A Toolbox for Unsupervised Change Detection Analysis

Nicola Falco[a], Prashanth R. Marpu[b], Jon A. Benediktsson[c]

[a]Faculty of Electrical and Computer Engineering, University of Iceland, Reykjavik, Iceland.

[b]Dept. of Information Engineering and Computer Science, University of Trento, Trento, Italy.

[c]Earth Observation and Hydro-Climatology Laboratory, Masdar Institute, Abu Dhabi, UAE.

Description of the software

Contents

1 Introduction

This tutorial describes the software which performs a change detection analysis on multispectral images. The function allows one to perform the cd analysis by choosing between two methods:

ITPCA is based on the application of the Iterated PCA (ITPCA) analysis. By analyzing the 1st principal component, it is possible to find a linear relation between the two input images and perform a recalibration between them. A windows based misregistration recovery is apply to the re-calibrated images in order to correct small misregistration errors. The results of ITPCA could be improved by using a initial mask of the strong change, which pixel are excluded from analysis. On the re-calibrated and recovered images, the chi square analysis is applied by using EH algorithm, which performs a multidimensional gaussian mixture estimation of both changed and unchanged classes.

IRMAD method that performs a *Iteratively Reweighted Multivariate Alteration Detection*. The obtained canonical variates are filtered in order to correct the small misregistration errors. Then, the chi square analysis is applied in order to estimate the multidimensional gaussian mixture of both changed and unchanged class.

All the functions are described in the next sections.

2 ChangeDetection function (main function)

It is the main function. Here is possible to decide which method to use between ITPCA and IRMAD. For only the ITPCA method is possible to choose:

- type of processing: in case of a huge data set, is possible to apply a line by line processing, in order to save memory;
- initial change mask: in order to eliminate the strong change form the analysis, it is possible to create a initial change mask.

Syntax:

CD_IRMAD_ITPCA()
CD_IRMAD_ITPCA(image_t1, image_t2, method)
CD_IRMAD_ITPCA(image_t1, image_t2, method, opts, save_name, path_name)

opts for ITPCA:

$$Tc, flag\_plot, size\_im, flag\_mask, low\_val, dimWin, flag\_save$$

opts fro IRMAD:

$$PCs, epsln, size\_im, flag\_mask, low\_val, dimWin, flag\_save$$

Inputs:

- *image_t1:* string of the whole path of the ENVI bip format image at data t1;
- *image_t2:* string of the whole path of the ENVI bip format image at data t2;
- *method:* string: choose between ITPCA and IRMAD;
- *save_name:* string of the name used as suffix when the files are saved;
- *path_name:* string of the path where to save the files;

opts for ITPCA (vector of integer)

- *Tc:* convergence threshold for the ITPCA method (0.002 default value).

- *flag_plot:* 0: no plot, 1: plot of the PCs.
- *size_im:* string: it permits to select a different processing in based on the dimension of the data set:
  - small images: ⇒ 0: the whole data set is read.
  - big images: ⇒ 1: the data set is read line by line (less memory used).

  0: default value.
- *flag_mask:* 1: to choose a strict mask thresholding based on EM algorithm, 2: to choose a relaxed mask thresholding based on EM algorithm, 3: to use the principal component instead of the original data set, 0: no mask. Default value = 0.
- *low_val:* value in to mask the low values of the histogram 0 no mask default value = 0.
- *dimWin:* window dimension for the filter used in the WRMisreg function (e.g.: 3 ⇒ for a 3x3 window, 5 ⇒ for a 5x5 window, and so on (3 default value).
- *flag_save:* 0 to save the chi square files, 1 to save the chi square and the intermediary files.

opts for IRMAD (vector of integer)

- *PCs:* 0 to use the original dataset, 1 to use the principal components, default value = 0.
- *epsln:* epsilon for iterations. default value 0.05. Insert 100 to apply 0 iterations.
- *size_im:* string: it permits to select a different processing in based on the dimension of the data set:
  - small images: ⇒ 0: the whole data set is read.
  - big images: ⇒ 1: the data set is read line by line (less memory used).

  0: default value.

- *flag_mask:* 1: to choose a strict mask thresholding based on EM algorithm, 2: to choose a relaxed mask thresholding based on EM algorithm, 3: to use the principal component instead of the original data set, 0: no mask. Default value = 0.
- *low_val:* value in to mask the low values of the histogram 0 no mask default value = 0.
- *dimWin:* window dimension for the filter used in the WRMisreg function (e.g.: 3 ⇒ for a 3x3 window, 5 ⇒ for a 5x5 window, and so on (3 default value).
- *flag_save:* 0 to save the chi square files, 1 to save the chi square and the intermediary files.

Outputs:

- *ITPCA method (stored in ENVI bip format):*
  - *ITPCA_img1:* re-calibrated image t1.
  - *ITPCA_img2:* re-calibrated image t2.
  - *PC1:* 1st principal component.
  - *PC2:* 2st principal component.
  - *rimg_Diff:* difference between the re-calibrated images.
  - *wrm_img:* misregistration recovery of the re-calibrated images.
  - *chi2_itpca_pc2:* 1st method: the Chi Square is obtained using the 2nd principal component.
  - *chi2_itpca_diff:* 2nd method: the Chi Square is obtained using the diff_rim components.
  - *chi2_itpca_wrm:* 3rd method: the Chi Square is obtained using the wrm_itp components.
  - *mask:* initial change mask.
- *IRMAD method (stored in ENVI bip format):*
  - *cv1:* canonical variates1 unit variance.
  - *cv2:* canonical variates2 unit variance.
  - *mads:* the MAD variates.
  - *wrm_img:* recovery misregistration of cvs.
  - *chi2_irmad:* Chi Square obtained by using cvs.
  - *chi2_irmad_wrm:* Chi Square obtained by using the recovered cvs.

3 ITPCA / ITPCALine function

The Iterated PCA (ITPCA / ITPCALine) function performs a spectral re-calibration of the images based on the linear relation between them. The linear relation is related to the first principal component.

Syntax:

ITPCA()
ITPCA(image_t1, image_t2)
ITPCA(image_t1, image_t2, opts, save_name, path_name)
opts for ITPCA:

$$Tc, flag\_mask, low\_val, flag\_plot,$$

Inputs:

- *image_t1:* string of the whole path of the ENVI bip format image at data t1.
- *image_t2:* string of the whole path of the ENVI bip format image at data t2.
- *save_name:* string of the name used as suffix when the files are saved.
- *path_name:* string of the path where to save the files.

opts for ITPCA (vector of integer)

- *Tc:* convergence threshold for the ITPCA method (0.002 default value).
- *flag_mask:* 1: to choose a strict mask thresholding based on EM algorithm, 2: to choose a relaxed mask thresholding based on EM algorithm, 3: to use the principal component instead of the original data set, 0: no mask. Default value = 0.
- *low_val:* value in to mask the low values of the histogram 0 no mask default value = 0.
- *flag_plot:* 0: no plot, 1: plot of the PCs.

Outputs:

- *ITPCA_img1:* re-calibrated image t1 / string of the whole path of the re-calibrated image t1 (line by line version).

- *ITPCA_img2:* re-calibrated image t2 / string of the whole path of the re-calibrated image t2 (line by line version).
- *PC1:* 1st principal component / string of the whole path of 1st principal component.
- *PC2:* 2st principal component / string of the whole path of 2nd principal component.
- *(stored in ENVI bip format):*
  - *ITPCA_img1:* re-calibrated image t1.
  - *ITPCA_img2:* re-calibrated image t2.
  - *PC1:* 1st principal component.
  - *PC2:* 2st principal component.
  - *mask:* initial change mask.
- *(MATLAB structs):*
  - *ITPCA.mat:* previous files in MATLAB format
  - *mask.mat:*

4 IRMAD / IRMADLine function

Iteratively Reweighted Multivariate Alteration Detection.

Syntax:

IRMAD()
IRMAD(image_t1, image_t2)
IRMAD(image_t1, image_t2, opts, save_name, path_name)
opts for IRMAD:

$$PCs, epsln, flag\_mask, low\_val, w, flag\_save$$

Inputs:

- *image_t1:* string of the whole path of the ENVI bip format image at data t1;
- *image_t2:* string of the whole path of the ENVI bip format image at data t2;
- *save_name:* string of the name used as suffix when the files are saved;
- *path_name:* string of the path where to save the files;

opts for IRMAD (vector of integer)

- *PCs:* 0 to use the original dataset, 1 to use the principal components, default value = 0.
- *epsln:* epsilon for iterations. default value 0.05. Insert 100 to apply 0 iterations.
- *flag_mask:* 1: to choose a strict mask thresholding based on EM algorithm, 2: to choose a relaxed mask thresholding based on EM algorithm, 3: to use the principal component instead of the original data set, 0: no mask. Default value = 0.
- *low_val:* value in to mask the low values of the histogram 0 no mask default value = 0.
- *w:* weights for stats calculation (available only for command line), 0 no weights. Default value = 0.

- *flag_save:* 0 to save the chi square files, 1 to save the chi square and the intermediary files.

Outputs:

- *cv1* canonical variates1 unit variance. String of the whole path of the canonical variates cv1 (line by line version).
- *cv2:* canonical variates2 unit variance. String of the whole path of the canonical variates cv2 (line by line version).
- *mads:* the MAD variates. String of the whole path of the mads (line by line version).
- *chi2:* Chi Square. String of the whole path of the chi2 (line by line version).

5 ICM / ICMLine function

The Initial Change Mask (ICM / ICMLine) function identifies strong changes and creates a mask of them. The mask is produced by using the threshold estimated with EM algorithm.

Syntax:

ICM()
ICM(image_t1, image_t2)
ICM(image_t1, image_t2,flag_mask,low_val, save_name, path_name)

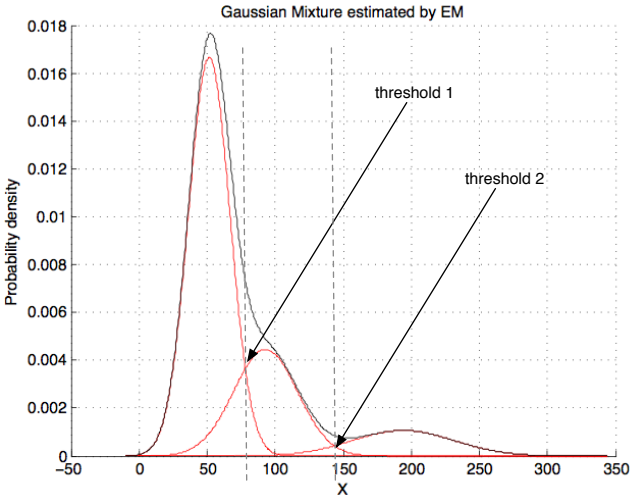
Inputs:

- *image_t1:* string of the whole path of the ENVI bip format image at data t1.
- *image_t2:* string of the whole path of the ENVI bip format image at data t2.
- *flag_mask:* 1: to choose a strict mask thresholding based on EM algorithm, 2: to choose a relaxed mask thresholding based on EM algorithm, 3: to use the principal component instead of the original data set. Default value = 1.
- *low_val:* value in to mask the low values of the histogram 0 no mask default value = 0.
- *save_name:* string of the name used as suffix when the files are saved.
- *path_name:* string of the path where to save the files.

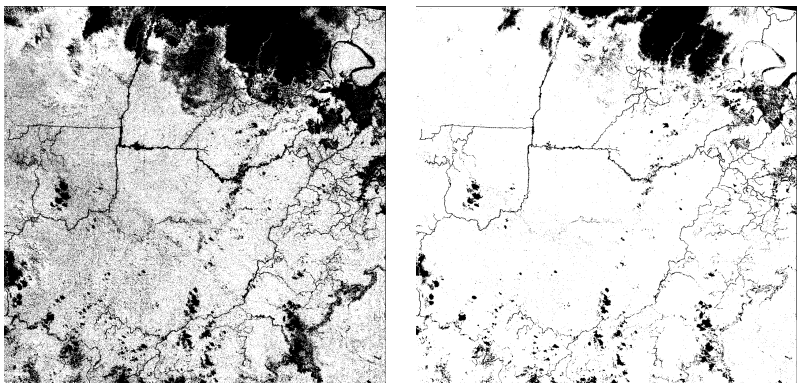
Outputs:

- *mask:* initial change mask / string of the whole path of the initial change mask (line by line version).

Example of results:



(a)



(b) (c)

Figure 1: *Initial Change Masks obtained by using the 1st threshold(b), and the 2nd one(c). The pixels in black are excluded from further analysis as they represent strong changes.*

6 WRM / WRMLine function

The Window Recovery Misregistration (WRM / WRMLine) function performs a window-based correction for small misregistrations errors between two multitemporal and multispectral images.

Syntax:

WRM()
WRM(image_t1, image_t2)
WRM(image_t1, image_t2, dimWin, save_name, path_name)

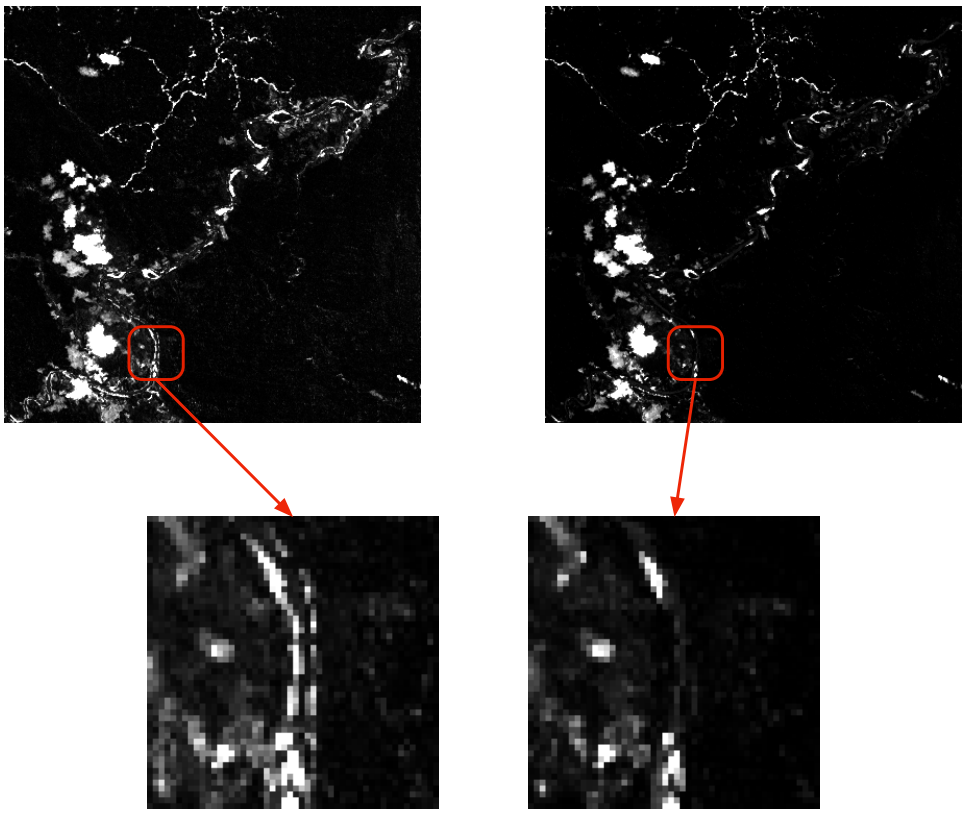
Inputs:

- *image_t1:* string of the whole path of the ENVI bip format image at data t1;
- *image_t2:* string of the whole path of the ENVI bip format image at data t2;
- *dimWin:* window dimension for the filter used in the WRMisreg function (e.g.: 3 ⇒ for a 3x3 window, 5 ⇒ for a 5x5 window, and so on (3 default value);
- *save_name:* string of the name used as suffix when the files are saved;
- *path_name:* string of the path where to save the files.

Outputs:

- *wrm_img:* recovered image / string of the whole path of the recovered image (line by line version);
- *(stored in ENVI bip format):*
  - *wrm_img:* recovered image;
- *(MATLAB structs):*
  - *wrm_img.mat:* recovered image

Example of results:



(a) (b)

Figure 2: *(a) (b) Without and with Window Recovery Misregistration.*