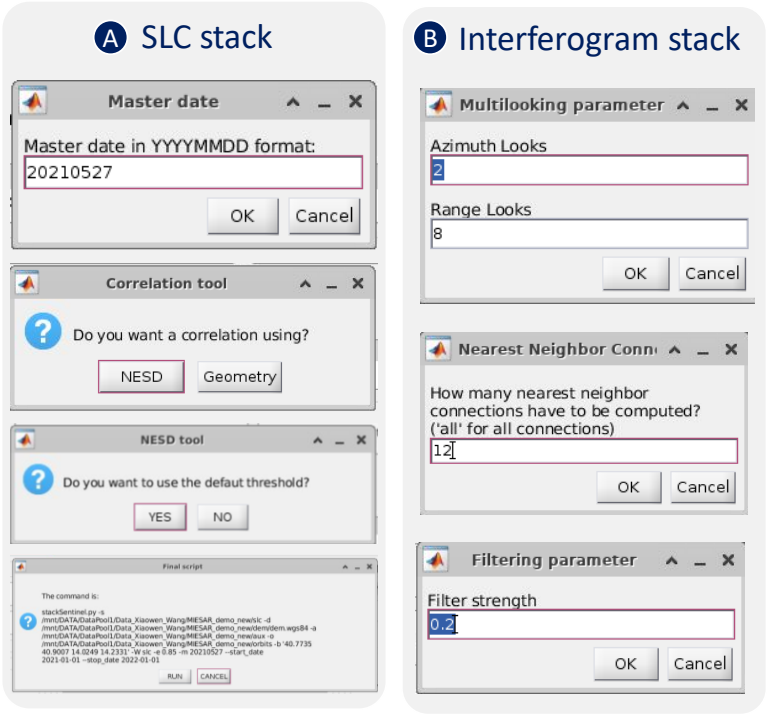


Figure 3.12 Setting parameters for preparing the SLC (A) or Interferogram stack (B) generation.

⑥Click "**run the selected step**": ISCE will conduct the processing of the step you selected in the selection list (**Figure 3.13**). The step-by-step processing is helpful for conducting tests and checking the results of each step immediately. Users can also click "**Run all the steps**" to let ISCE process all the steps at one time.

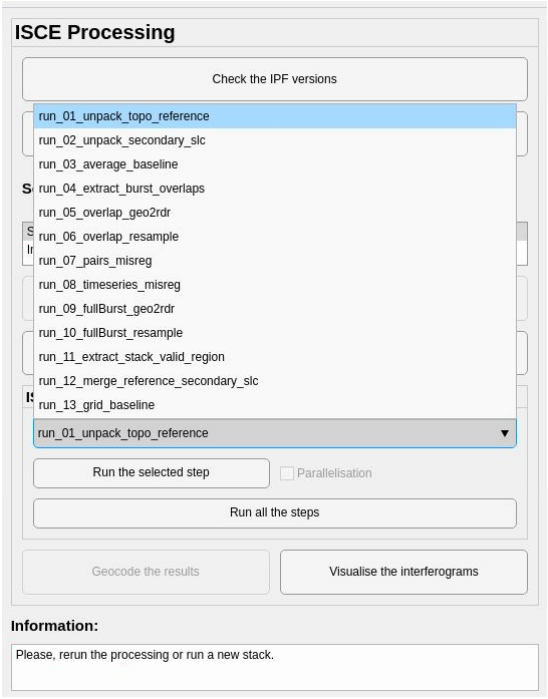


Figure 3.13 Processing steps of ISCE for generating the coregistered SLC stack.

⑦Click "**Visualize the interferograms**" (optional): Open the plotting toolbox to show the processing results (**Figure 3.14**).

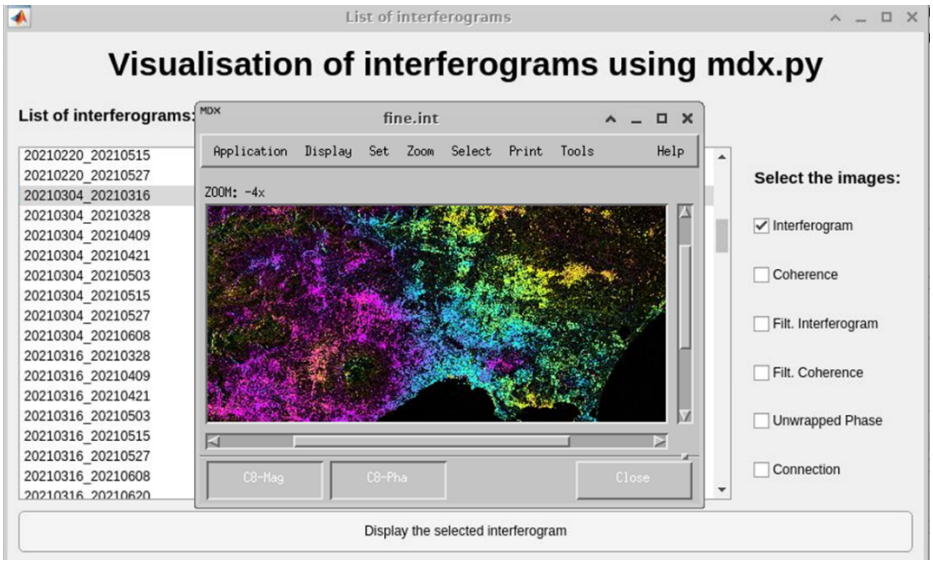


Figure 3.14 The visualization toolbox for checking the ISCE processing results.

3.4 InSAR time-series analysis using StaMPS

StaMPS provides two kinds of processing data processing strategy: the persistent scatterer (PS) approach and the small-baseline (SBAS) approach. The PS approach employs the single-reference network for generating interferograms and do displacement time-series analysis based on the stable PS points, while the SBAS approach employs a multiple-reference network for generating interferograms and do the post-analysis based on distributed points. StaMPS also provides an option of merging the results from the PS and SBAS approach. This will increase the measurement point density and explore both merits of the two approaches. Please refer to the StaMPS help document (see the references in **Part V**) to check the detailed differences of the two approaches.

EZ-INSAR employs the similar data processing workflow (i.e., "preparation of dataset ---> parameter setting---> run analysis") for all the three processing approaches mentioned above. Note the merged approach only applies after the computations using both the PS and SBAS methods. **Figure 3.15** shows the 10 steps of the StaMPS processor module.

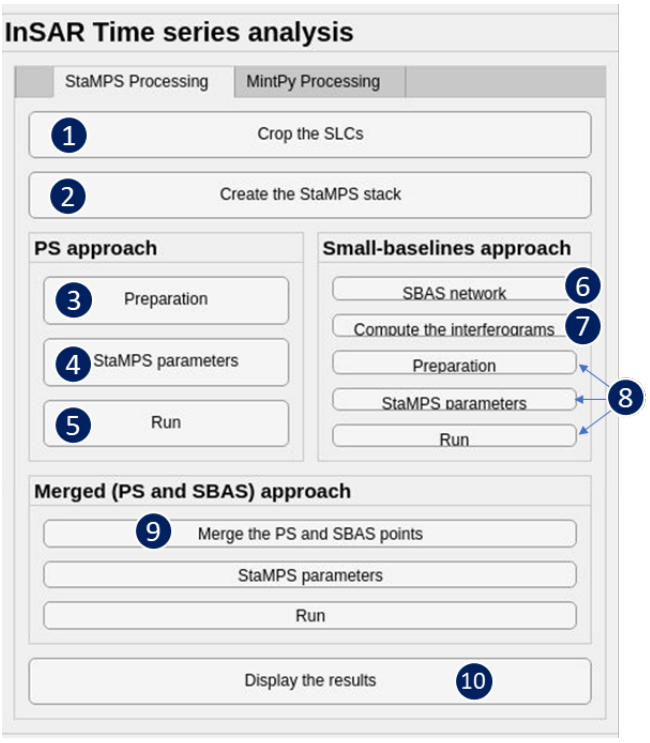


Figure 3.15 The steps for running the StaMPS processer.

① Click "**Crop the SLCs**": The SLC stack from ISCE will be cropped based on the polygon read from the KML file generated in the preparation step (See **Section 3.1**). This is useful because ISCE generates the SLC stack in the format of burst-by-burst, resulting the geographical coverage of the SLC stack generally larger than the defined AOI region. Cropping the SLC stack will accelerate the following data processing.

② Clik "**Create the StaMPS stack**": Input the parameter "**lambda**" (the wavelength of the radar sensor, default for Sentinel-1: 0.055 meter). EZ-INSAR will detect whether the cropping step has been done. A tip message window will be shown with "**YES**" to allow following processing only focusing on the AOI region. If you chose "**No, process the whole region**", the original merged SLC stack will be used (**Figure 3.16A**). Then set the path for storing the StaMPS processing results (**Figure 3.16B**).

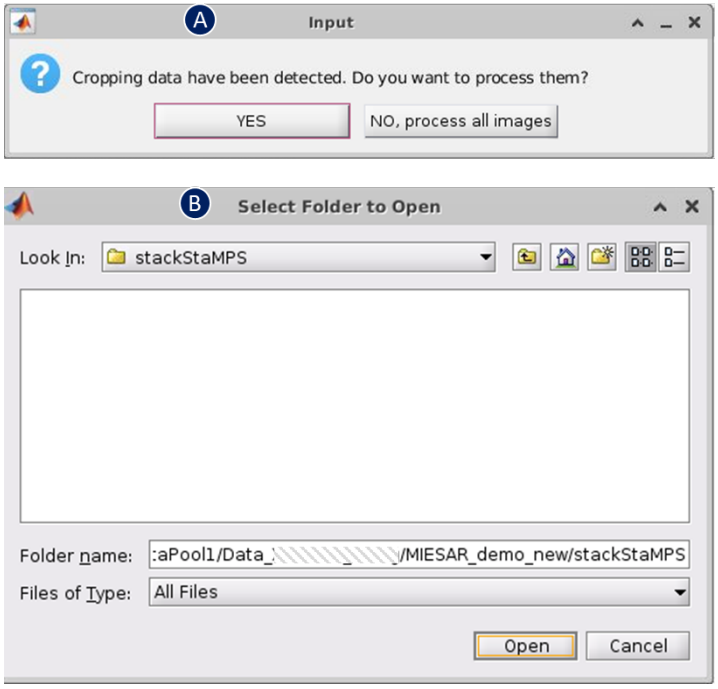


Figure 3.16 Chose the processing region and the result storing path.

Tips!

- If you run StaMPS based on the SLC stack at the first time, a dialog will show to let you select a directory where a directory "InSAR_reference-date" will be created and all the StaMPS processing will be located in this directory.
- If you have already finished the running of StaMPS and clik "**Create the StaMPS stack**" once again, there will be a warning message "*The Stack-StaMPS directory already exist. Do you want to continue?*". Select "**Yes**", and you can define a new directory and re-run the StaMPS processing. If you still specify the original folder, all the files in the old folder will be deleted before the beginning of the new processing.

- **For the PS approach**

③Click "**Preparation**": Set the required parameters for preparing the input dataset of StaMPS through the dialog window (**Figure 3.17**). This step runs the "*make_prepare_sice"* of ISCE and create a directory "stackStaMPS" and a sub-directory "INSAR_reference-date", which contains the sub-patches and datasets for StaMPS processing.

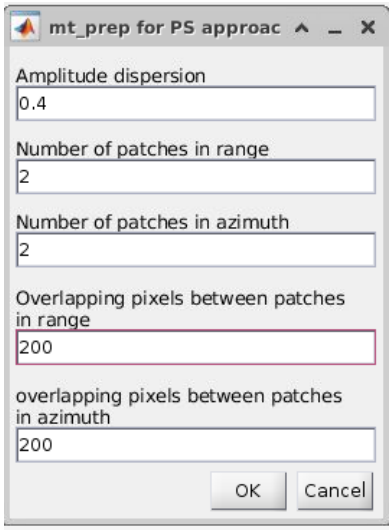


Figure 3.17 Parameter setting for running "*make_prepare_isce*".

④ Click "**StaMPS parameters**": Set the parameters for running StaMPS using the PS approach. Please refer to the StaMPS help document to adjust the parameters and click "**Validate the parameters**" to save the inputs (**Figure 3.18**).

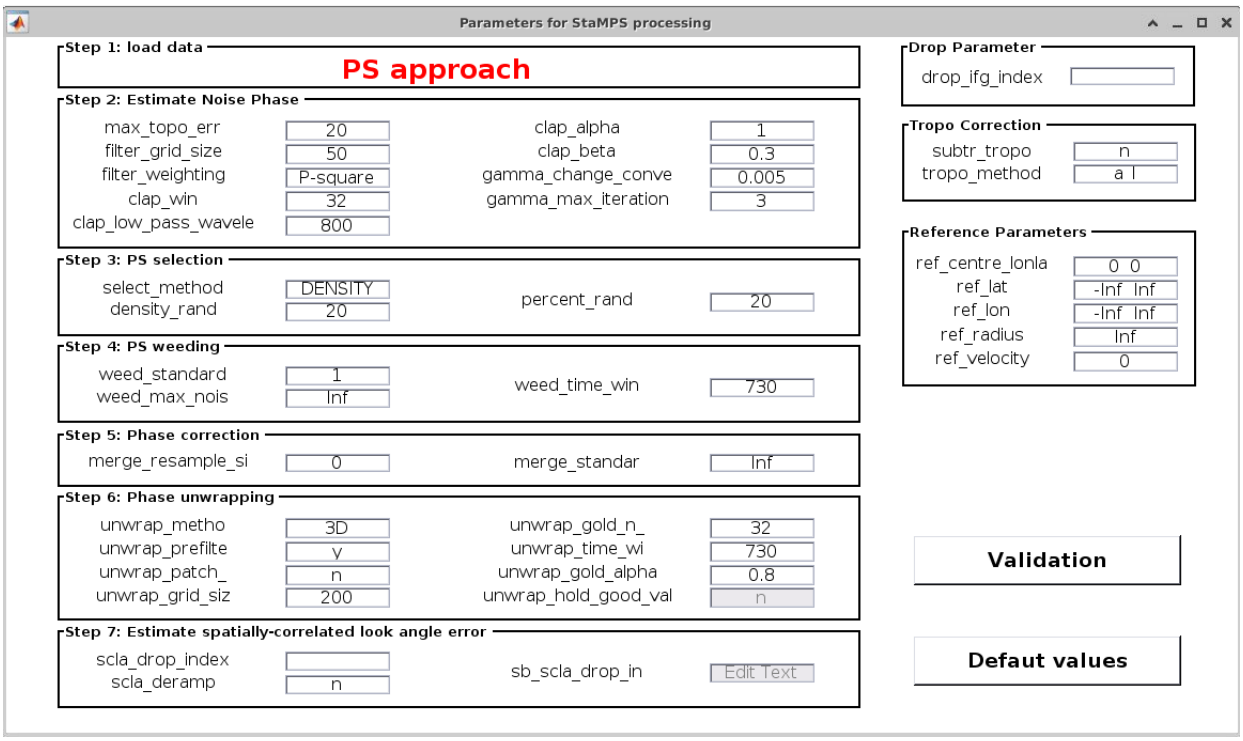


Figure 3.18 Parameter setting for the StaMPS PS approach.

⑤ Click "**Run**": Open a window showing the 8 steps (step 1 to 8) of StaMPS processing. Input the numbers to adjust steps you want to run or just leave it as default to run all the 8 steps (**Figure 3.19**).

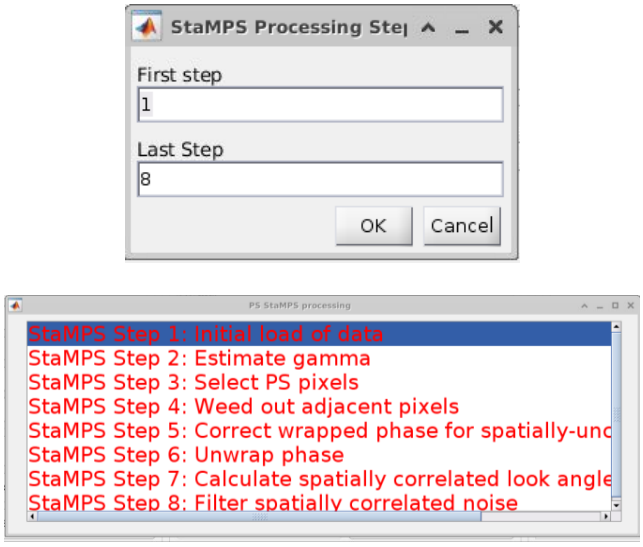


Figure 3.19 The input dialog of running StaMPS from step 1 to 8.

- **For Small-baseline approach**

⑥ Click "**SBAS network**": Check the network of SAR image pairs (**Figure 3.20A**). Users can adjust the network by "**StaMPS tool**" to open a dialog to set the thresholds of coherence, temporal baseline, and perpendicular baseline (**Figure 3.20B**). Users can also click "**Add an ifg**" or "**Remove an ifg**" in the network manager to add or delete the baseline connection manually.

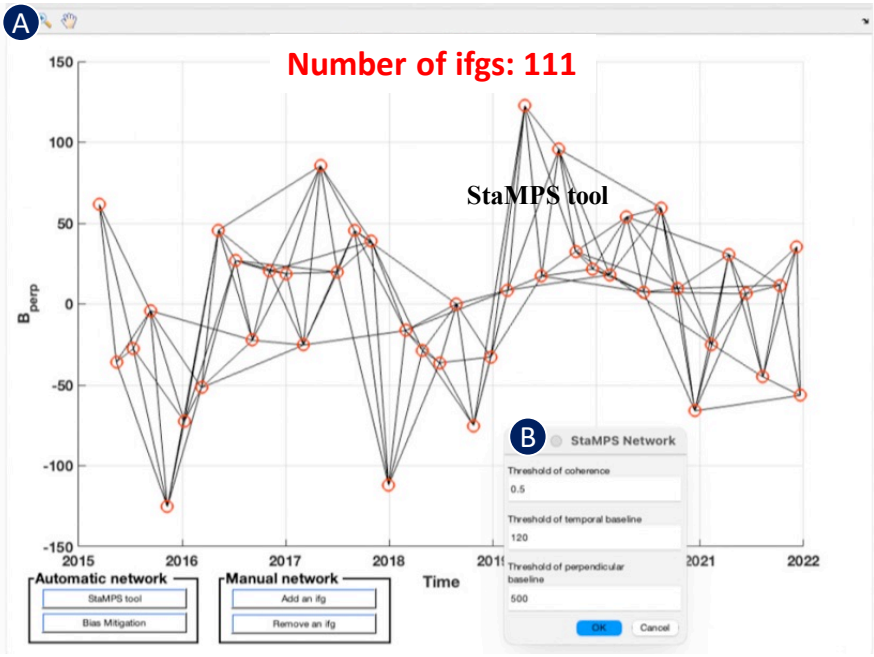


Figure 3.20 Tools for adjusting of the interferogram connection network.

⑦Click "**Compute the interferograms**": Generate the dataset required for SBAS analysis. This will create a folder "SBAS_BASELINE" in the directory "INSAR_reference-date", and the interferograms will be generated by the ISCE script "*make_small_baselines_isce*".

⑧ Click "**Preparation**--> **StaMPS parameters** --> **Run**" in the SBAS panel step-by-step like the PS approach (see Step 3-5).

- **For the Merged approach**

⑨ Click "**Merge the PS and SBAS points**": Create a new subdirectory MERGED in the "INSAR_reference-date" directory. The new "PATCH_X" directories contain the merged measurements points and the related datasets generated by the Step 1-5 from the PS and SBAS approaches. Then click "**StaMPS parameters**" and "**Run**" to finish the processing like the PS and SBAS approaches.

⑩ Click "Display the results": Show the tips for visualizing the StaMPS results.

Tips!

- StaMPS provides a script "ps_plot.m" to visualize the results such as mean velocity, displacement time series. Users can refer to the manual of StaMPS to check the results. For example, go to the "$work_path/stackStaMPS/InSAR_reference" directory, and type the following command in Matlab to show the mean displacement velocity. You can also add the parameter 'ts' for ps_plot.m to show the displacement time series at a selected point on the mean velocity map (**Figure 3.21**):

  >> ps_plot('v-do')

  >> ps_plot('v-do',1,'ts')

- There is also a GUI toolbox termed "viStamps" developed by Sousa et al. (2013) to visualize the StaMPS results. However, the toolbox has not been updated after 2013. Users may refer to the project website (*https://vistamps.utad.pt/?page_id=38*) to download and test the code if they are interested to use it.

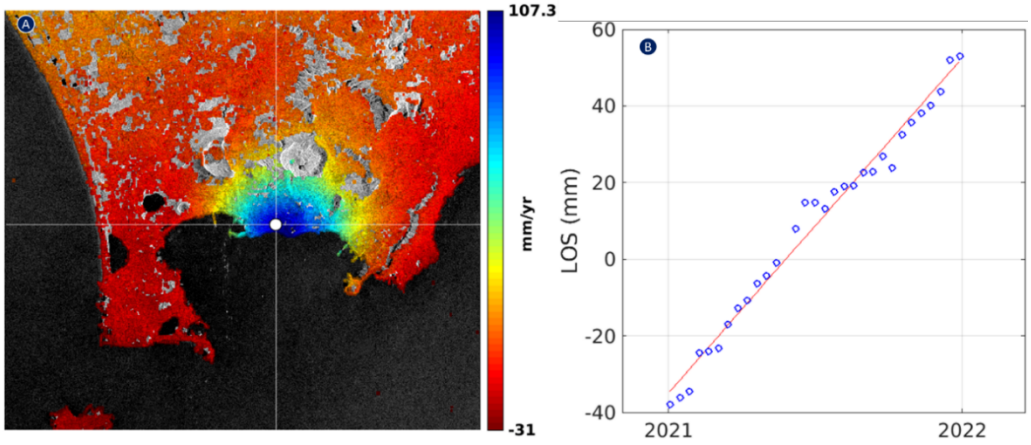


Figure 3.21 (A) Mean displacement velocity over the Campi Flegrei volcano. (B) The displacement time-series at the selected point (white circle).

3.5 InSAR time-series analysis using MintPy

MintPy has a straightforward data processing structure with all the processing controlled by the python application "*smallbaselineApp.py*". The MintPy processor module in EZ-INSAR is thus designed to following the logic of "*smallbaselineApp.py*". Figure 3.22 shows the workflow of MintPy consisting 9 steps in total.

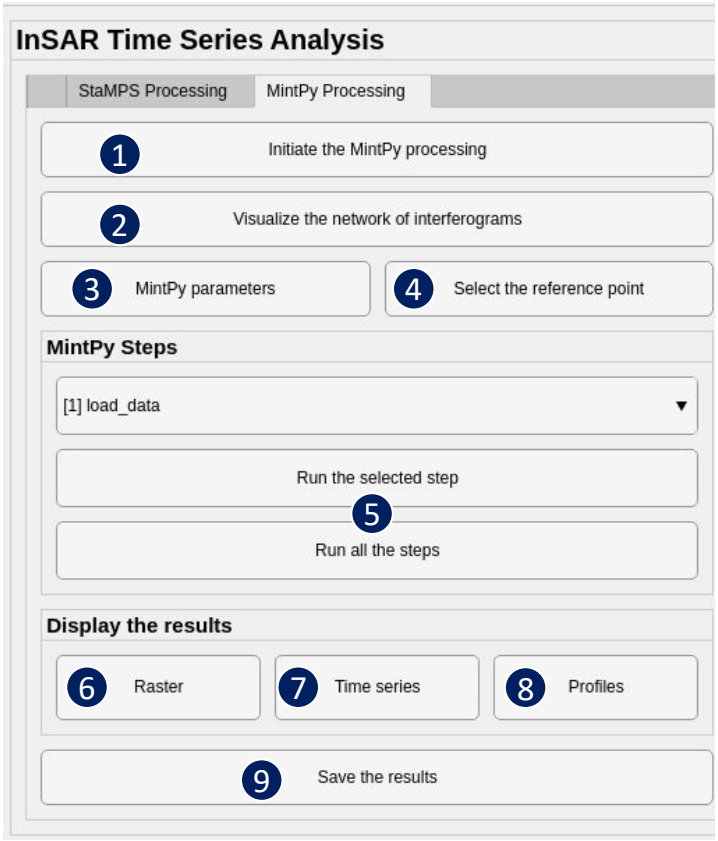
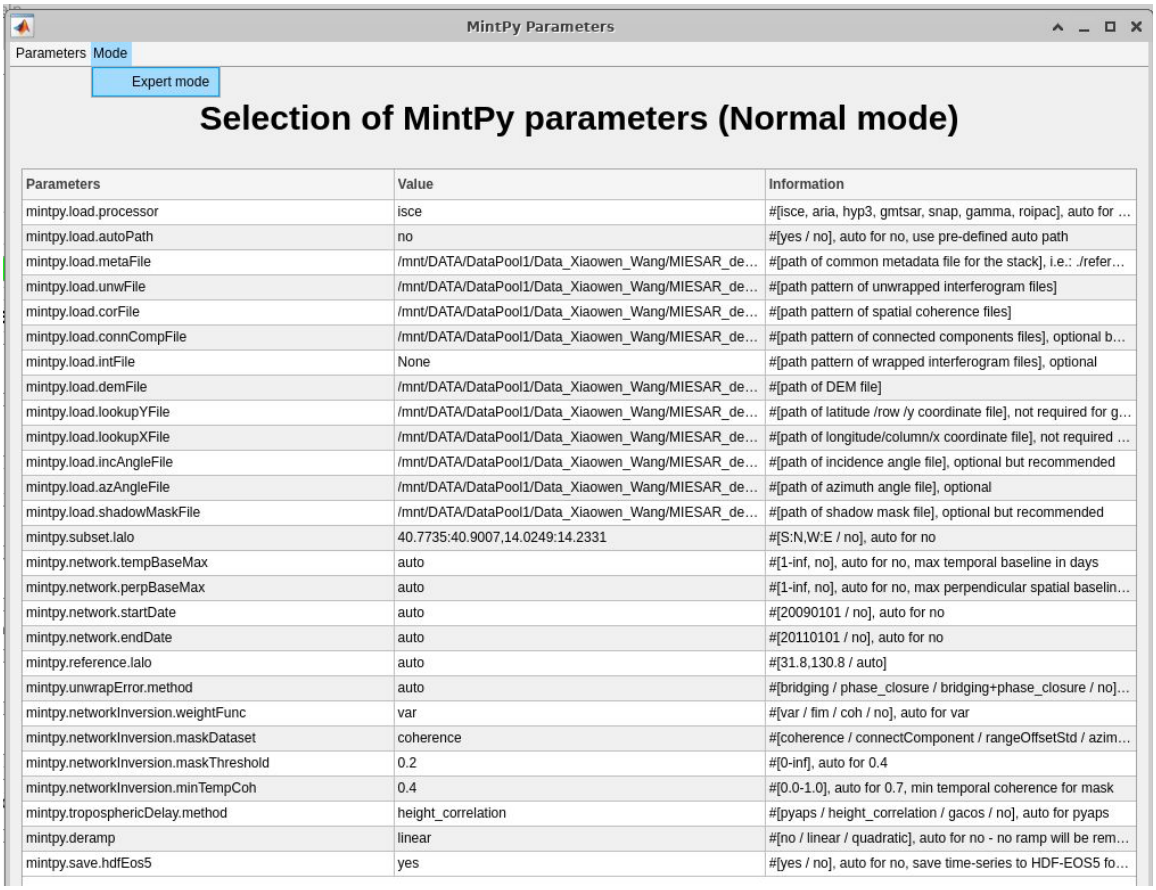


Figure 3.22 The workflow of MintPy interface.

①Click "**Initiate the MintPy processing**": Active the MintPy Processing module. This will create a directory "stackmintpy" in the work path, within which you can see a parameter template file of MintPy.

②Click "**Visualize the network of interferograms**": Check the interferogram network to ensure that the network is fully connected.

③ Click "**MintPy parameters**": Open an interactive window for setting the parameters (**Figure 3.23**). By default, it will show a short list of the selected parameters that are generally important for controlling the processing. One can also click "**Mode** --->**Expert mode**" in the window to check and modify the full list of parameters.



Parameters	Value	Information
mintpy.load.processor	isce	#[isce, aria, hyp3, gmtsar, snap, gamma, roipac], auto for ...
mintpy.load.autoPath	no	#[yes / no], auto for no, use pre-defined auto path
mintpy.load.metaFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path of common metadata file for the stack], i.e.: ./refer...
mintpy.load.unwFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path pattern of unwrapped interferogram files]
mintpy.load.corFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path pattern of spatial coherence files]
mintpy.load.connCompFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path pattern of connected components files], optional b...
mintpy.load.intFile	None	#[path pattern of wrapped interferogram files], optional
mintpy.load.demFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path of DEM file]
mintpy.load.lookupYFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path of latitude /row /y coordinate file], not required for g...
mintpy.load.lookupXFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path of longitude/column/x coordinate file], not required ...
mintpy.load.incAngleFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path of incidence angle file], optional but recommended
mintpy.load.azAngleFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path of azimuth angle file], optional
mintpy.load.shadowMaskFile	/mnt/DATA/DataPool1/Data_Xiaowen_Wang/MIESAR_de...	#[path of shadow mask file], optional but recommended
mintpy.subset.lalo	40.7735:40.9007,14.0249:14.2331	#[S:N,W:E / no], auto for no
mintpy.network.tempBaseMax	auto	#[1-inf, no], auto for no, max temporal baseline in days
mintpy.network.perpBaseMax	auto	#[1-inf, no], auto for no, max perpendicular spatial baselin...
mintpy.network.startDate	auto	#[20090101 / no], auto for no
mintpy.network.endDate	auto	#[20110101 / no], auto for no
mintpy.reference.lalo	auto	#[31.8,130.8 / auto]
mintpy.unwrapError.method	auto	#[bridging / phase_closure / bridging+phase_closure / no]...
mintpy.networkInversion.weightFunc	var	#[var / fim / coh / no], auto for var
mintpy.networkInversion.maskDataset	coherence	#[coherence / connectComponent / rangeOffsetStd / azim...
mintpy.networkInversion.maskThreshold	0.2	#[0-inf], auto for 0.4
mintpy.networkInversion.minTempCoh	0.4	#[0.0-1.0], auto for 0.7, min temporal coherence for mask
mintpy.troposphericDelay.method	height_correlation	#[pyaps / height_correlation / gacos / no], auto for pyaps
mintpy.deramp	linear	#[no / linear / quadratic], auto for no - no ramp will be rem...
mintpy.save.hdfEos5	yes	#[yes / no], auto for no, save time-series to HDF-EOS5 fo...

Figure 3.23 Parameter setting interface for MintPy.

④ Click "**Select reference point**": Select reference point manually with the assistance of optical satellite image (**Figure 3.24**).

Figure 3.24 Selecting the reference point manually.

Tips!

The full processing chain of MintPy has 16 steps. Some steps are optional, for example the correct_SET (Solid Earth Tides correction), which are marked with black text "==> OPT" following the step name. Also, the required steps are marked with green text and "==> OKAY" following the step name.

⑤ Run MintPy processing step-by-step or in a batch:

- Click "**Run the selected step**": Run the processing by choosing a specific step (**Figure 3.25**) .
- Click "**Run all the steps**": Selected the steps to be run and then click "**Launch the processing**".

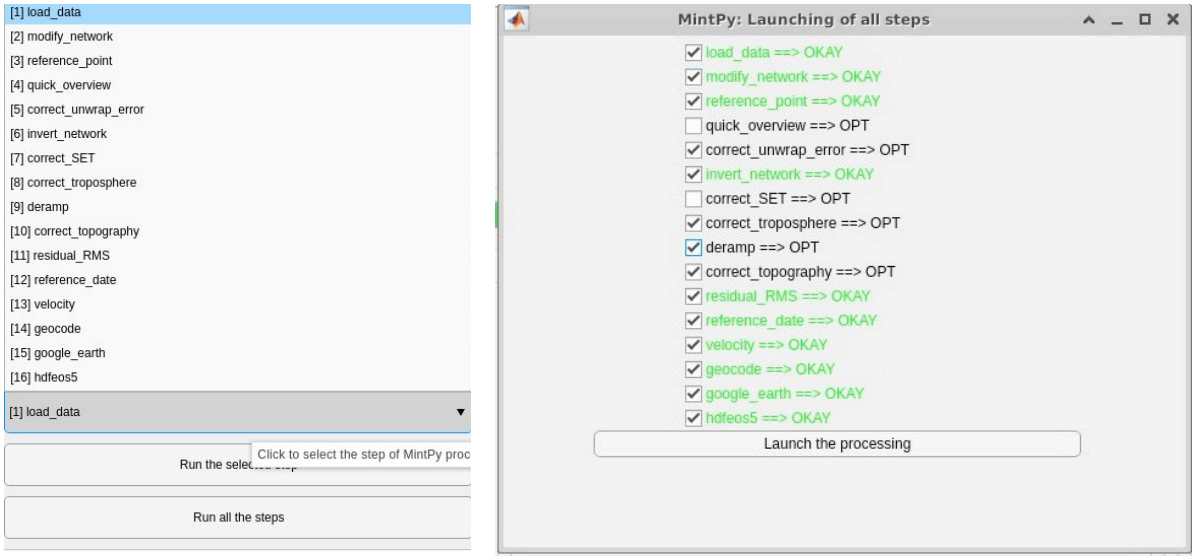


Figure 3.25 Launch the MintPy processing.

Tips!

MintPy provides a set of scripts to easily visualize the 2D raster (e.g., deformation map, residual map) results or time-series results (e.g., displacement time series). EZ-INSAR will call these scripts to visualize the data.

⑥ Click "**Raster**" to open the raster visualization window (**Figure 3.26**). One can select the "**Dataset**" you want to show then adjust the plotting parameters below in the "**Value**" column. The annotations of each parameters are also annotated. Finally, click "**Run**" to show the plot. **Figure 3.27** shows the mean displacement velocity of the demo data over the Campi Flegrei volcano, which is similar to the processing results from StaMPS (**Figure 3.20**).

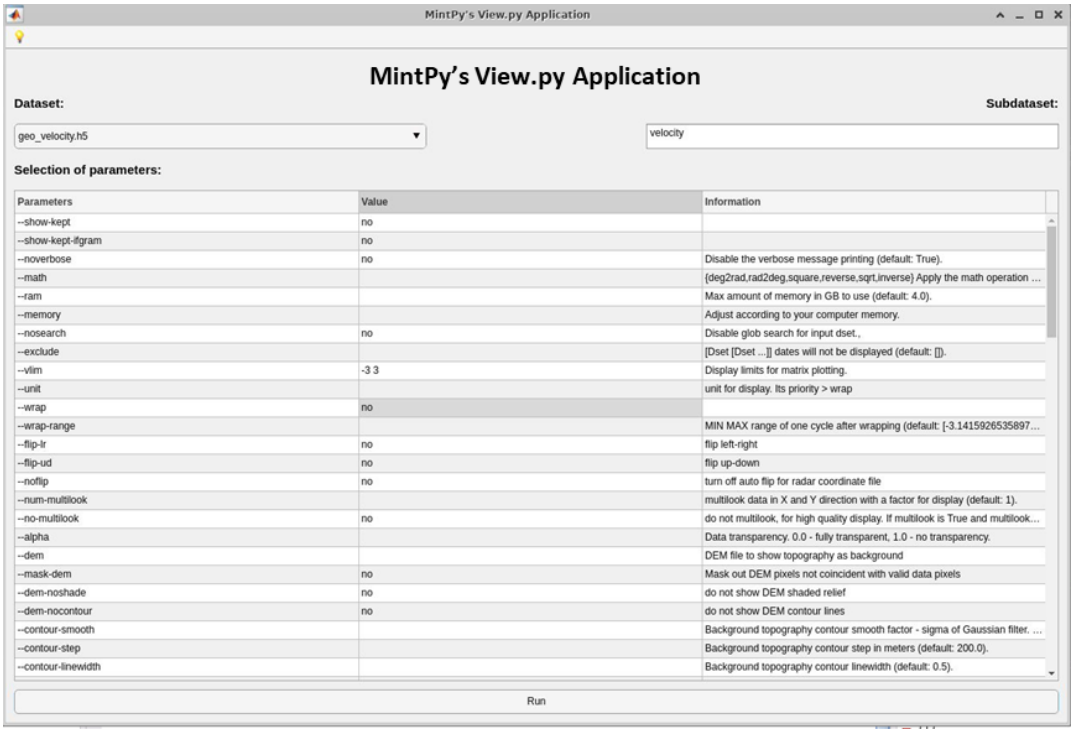


Figure 3.26 The interface for plotting the 2D raster data.

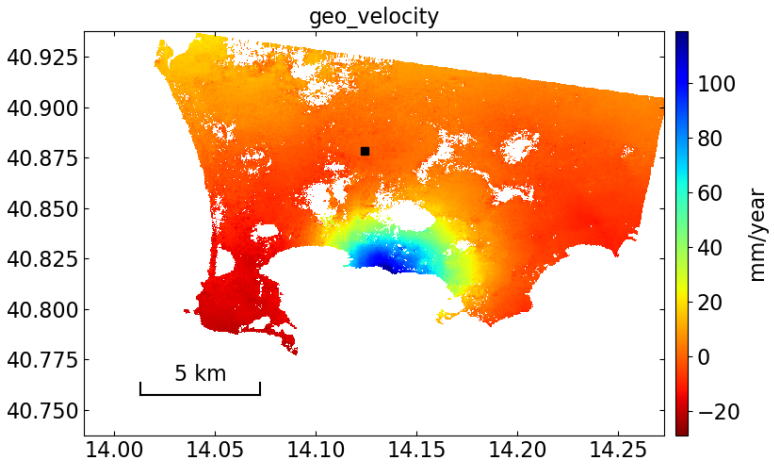


Figure 3.27 The mean surface displacement velocity from MintPy processing.

⑦ Click "**Time series**": Open the time series plotting interface (**Figure 3.28**). Similar to the "**Raster**" display window, one can select the dataset, modify the parameter, and click the "**Run**" button to show the plot. **Figure 29** shows the cumulative displacements between Jan. and Dec. 2021 over the Campi Flegrei volcano.

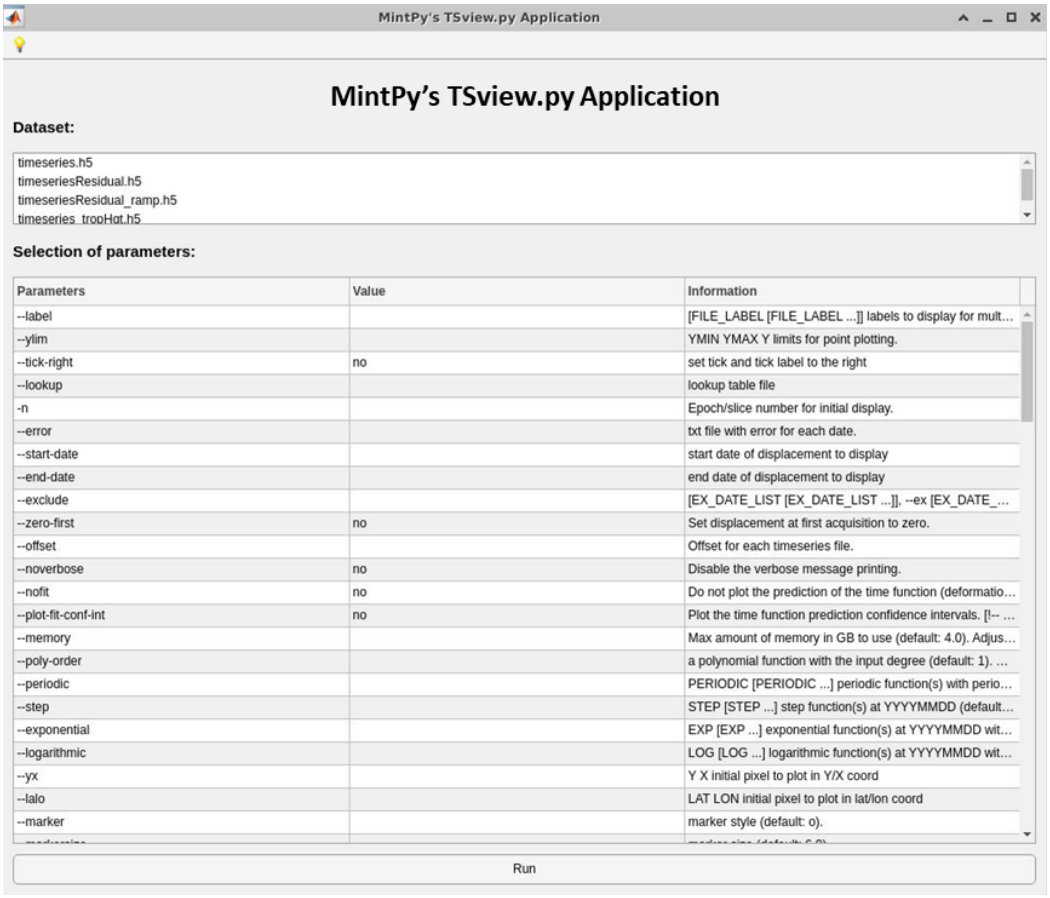


Figure 3.28 The interface for plotting the time series data.

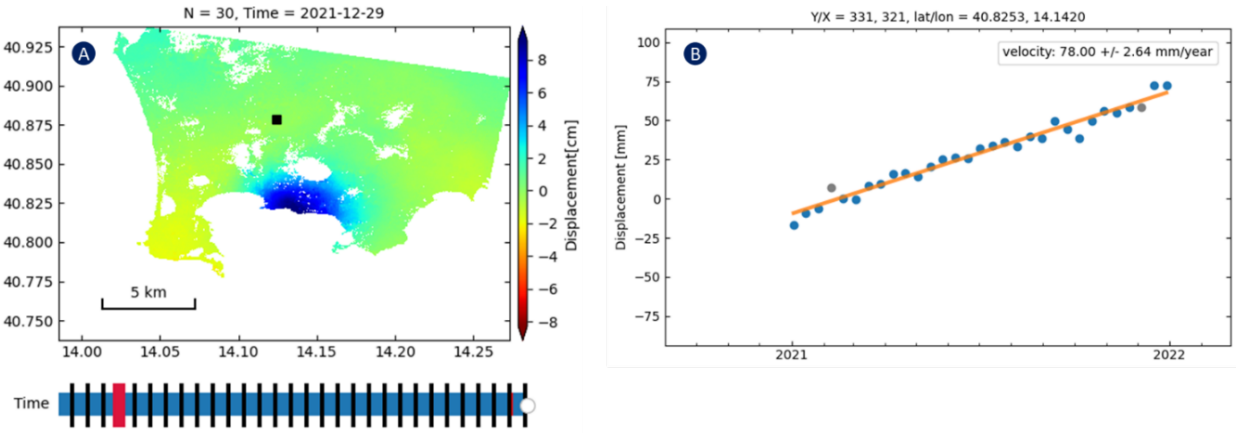


Figure 3.29 The cumulative displacements (A) and displacement time series (B) at the selected point (white circle in A) over the Campi Flegrei volcano from MintPy processing.

⑧ Click "**Profiles**": Open the interface to plot the results along a manually selected profile.

⑨ Click "**Save the results**": Open the interface for saving the results into the desired format (**Figure 3.30**). EZ-INSAR supported the saving of data in formats of GDAL, GMT, hdfeos5, KMZ, and ROIPAC. All the save

resulted are located in the "stackmintpy" directory, and one can visualize or edit the data using the other platform. **Figure 3.31** shows an example of overlapping the save KMZ file (i.e., displacement velocity map) in Google Earth.

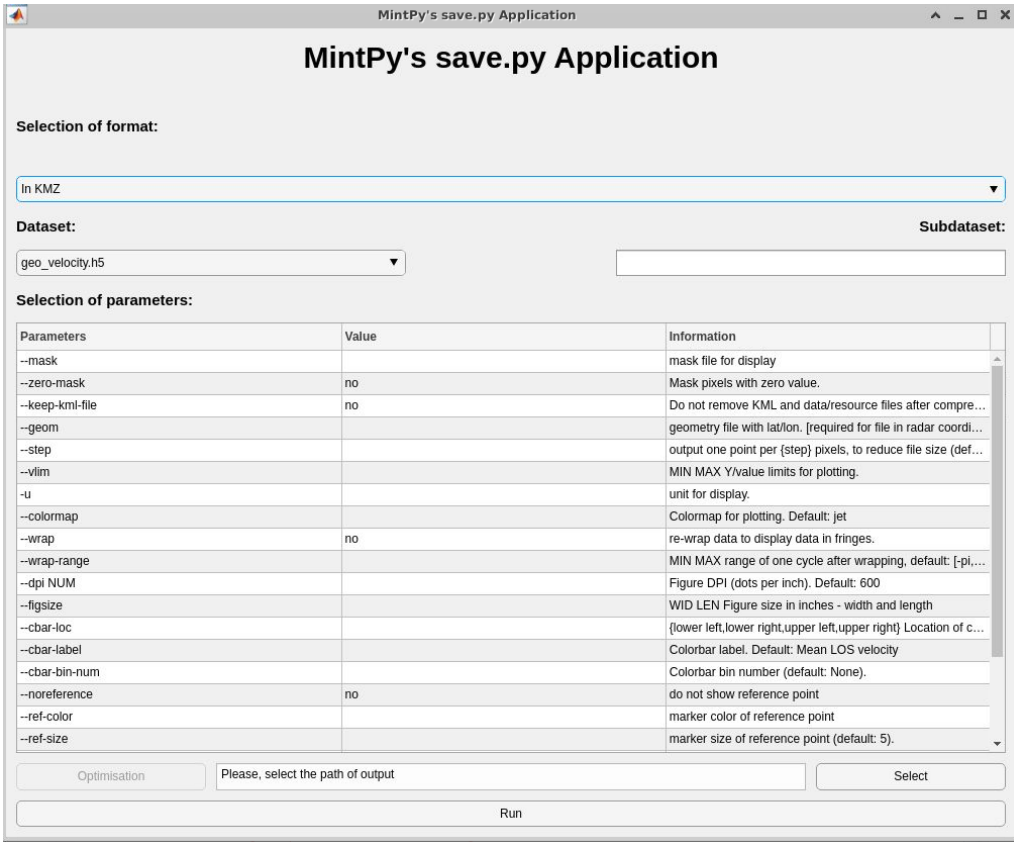


Figure 3.30 The interface for saving the results.

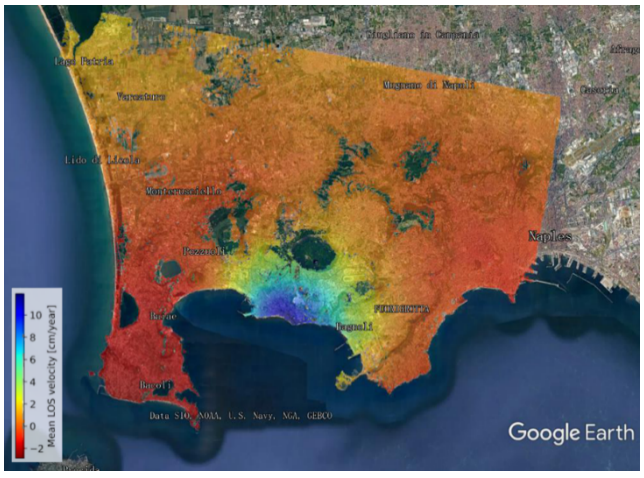


Figure 3.31 The mean displacement velocity overlapping on Google Earth.

Part IV

Processing with the other SAR sensor data

The processing with the other SAR sensor data is same as the Sentinel-1 demo as shown in Part III except for that users must prepare the SAR images in L1 level (SLC) in advance (except for Sentinel-1 Stripmap data, which can be downloaded). In detail, after setting the working path, users just need to put the SAR images into the slc folder. A KML file defining the area of interest is also needed, which can be easily crated with Google Earth.

EZ-InSAR supports the import of the SLC SAR images in several file formats, e.g., **.zip or *.tar or *.tar.gz*. With the SLC images and AOI KML file being prepared, click the buttons below from Step ① to ⑦ to finish the data preparation. Note, the selection of the "Mode" in step ④ should be consist with the data to be imported.

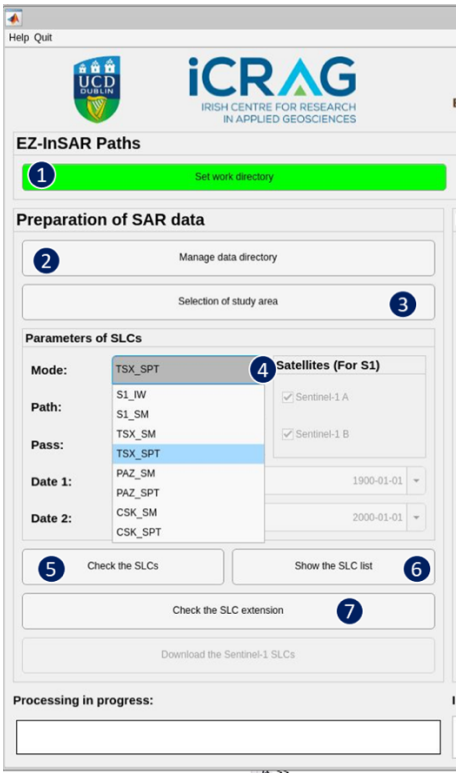


Figure 4.1 The workflow for importing the SAR data.

4.1 For Strimap data

Please follow the instructions in the next table regarding data formats and directories:

Satellite	*Mode*	*EZ-InSAR*	*SLC format*
Sentinel-1	IW	Ready	.zip or .SAFE in the slc directory
Sentinel-1	StripMap	To be tested	.zip or .SAFE in the slc directory
TerraSAR-X or PAZ	StripMap	To be tested	Unzipped PAZ1_* or TSX1_* directory in the slc directory
Cosmo-SkyMed	StripMap	To be tested	[directory of the acquisition]/CSK*.h5 in the slc directory
ALOS2	StripMap	No	NE

Please note that the processing with the Stripmap data has not been fully tested: only the preparation of data has been tested. However, the InSAR processors are compatible with them.

ISCE must change the PATH variable to process the Stripmap data, this is fully implemented in EZ-InSAR.

4.2 For Spotlight data

The instructions for Spotlight data are the same as for Stripmap data. **However, the processing of Spotlight is not ready in ISCE.** EZ-InSAR is therefore ready to prepare/create the stack of coregistered SLCs but it is not possible to estimate the ground surface displacements because of the residual Doppler phase on the Spotlight interferograms.

Part V

Change history

April, 2023 (Version 2.0.2 Beta)

- Fix for the compatibility with the last version of MintPy.

March, 2023 (Version 2.0.1 Beta)

- Some fixes concerning the creation of the Sentinel-1 IW list.

August 19, 2022 (Version 2.0.0 Beta)

- Support of SAR sensors with Stripmap acquisition mode (TerraSAR-X, COSMO-SkyMed, PAZ, Sentinel-1 in Stripmap mode)
- Modification of the main interface

March 01, 2022

- Version 1.0.0 beta

Part VI

Bibliography

Bekaert, D., A. Hooper, & T. Wright, A spatially-variable power-law tropospheric correction technique for InSAR data. J. Geophys. Res., 120, 2015a.

Bekaert, D., R. Walters, T. Wright, A. Hooper, & D. Parker, Statistical comparison of insar tropospheric correction techniques. Remote Sensing of Environment, 170, 40–47, 2015b.

Hooper, A., H. Zebker, P. Segall, & B. Kampes, A new method for measuring deformation on volcanoes and other natural terrains using InSAR persistent scatterers. Geophys. Res. Lett., 31, 2004.

Hooper, A. A multi-temporal InSAR method incorporating both persistent scatterer and small base-line approaches. Geophys. Res. Lett., 35, 2008.

Ortega Rodriguez, A., Carrilho Gomes, R., Correia, V., Pinto, C., Bodó, B., & Cseko, A. Interreg Atlantic Area AGEO Project-Explaining natural hazards and the role of citizen observatories through storytelling. In EGU General Assembly Conference Abstracts, EGU21-12644, 2021.04.

Pasquali, P., Atzori, S., Cantone, A., Riccardi, P., De Filippi, M., & Barbieri, M. SARscape®, a Commercial-Off-The-Shelf Software Package for the Measurement, Monitoring and Modeling of Geophysical Phenomena. In EAGE Workshop on Dead Sea Sinkholes–Causes, Effects and Solutions, European Association of Geoscientists & Engineers, 2012.09.

Perissin, D., Wang, Z., & Wang, T. The SARPROZ InSAR tool for urban subsidence/manmade structure stability monitoring in China. Proceedings of the ISRSE, Sidney, Australia, 1015, 2011.

Rosen, P. A., E. Gurrola, G. F. Sacco, & Zebker H. The insar scientific computing environment, in Synthetic Aperture Radar, 2012. EUSAR. 9th European Conference on, pp. 730–733, VDE, 2012.

Sousa, J. J., Magalhães, L. G., Ruiz, A. M., Sousa, A. M., & Cardoso, G. The viStaMPS tool for visualization and manipulation of time series interferometric results. Computers & Geosciences, 52, 409–421, 2013.

Zhang, Y., Fattahi, Heresh., & Amelung F. Small baseline InSAR time series analysis: Unwrapping error correction and noise reduction. Computers & Geosciences, 133, 104331, 2019.

Useful online help documents:

Andy Hooper, David Bekaert, Ekbal Hussain, and Karsten Spaans. *StaMPS/MTI Manual* (Version 4.1b). Available at: https://github.com/dbekaert/StaMPS/blob/master/Manual/StaMPS_Manual.pdf

David Bekaert, *ISCE to StaMPS Manual (Version 1.0.1).* Available at: https://github.com/isce-framework/isce2-docs/blob/master/StackToStaMPS_Manual/Manual.pdf

David Bekaert, *Toolbox for Reducing Atmospheric InSAR Noise (TRAIN)*, *Version 3beta*. Available at: http://davidbekaert.com/download/TRAIN_manual.pdf

Francisco Delgado. *ISCE2 Tutorial for Geological and Geophysical Applications*. Available at: https://www.overleaf.com/project/5babb2b0a601937d5d27c762

Serco Italia SPA. StaMPS: *Presistent Scatterer Interferometry Processing - Mexico City 2021 (Version 1.1).* Available at: https://rus-copernicus.eu/portal/wp-content/uploads/library/education/training/HAZA12_StaMPsPSI_Processing_Tutorial.pdf