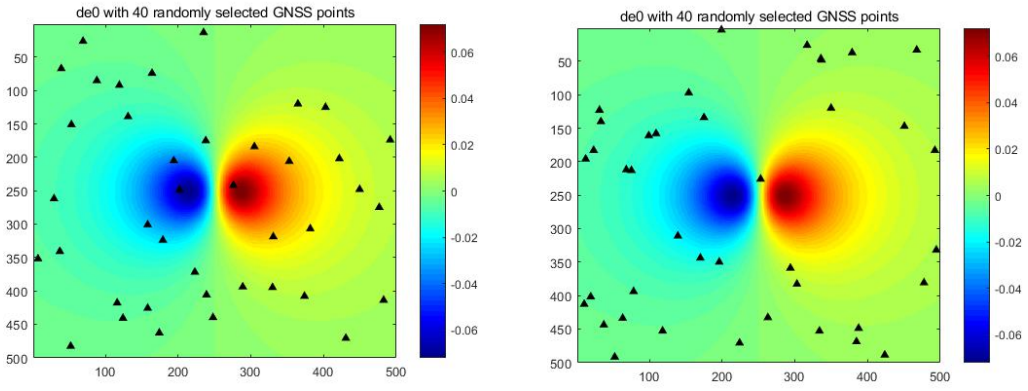
GA-SVM User Manual

This is a user manual of the genetic algorithm and support vector machine (GA-SVM) software package, which is used to estimate the three-dimensional surface deformation maps by integrating InSAR and GNSS data. The software package contains functions such as 3-D deformation simulation (*sim3d*), GNSS selection (*selgnss*), InSAR synthesis (*syninsar*), GA-SVM training and prediction of three-dimensional deformation (*gasvme/gasvmn/gasvmu*). The manual first introduces the relevant functions for estimating 3-D deformation, and then demonstrates its usage through a simulated example.

Main functions and methods are derived from the letter submitted to IEEE GRSL: *P. F. Ji, X. L. Lv, J. C. Yao and G. C. Sun, A New Method to Obtain 3-D Surface Deformations from InSAR And GNSS Data with Genetic Algorithm And Support Vector Machine, IEEE Geosci. Remote Sens. Lett., under reviewing after major revision, 2020.* Moreover, some functions in the software package come from the existing GA and SVM software libraries. The references are given on the last page.



(a) (b)

Fig. 1. (a) “Good” distribution of 40 GNSS (b) “Bad” distribution of 40 GNSS

The result predicted by the function *gasvme/gasvmn/gasvmu* may not be satisfactory. This is usually caused by three reasons: 1) insufficient GNSS number, 2) poor GNSS distribution, and 3) incorrect GA parameters. Fig. 1 (a) and (b) show the “good” distribution and the “bad” distribution when 40 GNSS points are selected using the function *selgnss*, respectively. In Fig. 1(a), some randomly selected GNSS points are located in the deformation

area (center of the figure), but not in (b). Therefore, if the GNSS points selected in Fig. 1(a) are used as the training model, the final estimated three-dimensional deformation accuracy will be significantly better than Fig. 1(b). I strongly recommend that users increase the number of GNSS or repeat the *selgnss* function until the distribution of the selected GNSS points is close to Fig. 1(a).

Please note that in the function example, the unit of all values is *m*. All functions are compiled in MATLAB2017b. Therefore, the software package may not work in an older version.

Author: Panfeng Ji.

Address: Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, China, also with the School of Electronic, Electrical, and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China.

Email: jipanfeng17@mails.ucas.ac.cn

➢ FUNCTION FILES

（1）sim3d

Description: Simulate 3-d surface deformation by Mogi's model (point source in elastic half-space).

Usage: *sim3d (err_e, err_n, err_u)*

"err_e", "err_n" and "err_u" are user-defined standard deviations of Gaussian noises added to the simulated deformation in the east-west, north-south, and up-down directions, respectively. The original 3-D deformation (de0, dn0, du0) and the 3-D deformation after adding noises (de0, dn0, du0) are saved to the file "sim3d.mat" in the current directory.

"sim3d.mat" contains:

‘de0’,‘dn0’,‘du0’: 3-D deformation without noises.

‘de’,‘dn’,‘du’: 3-D deformation with Gaussian noises.

‘err_e’, ‘err_n’, ‘err_u’: 3-D standard deviations of Gaussian noises.

Example: sim3d (0.004, 0.004, 0.008).

（2）selgnss

Description: Randomly select a user-defined number of GNSS points from the simulated 3-D deformation maps.

Usage: *selgnss (sim3d, num)*

"sim3d" is the name of the MAT file generated by the function *sim3d*. "num" is the user-defined number of GNSS points to be randomly selected.

The selected GNSS points are stored in the "gnss" variable and saved in the "gnssnum.mat" (num depends on the number of GNSS defined by the user) file in the current directory.

"gnss" is a variable with row num and column 8. The eight columns respectively indicate:

pos_y: The longitudinal position of the GNSS point y in the grid.

pos_x: The horizontal position of the GNSS point x in the grid.

disp_e: East-west displacements of GNSS point (x,y).

disp_n: North-south displacements of GNSS point (x,y).

disp_u: Up-down displacements of GNSS point (x,y).

sig_e: East-west errors of GNSS point (x,y).

sig_n: North-south errors of GNSS point (x,y).

sig_u: Up-down errors of GNSS point (x,y).

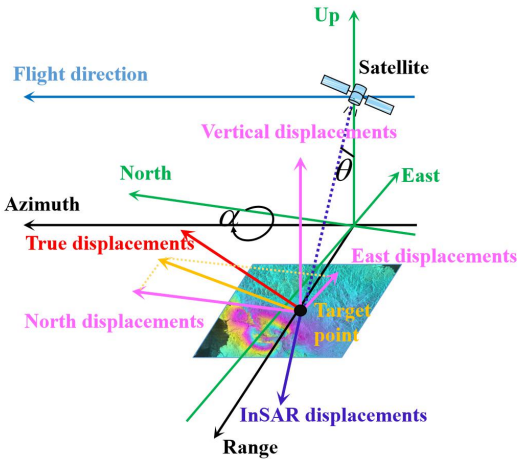
Example: selgnss ('sim3d.mat', 100).

（3）syninsar

Description: Synthesize the InSAR deformation in the LOS direction, according to the user-defined projection vector and the input 3-D deformation maps.

Usage: *syninsar(sim3d,sa,sd,err_a,err_d)*

"sim3d" is the name of the MAT file generated by the function sim3d. "sa" and "sd" are user-defined projection vectors in the ascending and descending orbit directions. "err_a" and "err_d" are user-defined standard deviations of Gaussian noises in the ascending and descending orbit directions.



The figure above shows the geometric relationship between ascending and descending InSAR tracks and 3-D surface deformations $[x_e \quad x_n \quad x_u]$. InSAR derformation d_{los} can be expressed by the three components:

$d_{los} = [s_e \quad s_n \quad s_u][x_e \quad x_n \quad x_u]^T$ where the unit projection vector $[s_e \quad s_n \quad s_u] = [-\sin\theta\sin(\alpha - 3/2\pi) \quad -\sin\theta\cos(\alpha - 3/2\pi) \quad \cos\theta]$. θ is the incidence angle of satellite sight and α is the heading angle, that is, the clockwise angle between the north and the flight of the satellite.

Syninsar produces absolute InSAR deformation, in the LOS direction, although real InSAR usually produces relative deformation with respect to the reference point.

The synthetic insar displacements of ascending and descending orbit are saved in the "insar.mat" file in the current directory.

"insar.mat" contains:

'la0', 'ld0': Synthesized original InSAR deformation, in the ascending and descending orbit directions.

'la', 'ld': InSAR deformation with Gaussian noise, in the ascending and descending orbit directions.

'sa', 'sd': Projection vectors, in the ascending and descending orbit directions.

Example: syninsar('sim3d.mat', [0.34 -0.095 0.935], [-0.34 0.095 0.935], 0.009, 0.009).

（4）makedata

Description: Make a training data set, using the weighted least square method.

Usage: *makedata(sim3d,gnss,insar)*

"sim3d","gnss" and "insar" are the names of the MAT files generated by the function *sim3d, selgnss, syninsar*, respectively.

makedata uses the weighted least square (WLS) method to calculate the response set *y* at the positions of the GNSS points. For details about the WLS method or analytical optimization method, users can refer to:

S. Samsonov, K. Tiampo, J. Rundle, and Z. Li, "Application of DInSAR-GPS Optimization for Derivation of Fine-Scale Surface Motion Maps of Southern California," IEEE Transactions on Geoscience and Remote Sensing, vol. 45, no. 2, pp. 512–521, 2007.

J. Hu, Z. W. Li, X. L. Ding, J. J. Zhu, and Q. Sun, "Derivation of 3-D coseismic surface displacement fields for the 2011 Mw 9.0 Tohoku-Oki earthquake from InSAR and GPS measurements," Geophysical Journal International, vol. 192, no. 2, pp. 573–585, 2012.

The weights of GNSS east-west (*ge*), north-south (*gn*), up-down (*gu*), InSAR ascending orbit (*la*), and descending orbit (*ld*) are defined as:

$$w_i = \frac{1}{2 \cdot err_i^2} \qquad i = ge, gn, gu, la, ld$$

Where err_i are the standard deviations of the five observations.

The optimal responses *y* are calculated using the singular value decomposition method (lssvd.p) to avoid instability of the solutions caused by rank deficiency.

The calculated results are stored in the "gnsssvm" variable and saved in the "gnsssvm.mat" file in the current directory.

"gnsssvm" is a variable with row num (GNSS number) and column 15. The first eight columns are copied from the variable "gnss". The last seven columns respectively indicate:

disp_a: Ascending track displacements of InSAR.

disp_d: Descending track displacements of InSAR.

sig_a: Ascending track errors of InSAR.

sig_d: Descending track errors of InSAR.

disp_wlse: Response variables in the east-west direction obtained by WLS.

disp_wlsn: Response variables in the north-south direction obtained by WLS.

disp_wlsu: Response variables in the up-down direction obtained by WLS.

Example: makedata('sim3d.mat', 'gnss100.mat','insar.mat').

（5）gasvme/gasvmn/gasvmu

GA-SVM trains a model in each direction of east-west, north-south, and up-down. The manual introduces *gasvme* as a typical example.

Description: Estimate the optimal East-west deformation map using the GA-SVM method.

Usage: *gasvme(sim3d,gnsssvm,insar,gaoption)*

"sim3d","gnss" and "insar" are the names of the MAT files generated by the function *sim3d, selgnss, syninsar*, respectively. "gaoption" indicates an initialization option for searching the optimal SVM parameters *c* and *g* using genetic algorithm (GA).

"gaoption" is a 7-dimensional vector, each of which indicates:

gaoption(1): Maximum number of evolutionary generations

gaoption(2): Maximum number of population

gaoption(3): Bound of penalty factor *c* (minimum)

gaoption(4): Bound of penalty factor *c* (maximum)

gaoption(5): Bound of kernel parameter *g* (minimum)

gaoption(6): Bound of kernel parameter *g* (maximum)

gaoption(7): SVM cross validation parameter

"*gasvme*" performs the following main steps:

1) Standardized data (Min-Max method).
2) Search the best SVM parameters *c&g* by GA.
3) Train the model using the parameters selected by GA.
4) Predict on the training set.
5) Predict displacements in the east-west direction at each point in the grid.
6) De-standardized data.
7) Display the predicted displacements in the east-west direction.

Example:

gaoption=[300 50 0 200 0 100 5];

gasvme('sim3d.mat','gnsssvm.mat','insar.mat',gaoption) ;

（6）*svmtain* and *svmpredict* are two functions from LIBSVM. They are C code files that

have been compiled into executable files by MATLAB's *mex* command.

（7）Other functions in the directory such as *cholesky* are auxiliary functions.

➢ EXAMPLE

```
clear;
close all;
sim3d(0.004, 0.004, 0.008);
selgnss('sim3d.mat', 100);
syninsar('sim3d.mat', [0.34 -0.095 0.935], [-0.34 0.095 0.935], 0.009, 0.009);
makedata('sim3d.mat', 'gnss100.mat','insar.mat');
gaoption=[300 50 0 200 0 100 5];
gasvme('sim3d.mat','gnsssvm.mat','insar.mat',gaoption);
gasvmn('sim3d.mat','gnsssvm.mat','insar.mat',gaoption);
gasvmu('sim3d.mat','gnsssvm.mat','insar.mat',gaoption);
```

➢ References

Faruto and Liyang , LIBSVM-faruto Ultimate Version a toolbox with implements for support vector machines based on libsvm, 2011.

Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines, 2001.