SETP-EMI Program Guide Book

1. Introduction

SETP-EMI is an open source software used for synthetic aperture radar distributed scatterer interferometry (DS-InSAR) analysis. It is suitable for dynamic monitoring of low-coherence surface deformation in vegetation environments, etc. It can greatly improve the signal-to-noise ratio of interference signals and efficiently provide high-precision data sources for geoscience researchers.

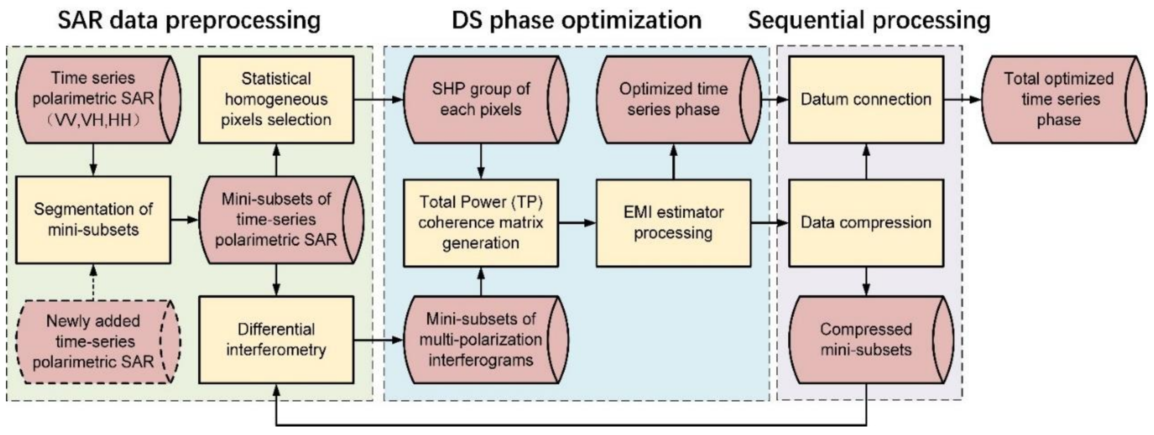
We combine the Sequential Estimator (SE) with Total Power (TP) polarization stacking method and solve it using eigen decomposition-based Maximum Likelihood Estimator (EMI), named SETP-EMI. Moreover, the SETP-EMI algorithm has dynamic processing capabilities, and there is no need to repeatedly process historical data after adding new SLC data.

2. Function introduction

The SETP-EMI method workflow is depicted in Figure 1. It comprises the following steps: (1) Divide the time-series multi-polarization SAR data into mini-subsets based on the user-defined number of images. Phase optimization will be performed on these mini-subsets and the compressed dataset. (2) Calculate the differential interferograms for each polarimetric channel from the co-registered SLC images. At the same time, we identify statistically homogeneous pixels within each subset. (3) Combine the multi-polarization subsets to create a complex Pauli basis scattering vector and the homogeneous pixels group to generate TP coherency matrices. (4) Apply the EMI estimator method to optimize the interferometric phase and obtain the optimized phases. (5) Utilize the optimized interferometric phases to compress the original multi-polarization Pauli basis scattering vectors. (6) Repeat steps (2) to (5) for the subsequent subsets in the dataset. (7) Perform EMI estimation on the optimized and compressed interferometric phases of all subsets, establish the reference connection, and obtain the total optimized time series phases.

By integrating multi-polarization SAR data and sequential estimation, this method balances accuracy and efficiency. The advantage lies in the ability to process new SAR data without repeating the analysis of previously processed one, thereby significantly improving the processing efficiency. Secondly, utilizing multi-polarization data offers redundant observation samples that lead to an overall enhanced phase quality once optimized. Last but not least, the identification of SHP sets is performed in each subset, which ensures the homogeneity of the estimated samples under long-term dynamic changes scenarios.

The specific details of the SETP-EMI method are elaborated below.



SAR data preprocessing
DS phase optimization
Sequential processing
Time series polarimetric SAR (VV,VH,HH)
Statistical homogeneous pixels selection
SHP group of each pixels
Optimized time series phase
Datum connection
Total optimized time series phase
Segmentation of mini-subsets
Mini-subsets of time-series polarimetric SAR
Total Power (TP) coherence matrix generation
EMI estimator processing
Data compression
Newly added time-series polarimetric SAR
Differential interferometry
Mini-subsets of multi-polarization interferograms
Compressed mini-subsets

Figure 1. The workflow of SETP-EMI algorithm.

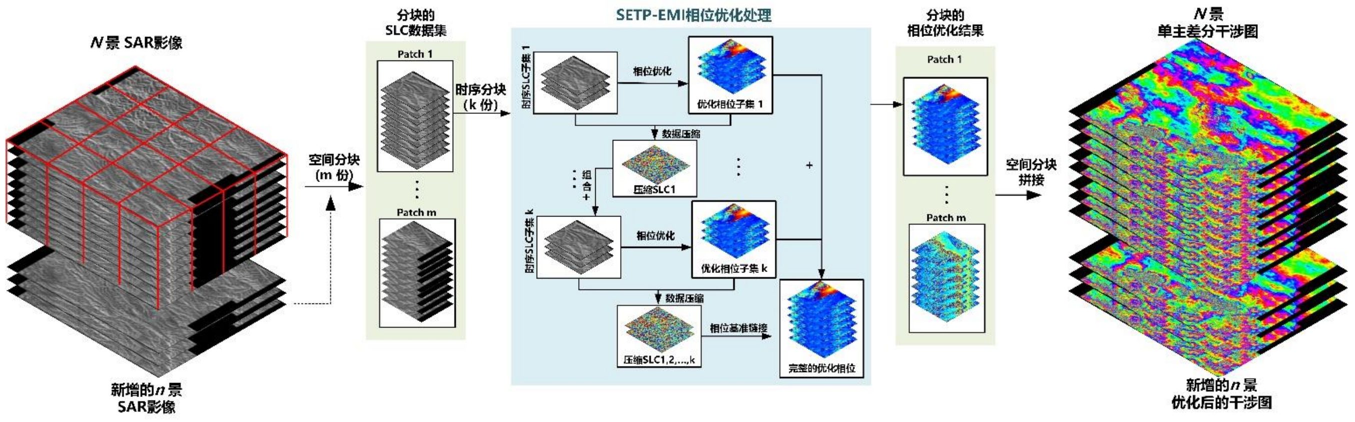


Figure 2 workflow diagram of SETP-EMI Program

The modular processing flow of the program is as follows:

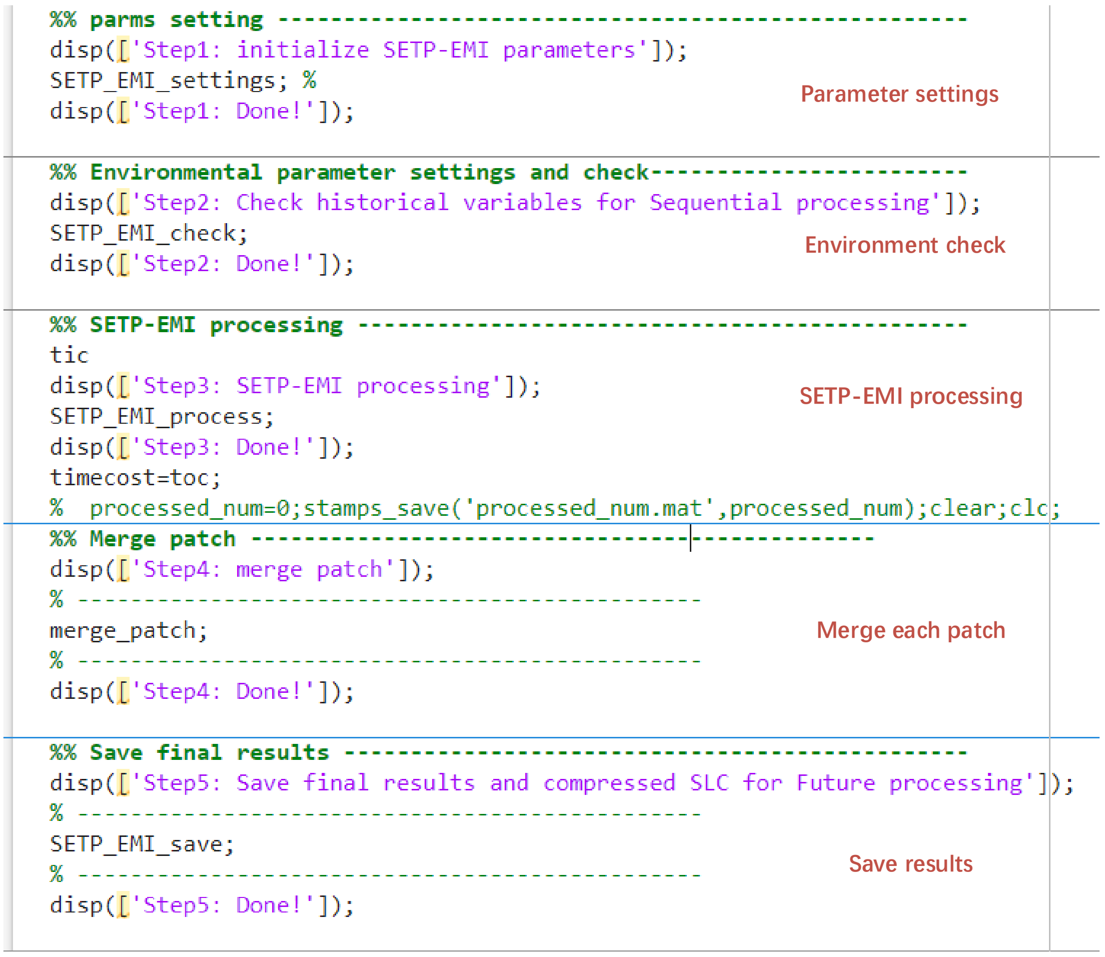
(1) Step one: Users need to define processing parameters and modify the SETP_EMI_settings file.

(2) Step 2: The program checks whether the environment settings meet the requirements, and creates the chunked processing folder patch according to the SETP_EMI_settings parameter to store the chunked result files.

(3) Step 3: Perform SETP-EMI processing on each sub-block and save the results in each patch folder.

(4) Step 4: Mosaic the results of the sub-blocks.

(5) Step 5: Save the final result；



```
%% parms setting -------------------------------------------------------
disp(['Step1: initialize SETP-EMI parameters']);
SETP_EMI_settings; %
disp(['Step1: Done!']);
```

```
%% Environmental parameter settings and check-------------------------
disp(['Step2: Check historical variables for Sequential processing']);
SETP_EMI_check;
disp(['Step2: Done!']);
```

```
%% SETP-EMI processing ----------------------------------------------
tic
disp(['Step3: SETP-EMI processing']);
SETP_EMI_process;
disp(['Step3: Done!']);
timecost=toc;
%  processed_num=0;stamps_save('processed_num.mat',processed_num);clear;clc;
%% Merge patch ----------------------------------------------------
disp(['Step4: merge patch']);
% ----------------------------------------------
merge_patch;
% ----------------------------------------------
disp(['Step4: Done!']);
```

```
%% Save final results -----------------------------------------------
disp(['Step5: Save final results and compressed SLC for Future processing']);
% ----------------------------------------------
SETP_EMI_save;
% ----------------------------------------------
disp(['Step5: Done!']);
```

Figure 3 demo_SETP_EMI

2.1 Input and output

SETP-EMI selected the GAMMA software format as the default input format, including intensity image sequences (float), single master differential interferograms sequences (complex) and SLC sequences (complex) with the byte order being big-endian.

Since multi-polarization data is used for phase optimization, this program supports dual-polarization (HH+VV), cross-polarization (VV+VH or HH+VH) and full-polarization (VV+HH+VH) data input. In addition, this program also supports using only single polarization data (VV or HH). When using single polarization data, the processing flow of the program is similar to the sequential EMI algorithm, except that the SHP can be dynamically updated for estimation.

Users such as SNAP and ISCE can preprocess the data according to the variable definitions in Table 1, and then use this program to input.

Input	**Description**	**Type**	**Remark**	**Output**
Time series intensity images	$\lvert S_i \rvert^2$	Float	$\lvert S_i \rvert^2$ represents a single-look complex image, i represents the i-th scene image	-
Time series differential interferograms	$\frac{s_1 \cdot (s_2)^* \cdot e^{-j\phi_{12}}}{\lvert s_1 \rvert \cdot \lvert s_2 \rvert}$	Complex float	$*$ represents conjugate multiplication, ϕ_{12} represents the terrain phase of images 1 and 2. Interferograms of a single master image is required here.	Optimized interferograms
Time series SLCs	S_i	Complex float	represents a single-look complex image.	Compressed SLC

The SETP-EMI output files are phase-optimized single-main differential interferograms, compressed SLC images, and processed image count files. Single master differential interferograms can be directly used in subsequent time-series DInSAR processing or other interferogram-based procedures. The compressed SLC image is used to assist in phase optimization when adding new data later. The image count file is used to record the number of images that have been processed.

2.2 Code explanation

2.1 Step1: initialize SETP-EMI parameters

(1) Working path settings:

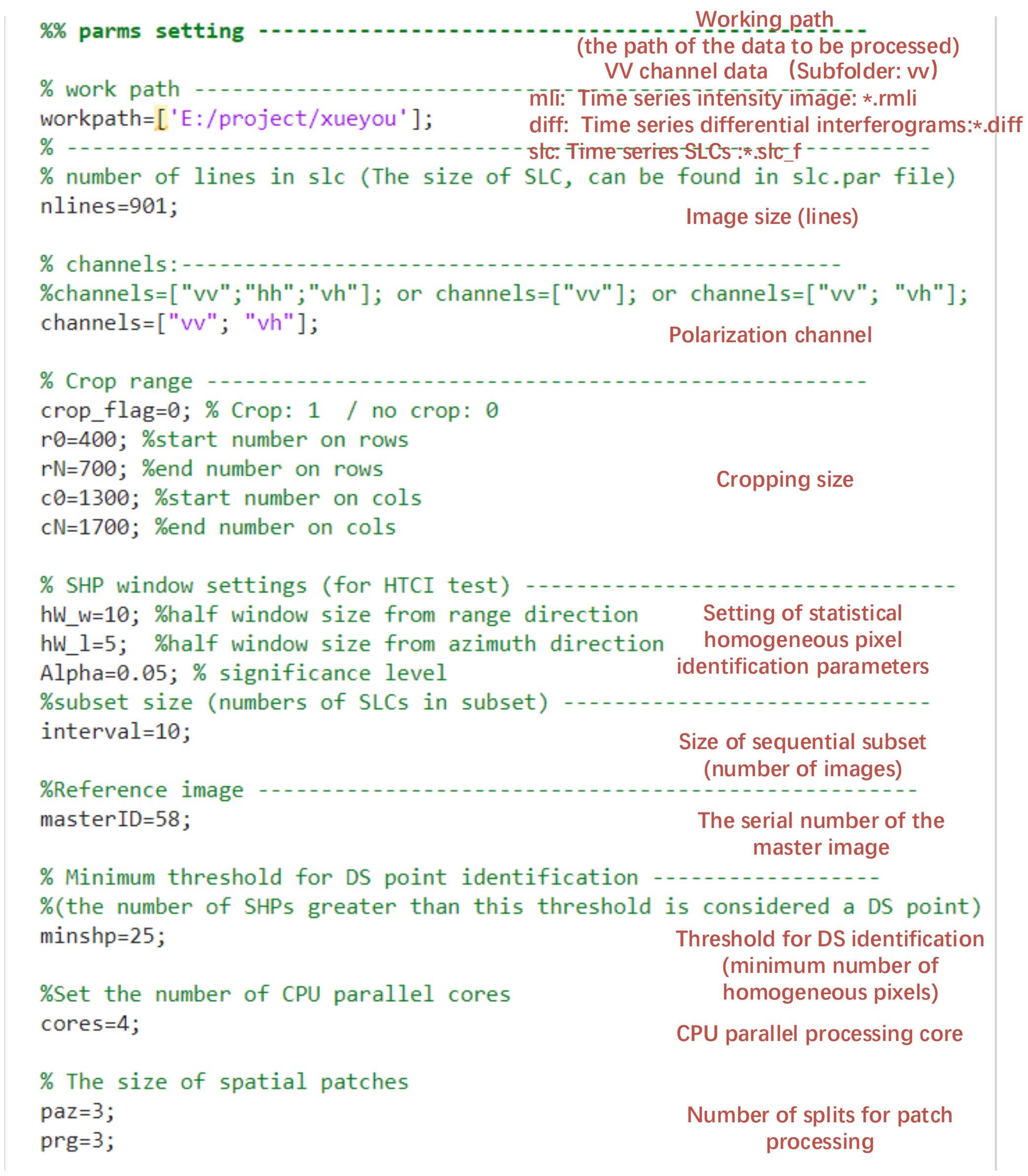
Create a folder (lowercase) with the name of the polarization channel in the workpath path, such as "vv". Then create 3 subfolders in the "vv" folder, named rmli, diff and slc respectively. The format of “vh” or “hh”channel file are same as “vv”.

- The rmli folder stores the time series SAR intensity images, and the naming format is "yyyymmdd.rmli".
- The diff folder stores the time series differential interference pattern based on the super master image, and the naming format is "yyyymmdd_yyyymmdd.diff".
- The time series SLC is stored in the slc folder, and the naming format is "yyyymmdd.rslc_f".

(2) Data processing parameters:

It is necessary to set the image size nlines (number of lines), polarization channel (channel) type channels to be processed, cropping range (crop_flag=0 is not cropped by default), SHP recognition parameter settings (window size and significance level), and the size of the time series subset intercal (The default is 10 scenes as a subset), refer to the serial number of the main image master_ID (the sequence number of the main image in the time series image, such as scene 58), and the SHP number threshold minshp for DS phase optimization (when the number of SHP is less than the threshold, This pixel does not perform phase optimization), the number of CPU cores for parallel processing, and the number of divisions for block processing (paz is the number of divisions in the azimuth direction, prg is the number of divisions in the distance direction)

After setting the parameters, the program can read the data in sequence according to the user-defined sequential subset size and perform subsequent processing.



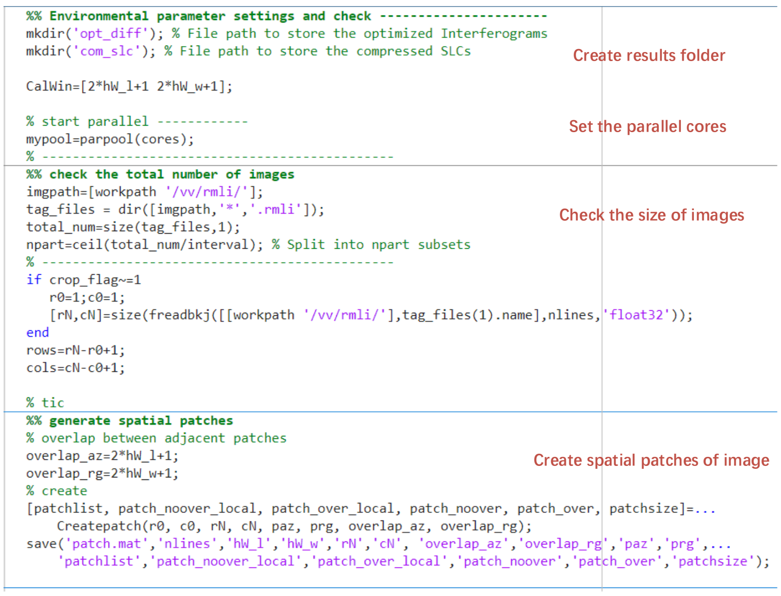
```
%% parms setting ------------------------------------------------

% work path ---------------------------------------------------------
workpath=['E:/project/xueyou'];
% ---------------------------------------------------------------------------
% number of lines in slc (The size of SLC, can be found in slc.par file)
nlines=901;

% channels:----------------------------------------------------
%channels=["vv";"hh";"vh"]; or channels=["vv"]; or channels=["vv"; "vh"];
channels=["vv"; "vh"];

% Crop range ---------------------------------------------------
crop_flag=0; % Crop: 1  / no crop: 0
r0=400; %start number on rows
rN=700; %end number on rows
c0=1300; %start number on cols
cN=1700; %end number on cols

% SHP window settings (for HTCI test) ----------------------------------
hW_w=10; %half window size from range direction
hW_l=5;  %half window size from azimuth direction
Alpha=0.05; % significance level
%subset size (numbers of SLCs in subset) ----------------------------
interval=10;

%Reference image ------------------------------------------------------
masterID=58;

% Minimum threshold for DS point identification ------------------
%(the number of SHPs greater than this threshold is considered a DS point)
minshp=25;

%Set the number of CPU parallel cores
cores=4;

% The size of spatial patches
paz=3;
prg=3;
```

Working path
(the path of the data to be processed)
VV channel data （Subfolder: vv)
mli: Time series intensity image: *.rmli
diff: Time series differential interferograms:*.diff
slc: Time series SLCs :*.slc_f
Image size (lines)
Polarization channel
Cropping size
Setting of statistical homogeneous pixel identification parameters
Size of sequential subset (number of images)
The serial number of the master image
Threshold for DS identification (minimum number of homogeneous pixels)
CPU parallel processing core
Number of splits for patch processing

Figure 4 SETP_EMI_settings

2.2 Step2: Environmental parameter settings and check

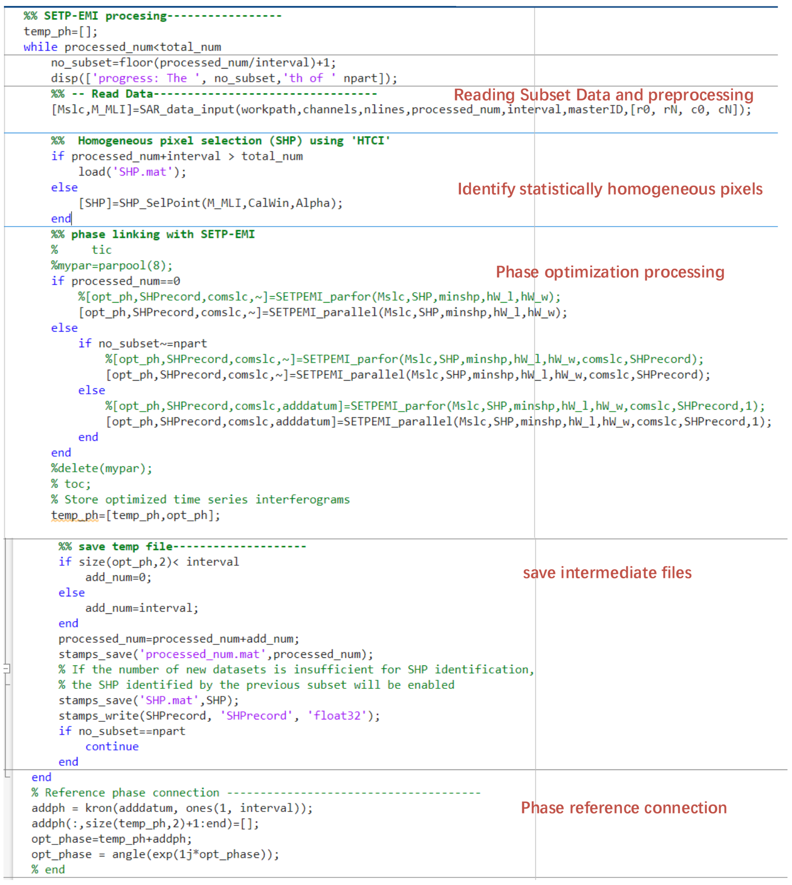
Check the file environment, create a result folder, set the parallel cores, create spatial patches, and prepare for new data processing.



```
%% Environmental parameter settings and check ----------------------
mkdir('opt_diff'); % File path to store the optimized Interferograms
mkdir('com_slc'); % File path to store the compressed SLCs

CalWin=[2*hW_l+1 2*hW_w+1];

% start parallel ------------
mypool=parpool(cores);
% ---------------------------------------------
%% check the total number of images
imgpath=[workpath '/vv/rmli/'];
tag_files = dir([imgpath,'*','.rmli']);
total_num=size(tag_files,1);
npart=ceil(total_num/interval); % Split into npart subsets
% ---------------------------------------------
if crop_flag~=1
   r0=1;c0=1;
   [rN,cN]=size(freadbkj([[workpath '/vv/rmli/'],tag_files(1).name],nlines,'float32'));
end
rows=rN-r0+1;
cols=cN-c0+1;

% tic
%% generate spatial patches
% overlap between adjacent patches
overlap_az=2*hW_l+1;
overlap_rg=2*hW_w+1;
% create
[patchlist, patch_noover_local, patch_over_local, patch_noover, patch_over, patchsize]=...
    Createpatch(r0, c0, rN, cN, paz, prg, overlap_az, overlap_rg);
save('patch.mat','nlines','hW_l','hW_w','rN','cN', 'overlap_az','overlap_rg','paz','prg',...
    'patchlist','patch_noover_local','patch_over_local','patch_noover','patch_over','patchsize');
```

Figure 5 SETP_EMI_check

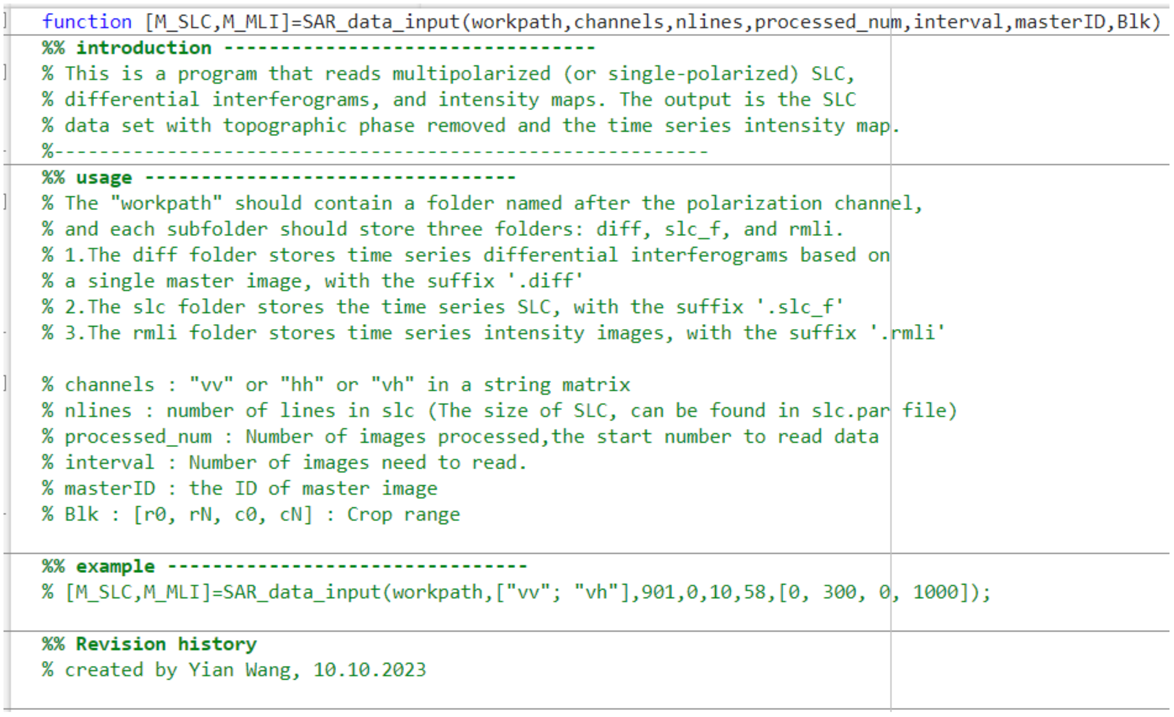
2.3 Step3: SETP-EMI processing


Reading Subset Data and preprocessing
Identify statistically homogeneous pixels
Phase optimization processing
save intermediate files
Phase reference connection

```
%% SETP-EMI procesing----------------
temp_ph=[];
while processed_num<total_num
    no_subset=floor(processed_num/interval)+1;
    disp(['progress: The ', no_subset,'th of ' npart]);
    %% -- Read Data---------------------------------
    [Mslc,M_MLI]=SAR_data_input(workpath,channels,nlines,processed_num,interval,masterID,[r0, rN, c0, cN]);

    %%  Homogeneous pixel selection (SHP) using 'HTCI'
    if processed_num+interval > total_num
        load('SHP.mat');
    else
        [SHP]=SHP_SelPoint(M_MLI,CalWin,Alpha);
    end
    %% phase linking with SETP-EMI
    %     tic
    %mypar=parpool(8);
    if processed_num==0
        %[opt_ph,SHPrecord,comslc,~]=SETPEMI_parfor(Mslc,SHP,minshp,hW_l,hW_w);
        [opt_ph,SHPrecord,comslc,~]=SETPEMI_parallel(Mslc,SHP,minshp,hW_l,hW_w);
    else
        if no_subset~=npart
            %[opt_ph,SHPrecord,comslc,~]=SETPEMI_parfor(Mslc,SHP,minshp,hW_l,hW_w,comslc,SHPrecord);
            [opt_ph,SHPrecord,comslc,~]=SETPEMI_parallel(Mslc,SHP,minshp,hW_l,hW_w,comslc,SHPrecord);
        else
            %[opt_ph,SHPrecord,comslc,adddatum]=SETPEMI_parfor(Mslc,SHP,minshp,hW_l,hW_w,comslc,SHPrecord,1);
            [opt_ph,SHPrecord,comslc,adddatum]=SETPEMI_parallel(Mslc,SHP,minshp,hW_l,hW_w,comslc,SHPrecord,1);
        end
    end
    %delete(mypar);
    % toc;
    % Store optimized time series interferograms
    temp_ph=[temp_ph,opt_ph];

     %% save temp file--------------------
     if size(opt_ph,2)< interval
         add_num=0;
     else
         add_num=interval;
     end
     processed_num=processed_num+add_num;
     stamps_save('processed_num.mat',processed_num);
     % If the number of new datasets is insufficient for SHP identification,
     % the SHP identified by the previous subset will be enabled
     stamps_save('SHP.mat',SHP);
     stamps_write(SHPrecord, 'SHPrecord', 'float32');
     if no_subset==npart
         continue
     end
end
% Reference phase connection --------------------------------------
addph = kron(adddatum, ones(1, interval));
addph(:,size(temp_ph,2)+1:end)=[];
opt_phase=temp_ph+addph;
opt_phase = angle(exp(1j*opt_phase));
% end
```

Figure 6 SETP_EMI_process

① Read slc, diff, and mli data and calculate the SLC after removing the terrain phase

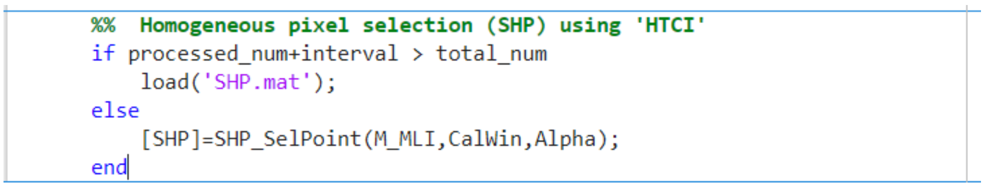


```
function [M_SLC,M_MLI]=SAR_data_input(workpath,channels,nlines,processed_num,interval,masterID,Blk)
%% introduction --------------------------------
% This is a program that reads multipolarized (or single-polarized) SLC,
% differential interferograms, and intensity maps. The output is the SLC
% data set with topographic phase removed and the time series intensity map.
%-------------------------------------------------------
%% usage --------------------------------
% The "workpath" should contain a folder named after the polarization channel,
% and each subfolder should store three folders: diff, slc_f, and rmli.
% 1.The diff folder stores time series differential interferograms based on
% a single master image, with the suffix '.diff'
% 2.The slc folder stores the time series SLC, with the suffix '.slc_f'
% 3.The rmli folder stores time series intensity images, with the suffix '.rmli'

% channels : "vv" or "hh" or "vh" in a string matrix
% nlines : number of lines in slc (The size of SLC, can be found in slc.par file)
% processed_num : Number of images processed,the start number to read data
% interval : Number of images need to read.
% masterID : the ID of master image
% Blk : [r0, rN, c0, cN] : Crop range

%% example --------------------------------
% [M_SLC,M_MLI]=SAR_data_input(workpath,["vv"; "vh"],901,0,10,58,[0, 300, 0, 1000]);

%% Revision history
% created by Yian Wang, 10.10.2023
```

Figure 7 SAR_data_input

The purpose of reading time series intensity images is to identify SHP and dynamically update SHP. The purpose of reading the differential interference pattern and SLC is to remove the topographic phase contribution in the SLC image. In this way, the interferograms generated when SLC performs interference combination will not contain the terrain phase, which is beneficial to ensuring the stability of the local phase. Since the terrain phase is triangularly closed, removing the terrain phase in advance has no effect on the phase optimization estimation.

② Statistically homogeneous pixel identification in subset

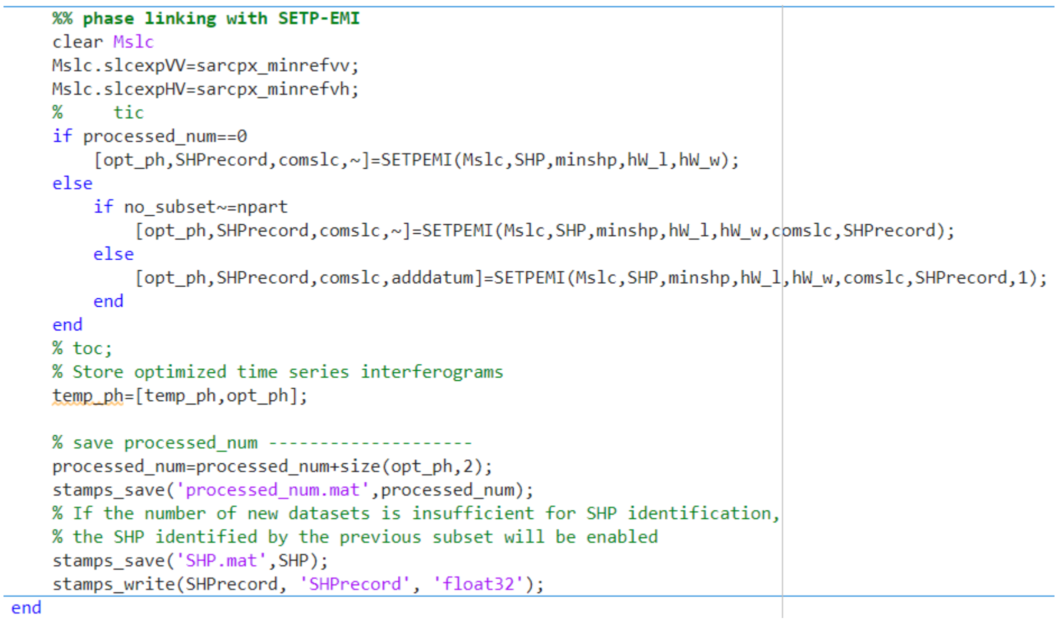


```
%%  Homogeneous pixel selection (SHP) using 'HTCI'
if processed_num+interval > total_num
    load('SHP.mat');
else
    [SHP]=SHP_SelPoint(M_MLI,CalWin,Alpha);
end
```

Figure 8 SHP_SelPoint_HTCI

Mi Jiang 's SHP recognition program is used here to perform SHP recognition based on intensity images. Considering the GLR test combined with polarization information can obtain a more accurate SHP set in small data sets.

③ Phase linking



```
    %% phase linking with SETP-EMI
    clear Mslc
    Mslc.slcexpVV=sarcpx_minrefvv;
    Mslc.slcexpHV=sarcpx_minrefvh;
    %     tic
    if processed_num==0
        [opt_ph,SHPrecord,comslc,~]=SETPEMI(Mslc,SHP,minshp,hW_l,hW_w);
    else
        if no_subset~=npart
            [opt_ph,SHPrecord,comslc,~]=SETPEMI(Mslc,SHP,minshp,hW_l,hW_w,comslc,SHPrecord);
        else
            [opt_ph,SHPrecord,comslc,adddatum]=SETPEMI(Mslc,SHP,minshp,hW_l,hW_w,comslc,SHPrecord,1);
        end
    end
    % toc;
    % Store optimized time series interferograms
    temp_ph=[temp_ph,opt_ph];

    % save processed_num --------------------
    processed_num=processed_num+size(opt_ph,2);
    stamps_save('processed_num.mat',processed_num);
    % If the number of new datasets is insufficient for SHP identification,
    % the SHP identified by the previous subset will be enabled
    stamps_save('SHP.mat',SHP);
    stamps_write(SHPrecord, 'SHPrecord', 'float32');
end
```

Figure 9 SETPEMI_parallel

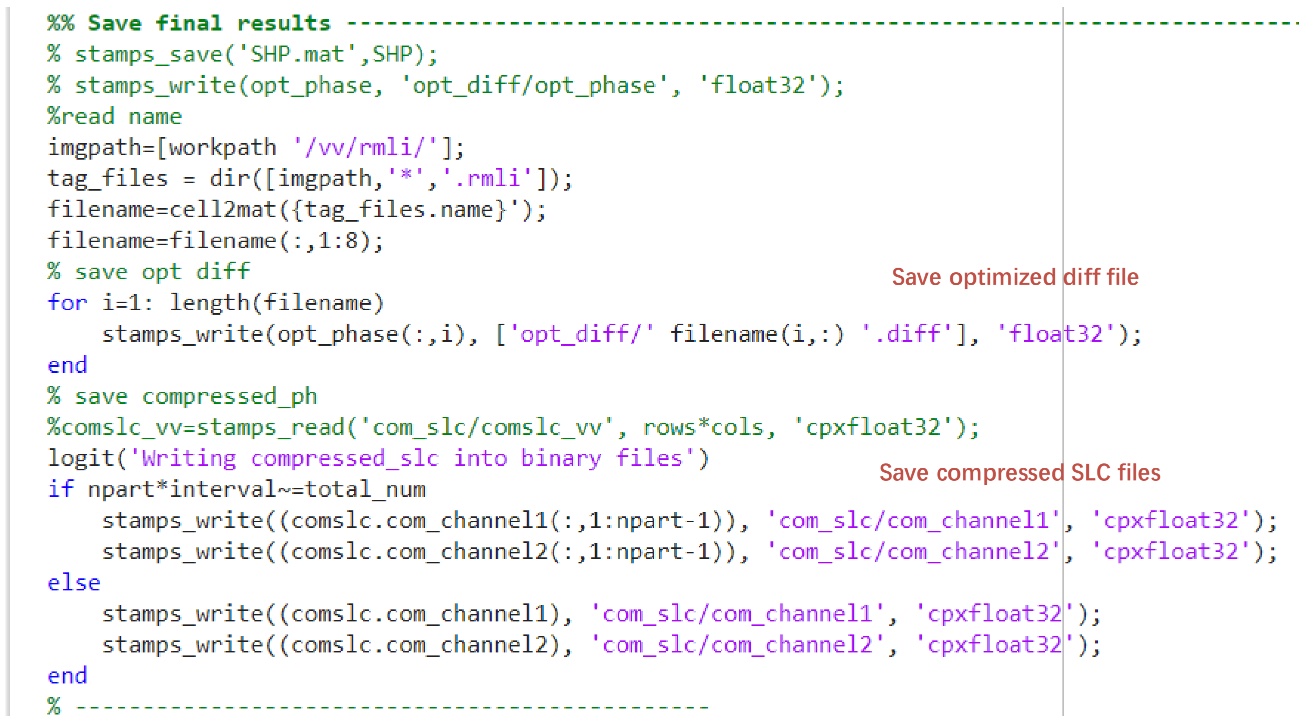
Here, the structure 'Mslc' is used to store the preprocessed SLC data as the input source of the SETPEMI function. The SETPEMI function supports the sequential processing of single-polarization or multi-polarization data sets, generating optimized interferometric phases, SHP point number records, and Compressed SLC. During sequential processing, the compressed SLC and the newly added subset of data are jointly processed for phase optimization. This ensures full connectivity of the interference pattern to avoid the impact of systematic bias caused by multi-looking. During the last sequential subset processing, the phase optimization estimation of the compressed SLC is performed to obtain the reference connection phase.

⑤ Datum connection

Calculate the phase offset between the compressed SLC data sets and reconnect the optimized timing phases of each sub-data set. After sequential phase optimization of the phases of each sequential subset, the phase references of each subset are connected to form a complete time-series differential interference phase.

2.4 Step4: Save final results and compressed SLC for Future processing

Finally, the optimized phase and compressed SLC are stored in binary form in a new folder.



```
%% Save final results ------------------------------------------------------------------
% stamps_save('SHP.mat',SHP);
% stamps_write(opt_phase, 'opt_diff/opt_phase', 'float32');
%read name
imgpath=[workpath '/vv/rmli/'];
tag_files = dir([imgpath,'*','.rmli']);
filename=cell2mat({tag_files.name}');
filename=filename(:,1:8);
% save opt diff
for i=1: length(filename)
    stamps_write(opt_phase(:,i), ['opt_diff/' filename(i,:) '.diff'], 'float32');
end
% save compressed_ph
%comslc_vv=stamps_read('com_slc/comslc_vv', rows*cols, 'cpxfloat32');
logit('Writing compressed_slc into binary files')
if npart*interval~=total_num
    stamps_write((comslc.com_channel1(:,1:npart-1)), 'com_slc/com_channel1', 'cpxfloat32');
    stamps_write((comslc.com_channel2(:,1:npart-1)), 'com_slc/com_channel2', 'cpxfloat32');
else
    stamps_write((comslc.com_channel1), 'com_slc/com_channel1', 'cpxfloat32');
    stamps_write((comslc.com_channel2), 'com_slc/com_channel2', 'cpxfloat32');
end
% -----------------------------------------------
```

Figure 10 SETP_EMI_save

3. Demo

3.1 Data introduction

An unstable slope in the reservoir area of a hydropower station was selected as an example. We collected 117 scenes of Sentinel-1 dual polarization data (VV and VH) from January 2, 2019 to November 06, 2022.

3.2 Data processing results

3.2.1 Step1 and Step2 Parameter settings and check

See Section 2.2 for parameter settings.

3.2.2 Step3: SETP-EMI processing

① Data preprocessing result

The figure below shows the preprocessed SLC data set. It can be seen that the SLC pairwise interferograms with the topographic phase removed can directly obtain a differential interferograms that is consistent with the original differential interferograms. Such an operation ensures the phase stability of the DS local area without destroying the triangle closure of the interference phase.

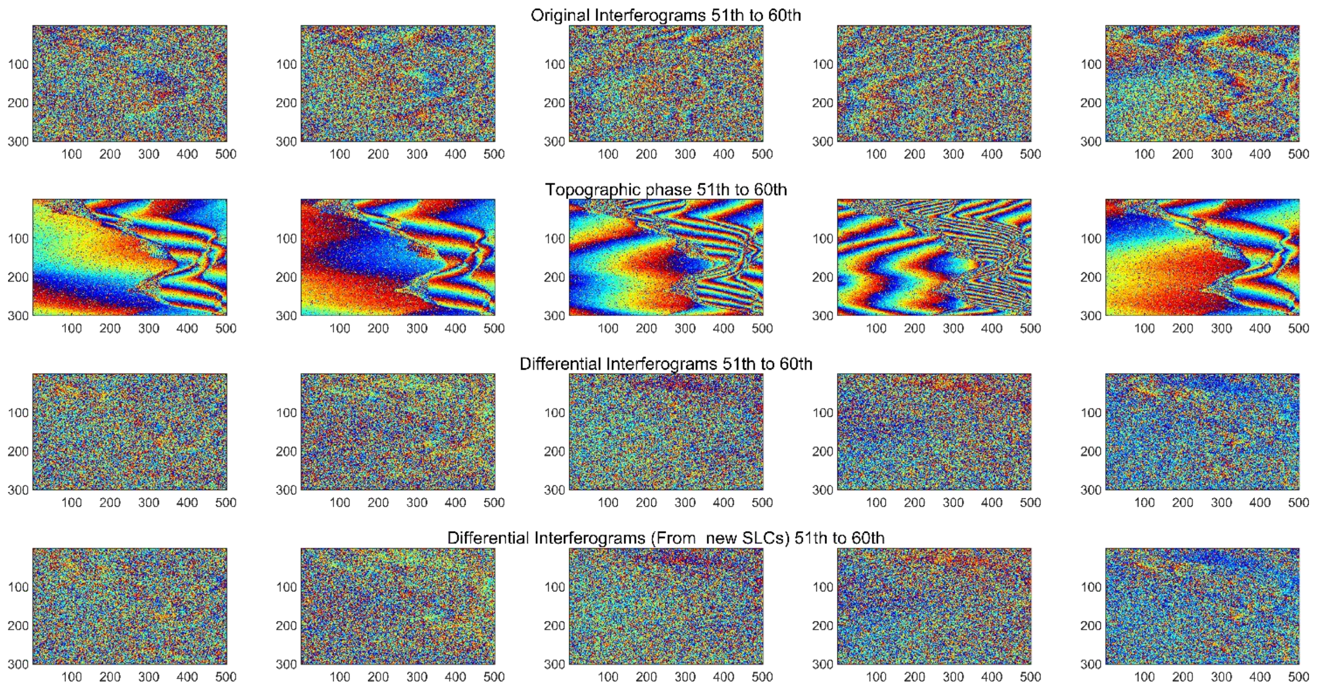


Figure 11 Interferograms with terrain phase removed

② Statistically homogeneous pixel identification

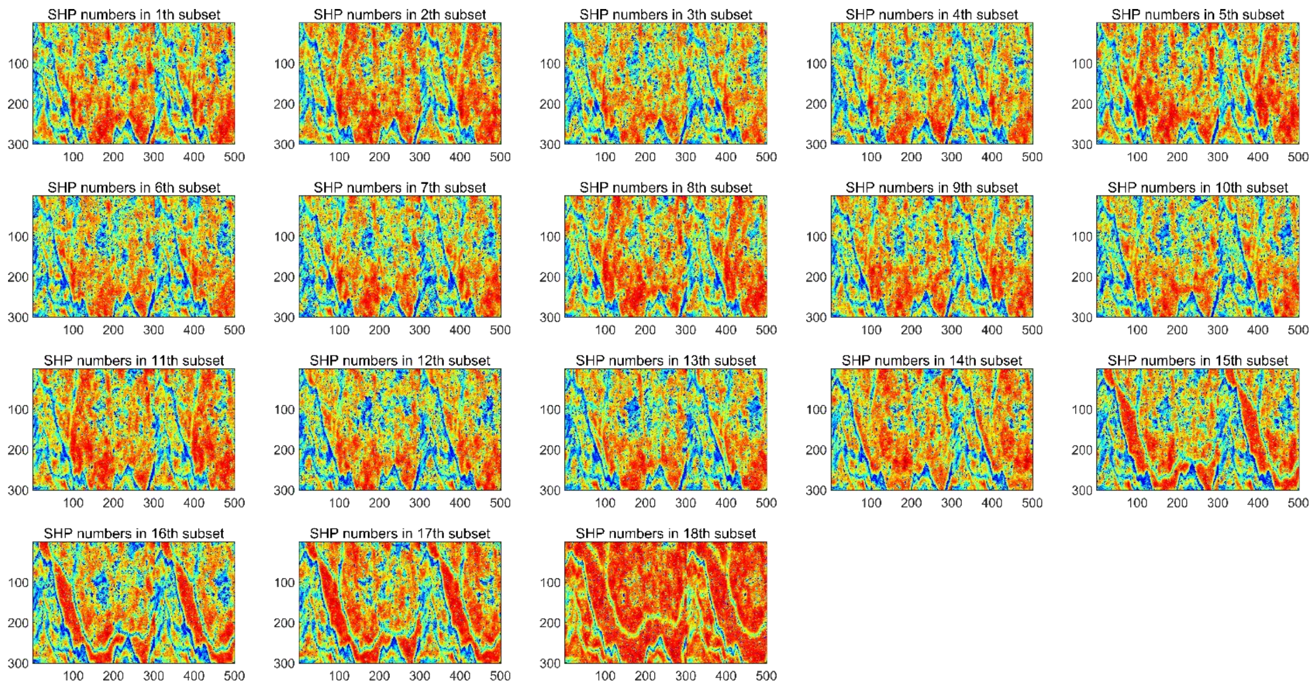


Figure 12 Number of Statistically homogeneous samples

③ **Phase linking**

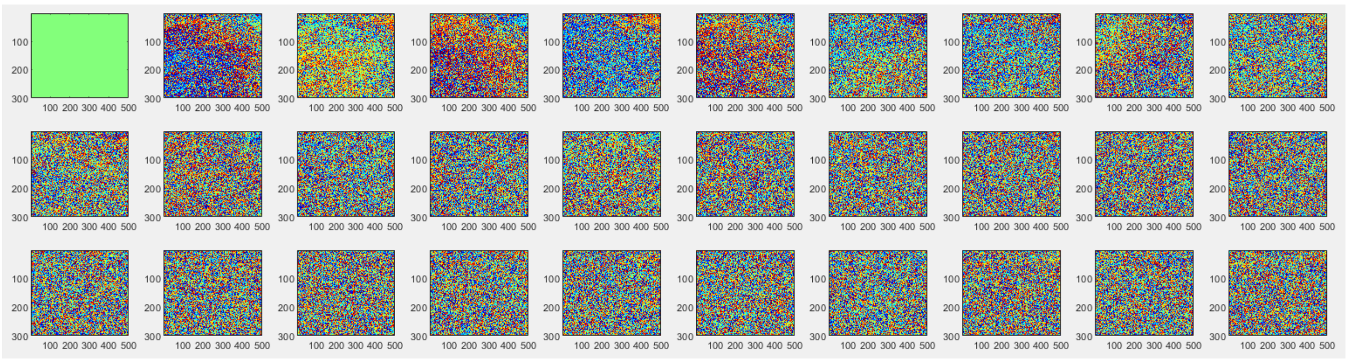


Figure 13 Original differential interferograms

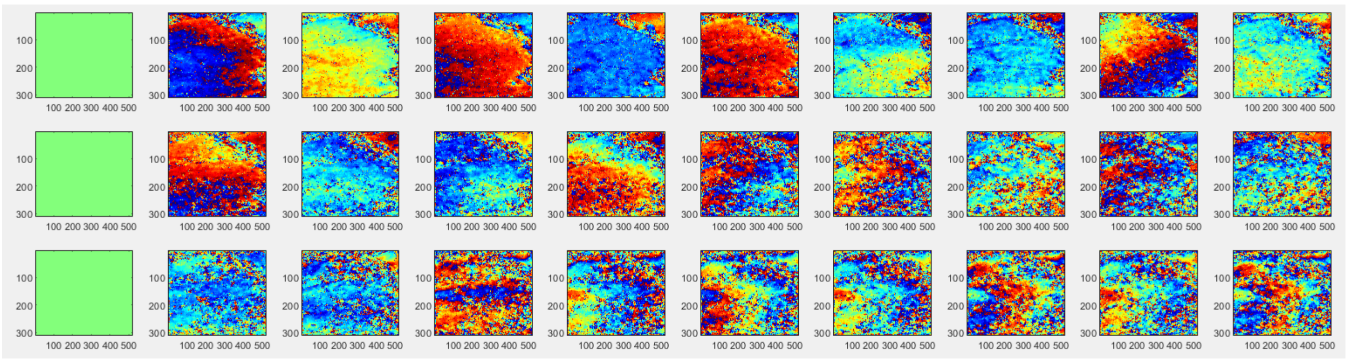


Figure 14 Optimized differential interferograms

④ **Datum connection**

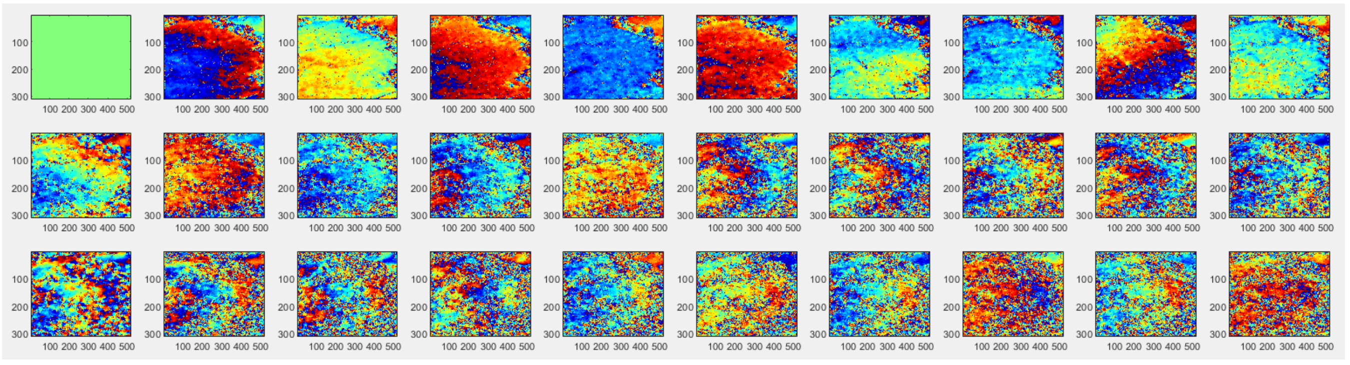


Figure 15 Total Optimized differential interferograms

3.2.3 Step4: Save final results