C

ENVI Extensions in IDL

This appendix gives installation instructions and a brief documentation of all of the ENVI/IDL extensions described in the text.

C.1 Installation

To install the complete extension package:

1. Place the program files (extension `.PRO`) anywhere in the IDL `!PATH`.

2. Place the file `MADVIEWHELP.PDF` in anywhere in the IDL `!HELP_PATH`.

3. Place the dynamic link library (DLL) file `PROV_MEANS.DLL` in the operating system path (e.g., the `WINDOWS` directory for Windows systems). The compiled DLL is provided for Windows only. Unix flavors and Mac OS will need to create it from the C source code `PROV_MEANS.C` provided. See the IDL documentation for `MAKE_DLL`.

4. Under `Preferences` in the ENVI main menu select the tab `User Defined Files`. Under `Envi Menu File` and `Display Menu File` enter the paths to the two menu files `ENVI.MEN` and `DISPLAY.MEN` provided in the package.

5. (This step is only necessary for the calculation of structure heights with `CALCHEIGHT_RUN` and can be skipped if desired.) Place the file `CH_CURSOR_MOTION.PRO` in your ENVI `SAVE_ADD` directory. In `File/ Preferences/ User Defined Motion Routine` in the ENVI main menu enter: `CH_CURSOR_MOTION`.

Many of the routines make use of David Fanning's object class `PROGRESSBAR__DEFINE` and associated dependencies, so his `COYOTE` library must also be in the IDL `!PATH`. The library may be downloaded from

`http://www.dfanning.com/documents/programs.html`

Also Matt Craig's `TeXtoIDL` package is required, see

`http://physweb.mnstate.edu/mcraig/TeXtoIDL/`

C.2 Panchromatic sharpening

C.2.1 Discrete wavelet transform fusion

`DWT_RUN` is an ENVI/IDL extension for panchromatic sharpening with the discrete wavelet transform using Daubechies wavelets as discussed in Section 5.3.4.

C.2.1.1 Usage

The routine is invoked from the ENVI main menu as

`Transform/Image Sharpening/Wavelet(ARSIS Model)/DWT`

The low and high resolution images must be georeferenced (usually acquired simultaneously from the same platform). In the

`Select low resolution multi-band input file`

window choose the (spatial/spectral subset of the) image to be sharpened. In the

`Select high resolution (pan) input band band`

window choose the corresponding panchromatic or high resolution image. This image must overlap the low resolution image completely. Then in the

`ARSIS Fusion Output`

box select an output file name, or choose memory. During the calculation, scatter plots are shown of the wavelet coefficients for the low and high resolution bands.

C.2.1.2 Source headers

Listing C.1: `DWT_RUN.PRO`

```
1  ;+
2  ; NAME:
3  ;        DWT_RUN
4  ; PURPOSE:
5  ;        ENVI extension for panchromatic sharpening
6  ;        under ARSIS model with Mallat's discrete wavelet
7  ;        transform and Daubechies wavelets
8  ;        Ref: Ranchin and Wald, Photogramm. Eng. Remote.
9  ;        Sens. 66(1), 2000, 49-61
10 ; AUTHOR
11 ;        Mort Canty (2006)
12 ;        Juelich Research Center
13 ;        m.canty@fz-juelich.de
14 ; CALLING SEQUENCE:
15 ;        DWT_RUN
16 ; ARGUMENTS:
17 ;        Event (if used as a plug-in menu item)
```

```
18 ; KEYWORDS:
19 ;       None
20 ; DEPENDENCIES:
21 ;       ENVI
22 ;       DWT__DEFINE
23 ;       ORTHO_REGRESS
24 ;-------------------------------------------------------
```

Listing C.2: DWT__DEFINE.PRO

```
1  ;+
2  ; NAME:
3  ;       DWT__DEFINE
4  ; PURPOSE:
5  ;       Discrete wavelet transform object class using
6  ;       Daubechies wavelets for construction of pyramid
7  ;       representations of images.
8  ;       Ref: T. Ranchin, L. Wald, Photogammetric Engineering
9  ;            and Remote Sensing 66(1) (2000) 49-61.
10 ; AUTHOR
11 ;       Mort Canty (2006)
12 ;       Juelich Research Center
13 ;       m.canty@fz-juelich.de
14 ; CALLING SEQUENCE:
15 ;       dwt = Obj_New("DWT",image)
16 ; ARGUMENTS:
17 ;       image: grayscale image to be transformed
18 ; KEYWORDS
19 ;       None
20 ; METHODS:
21 ;       SET_COEFF: choose the Daubechies wavelet
22 ;         dwt -> Set_Coeff, n
23 ;         n = 4,6,8
24 ;       SHOW_IMAGE: display the image pyramid in a window
25 ;         dwt -> Show_Image, wn
26 ;       INJECT: overwrite innermost quadrants
27 ;         dwt -> Inject, array, quadrant=quadrant (default 0)
28 ;       SET_COMPRESSIONS: set the number of compressions
29 ;         dwt -> Set_Compressions, nc
30 ;       GET_COMPRESSIONS: get the number of compressions
31 ;         nc = dwt -> Get_Compressions()
32 ;       GET_NUM_COLS: get number of columns in compressed image
33 ;         cols = dwt -> Get_Num_Cols()
34 ;       GET_NUM_ROWS: get number of rows in compressed image
35 ;         cols = dwt -> Get_Num_Rows()
36 ;       GET_IMAGE: return the pyramid image
37 ;         im = dwt -> Get_Image()
38 ;       GET_QUADRANT: get compressed image (as 2D array) or
39 ;                     innermost wavelet coefficients as vector
```

```
40 ;           wc = dwt -> Get_Quadrant(n)
41 ;           n  = 0,1,2,3
42 ;        NORMALIZE_WC: normalize all wavelet coefficients
43 ;          dwt -> Normalize, a, b
44 ;            a, b are normalization parameters
45 ;        COMPRESS: perform a single compression
46 ;          dwt -> Compress
47 ;        EXPAND: perfrom a single expansion
48 ;          dwt -> Expand
49 ; DEPENDENCIES:
50 ;        None
51 ; -------------------------------------------------------------
```

Listing C.3: `ORTHO_REGRESS.PRO`

```
1  ;+
2  ; NAME:
3  ;        ORTHO_REGRESS
4  ; PURPOSE:
5  ;        Orthogonal regression between two vectors
6  ;        Ref: M. Canty et al. Remote Sensing of Environment
7  ;             91(3,4) (2004) 441-451
8  ; AUTHOR
9  ;        Mort Canty (2006)
10 ;        Juelich Research Center
11 ;        m.canty@fz-juelich.de
12 ; CALLING SEQUENCE:
13 ;        Ortho_Regress, X, Y, b, Xm, Ym, sigma_a, sigma_b
14 ;        regression line is Y = Ym + b(X-Xm)
15 ;                             = (Ym-bXm) + bX = a + bX
16 ; ARGUMENTS:
17 ;        input column vectors X and Y
18 ;        returns b, Xm, Ym, sigma_a, sigma_b, sigma(RMSE)
19 ; KEYWORDS:
20 ;        None
21 ; DEPENDENCIES:
22 ;        None
23 ;-------------------------------------------------------------
```

C.2.2 *À trous* wavelet transform fusion

`ATWT_RUN` is an ENVI/IDL extension for panchromatic sharpening with the ATWT with a cubic B-spline filter, Section 5.3.5.

C.2.2.1 Usage

The routine is invoked from the ENVI main menu as

`Transform/Image Sharpening/Wavelet(ARSIS Model)/ATWT`

The low and high resolution images must be georeferenced (usually acquired simultaneously from the same platform). In the

`Select low resolution multi-band input file`

window choose the (spatial/spectral subset of the) image to be sharpened. In the

`Select high resolution (pan) input band`

window choose the corresponding panchromatic or high resolution image. This image must overlap the low resolution image completely. Then in the

`ARSIS Fusion Output`

box select an output file name, or choose memory. During the calculation scatter plots of the wavelet coefficients are shown for the low and high resolution bands.

C.2.2.2 Source headers

Listing C.4: `ATWT_RUN.PRO`

```
;+
; NAME:
;       ATWT_RUN
; PURPOSE:
;       ENVI extension for panchromatic sharpening under
;       ARSIS model with "A trous" wavelet transform.
;       Ref: Aiazzi et al, IEEE Transactions on Geoscience
;       and Remote Sensing, 40(10) 2300-2312, 2002
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       ATWT_RUN
; ARGUMENTS:
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       ATWT__DEFINE
;       ORTHO_REGRESS
;-----------------------------------------------------
```

Listing C.5: `ATWT__DEFINE.PRO`

```
;+
; NAME:
;       ATWT__DEFINE
; PURPOSE:
;       A Trous wavelet transform class using cubic B-
```

```
;       spline wavelets. For shift invariant image fusion
;       Ref: Aiazzi et al. IEEE Transactions on Geoscience
;            and Remote Sensing 40(10) (2002) 2300-2312
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       atwt = Obj_New("ATWT",image)
; ARGUMENTS:
;       image: grayscale image to be processed
; KEYWORDS
;       None
; METHODS:
;       SHOW_IMAGE: display the image pyramid in a window
;         atwt -> Show_Image, wn
;       INJECT: overwrite the filtered image
;         atwt -> Inject, im
;       SET_TRANSFROMS: set the number of transformations
;         atwt -> Set_Transforms, nc
;       GET_TRANSFROMS: get the number of transformations
;         nc = atwt -> Get_Transforms()
;       GET_NUM_COLS: get the number of columns in the
;       compressed image
;         cols = atwt -> Get_Num_Cols()
;       GET_NUM_ROWS: get the number of rows in the
;       compressed image
;         cols = atwt -> Get_Num_Rows()
;       GET_IMAGE: return filtered image or details
;         im = atwt -> Get_Image(i)
;           i = 0 for filters image, i > 0 for details
;       NORMALIZE_WC: normalize details at all levels
;         atwt -> Normalize, a, b
;           a, b are normalization parameters
;       COMPRESS: perform a single transformation
;         atwt -> Compress
;       EXPAND: perfrom a single reverse transformation
;         atwt -> Expand
; DEPENDENCIES:
;       None
; -----------------------------------------------------
```

C.2.3 Quality index

`QUALITY_INDEX_RUN` is an ENVI extension to determine the Wang-Bovik quality index of a pan-sharpened image, see Section 5.3.6.

C.2.3.1 Usage

The routine is invoked from the ENVI main menu as

`Transform/Image Sharpening/Quality Index`

From the

`Choose reference image`

menu select the multispectral image to which the sharpened image is to be compared. In the

`Choose pan-sharpened image`

menu, select the image whose quality is to be determined. The reference image must completely overlap the pan-sharpened image. The image bands are matched to the nearest pixel using phase correlation, plots of which are shown during the calculation.

C.2.3.2 Source headers

Listing C.6: `QUALITY_INDEX_RUN.PRO`

```
;+
; NAME:
;       QUALITY_INDEX_RUN
; PURPOSE:
;       ENVI extension for radiometric comparison of two
;       multispectral images
;       Ref: Wang and Bovik, IEEE Signal Processing
;       Letters 9(3) 2002, 81-84
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Quality_Index_Run
; ARGUMENTS:
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       PHASE_CORR
;       QUALITY_INDEX
;-------------------------------------------------------
```

Listing C.7: `PHASE_CORR.PRO`

```
;+
; NAME:
;       PHASE_CORR
; PURPOSE:
;       Returns relative offset [xoff,yoff] of two images
```

```
6 ;        using phase correlation
7 ;        Maximum offset should not exceed +- 5 pixels
8 ;        in each dimension
9 ;        Returns -1 if dimensions are not equal
10 ;       Ref: H, Shekarforoush et al. (1995) INRIA 2707
11 ; AUTHOR
12 ;       Mort Canty (2006)
13 ;       Juelich Research Center
14 ;       m.canty@fz-juelich.de
15 ; CALLING SEQUENCE:
16 ;       shft = Phase_Corr(im1,im2,display=display, $
17 ;       subpixel=subpixel)
18 ; ARGUMENTS:
19 ;       im1, im2: the images to be correlated
20 ; KEYWORDS:
21 ;       Display: (optional) show a surface plot if the
22 ;                correlation in window with display
23 ;                number display
24 ;       Subpixel: returns result to subpixel accuracy if
25 ;                 set, otherwise nearest integer (default)
26 ; DEPENDENCIES:
27 ;       None
28 ;-------------------------------------------------------
```

Listing C.8: QUALITY_INDEX.PRO

```
1 ;+
2 ; NAME:
3 ;       QI
4 ; PURPOSE:
5 ;       Determine the Wang-Bovik quality index for a
6 ;       pan-sharpened image band
7 ;       Ref: Wang and Bovik, IEEE Signal Processing
8 ;       Letters9(3) 2002, 81-84
9 ; AUTHOR
10 ;       Mort Canty (2006)
11 ;       Juelich Research Center
12 ;       m.canty@fz-juelich.de
13 ; CALLING SEQUENCE:
14 ;       index = QI(band1,band2)
15 ; ARGUMENTS:
16 ;       band1: reference band
17 ;       band2: degraded pan-sharpened band
18 ; KEYWORDS:
19 ;       None
20 ; DEPENDENCIES:
21 ;       None
22 ;-------------------------------------------------------
```

C.3 Topographic modeling and registration

C.3.1 Calculating heights of man-made structures in high resolution imagery

`CALC_HEIGHT_RUN` is an ENVI extension to determine heights of vertical structures in QuickBird and IKONOS images using rational function models (RFMs) accompanying ortho-ready imagery distributed by the image providers, see Section 5.4.3. It is implemented as a small graphical user interface (GUI) using IDL widgets.

C.3.1.1 Usage

After displaying a high resolution (IKONOS or QuickBird) image in a display window, invoke the tool as

`Tools/Structure Height`

from the ENVI display menu. Load an RFM file in the

`CalcHeight`

window with

`File/Load RPC File`

(extension RPC or RPB). If a digital eleveation model (DEM) is available for the scene, this can also be loaded with

`File/Load DEM File`

A DEM is not required, however. Click on the bottom of a vertical structure to set the base height and then shift-click on the top of the structure. Press the `CALC` button to display the structure's height, latitude, longitude and base elevation. The number in brackets next to the height is the minimum distance (in pixels) between the top pixel and a vertical line through the bottom pixel. It should be of the order of 1 or less. If no DEM is loaded, the base elevation is the average value for the whole scene. If a DEM is used, the base elevation is taken from it. The latitude and longitude are then orthorectified values.

C.3.1.2 Source headers

Listing C.9: `CALCGEIGHT_RUN.PRO`

```
1 ;+
2 ; NAME:
3 ;        CALCHEIGHT_RUN
4 ; PURPOSE:
5 ;        ENVI extention to determine height of vertical
6 ;        structures in QuickBird/Ikonos images using RFM
7 ; AUTHOR;
8 ;        Mort Canty (2006)
9 ;        Juelich Research Center
```

```
10 ;       m.canty@fz-juelich.de
11 ; CALLING SEQUENCE:
12 ;       CalcHeight_Run
13 ; ARGUMENTS:
14 ;       Event (if used as a plug-in menu item)
15 ; KEYWORDS:
16 ;       None
17 ; COMMON BLOCKS:
18 ;       Shared, RFM, Cb, Rb, Ct, Rt, elev
19 ;       Cursor_Motion_C, dn, Cbtext, Rbtext, Cttext, Rttext
20 ;          RFM: IDL structure containing RFM camera model
21 ;          Cb, Rb: coordinates of building base
22 ;          Ct, Rt: coordinates of building top
23 ;          elev: elevation of base
24 ;          dn: display number
25 ;          Cbtext etc : Edit widgets
26 ; DEPENDENCIES:
27 ;       ENVI
28 ;       CH_CURSOR_MOTION
29 ; ------------------------------------------------------
```

Listing C.10: CH_CURSOR_MOTION.PRO

```
1  ;+
2  ; NAME:
3  ;       CH_CURSOR_MOTION
4  ; PURPOSE:
5  ;       Cursor communication with ENVI image windows
6  ; AUTHOR;
7  ;       Mort Canty (2006)
8  ;       Juelich Research Center
9  ;       m.canty@fz-juelich.de
10 ; CALLING SEQUENCE:
11 ;       CH_Cursor_Motion, dn, xloc, yloc, $
12 ;       xstart=xstart, ystart=ystart, event=event
13 ; ARGUMENTS:
14 ;       dn: display number
15 ;       xloc,yloc: mouse position
16 ; KEYWORDS
17 ;       xstart, ystart: display origin
18 ;       event: mouse event
19 ; COMMON BLOCKS:
20 ;       Cursor_Motion_C,dn,Cbtext,Rbtext,Cttext,Rttext
21 ; DEPENDENCIES:
22 ;       ENVI
23 ;-------------------------------------------------------
```

C.3.2 Illumination correction

`C_CORRECTION_RUN` is an ENVI/IDL extension for performing local solar incidence angle corrections for multispectral images over rough terrain, Section 5.4.6.

C.3.2.1 Usage

The routine is invoked from the ENVI main menu as

`Topographic/Illumination Correction`

From the

`Choose image to correct`

menu select the (spectral/spatial subset of the) image to be corrected and (optionally) a mask. Then in the

`C-correction parameters`

box enter the solar elevation and azimuth in degrees and a size for the kernel used for slope/aspect determination (default 9×9). In the

`Choose digital elevation file`

window select the corresponding DEM file. Finally in the

`Output corrected image`

box choose an output file name or select memory.

C.3.2.2 Source headers

Listing C.11: `C_CORRECTION_RUN.PRO`

```
1  ;+
2  ; NAME:
3  ;       C_CORRECTION_RUN
4  ; PURPOSE:
5  ;       ENVI extension for c-correction for solar
6  ;       illuminatin in rough terrain
7  ;       Ref: P. M. Teillet et al., Canadian Journal of
8  ;       Remote Sensing 8(2) 84-106
9  ; AUTHOR
10 ;       Mort Canty (2006)
11 ;       Juelich Research Center
12 ;       m.canty@fz-juelich.de
13 ; CALLING SEQUENCE:
14 ;       C_Correction_Run
15 ; ARGUMENTS:
16 ;       Event (if used as a plug-in menu item)
17 ; KEYWORDS:
18 ;       None
19 ; DEPENDENCIES:
20 ;       ENVI
21 ;-------------------------------------------------------
```

C.3.3 Image registration

`CONTOUR_MATCH_RUN` is an ENVI/IDL extension for extracting tie-points for image-image registration, Section 5.5.2. It is primarily intended for fine adjustment, assuming that the images are already georeferenced and, if necessary, resampled to the same ground sample distance (GSD).

C.3.3.1 Usage

The routine is invoked from the ENVI main menu as

`Map/Registration/contour Matching`

From the dialog windows choose the base and warp image bands. Then enter the σ parameter for the LoG filter (default value is 2.5 pixels). Finally give a filename (extension .PTS) for the tie-points (ground control points). If tie-points are found, they can be read in from this file into ENVI's

`Ground Control Points Selection`

dialog. It may be necessary to edit these further to eliminate outliers.

C.3.3.2 Source headers

Listing C.12: `CONTOUR_MATCH_RUN.PRO`

```
;+
; NAME:
;       CONTOUR_MATCH_RUN
; PURPOSE:
;       ENVI extension for extraction of tie points for
;       image-image registration.Images may be already
;       georeferenced, in which case GCPs are for "fine
;       adjustement". Uses Laplacian of Gaussian and Sobel
;       filter and contour tracing to match contours
;       Ref: Li et al, IEEE Transactions on
;       Image Processing, 4(3) (1995) 320-334
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Contour_Match_Run
; ARGUMENTS:
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       CI_DEFINE
;       PROGRESSBAR_DEFINE (FSC_COLOR)
;-------------------------------------------------
```

Listing C.13: `CI__DEFINE.PRO`

```
;+
; NAME:
;       CI__DEFINE
; PURPOSE:
;       Object class to find thin closed contours in an
;       image band with combined Sobel-LoG filtering
;       Ref: Li et al, IEEE Transactions on Image
;       Processing, 4(3) (1995) 320-334
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       ci = Obj_New("CI",image,sigma)
; ARGUMENTS:
;       image: grayscale image to be searched for contours
;       sigma: radius for LoG filter
; KEYWORDS
;       None
; METHODS:
;       GET_MAX_CONTOURS     return maximum number of contours
;       GET_MAX_LENGTH       return maximum contour length
;       GET_CONTOUR_IMAGE    return contour (filtered) image
;       CLEAR_CONTOUR_IMAGE  erase contour image
;       TO_PIXELS            read contour structure and return
;                            its pixel array
;       TO_FILTERED_CC       read contour structure and return
;                            filtered chain code
;       TO_MOMENTS           read contour structure and return
;                            Hu invariant moments
;       WRITE_CONTOUR        read contour stucture and display
;                            on contour_image
;       TRACE_CONTOUR        search contour image for next
;                            closed contour
; DEPENDENCIES:
;       None
; Contour structure
;   c = {
;    sp:       intarr(2), $ ; starting point coordinates
;    length:   0L,        $ ; number of pixels
;    closed:   -1L,       $ ; set to zero while tracing, 1 if
;                             closed, -1 if no more contours
;    code:     bytarr(self.max_length),   $   ; chain code
;    icode:    bytarr(self.max_length)    }   ; rotationally
;                                               invariant
;                                               chain code
; ------------------------------------------------------
```

C.4 Supervised classification

C.4.1 Maximum likelihood classifier

`MAXLIKE_RUN` is a modification of ENVI's maximum likelihood classifier, discussed in Section 6.3.2, which reserves test data from the training regions of interest (ROIs) in the ratio 2:1 for training:test.

C.4.1.1 Usage

The routine can be called from the ENVI command prompt with

```
MaxLike_Run
```

The user is prompted for an input file, the training ROIs (which must be present beforehand) and output destinations for the classified image, the probability image (optional) and the test results. The test result file can be processed with the procedures `CT_RUN` and `McNEMAR_RUN` discussed below.

C.4.1.2 Source headers

Listing C.14: `MAXLIKE_RUN.PRO`

```
1  ;+
2  ; NAME:
3  ;       MAXLIKE_RUN
4  ; PURPOSE:
5  ;       ENVI extension for classification of
6  ;       a multispectral image with maximum likelihood
7  ; AUTHOR;
8  ;       Mort Canty (2006)
9  ;       Juelich Research Center
10 ;       m.canty@fz-juelich.de
11 ; CALLING SEQUENCE:
12 ;       Maxlike_Run
13 ; ARGUMENTS
14 ;       Event (if used as a plug-in menu item)
15 ; KEYWORDS:
16 ;       None
17 ; DEPENDENCIES:
18 ;       ENVI
19 ;-----------------------------------------------------
```

C.4.2 Gaussian Kernel classifier

`KERNEL_RUN` performs nonparametric classification of a multispectral image using the Gaussian kernel algorithm described in Section 6.4.

C.4.2.1 Usage

The routine is invoked from the ENVI main menu as

```
Classification/Supervised/Gaussian Kernel
```

The user is prompted for an input file, the training ROIs (which must be present beforehand) and output destinations for the classified image, the probability image (optional) and the test results. During calculation a plot window shows the minimization of the misclassification rate with respect to the parameter σ.

C.4.2.2 Source headers

Listing C.15: `KERNEL_RUN.PRO`

```
;+
; NAME:
;       KERNEL_RUN
; PURPOSE:
;       ENVI extension for classification of
;       a multispectral image with a kernel-based
;       classifier
;       Ref: T. Masters, Advanced Algorithms for
;       Neural Networks (1995) Wiley
; AUTHOR;
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Kernel_Run
; ARGUMENTS
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       MINF_BRACKET
;       MINF_PARABOLIC (PROGRESSBAR_DEFINE FSC_COLOR)
;-------------------------------------------------
```

C.4.3 Neural network classifier

`FFN_RUN` classifies a multispectral image with a two-layer, feed-forward neural network trained with a combination of the Kalman filter algorithm of Appendix B.3 and the scaled conjugate gradient algorithm of Section B.2, see also Section B.4.

C.4.3.1 Usage

The routine is invoked from the ENVI main menu as

`Classification/Supervised/Neural Net/Kalman-Congrad`

The user is prompted for an input file, the training ROIs (which must be present beforehand), output destinations for the classified image, the probability image (optional) and the test results, the number of hidden neurons and, finally, whether or not to use validation. During the calculation the cross entropy cost function is displayed. Training can be interrupted at any time, whereupon processing continues to the next phase. After training, the Hessian matrix may be calculated, in which case its eigenvalues are written to a text window.

C.4.3.2 Source headers

Listing C.16: `FFN_RUN.PRO`

```
;+
; NAME:
;       FFN_RUN
; PURPOSE:
;       ENVI extension for classification of a
;       multispectral image  with a feed forward
;       neural network using Kalman filter
;       plus scaled conjugate gradient training
; AUTHOR;
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Ffn_Run
; ARGUMENTS
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       PROGRESSBAR_DEFINE (FSC_COLOR)
;       FFNKAL__DEFINE (FFN__DEFINE)
;       FFNCG__DEFINE
;-------------------------------------------------
```

Listing C.17: `FFNKAL__DEFINE.PRO`

```
;+
; NAME:
;       FFNKAL__DEFINE
; PURPOSE:
;       Object class for implementation of a
```

```
;       two-layer, feed-forward neural network
;       for classification of multi-spectral images.
;       Implements Kalman filter training.
;       Ref:Shaw and Palmieri, Proc. Int. Joint Conf.
;       on Neural Networks, San Diego (1990) I(3), 41-46
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       ffn = Obj_New("FFNKAL",Xs,Ls,L)
; ARGUMENTS:
;       Xs:  array of observation column vectors
;       Ls:  array of class label column vectors
;            of form (0,0,1,0,0,...0)^T
;       L:   number of hidden neurons
; KEYWORDS
;       None
; METHODS:
;       TRAIN: train the network with validation
;              if key=1 (default 0)
;         ffn -> train, key=key
; DEPENDENCIES:
;       FFN__DEFINE
;       PROGRESSBAR(FSC_COLOR)
;-------------------------------------------------
```

Listing C.18: FFNCG__DEFINE.PRO

```
;+
; NAME:
;       FFNCG__DEFINE
; PURPOSE:
;       Object class for implementation of a
;       two-layer, feed-forward neural network
;       for classification of multi-spectral images.
;       Implements scaled conjugate gradient training.
;       Ref: C. Bishop, Neural Networks for
;       Pattern Recognition, (1995) Oxford Univ. Press
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       ffn = Obj_New("FFNCG",Xs,Ls,L)
; ARGUMENTS:
;       Xs:  array of observation column vectors
;       Ls:  array of class label column vectors
;            of form (0,0,1,0,0,...0)^T
```

```
;       L:   number of hidden neurons
; KEYWORDS
;       None
; METHODS:
;       ROP: determine the matrix product v^t.H,
;       where H is the Hessian of
;       the cost function wrt the weights,
;       using the R-operator
;         r = ffn -> Rop(v)
;       HESSIAN: claculate the Hessian
;         h = ffn -> Hessian()
;       EIGENVALUES: calculate the eigenvalues
;       of the Hessian
;         e = ffn -> Eigenvalues()
;       GRADIENT: calculate the gradient of the
;       global cost function
;         g = ffn -> Gradient()
;       TRAIN: train the network with validation
;              if key=1 (default 0)
;         ffn -> train, key=key
; DEPENDENCIES:
;       FFN__DEFINE
;       PROGRESSBAR (FSC_COLOR)
;-------------------------------------------------
```

Listing C.19: FFNBP__DEFINE.PRO

```
;+
; NAME:
;       FFNBP__DEFINE
; PURPOSE:
;       Object class for implementation of a
;       two-layer, feed-forward neural network
;       for classification of multi-spectral images.
;       Implements ordinary backpropagation training.
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       ffn = Obj_New("FFNBP",Xs,Ls,L)
; ARGUMENTS:
;       Xs:  array of observation column vectors
;       Ls:  array of class label column vectors
;            of form (0,0,1,0,0,...0)^T
;       L:   number of hidden neurons
; KEYWORDS
;       None
; METHODS:
```

```
;        TRAIN: train the network
;          ffn -> train
; DEPENDENCIES:
;        FFN__DEFINE
;        PROGRESSBAR (FSC_COLOR)
;-------------------------------------------------
```

Listing C.20: FFN__DEFINE.PRO

```
;+
; NAME:
;       FFN__DEFINE
; PURPOSE:
;     Object class for implementation of a
;     two-layer, feed-forward neural network classifier.
;     This is a generic class with no training methods.
; AUTHOR
;     Mort Canty (2006)
;     Juelich Research Center
;     m.canty@fz-juelich.de
; CALLING SEQUENCE:
;     ffn = Obj_New("FFN",Xs,Ls,L)
; ARGUMENTS:
;     Xs:  array of observation column vectors
;     Ls:  array of class label column vectors
;          of form (0,0,1,0,0,...0)^T
;     L:   number of hidden neurons
; KEYWORDS
;     None
; METHODS (external):
;     FORWARDPASS: propagates a biased input
;                  observation vector through
;                  the network. Returns the softmax
;                  probabilities vector, sets the
;                  hidden layer outputs as side effect
;       m = ffn -> ForwardPass()
;     VFORWARDPASS: propagates an array of biased input
;                   vectors through the network,
;                   returns the corresponding softmax
;                   probability vectors and hidden
;                   layer outputs in variable Ns
;       Ms = ffn -> vForwardPass(Ns)
;     CLASSIFY: returns the classes for an array
;               of observation vectors X
;               returns the class probabilities in
;               array variable PROBS
;       c = ffn -> Classify(X,Probs)
;     COST: returns the current cross entropy
;       c = ffn -> Cost(key)
```

```
41 ;        key = 0 for all training data,
42 ;              1 for odd-numbered,
43 ;              2 for even numbered
44 ;      GET_WEIGHTS: return synaptic weights as a vector
45 ;        w = ffn -> Get_Weights()
46 ;      put_WEIGHTS: insert synaptic weights into network
47 ;        ffn -> Put_Weights, w
48 ; DEPENDENCIES:
49 ;      None
50 ;-------------------------------------------------------
```

C.4.4 Probabilistic label relaxation

`PLR_RUN` is an ENVI/IDL extension for probabilistic label relaxation (PLR) post-processing of supervised or unsupervised classification images, see Section 6.6.2. It takes as input a class probability vector image generated by any of the supervised classification extensions described in this appendix, as well as from the clustering routine `EM_RUN` described in the Section C.5.3, and generates a modified class probability vector image. This can be processed with `PLR_RECLASS` to generate an improved classification image with better spatial coherence.

C.4.4.1 Usage

The routines are invoked from the ENVI main menu as

```
Classification/Post Classification/Probabilistic Label
  Relaxation/Run PLR
```

and

```
Classification/Post Classification/Probabilistic Label
  Relaxation/Reclassify
```

At the prompt for `PLR_RUN`, choose a class membership probabilities image and the number of iterations (default = 3). Then choose a destination either in memory or as a named file. At the prompt for `PLR_RECLASS`, choose a previous classification image if available (to synchronize the class colors) and then a class membership probabilities image. At next prompt for a dummy unclassified class choose "Yes." Then choose a destination either in memory or as a named file.

C.4.4.2 Source headers

Listing C.21: `PLR_RUN.PRO`

```
1 ;+
2 ; NAME:
3 ;        PLR_RUN
4 ; PURPOSE:
```

```
;       ENVI extension for postclassification with
;       Probabilistic Label Relaxation
;       Ref. Richards and Jia, Remote Sensing Digital
;       Image Analysis (1999) Springer
;       Processes a rule image (class membership
;       probabilities), outputs a new rule image
; AUTHOR;
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Plr_Run
; ARGUMENTS
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       PROGRESSBAR_DEFINE (FSC_COLOR)
;-------------------------------------------------
```

Listing C.22: PLR_RECLASS.PRO

```
;+
; NAME:
;       PLR_RECLASS
; PURPOSE:
;       ENVI extension for postclassification with
;       Probabilistic Label Relaxation
;       Ref. Richards and Jia, Remote Sensing
;       Digital Image Analysis (1999) Springer
;       Processes a rule image (class membership
;       probabilities), outputs a
;       new classification file
; AUTHOR;
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Plr_Reclass
; ARGUMENTS
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       PROGRESSBAR_DEFINE (FSC_COLOR)
;-------------------------------------------------
```

C.4.5 Classifier evaluation and comparison

The procedures `CT_RUN` and `McNEMAR_RUN` are used to evaluate and compare test results generated by the supervised classifiers `MAXLIKE_RUN`, `KERNEL_RUN` and `FFN_RUN`, see Section 6.7.

C.4.5.1 Usage

Both routines may be called from the ENVI command prompt. They request an input file with extension `TST` and generate their outputs (contingency table etc.) in the IDL Log window.

C.4.5.2 Source headers

Listing C.23: `CT_RUN.PRO`

```
;+
; NAME:
;       CT_RUN
; PURPOSE:
;       ENVI extention to determine contingency
;       table (confusion matrix) and classification
;       accuracies from test classifcation results
;       Ref. Richards and Jia, Remote Sensing
;       Digital Image Analysis (3rd Ed), Springer, 1999
; AUTHOR;
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       CT_Run
; ARGUMENTS:
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
; ------------------------------------------
```

Listing C.24: `MCNEMAR_RUN.PRO`

```
;+
; NAME:
;       MCNEMAR_RUN
; PURPOSE:
;       ENVI extention to compare classifiers
;       on the basis of misclassifications
;       using McNemar's statistic
;       Ref. T. G. Dietterich, Neural Computation
```

```
;        10 (1998) 1895-1923
; AUTHOR;
;        Mort Canty (2006)
;        Juelich Research Center
;        m.canty@fz-juelich.de
; CALLING SEQUENCE:
;        McNemar_Run
; ARGUMENTS:
;        Event (if used as a plug-in menu item)
; KEYWORDS:
;        None
; DEPENDENCIES:
;        ENVI
; ------------------------------------------
```

C.5 Unsupervised classification

C.5.1 Agglomerative hierarchical clustering

`HCL_RUN` is an ENVI/IDL extension for agglomerative hierarchical clustering (HCL) of multispectral imagery as discussed in Section 7.2.3. Clustering is performed on a small random sample of image pixels (≤ 1500), and generalized to the entire image by using ENVI's built-in maximum likelihood supervised classification algorithm.

C.5.1.1 Usage

The routine is invoked from the ENVI main menu as

`Classification/Unsupervised/Hierarchic`

Choose (a spatial/spectral subset of) a multispectral image in the first dialog. Then enter the number of random samples and number of clusters in the corresponding prompts. Finally indicate whether the classified image is to be saved to disk or memory. During computation, the current number of clusters is displayed in the progress bar.

C.5.1.2 Source headers

Listing C.25: `HCL_RUN.PRO`

```
;+
; NAME:
;        HCL_RUN
; PURPOSE:
;        ENVI extension for agglomerative hierarchical
```

```
;        clustering
; AUTHOR
;        Mort Canty (2006)
;        Juelich Research Center
;        m.canty@fz-juelich.de
; CALLING SEQUENCE:
;        HCL_Run
; ARGUMENTS:
;        Event (if used as a plug-in menu item)
; KEYWORDS:
;        None
; DEPENDENCIES:
;        ENVI
;        HCL (PROGRESSBAR__DEFINE (FSC_COLOR))
;        CLASS_LOOKUP_TABLE
;-------------------------------------------------------
```

Listing C.26: HCL.PRO

```
;+
; NAME:
;        HCL
; PURPOSE:
;        Agglomerative hierarchic clustering with
;        sum of squares cost function.
;        Takes data array Xs (column vectors) and number
;        of clusters K as input.
;        Returns cluster memberships Cs.
;        Ref. Fraley Technical Report 311, Dept. of Stat.,
;        University of Washington, Seattle (1996).
; AUTHOR
;        Mort Canty (2006)
;        Juelich Research Center
;        m.canty@fz-juelich.de
; CALLING SEQUENCE:
;        HCL, Xs, K, Cs
; ARGUMENTS:
;         Xs: input observations array (column vectors)
;         K:  number of clusters
;         Cs: Cluster labels of observations
; KEYWORDS:
;         None
; DEPENDENCIES:
;        PROGRESSBAR__DEFINE (FSC_COLOR)
;-------------------------------------------------------
```

Listing C.27: CLASS_LOOKUP_TABLE.PRO

```
;+
; NAME:
```

```
3 ;        CLASS_LOOKUP_TABLE
4 ; PURPOSE:
5 ;        Provide 16 class colors for supervised
6 ;        and unsupervised classification programs
7 ; AUTHOR;
8 ;        Mort Canty (2006)
9 ;        Juelich Research Center
10 ;       m.canty@fz-juelich.de
11 ; CALLING SEQUENCE:
12 ;       colors = Class_Lookup_Table(Ptr)
13 ; ARGUMENTS:
14 ;       Ptr: a vector of pointers into the table
15 ; KEYWORDS:
16 ;       None
17 ; DEPENDENCIES:
18 ;       None
19 ;-------------------------------------------------------
```

C.5.2 Fuzzy K-means clustering

`FKM_RUN` is an ENVI/IDL extension for fuzzy K-means (FKM) clustering of multispectral imagery (Section 7.2.4). Clustering is performed on a random sample of image pixels ($\leq 10^5$), and generalized to the entire image by using a modification of the IDL distance classifier `CLUSTER_FKM`.

C.5.2.1 Usage

The routine is invoked from the ENVI main menu as

`Classificatio/Unsupervised/Fuzzy-K-Means`

Choose (a spatial/spectral subset of) a multispectral image at the prompt. Then enter the desired number of clusters and whether or not they should be saved as ROIs (ENVI regions of interest) at the corresponding prompts. Finally indicate whether the class image is to be saved to disk or memory.

C.5.2.2 Source headers

Listing C.28: `FKM_RUN.PRO`

```
1 ;+
2 ; NAME:
3 ;        FKM_RUN
4 ; PURPOSE:
5 ;        ENVI extension for fuzzy K-means clustering
6 ;        with sampled data
7 ; AUTHOR
8 ;        Mort Canty (2006)
9 ;        Juelich Research Center
```

```
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       FKM_Run
; ARGUMENTS:
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       FKM (PROGRESSBAR_DEFINE (FSC_COLOR))
;       CLUSTER_FKM
;       CLASS_LOOKUP_TABLE
;-----------------------------------------------------
```

Listing C.29: FKM.PRO

```
;+
; NAME:
;       FKM
; PURPOSE:
;       Fuzzy Kmeans clustering algorithm.
;       Takes data array Xs and initial fuzzy
;       membership matrix (as column vectors).
;       Returns fuzzy membership matrix U and
;       the class centers Ms.
;       Ref: J. C. Dunn, Journal of Cybernetics,
;       (1973) PAM1:32-57
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       FKM, Xs, K, U, Ms, niter=niter, seed=seed
; ARGUMENTS:
;        Xs: input observations array (column vectors)
;        U:  class probability membership matrix
;            (input and output)
;        Ms: cluster means (output)
; KEYWORDS:
;        niter: number of iterations (optional)
;        unfrozen: observations taking part in
;        iteration (default all)
; DEPENDENCIES:
;       PROGRESSBAR__DEFINE (FSC_COLOR)
;-----------------------------------------------------
```

Listing C.30: CLUSTER_FKM.PRO

```
;+
; NAME:
```

```
;       CLUSTER_FKM
; PURPOSE:
;       Distance clusterer from IDL library
;       Modified for FKM (Mort Canty (2006))
; CALLING SEQUENCE:
;       labels =  Cluster_fkm(Array,Weights, $
;             Double=Double,N_clusters=N_clusters)
;-----------------------------------------------
```

C.5.3 Fuzzy maximum likelihood estimation clustering

`EM_RUN` is an ENVI/IDL extension for fuzzy maximum likelihood estimation (FMLE) — or equivalently expectation maximization (EM) — clustering of multispectral imagery. It is discussed in Sections 7.3 and 7.4. Clustering occurs optionally at different scales, and both simulated annealing and spatial memberships can be included if desired. The program will also optionally generate class membership probability images which can be postprocessed with probabilistic label relaxation (see Section C.4.4).

C.5.3.1 Usage

The routine is invoked from the ENVI main menu as

`Classification/Unsupervised/FMLE(EM)`

Choose (a spatial/spectral subset of) a multispectral image at the prompt. Then enter the desired number of clusters and compressions (pyramid depth) at the corresponding prompts. In the `Clustering parameters` dialog enter the initial annealing temperature (zero for no annealing) and spatial membership parameter `Beta` (zero for no spatial memberships). At the next prompt indicate whether or not to save the clusters as ROIs. Finally say whether the class image is to be saved to disk or memory and similarly for the membership probability image (optional).

C.5.3.2 Source headers

Listing C.31: `EM_RUN.PRO`

```
;+
; NAME:
;       EM_RUN
; PURPOSE:
;       ENVI extension for FMLE(EM) clustering at
;       different scales using DWT compression
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
```

```
12 ;        EM_run
13 ; ARGUMENTS:
14 ;        Event (if used as a plug-in menu item)
15 ; KEYWORDS:
16 ;        None
17 ; DEPENDENCIES:
18 ;        ENVI
19 ;        EM (PROGRESSBAR__DEFINE (FSC_COLOR))
20 ;        DWT__DEFINE
21 ;        CLASS_LOOKUP_TABLE
22 ;-----------------------------------------------
```

Listing C.32: EM.PRO

```
1  ;+
2  ; NAME:
3  ;        EM
4  ; PURPOSE:
5  ;        FMLE (EM) clustering for Gaussian mixtures.
6  ;        Takes data array Xs (column vectors) and initial
7  ;        class membership probability matrix U as input.
8  ;        Returns U, the class centers Ms, Priors Ps and
9  ;        final class covariances Fs.
10 ;        Allows for simulated annealing
11 ;        Ref: Gath and Geva, IEEE Trans. Pattern Anal. and
12 ;             Mach. Intell. 3(3):773-781, 1989
13 ;             Hilger, Exploratory Analysis of Multivariate
14 ;             Data, PhD Thesis, Tech. Univ. Denmark, 2001
15 ; AUTHOR
16 ;        Mort Canty (2006)
17 ;        Juelich Research Center
18 ;        m.canty@fz-juelich.de
19 ; CALLING SEQUENCE:
20 ;        EM, Xs, U, Ms, Ps, Fs, $
21 ;         unfrozen=unfrozen,wnd=wnd, $
22 ;         maxiter=maxiter,miniter=miniter,verbose=verbose,$
23 ;         pdens=pdens,pd_exclude=pd_exclude,fhv=fhv,T0=T0,$
24 ;         pb_msg=pb_msg,beta=beta, $
25 ;         num_cols=num_cols,num_rows=num_rows
26 ; ARGUMENTS:
27 ;         Xs: input observations array (column vectors)
28 ;         U:  initial class probability membership matrix
29 ;             (column vectors)
30 ;         Ms: cluster means (output)
31 ;         Ps: cluster priors (output)
32 ;         Fs: cluster covariance matrices (output)
33 ; KEYWORDS:
34 ;         unfrozen: Indices of the observations which
35 ;                 take part in the iteration (default all)
```

```
;         wnd: window for displaying the log likelihood
;              (optional)
;         maxinter: maximum iterations (optional)
;         minimter: minimum iterations (optional)
;         pdens: partition density (output, optional)
;         fhv: fuzzy hypervolume (output, optional)
;         T0: initial annealing temperature (default 1.0)
;         verbose: set to print output info to IDL log
;         pb_msg: string for progressbar
;         beta: spatial field parameter
;         num_cols, num_rows: image dimensions (if beta>0)
; DEPENDENCIES:
;        PROGRESSBAR__DEFINE (FSC_COLOR)
;-------------------------------------------------------
```

C.5.4 Kohonen self-organizing map

`SOM_RUN` is an ENVI/IDL extension for unsupervised image classification using the Kohonen self-organizing map, see Section 7.6.

C.5.4.1 Usage

The routine is invoked from the ENVI main menu as

`Classification/Unsupervised/SOM`

At the prompts enter the (spectral/spatial subset of the) image to be classified, the dimension of the neuron cube (default $6\times6\times5$), and the output destination for the classified image. A three-dimensional plot of the neuron cube projected onto the space of the first three image bands is generated as a side effect.

C.5.4.2 Source headers

Listing C.33: `SOM_RUN.PRO`

```
;+
; NAME:
;        SOM_RUN
; PURPOSE:
;        ENVI extension for Kohonen Self Organizing Map
;        Ref. T. Kohonen, Self Organization and
;        Associative Memory (1989) Springer.
; AUTHOR
;        Mort Canty (2006)
;        Juelich Research Center
;        m.canty@fz-juelich.de
; CALLING SEQUENCE:
;        SOM_Run
; ARGUMENTS:
;        Event (if used as a plug-in menu item)
```

```
16 ; KEYWORDS:
17 ;       None
18 ; DEPENDENCIES:
19 ;       ENVI
20 ;       PROGRESSBAR__DEFINE (FSC_COLOR)
21 ;-------------------------------------------------------
```

C.6 Change detection

C.6.1 Multivariate alteration detection

`MAD_RUN` is an ENVI/IDL extension for the multivariate alteration detection (MAD) method of Section 8.4. It allows for iterative reweighting. It (optionally) generates two canonical variate images and it (optionally) generates the MAD variates together a chi-square image. The latter can be used for masking invariant pixels for radiometric normalization with `RADCAL_RUN` discussed in Section C.6.3 below.

C.6.1.1 Usage

The routine is invoked from the ENVI main menu as

`Basic Tools/Change Detection/MAD`

Choose (a spatial/spectral subset of) the two multispectral images at the prompt. Then enter the number of iterations (minimum zero for uniterated MAD, maximum 50). Enter the output destinations for the MAD and canonical variates (CVs). If the latter are to be output to a file, the program will append `_1` and `_2` to the file name, to distinguish the two CV files which are created. If the images are not of the same spectral/spatial dimension, the program will abort. If an image is not in band interleaved by pixel (BIP) or band interleaved by line (BIL) format, the user will be prompted to allow it to be converted to BIP in place (the image must reside on disk in this case).

C.6.1.2 Source headers

Listing C.34: `MAD_RUN.PRO`

```
1 ;+
2 ; NAME:
3 ;       MAD_RUN
4 ; PURPOSE:
5 ;       ENVI extension for Iteratively Re-weighted
6 ;       Multivariate Alteration Detection (IR-MAD).
7 ;       Ref: A. A. Nielsen et al. Remote Sensing of
8 ;            Environment 64 (1998), 1-19
```

```
;            A. A. Nielsen, submitted for publication
;       Uses spectral tiling and therefore suitable
;       for large datasets.
;       Reads in two registered multispectral images.
;       Image subsets must have the same spatial/spectral
;       dimensions, spectral subset size must be at least 2,
;       If an input image is in BSQ format, it is, after
;       a warning, converted in place to BIP.
;       Writes the IR-MADs, canonical variates and
;       chi-square statistic to memory or disk.
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Mad_Run
; ARGUMENTS:
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       MAD_ITER (COVPM_DEFINE, GEN_EIGENPROBLEM)
;       PROGRESSBAR_DEFINE (FSC_COLOR)
;-------------------------------------------------------
```

Listing C.35: MAD_ITER.PRO

```
;+
; NAME:
;       MAD_ITER
; PURPOSE:
;       Function for Iteratively Re-weighted Multivariate
;       Alteration Detection (IR-MAD).
;       Ref: A. A. Nielsen et al. Remote Sensing
;            of Environment 64 (1998), 1-19
;            A. A. Nielsen submitted for publication
;       Input files must be BIL or BIP format and
;       have equal spatial/spectral dimensions.
;       On error or if interrupted during the first
;       iteration, returns = -1 else 0
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       result = Mad_Iter(fid1,fid2,dims1,dims2,pos1,pos2,$
;         A=A,B=B,EM=EM,NCP=NCP,chi_sqr=chi_sqr,niter=niter,$
;         means1=means1, means2=means2, rho=rho)
```

```
22 ; ARGUMENTS:
23 ;       fid1, fid2      input file specifications
24 ;       dims1, dims2
25 ;       pos1, pos2
26 ; KEYWORDS:
27 ;       A, B            transformation eigenvectors,
28 ;                       smallest correlation first
29 ;       means1,means2   weighted mean values, row-replicated
30 ;       rho             canonical correlations in decreasing
31 ;                       order
32 ;       NCP             no change probabilities image
33 ;       chi_sqr         chi-square image
34 ;       niter           number of iterations, default is 5
35 ; DEPENDENCIES:
36 ;       ENVI
37 ;       COVPM_DEFINE
38 ;       GEN_EIGENPROBLEM
39 ;       DWT__DEFINE
40 ;       PROGRESSBAR_DEFINE (FSC_COLOR)
41 ;-------------------------------------------------------------
```

Listing C.36: COVPM__DEFINE.PRO

```
1  ;+
2  ; NAME:
3  ;       COVPM__DEFINE
4  ; PURPOSE:
5  ;       Object class for sequential covariance matrix
6  ;       calculation using the method of provisional means.
7  ; AUTHOR
8  ;       Mort Canty (2006)
9  ;       Juelich Research Center
10 ;       m.canty@fz-juelich.de
11 ; CALLING SEQUENCE:
12 ;       covpm = Obj_New("COVPM",p)
13 ; ARGUMENTS:
14 ;       p: dimension of the covariance matrix
15 ; KEYWORDS
16 ;       None
17 ; METHODS:
18 ;       UPDATE: update the covariance matrix
19 ;       with an array (spectral tile) of observations
20 ;       covpm -> Update, Xs, weights = Ws
21 ;         Xs is an array of observation row vectors
22 ;         Ws is an optional array of weights for
23 ;         the observations
24 ;       COVARIANCE: return the covariance matrix
25 ;       cov = covpm -> Covariance()
26 ;       MEANS: return the observation means
```

```
;       mns = covpm -> Means()
; DEPENDENCIES:
;       PROV_MEANS.DLL
;-------------------------------------------------------
```

Listing C.37: PROV_MEANS.C

```
#include <stdio.h>
#include "idl_export.h"

/*
;DLL for provisional means algorithm, called from the object
;class DEFINE__COVPM in order to avoid IDL for-loop.
;Compile using IDL procedure MAKE_DLL.
;Under MS Windows OS, use MAKE_PROV_MEANS.PRO,
;assuming VC++ 6.0 or later compiler is installed.
;For other OS, see IDL documentation for MAKE_DLL.
-------------------------------------------------------*/
```

Listing C.38: GEN_EIGENPROBLEM.PRO

```
;+
; NAME:
;       GEN_EIGENPROBLEM
; PURPOSE:
;       Solve the generalized eigenproblem
;                       C##a = lambda*B##a
;       using Cholesky factorization
; AUTHOR:
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Gen_Eigenproblem, C, B, A, lambda
; ARGUMENTS:
;       C and B are real, square, symmetric matrices
;       B is positive definite
;       returns the eigenvalues in the row vector lambda
;       returns the eigenvectors a in the columns of A
; KEYWORDS:
;       None
; DEPENDENCIES
;       None
;-------------------------------------------------------
```

C.6.2 Viewing changes

MAD_VIEW is an ENVI/IDL GUI for viewing MAD or MAD/MNF variates, see Section 8.5. It is provided with a rudimentary online help PDF file.

C.6.2.1 Usage

The routine is invoked from the ENVI main menu as

```
Basic Tools/ Change Detection/MAD View
```

C.6.2.2 Source headers

Listing C.39: `MAD_VIEW.PRO`

```
;+
; NAME:
;       MAD_VIEW
; PURPOSE:
;       GUI for viewing and thresholding MAD images
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Mad_View
; ARGUMENTS:
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       EM
;       PROGRESSBAR_DEFINE (FSC_COLOR)
;-----------------------------------------------
```

C.6.3 Radiometric normalization

`RADCAL_RUN` is an ENVI/IDL extension for radiometric normalization of two multispectral images using iteratively reweighted MAD to determine time invariant pixels, see Section 8.6.

C.6.3.1 Usage

The routine is invoked from the ENVI main menu as

```
Basic Tools/Change Detection/MAD Radiometric Normalization
```

Choose (a spatial/spectral subset of) the two multispectral images at the prompt. Then enter the output destination. If the image(s) are not of the same spectral/spatial dimension, the program will abort. Then the user is prompted for the chi-square image generated previously by `MAD_RUN` on the same image pair, the minimum probability to use to identify no-change pixels and finally for the output destination (file or memory). During the calculation, the orthogonal regressions are plotted in separate plot windows. After

completion, another image (e.g., a full scene) may be normalized with the regression coefficients that were determined. The output is then to file only. Regression statistics and results of statistical tests calculated with the invariant pixels are written to a text window.

C.6.3.2 Source headers

Listing C.40: RADCAL_RUN.PRO

```
;+
; NAME:
;       RADCAL_RUN
; PURPOSE:
;       Radiometric calibration using MAD
;       Ref: M. Canty et al. Remote Sensing of
;       Environment 91(3,4) (2004) 441-451
;       Reference and target images must have
;       equal spatial and spectral dimensions,
;       at least 2 spectral components, and be
;       registered to one another.
;       Once the regression coefficients have been
;       determined, they can be used to
;       calibrate another file, for example a full
;       scene, which need not be registered
;       to the reference image.
; AUTHOR
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       Radcal_Run
; ARGUMENTS:
;       Event (if used as a plug-in menu item)
; KEYWORDS:
;       None
; DEPENDENCIES:
;       ENVI
;       ORTHO_REGRESS
;       WISHART
;       PROGRESSBAR_DEFINE (FSC_COLOR)
;-------------------------------------------------------
```

Listing C.41: WISHART.PRO

```
;+
; NAME:
;       WISHART
; PURPOSE:
;       Test two multivariate distributions for
```

```
;       equal means and covariance matrices
;       assuming both parameter sets are estimated
;       with the same sample size
;       Ref. Anderson (2003) pp. 416-426
; AUTHOR:
;       Mort Canty (2006)
;       Juelich Research Center
;       m.canty@fz-juelich.de
; CALLING SEQUENCE:
;       pVal=Wishart(M1,M2,S1,S2,n,statistic=statistic)
;       returns the probability of observing a
;       larger value of the realization of the
;       test statistic Q = -2*rho*log(lambda)
;       under the null hypothesis
; ARGUMENTS:
;       M1, M2 are the means
;       S1, S2 are the covariance matrices
;       n is the (common) number of samples
;       used to estimate them
; KEYWORDS:
;       statistic
;       the test statistic -2*rho*log(lambda)
; DEPENDENCIES
;       None
;-------------------------------------------------------
```