List of m-files, with initial comments lines, from `D:\Karl\Matlab\FrameTools\*.m`.
This list was printed 22-Sep-2003 15:59:53 by the MakeTex.m function.

contents.m	4330 bytes	22-Sep-2003 15:10:22

```
% Text describing the m-files in directory D:\Karl\Matlab\FrameTools
% File generated by mkcontnt.m 22-Sep-2003 15:10:21
%
% BulidFg   Build the Fg matrix from F vector and G matrix
% BulidG    Build the G matrix and F vector from Fg matrix
% C_ana     The analysis part of an IIR filter bank based on allpass filters
% C_anablock  Analysis block in a FIR two channel filterbank
% C_anafb   Tree structured analysis filter bank using IIR filter.
% C_syn     The synthesis part of an IIR filter bank based on allpass filters
% C_synblock  Synthesis block in a FIR two channel filterbank
% C_synfb   Tree structured synthesis filter bank using IIR filter.
% DataFile    Explain the contents of a mat-file that stores a set of training data.
% Decom1D   Decompose a 1D-signal into expansion coefficients,
% Decom2D    Decompose a 2D-signal (image) into expansion coefficients,
% DesignF   Design the frame and store it in FrameFile, based on training data
% DesignFb  This is a stripped version of DesignF, only for block-oriented frame
% FrameDesignEx   Here we design an example frame for an AR(1) signal.
% FrameFile   Explain the contents of a mat-file that stores a frame.
% FrameInfo Display information about the frame stored in a mat-file,
% FrameProperties   Find, display and return some properties of a frame F
% FramePropertiesObjFun  Objective function for finding the x that is
% GenIntPol  General Interpolation of Images/Patterns, generate textures
% GenLloyd  Design a codebook (frame) by the Generalized Lloyd Algorithm
% GetELT    Extendend Lapped (orthogonal) Transform, ELT, of size 4NxN
% GetHaar   Find the NxN Haar matrix
% GetLOT    Lapped Orthogonal Transform of size 2NxN
% GetWave   Get a synthesis transform matrix based on a dyadic wavelet filter bank
% ImDC       Split an image into DC component and residual, or opposite.
% MapCD     Makes the tables that map from WWT (or W*W') to C
% MCsimF    Monte Carlo simulation for the frame F, selecting S vectors
% NormalizeF   Normalize the synthesis vectors in frame F
% PDFpoly   Estimate the pdf (probability density function) by polynomials
% PlotF     Plot the column vectors of the one-dimensional frame F
% PlotF2D    Plot the image blocks in frame F, F is size NxK
% PlotFt     Plot the image blocks in frame F, F is size NxK
% PlotSNR   Plot development in SNR after each iteration during training
% PolyInterPol Polynomial interpolation of a function given as points.
% Reorder    Reorder distinct image blocks into columns, or vice versa
% SetF03    Set F using initial values from DCT/Haar/LOT/ELT/eye/randn
% SetF05    Set F as a general filter bank, using initial values from FIR filters
% SetF06    Set F as a general filter bank, using initial values from a wavelet
% SetF07    Set F as a general filter bank with N=8, K=16, P=6 and Q=208
% SetFfromImage    Set F as a frame of randomly selected blocks from image B.
% SignalExpansion   Here we test many different methods for signal expansion.
% TestGenLloyd  Test of the function GenLloyd
% TestPDFpoly  Test of the function PDFpoly and PolyInterPol
% TestVS  test of Vector Selection algorithms
% TestVSblockC     Test of VSblockC, vector selection done by program
% Ttimes    Do the operation Y=T*X, where T represent a filterbank or a transform.
% VSab2     Vector Selection algorithm that Always returns a Better (or the same) w
% VSblock   Vector Selection for block-oriented frame,
% VSblockC  Vector Selection for block-oriented frame, (special version)
% VSfomp2   Fast Orthogonal Matching Pursuit 2, Vector Selection algorithm
% VSfs      Vector Selection algorithm doing Full Search
% VSmp2     Matching Pursuit variant 2, Vector Selection algorithm
```

```
% VSolap1   Vector Selection for overlapping frame and 1D signal
% VSolap2    Vector Selection for overlapping frame and 2D signal
% VSomp     Orthogonal Matching Pursuit with QR decomposition, Vector Selection algorithm
% VSompJHH  Orthogonal Matching Pursuit, Vector Selection algorithm
% VSormp    Order Recursive Matching Pursuit with QR decomposition, Vector Selection algorithm
% VSormp2   Order Recursive Matching Pursuit with QR decomposition, Vector Selection algorithm
% VSps      Vector Selection algorithm doing Partial Search
% VSps2     Partial Search 2, Vector Selection algorithm
```

BuildFg.m	1893 bytes	27-Nov-2002 14:46:04

```
% BulidFg   Build the Fg matrix from F vector and G matrix
%           (Fg for type 'g' correspond to F for type 'b' and 'o')
% Fg(n,k,p)=sign*F(q)  <==>  G(n,k,p)=sign*q     and
% Fg(n,k,p)=0          <==>  G(n,k,p)=0
% Fg=BuildFg(F,G);
% --------------------------------------------------------------------------------
% Arguments:
%   Fg        - the general frame Fg, size NxKxP.
%   F         - a vector of all the different variables in F, size Qx1
%   G         - the "mapping" from F to Fg, Fg(n,k,p)=sign*F(q) <==> G(n,k,p)=sign*q
%               size NxKxP
% --------------------------------------------------------------------------------
% Note that if G is sparse then Fg will be sparse
%-------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  27.04.1999  KS: Function made
% Ver. 1.1  18.06.1999  KS: May use three dimensional G and Fg
% Ver. 1.2  10.03.2000  KS: may have negative values in G
% Ver. 1.3  09.01.2001  KS: changed name to BuildFg (from BuildF)
% Ver. 1.4  27.11.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------
```

BuildG.m	2238 bytes	04-Dec-2002 19:08:42

```
% BulidG    Build the G matrix and F vector from Fg matrix
%           (Fg for type 'g' correspond to F for type 'b' and 'o')
% Fg(n,k,p)=sign*F(q)  <==>  G(n,k,p)=sign*q     and
% Fg(n,k,p)=0          <==>  G(n,k,p)=0
% [F,G]=BuildG(Fg,delta);
% --------------------------------------------------------------------------------
% Arguments:
%   F         - a vector of all the different variables in F, size Qx1
%   G         - the "mapping" from F to Fg, Fg(n,k,p)=sign*F(q) <==> G(n,k,p)=sign*q
%               size NxKxP
%   Fg        - the general frame Fg, size NxKxP.
%   delta     - a small number, numbers which differece is smaller than
%               delta are considered to be equal. Default is 5*eps
%               use delta=0 to let all non-zero variables be free
% --------------------------------------------------------------------------------
%-------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
```

```
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  09.01.2001  KS: functio made
% Ver. 1.1  04.12.2002  KS: moved from ..\Frames\ to ..\FrameTools
%--------------------------------------------------------------------
```

C-ana.m	3047 bytes	02-Dec-2002 14:52:13

```
% C_ana     The analysis part of an IIR filter bank based on allpass filters
%           Each column of X is filtered independently of the other columns
% to give two columns in Y.
% Block diagram, x is a column of X. y0 and y1 are columns of Y.
%
%         even x       +-------+  v0             y0
%      +-------------->| A0(z) |------------>(+)----->
%  x   |               +-------+      \     /
% -----+                               x
%      |  odd x        +-------+  v1  /     \    y1
%      +-------------->| A1(z) |-------->--->(+)----->
%                      +-------+       -1
% Y = C_ana(X,a0,a1)
% ------------------------------------------------------------------------
% arguments:
%  X        the signal to be filtered
%  a0,a1    the values of a0 and a1 used in the IIR filterbank.
% ------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 1998.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  27.03.1998  Karl Skretting, used in Master thesis 1998, HIS:
%                       "Kompresjon av seismiske data"
% Ver. 1.1  20.07.2000  KS: added optional arguments a0, a1
% Ver. 1.2  02.12.2002  KS: moved from ..\Frames to ..\FrameTools
%--------------------------------------------------------------------
```

C-anablock.m	2185 bytes	02-Dec-2002 15:46:10

```
% C_anablock  Analysis block in a FIR two channel filterbank
% y=C_anablock(x,h1,h2);
% ----------------------------------------------
% Arguments:
%   x    - the signal, a column vector (or a matrix of column vectors)
%          each column is filtered independently of each other
%          [N,L]=size(x);   and we should have mod(N,2)==0
%   h1   - the FIR filter for the upper branch (low pass)
%   h2   - the FIR filter for the lower branch (high pass)
%   y    - the subbands, size(y)=[N/2,2*L], where odd columns correspond to
%          the upper branch, and even columns to the lower branch
% ----------------------------------------------
% Note that we assume circular expansion of the signal x
%--------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
```

```
% HISTORY:
% Ver. 1.0  20.08.1999  KS: Function made
% Ver. 1.1  02.12.2002  KS: moved from ..\Frames to ..\FrameTools\
%-------------------------------------------------------------------------
```

C-anafb.m	2265 bytes	02-Dec-2002 14:54:20

```
% C_anafb   Tree structured analysis filter bank using IIR filter.
% Y = C_anafb(X, N);
% Y = C_anafb(X, N, a0, a1);
% -------------------------------------------------------------------------
% arguments:
%  X        the signal to be filtered, a real column vector of size Lx1
%           L should have a factor N.
%  N        number of subbands, we should have N=2^k, k=1,2,..8,
%           default value of N is 2.
%  a0,a1    two optional parameters, give both or none.
%           These are the values of a0 and a1 used in the IIR filterbank.
% -------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 1998.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  27.03.1998  Karl Skretting, used in Master thesis 1998, HIS:
%                       "Kompresjon av seismiske data"
% Ver. 1.1  20.07.2000  KS: added optional arguments a0, a1
% Ver. 1.2  02.12.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------------
```

C-syn.m	3303 bytes	02-Dec-2002 14:56:02

```
% C_syn     The synthesis part of an IIR filter bank based on allpass filters
%           Two and two columns of Y are filtered independently of the other columns
% to give one column in X.
% Block diagram, x is a column of X. y0 and y1 are columns of Y.
%
%     y0              v0  +----------+   even x
%   ----------->(+)------>| A0(z^-1) |-----------+
%          \   /          +----------+           |       x
%            x                                   +--->--->
%     y1   /   \      v1  +----------+           |  0.5
%   -------->-->(+)------>| A1(z^-1) |-----------+
%           -1            +----------+   odd x
%
% X = C_syn(Y,a0,a1);
% -------------------------------------------------------------------------
% arguments:
%  Y        the subband signal to be filtered, number of columns should be even.
%  a0,a1    the values of a0 and a1 used in the IIR filterbank.
% -------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 1998.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
```

```
% Ver. 1.0  27.03.1998  Karl Skretting, used in Master thesis 1998, HIS:
%                       "Kompresjon av seismiske data"
% Ver. 1.1  20.07.2000  KS: added optional arguments a0, a1
% Ver. 1.2  02.12.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------------
```

C-synblock.m	2297 bytes	02-Dec-2002 15:46:15

```
% C_synblock  Synthesis block in a FIR two channel filterbank
% x=C_synblock(y,g1,g2);
% ----------------------------------------------
% Arguments:
%   y    - the subbands, odd columns correspond to the upper branch (low pass)
%          and even columns correspond to the lower branch (high pass)
%          We should have [N,L]=size(y) and L an even number
%   x    - the resulting signal, two and two columns of y are filtered to
%          give one column of x, size(x)=[2*N,L/2]
%   g1   - the FIR filter for the upper branch (low pass)
%   g2   - the FIR filter for the lower branch (high pass)
% ----------------------------------------------
% Note that we assume circular expansion of the sunbands y (and signal x)
%-------------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  20.08.1999  KS: Function made
% Ver. 1.1  02.12.2002  KS: moved from ..\Frames to ..\FrameTools\
%-------------------------------------------------------------------------
```

C-synfb.m	1981 bytes	02-Dec-2002 14:57:41

```
% C_synfb   Tree structured synthesis filter bank using IIR filter.
% X = C_synfb(Y);
% X = C_synfb(Y, a0, a1);
% ------------------------------------------------------------------------
% arguments:
%  Y        the subband signals to be filtered, a real matrix
%           this is a matrix of real numbers of size size LxN
%           N is the number of subbands, we should have N=2^k, k=1,2,..8,
%  a0,a1    two optional parameters, give both or none.
%           These are the values of a0 and a1 used in the IIR filterbank.
% ------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 1998.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  27.03.1998  Karl Skretting, used in Master thesis 1998, HIS:
%                       "Kompresjon av seismiske data"
% Ver. 1.1  20.07.2000  KS: added optional arguments a0, a1
% Ver. 1.2  02.12.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------------
```

DataFile.m	1950 bytes	26-Nov-2002 17:50:21

```
% DataFile    Explain the contents of a mat-file that stores a set of training data.
%            These files are usually temporary files.
% With the programs in this catalog it is possible to use very large sets
% of training data. This makes it necessary to store these in temporary mat-files,
% one file for each signal set (image). The filename is DataXmmm.mat
% mmm is three digits from 001 to Mdat. The file should contain
%  Name      - name of this actual signal, ex: lena, baboon, beethoven5th,
%  X_DC      - a number giving the DC component subtracted from the signal
%  X_LP      - the low-pass component of the image, as made by PreProc.Prog1
%  X         - the signal after preprocessing, the signal should be
%              reshaped into size NxL where N is given by the frame size.
%  SizeX     - the original size of X, (N1L1)x(N2L2)x...
%  ss2       - for bwImages 255^2*pixels, else sum((x-mean(x))^2)
%              for zero-mean signals this is the same as the energy of X
%  S         - how many frame vectors to use for each signal vector, size 1xL
%  WWT       - this is W*W' or Wstar*Wstar', size is KxKxIc
%  XWT       - this is X*W' or X*Wstar', size is NxKxP
% Example that save the signal X into the temporary data file numbered m:
% t1=int2str(m);t1=['0000',t1];t1=t1((length(t1)-2):length(t1));
% DataFile=['DataX',t1];
% save(DataFile,'Name','X_DC','X_LP','X','SizeX','ss2','S','WWT','XWT');
%-----------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  25.12.2002  KS: text moved from ..\Frames\readme.m to ..\FrameTools
%-----------------------------------------------------------------------
```

Decom1D.m	16517 bytes	03-Dec-2002 19:19:30

```
% Decom1D   Decompose a 1D-signal into expansion coefficients,
%           or the inverse function if Method is negative.
% Most decompositions methods suppurted by this function are orthogonal.
% Many of these methods also use other functions to do the real work.
% A special non-orthogonal case, frames, is also supported, the frame should
% then be stored in a mat-file with variables as specified in FrameFile.m
% Examples:
% W = Decom1D(Method, X);            % decomposition by Method
% X = Decom1D(-Method, W);           % inverse decomposition by Method
% W = Decom1D(Method, X, p1, p2);    % also using parameters p1, p2
% X = Decom1D(-Method, W, p1, p2);   % inverse, using parameters p1, p2
% ------------------------------------------------------------------------
% arguments:
%  X        the signal, a column vector of real data, size Samplesx1
%  Method   An integer for the decomposition method
%           Negative sign is used for the inverse function
%           1        - No decomposition
%           2-128    - 2x2 to 128x128 Discrete Cosine Transform, DCT
%           201-207  - Tree structure IIR filter bank, 2^(Method-200) subbands
%                      Two additional arguments may be given, they are optional
%             p1       a0 used in the IIR filterbank.
%             p2       a1 used in the IIR filterbank.
%                      NOTE: energy of W will be 2^(Method-200) times energy of X
%           210      - 32x16 Lapped Orthogonal Transform, LOT.
%                      (this is the same as Method 225)
%           211-217  - Dyadic filter bank, Daubechies 7-9 biorthogonal filters
%                      1-7 (Method-210) levels, 2-8 (Method-209) subbands
%           218      - wavelet decomposition (The wavelet toolbox is needed!)
```

```
%                       Two additional arguments should be given (default 'db6', 4 levels)
%                       db6 is the Daubechies wavlets, filterlength is 12.
%             p1        the wavelet name, as 'wname' used in 'wfilters' command
%             p2        The number of levels, 1-7.
%           220-229  - 2NxN Lapped Orthogonal Transform, LOT from GetLOT
%                       where we use N=[4,6,8,10,12,16,20,24,32,64].
%                       A third argument may be given, it is optional
%             p1        autocorrelation of signal, as 'rxx' used in 'GetLOT.m'
%           230-239  - 4NxN Extended Lapped Transform, ELT from GetELT
%                       where we use N=[4,6,8,10,12,16,20,24,32,64].
%             p1        free variable p, as 'p' used in 'GetELT.m'
%             p2        autocorrelation of signal, as 'rxx' used in 'GetELT.m'
%           255      - Use the frame stored in FrameFile
%             p1        the name of the mat-file for the frame, FrameFile
%             p2        the sparseness factor to use (override Savg in FrameFile)
%  W        the expansion coefficients, this matrix will be of size KxL
%           where K depends on the decomposition method, and L also
%           depends on the number of samples, often we will have Samples=K*L
% --------------------------------------------------------------------------
% This function needs the following functions to be available:
% C_anafb, C_synfb, (which again use C_ana and C_syn), C_anablock, C_synblock
% GetLOT, GetELT, and wfilters in Matlab Wavelet Toolbox.
% For frames: BuildFg, VSblock, GlobalMP  ??
```

Decom2D.m	5866 bytes	02-Dec-2002 20:10:44

```
% Decom2D    Decompose a 2D-signal (image) into expansion coefficients,
%            or the inverse function if TransMet is negative.
% Note: if X is uint8 and Y is uint16, then Y is multiplied by 2 (to keep
% it more accurate, and rounded) and thus energy of Y is approx. 4 times energy
% in X, if exact values are needed use double type.
% Note: Decom2D is "more similar to" Ttimes than to Decom1D
% Examples:
% Y = Decom2D(TransMet, X);      % decomposition (analysis) by TransMet
% X = Decom2D(-TransMet, Y);     % inverse decomposition (reconstruction) by TransMet
% --------------------------------------------------------------------------
% arguments:
%  X        the signal, usually an MxN image, may be uint8 or double
%  Y        the decomposition coefficients, usually also of size MxN
%           may be uint16 or double
%  TransMet An integer for the decomposition method
%           use negative integer for the inverse
%            0    : identity transform, Y=X
%            1    : 2x2 Haar wavelet, LP component is split recursively
%            2    : a dyadic wavelet (given by arg3, name as in wfilters.m)
%                   number of levels is given by arg4,
%                   arg3 and arg4 are used in GetWave.m
%                   works only for orthogonal wavelets and the db79 filters!
%            3- 4 : may be used for other special transforms
%            5- 8 : NxN DCT transform, N=[2,4,8,16]
%            9-12 : 2N*N LOT,          N=[2,4,8,16]
%           13-16 : 4N*N ELT,          N=[2,4,8,16]
% --------------------------------------------------------------------------
% some problems exist for biorthogonal wavelets, TransMet==2
% the function works only for orthogonal wavelets and the db79 filters!
% The problems may be in GetWave
% --------------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
```

```
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  03.08.2000  Karl Skretting, function made
% Ver. 1.1  02.12.2002  KS: moved from ..\Frames2D to ..\FrameTools
% ------------------------------------------------------------------------
```

DesignF.m	23063 bytes	04-Dec-2002 20:19:40

```
% DesignF   Design the frame and store it in FrameFile, based on training data
%           stored in mat-files, DataXnnn.mat, in current directory.
% The function will return after a total of TotIt iterations.
%
% ErrCode=DesignF(FrameFile,TotIt,MaxDays);
%-------------------------------------------------------------------------------
% arguments:
%  ErrCode   - 0 is returned if the function execute without error
%              1 an 'unexpected' error occurred, lasterr may explain it
%              2 or larger, another error occurred,
%  FrameFile - the name of the mat-file used to store the frame
%  TotIt     - total number of iterations to do
%              If more than TotIt iterations already has been done on this frame
%              then no iterations will be done now, the function just return and
%              do not change the frame in FrameFile.
%  MaxDays   - how many days the design process should be allowed to run
%              if MaxDays=0.25 it is 6 hours, 0.01 is 14 minutes and 24 seconds
%-------------------------------------------------------------------------------
%-----------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  11.12.2000  KS: function made
% Ver. 1.1  04.10.2001  KS: changed how vector selection is used
%   also the new function NormalizeF is used now
% Ver. 1.2  05.11.2001  KS: Overlapping frame for 2D added, (Mdim==2) & (P>1)
% Ver. 1.3  06.11.2001  KS: Separable frame for 2D added
% Ver. 1.4  03.12.2002  KS: moved from ..\Frames to ..\FrameTools
%-----------------------------------------------------------------------
```

DesignFb.m	9444 bytes	16-May-2003 18:23:22

```
% DesignFb  This is a stripped version of DesignF, only for block-oriented frame
%           DesignF is the complete program, and works also for block-oriented
% frames. This much more simple version also only use VSblock for vector selection.
% DesignFb is used exactly as DesignF.
%
% ErrCode=DesignFb(FrameFile,TotIt,MaxDays);
%-------------------------------------------------------------------------------
% arguments:
%  ErrCode   - 0 is returned if the function execute without error
%              1 an 'unexpected' error occurred, lasterr may explain it
%              2 or larger, another error occurred,
%  FrameFile - the name of the mat-file used to store the frame
%  TotIt     - total number of iterations to do
%              If more than TotIt iterations already has been done on this frame
%              then no iterations will be done now, the function just return and
%              do not change the frame in FrameFile.
```

```
%  MaxDays   - how many days the design process should be allowed to run
%              if MaxDays=0.25 it is 6 hours, 0.01 is 14 minutes and 24 seconds
%-------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.3  07.11.2001  KS: function made based on DesignF version 1.3
% Ver. 1.4  25.11.2002  KS: moved from ..\Frames to ..\FrameTools
%--------------------------------------------------------------------
```

FrameDesignEx.m	12857 bytes	06-Dec-2002 18:25:50

```
% FrameDesignEx   Here we design an example frame for an AR(1) signal.
%                 The different steps of the design process are briefly
% explained and done. The steps are:
% 1.  Decide for the frame structure to use, i.e. size of F, and
%     target sparseness factor.
% 2.  Prepare the set training vectors, make X.
%     also these vectors are stored in a mat-file used during frame design
%     see DataFile.m, ex: help DataFile
% 3.  Set the initial frame and store it in FrameFile, see: help FrameFile.
%     The initial values for the frame may be generated in many ways,
%     for example one of the files SetF03, SetF05, ..., could be used,
%     see help text for these. SetFnn generates the FrameFile. It may be
%     needed to update som of the variables stored in the mat-file.
%     An alternative is to use GenLloyd to find the initial values of F,
%     then we must remember to store the variables in the mat-file.
% 4.  Design the frame using DesignF or DesignFb
% 5.  We may now look at the designed frame and plot training results.
% The frame is now ready to use for sparse representation, for examples see
% SignalExpansion.m, ex: Y=SignalExpansion(5,1,255,'FrameEx2s20',0.25);
% Some frame properties can be found by for example
% [a,b,c,d,e,f,g,Ang]=FrameProperties('FrameEx1s20',[2,3,7,8,9,27,28,29,31,32,33]);
% F=FrameDesignEx(TestNo,arg1,arg2);
% F=FrameDesignEx(1,0.2);
% F=FrameDesignEx(2,0.2);
%-------------------------------------------------------------------------------
% arguments:
%  F          - the designed frame
%  TestNo     - which test or example to do
%           1 - Design a block-oriented 16x32 frame for an AR(1) signal
%           2 - Continue design from TestNo 1, but now change vector selection method
%           3 - Design a general overlapping frame for an AR(1) signal
%           4 - Design an overlapping frame for an AR(1) signal
%  arg1       - for most tests it is target sparseness factor
%-------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  03.12.2002  KS: function made
%--------------------------------------------------------------------
```

FrameFile.m	5097 bytes	03-Dec-2002 17:20:15

```
% FrameFile   Explain the contents of a mat-file that stores a frame.
%             A frame, as used by the files in this catalog, is a quite
% general concept, and can have a lot of variants. A simple frame does not
% need all these variables, in that case the variables are stored as empty
% arrays, or as variables with value zero.
% For information on a given mat-file, see FrameInfo
% The specification or structure for a frame is stored in a mat-file, FrameFile,
% this file should contain the following variables
%  Class     - a character array containing the name of the signal class
%              the frame is adapted for, ex: 'bwImages','ecg','AR(1)'
%  Type      - a single character indicating the type of the frame, [1]
%              The first tree are all frames where the synthesis equation
%              is linear also with regard to the frame coefficients
%               b: block-oriented frame
%               o: overlapping frame
%               g: general frame
%              The last two are special in the way that the (total) frame is
%              not linear with regard to the frame coefficients.
%               s: separable frame
%               t: tree structured filter bank, i.e. a tree of general filter banks
%                  or general frame, this is not implemented yet
%  Mdim      - number of dimensions in the signal, usually 1 or 2
%  F         - this one is used to store the free variables for the last frame.
%              If Type='b' the size is NxK no matter the dimension.
%              If Type='o' the size of F is NxKxP no matter the dimension.
%              If Type='g' the size is Qx1 no matter the dimension.
%              If Type='s' F is a cell array of Mdim matrices
%              If Type='t' F is a cell array of vectors, size (Qi)x1
%  SizeF     - This should be the 'original' size of F, ex 2D signal: N1xN2xKxP1xP2
%  G         - the "mapping" from F to Fg, only used if Type='g', size is NxKxP
%  Ctab      - a map from WWT (size KxKxIc) to C (size QxQ)
%  Dtab      - a map from XWT (size NxKxP) to D (size Qx1)
%  Fbest     - as F, but the best values during training are stored here
%  Savg      - the average sparseness factor for each signal, 0<Savg<1
%  Mdat      - Number of signals, (images), used during training of the frame
%  PreProc   - a structure array telling how the signal is preprocessed
%              Prog1   - name of program used to subtract LP part of signal
%                        if DC no other program is needed, mean is subtracted
%                        ImDC is another alternative.
%              Prog2   - name of preprocess program, possible programs are:
%                        Decom2D, Decom1D, Ttimes, or another m-file used to
%                        generate the set of training vectors.
%              Method  - Method used by Prog2, integer (not necessarily used)
%              argx    - x=1,2,3 arguments passed to Prog2
%  VecSel    - a structure array telling how vector selection is done during training
%              The structure has 6 fields:
%              Prog1   - name of vector selection program, possible programs are:
%                        BlockVS, GlobalMP, VSblock, VSolap1
%              argx    - x=1,2,3,4,5  arguments passed to Prog1 after X,F,S
%                        see help for each function for these.
%  InitialF  - a structure array telling how the initial values of F was set
%              Prog1   - name of program, Possible programs are:
%                        SetFxx
%              argx    - x=1,2,3,4,5  arguments passed to Prog1
%  History   - a character array telling when and how the frame was made and trained
%  SNRtot    - an array of size 1xItNo or (Mdat+1)xItNo which stores the SNR or PSNR
%              after each time Vector Selection is done. ItNo is
%              number of iterations done
%
% The commands that define an empty frame could be like this:
% FrameFile='test';
% PreProc=struct('Prog1','','Prog2',Mfile,'Method',[],'arg1',0,'arg2',0,'arg3',[]);
```

```
% VecSel=struct('Prog1','VSblock','arg1','VSfomp2','arg2',1,'arg3',[],'arg4',[],'arg5',[]);
% InitialF=struct('Prog1','','arg1',[],'arg2',[],'arg3',[],'arg4',[],'arg5',[]);
% History='';SNRtot=[];Type='b';Mdat=1;F=[];G=[];Dtab=[];Ctab=[];Savg=0.1;
% SizeF=[5,5,50,1,1];Mdim=2;Class='test';Fbest=[];
% save(FrameFile,'Class','Type','Mdim','F','SizeF','G','Ctab','Dtab',...
%      'Fbest','Savg','Mdat','PreProc','VecSel','InitialF','History','SNRtot');
%-------------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  25.12.2002  KS: text moved from ..\Frames\readme.m to ..\FrameTools
%-------------------------------------------------------------------------
```

FrameInfo.m	12992 bytes	27-Nov-2002 14:46:42

```
% FrameInfo Display information about the frame stored in a mat-file,
%          The mat-file is opened and some information is displayed.
% Only files with the 'mat' extension are considered
% [t1,t2]=FrameInfo(MatFile,Display);
%-------------------------------------------------------------------------------
% arguments:
%   t1      - a char array containing one line of information for every
%             mat-file that match MatFile
%   t2      - contains all information about the last mat-file that match MatFile
%   MatFile - the filename, wildchars may be used
%             also the file extension may be omitted, ex MatFile='F*';
%   Display - 1: display information on screen
%             0: or omitted, do not display information on screen
%-------------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  15.12.2000  KS: function made
% Ver. 1.1  27.11.2002  KS: moved from ..\Frames\MFcontent.m to ..\FrameTools
%-------------------------------------------------------------------------
```

FrameProperties.m	21584 bytes	04-Dec-2002 18:50:43

```
% FrameProperties   Find, display and return some properties of a frame F
%                   Many frame properties are only defined for
% block oriented frames. The number of out arguments must correspond
% to the number of entries in 'outargno' (that return output arguments).
% For more information on the different frame properties you should read
% chapter 7 in the thesis "Sparse Signal Representation using Overlapping
% Frames" by Karl Skretting, available at: http://www.ux.his.no/~karlsk/
% outarg=FrameProperties(F,out);    % the general way of calling this function
% outarg=FrameProperties(FrameFile,out);
% A=FrameProperties(F,1);            % the lower frame bound A
% B=FrameProperties(F,2);            % the upper frame bound B
% [A,B]=FrameProperties('FrameEx1s20',[1,2,31,32,33]);  % more examples
% [A,B,betamin,r1max]=FrameProperties('FrameEx1s20',[1,2,7,11,31,32,33]);
% [lambda,r2max,r3max,betaavg,betamse,betaavg2]=FrameProperties('FrameEx1s20',[3,15,19,8,9,27]);
% [a,b,c,d,e,f,g,Ang]=FrameProperties('FrameEx1s20',[2,3,7,8,9,27,28,29,31,32,33]);
```

```
% [a,b]=FrameProperties('FrameEx1s20',[11,15]);
%-------------------------------------------------------------------------------
% arguments:
%  F         - The frame, size is NxKxP
%  FrameFile - the name of the mat-file used to store the frame
%  out       - An array telling which properties to return
%              The first six values are based on the eigenvalues of the frame operator
%                1 - A, lower frame bound
%                2 - B, upper frame bound
%                3 - lambda, eigenvalues of frame operator
%               25 - fA, eigenvector corresponding to smallest eigenvalue
%               26 - fB, eigenvector corresponding to largest eigenvalue
%                    For overlapping frames, P>1, the following properties may be relevant
%                4 - Atheta, lower frame bounds for different values of theta
%                5 - Btheta, upper frame bounds, theta= 0:0.002:0.998  (when length=500)
%                6 - lambdatheta, eigenvalues of frame operator, size is (N x length)
%              The following properties are for angles (in degrees) between frame vectors
%                7 - betamin, the smallest angle between two frame vectors,
%                8 - betaavg, average for all frame vectors to closest neighbor
%                9 - betamse, average of all angles in mean square error sense
%               27 - betaavg2, average of angles between fB and the frame vectors
%               28 - betaavg3, average of angles between f-mean and the frame vectors
%               29 - Ang,  the angle in degrees between two vectors in F.
%              The representation error when selectiong 1, 2 or 3 frame vectors
%              Now we use L=10000 gaussian test vectors and estimate the properties
%              These properties are only returned for block-oriented frames, P=1
%               10 - r1, returns all the L errors r1(l)= || x(l)-F*w(l) ||
%               11 - r1max, estimate for max error
%               12 - r1avg, estimate for mean (average) error
%               13 - r1mse, estimate for square root of the mean square error
%               14,15,16,17  - r2, r2max, r2avg and r2mse
%               18,19,20,21  - r3, r3max, r3avg and r3mse
%              The minimum lower frame bound for all combinations of s vectors
%              These properties are only returned for block-oriented frames, P=1
%               22,23,24 - A2, A3 and A4
%              The following numbers are allowed in the end of outargno, they
%              do not give an output argument but produce a figure
%               31 - Plot frame vectors in figure 1
%               32 - Plot frame angles in figure 2
%               33 - Plot frequency response of frame vectors in figure 3
%  outarg    - the respective output arguments
%-------------------------------------------------------------------------------
%-------------------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  19.03.2002  KS  made function
% Ver. 1.1  10.04.2002  KS  added some of the eigenvectors, fA, fB, and betaav2
% Ver. 1.2  26.04.2002  KS  property 13, 17 and 21 changed from: estimate for the mean square error
% Ver. 1.3  30.04.2002  KS  added betaav3 property
% Ver. 1.4  30.04.2002  KS  swapped the (definitions for) betaav2 and betaav3 properties
% Ver. 1.5  21.06.2002  KS  swapped betaavg2 and betaavg3 properties again
% Ver. 1.6  04.12.2002  KS: moved from ..\Frames\ to ..\FrameTools, and added figures
%-------------------------------------------------------------------------------
```

FramePropertiesObjFun.m	431 bytes	04-Dec-2002 18:13:32

```
% FramePropertiesObjFun  Objective function for finding the x that is
```

```
% most difficult to represent by a sparse approximation using the frame F
% y is 1-r where r is error. This function is used during optmazation in
% FrameProperties.m (fminunc.m function use this object function).
```

GenIntPol.m	4923 bytes	08-Jan-2003 15:48:10

```
% GenIntPol  General Interpolation of Images/Patterns, generate textures
%            Nearest Neighbor or linear interpolation
% Interpolation steps are equal for each direction.
% Interpolation grid may be rotated, and/or translated by an offset.
% If initial image/pattern is too small, it will be periodically extended.
% For images (and matrices) the first axis (x-axis) points downwards and the
% second axis (y-axis) points to the right.
% Used as in mainT01 this function generates different textures.
% Y=GenIntPol(X,x0,y0,dx,dy,alpha,type,M,N);
%--------------------------------------------------------------------------------
% arguments:
%   X     - input image/pattern, this is scaled to fill the box given by
%           its four corners; (0,0), (0,1), (1,1) and (1,0).
%   x0    - offset in x-axis for the first point of the grid in Y
%   y0    - offset in y-axis for the first point of the grid in Y
%   dx    - increment in x-axis for the grid in Y
%   dy    - increment in y-axis for the grid in Y
%   alpha - angle (in radians) the grid of Y is rotated, center of rotation is (x0,y0)
%           negative angle of the grid gives 'positive' rotation of image
%   type  - 0 for nearest neighbor, or 1 for linear interpolation
%   M,N   - Y will be of size MxN (M in x-direction, and N in y direction)
%--------------------------------------------------------------------------------
%--------------------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  10.12.2002  KS: function made
%--------------------------------------------------------------------------------
```

GenLloyd.m	6195 bytes	26-Nov-2002 15:43:49

```
% GenLloyd  Design a codebook (frame) by the Generalized Lloyd Algorithm
%           If initial codebook is not given it will be made by first assuming
% that all vectors of X are in the same class, and the split the class with the
% largest error into two classes until K classes are made.
% IsNormalized is used to indicate if the vectors of X and F are normalized.
% Normalized: norm(f)==1 and sum(f)>0 where f is a column of F
% Then the codebook will be the shapes in a shape-gain quantizer
% F=GenLloyd(X,K,IsNormalized);        % K is a number
% F=GenLloyd(X,F0,IsNormalized);       % F0 is a matrix of size NxK
%--------------------------------------------------------------------------
% Arguments:
%  F            - the codebook vectors, size is NxK
%  X            - the training vectors, a matrix of size NxL
%  K            - number of vectors in codebook
%  F0           - the initial codebook vectors, size is NxK
%  IsNormalized - could be 1 or 0, default is 0.
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
```

```
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  25.12.2002  KS: function made based on ..\Frames\GenLloyd.m
%-------------------------------------------------------------------------
```

GetELT.m	3454 bytes	28-Nov-2002 15:46:30

```
% GetELT    Extendend Lapped (orthogonal) Transform, ELT, of size 4NxN
%           This function returns the synthesis matrix for the extendend lapped
% (orthogonal) transform of size 4NxN, where we should have N even.
% The algorithm is based on the article: Henrique S. Malvar:
% 'Extendend Lapped Transforms: Fast algorithms and applications'
% F=GetELT(N);
% F=GetELT(N,p);
% F=GetELT(N,p,rxx);
% ------------------------------------------------------------------------
% arguments:
%  F        The synthesis matrix for the ELT, size is 4NxN
%  N        The size of F, should be even
%  p        The free parameter in the design, 0<=p<=1, default p=0.7
%  rxx      Optional argument for the signal autocorrelation function
%           this could be of size 4Nx1, where rxx(1) is for zero lag (rxx0)
%           or it could be of size 1x1, then it is assumed to be rho for an AR-1 process
%           if omitted the F matrix is not multiplied by unitary matriz Z.
% ------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  24.07.2000  KS: m-file made
% Ver. 1.1  28.11.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------------
```

GetHaar.m	1627 bytes	18-Jun-2003 12:05:33

```
% GetHaar   Find the NxN Haar matrix
% F = GetHaar(N);
% ------------------------------------------------------------------------------
% Arguments:
%   N    - a positive power of 2, N=2^K  (K=1,2,..12)
%          if N is not a power of 2, the upper left part of a larger
%          Haar matrix is returnd
%   F    - the Haar matrix. (normalized matrix)
% ---------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  30.03.2000  KS: m-file made
% Ver. 1.1  15.12.2000  KS: sort vectors by frequency (zero crossings)
% Ver. 1.2  05.12.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------------
```

GetLOT.m 3589 bytes 28-Nov-2002 15:42:10

```
% GetLOT    Lapped Orthogonal Transform of size 2NxN
%           This function returns the synthesis matrix for the lapped orthogonal
% transform of size 2NxN, where we should have N even. The algorithm is based on
% the article: Malvar, 'The LOT: Transform Coding without Blocking Effects'
% in IEEE Trans on ASSP. vol 37 No 4 April 1989.
% F=GetLOT(N);
% F=GetLOT(N,rxx);
% ------------------------------------------------------------------------
% arguments:
%  F        The synthesis matrix for the lapped orthogonal transform, size is 2NxN
%  N        The size of F, should be even
%  rxx      Optional argument for the signal autocorrelation function
%           this could be of size 2Nx1, where rxx(1) is for zero lag (rxx0)
%           or it could be of size 1x1, then it is assumed to be rho for an AR-1 process
%           if omitted the F matrix is not multiplied by unitary matriz Z.
% ------------------------------------------------------------------------
%------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  21.07.2000  KS: m-file made
% Ver. 1.1  28.11.2002  KS: moved from ..\Frames to ..\FrameTools
%------------------------------------------------------------------------
```

GetWave.m 6057 bytes 03-Dec-2002 13:54:05

```
% GetWave   Get a synthesis transform matrix based on a dyadic wavelet filter bank
%           Note the NormLim parameter given in this file, which may be set to
% trucate the filters, i.e. remove ends that are close to zero.
% This function returns the synthesis filters, which are derived from the
% wavelet (reconstruction) filters found by wfilters, as columns in a matrix F.
% This is like the matrices used for overlapping frames. For orthogonal filter banks
% (orthogonal wavelets) we have F'*F=I. The normalization of the filters in
% the last part of the function cause problems for biorthogonal wavelets, but
% biorthogonal wavelets are in general problematic.
% F=GetWave(wname,Level);
% F=GetWave('db3',1);  % returns the filters in 'db3' as columns in F, 6x2
% F=GetWave('db3',3);  % F is size 40x8, N=8 and P=5.
% F=GetWave('db79s',3);G=GetWave('db79a',3);  % F'*F~=I, but G'*F=I (and F'*G=I)
% ------------------------------------------------------------------------
% arguments:
%  F        The matrix with the synthesis filters, size of F (without truncation)
%           is NPxN, where N=2^Level and P depends on the length of the wavelet
%           filters and the number of Levels in the filter bank.
%  wname    name of the wavelet filter, as used in wfilters.m,
%           the synthesis (or analysis) part is returned.
%           also Daubechies 7-9 biorthogonal filter is possible
%  Level    number of levels to use,
%           use Level=1 to return just the synthesis filters,
% ------------------------------------------------------------------------
% The Daubechies 7-9 biorthogonal wavelet is like in table 7.2 page 119
% in C.S.Burrows et.al "Introduction to Wavelets and Wavelet Transforms"
% It is normalized. Use 'db79a' for analysis and 'db79s' for synthesis
% or only 'db79' (which use the synthesis filters).
```

ImDC.m 3316 bytes 27-Nov-2002 18:53:13

```
% ImDC       Split an image into DC component and residual, or opposite.
%           We use a separable low-pass filter whith length 48 and downsample
% by 16 (in each dimension). (that is the synthesis filter is of length 48
% smooth without ringing, but the analysis filter is 'infinite')
% Thus it is a biorthogonal filter.
% This is an ad-hoc filter, and is not optimal in any way.
% [I,Idc]=ImDC(I);             % split image
% I=ImDC(I,Idc);               % put images together again
% ------------------------------------------------------------------------
% arguments:
%  I        the image, size MxN wher M and N have 16 as factors
%           image should be type double
%  Idc      the DC component image, size (M/16)x(N/16)
% ------------------------------------------------------------------------
% ------------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  20.09.2000 Karl Skretting, function made
% Ver. 1.1  27.11.2002  KS: moved to ..\FrameTools
% ------------------------------------------------------------------------
```

MapCD.m	3639 bytes	04-Dec-2002 19:11:11

```
% MapCD     Makes the tables that map from WWT (or W*W') to C
%           and from XWT (or X*W') to D.
% [Ctab,Dtab]=MapCD(G);
% ---------------------------------------------
% arguments:
%  G         - The structure of the frame, only used if Type='g', size is NxKxP
%              Note that Q is given by max(G(:))
%  Ctab      - a map from WWT (size KxKxIc) to C (size QxQ)
%              Now we assume 1D signal, then Ic=P.
%              The structure is more complex when we have 2D or multi-D signals.
%              Note that we use single indexing into both C and A
%              C(q1,q2)=C(ci)  where ci=(q2-1)*Q+q1 and
%              WWT(k1,k2,p)=WWT(ai)  where ai=((p-1)*K+(k2-1))*K+k1
%              We may then build C from WWT with the following statements:
%                  C=zeros(Q,Q);
%                  for i=1:size(Ctab,1)
%                     ci=Ctab(i,1);ai=Ctab(i,2)
%                     C(ci)=C(ci)+sign(ai)*WWT(abs(ai));
%                  end
%                  C=C+C'-diag(diag(C));   % IMPORTANT!
%  Dtab      - a map from XWT (size NxKxP) to D (size Qx1)
%               We also use single indexing into XWT
%               XWT(n,k,p)=XWT(bi)  where bi=((p-1)*K+(k-1))*N+n
%               We may then build D from XWT with the following statements:
%                  D=zeros(Q,1);
%                  for i=1:size(Dtab,1)
%                     di=Dtab(i,1);bi=Dtab(i,2)
%                     D(di)=D(di)+sign(bi)*XWT(abs(bi));
%                  end
%-------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
```

```
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  15.03.2000  KS  made function
% Ver. 1.1  09.01.2001  KS  function renamed MapCD (from MapACBd)
% Ver. 1.2  04.12.2002  KS: moved from ..\Frames\ to ..\FrameTools
%--------------------------------------------------------------------
```

MCsimF.m	5674 bytes	04-Dec-2002 17:03:45

```
% MCsimF    Monte Carlo simulation for the frame F, selecting S vectors
%           returning the norm of the error for each
% training vector (or each simulation). The data to test is given in the
% X matrix, or X may indicate which kind of random data to use.
% This function is quite similar to VSblock, but does not return the weights.
% r=MCsimF(X,F,S);
% r=MCsimF(X,F,S,VSalg);
%-------------------------------------------------------------------------------
% arguments:
%   X     - the data, a matrix of siz NxL
%           or if X is a 2x1 (or 1x2) vector then X(2) is L and
%           X(1)==1  for random distribution within unit ball (radius==1)
%           X(1)==2  for random distribution within cube, uniform in range [-1,1]
%           X(1)==3  random distribution, each variable is Gaussian.
%                    Note, this is uniform on the unit ball!
%   F     - the normalized dictionary (or F matrix), size NxK
%   S     - number of vectors to select or non-zero weights, 1<=S<=N
%   VSalg - Which vector selection algorithm to use, alternatives are:
%           VSfs (default), VSfomp2
%   r     - norm of error for each try, size 1xL
%-------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2001.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  06.12.2001  KS: function made
% Ver. 1.1  19.12.2001  KS: changed arguments
% Ver. 1.2  04.12.2002  KS: moved from ..\Frames\ to ..\FrameTools
%--------------------------------------------------------------------
```

NormalizeF.m	2153 bytes	03-Dec-2002 17:07:54

```
% NormalizeF   Normalize the synthesis vectors in frame F
% F=NormalizeF(F);
% [F,nf]=NormalizeF(F,G);
%-------------------------------------------------------------------------------
% arguments:
%   F      the frame, a NxKxP matrix or a Qx1 vector
%          as input argument it is "unnormalized" values
%          as output argument it is "normalized" values
%   G      the mapping matrix, size NxKxP if F is a Qx1 vector, else G=[]
%          or G may be omitted
%   nf     the normalization factors, for each of the K synthesis vectors
%          in the frame the frame vector is updated like: F(:,k)=F(:,k)/nf(k);
%-------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2001.  Karl Skretting.  All rights reserved.
```

```
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  04.10.2001  KS: function made
% Ver. 1.1  25.11.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------
```

PDFpoly.m	5344 bytes	18-Dec-2002 17:52:56

```
% PDFpoly   Estimate the pdf (probability density function) by polynomials
%           i.e, a function in the pp-form (piecewise polynomial).
% The input data should be some one-dimensional outcomes (results)
% of the random process for which the pdf is to be estimated.
% The pdf function is returned in piecewise polynomial (pp) form,
% the pp-form and the polynomial approximation is done by the PolyInterPol function.
% The number of line (polynomial) segments is N.
% Note the lengths of bs, deg and cont which are (N+1), N and (N-1) respectively.
% A problem with this function is that the resulting function may be negative,
% this should never happen to a real probability density function.
% pdffun=PDFpoly(data,L,bs,deg);
% pdffun=PDFpoly(data,L,bs,deg,cont,fixval);
% ------------------------------------------------------------------------------
% Arguments:
%   pdffun   - probability density function in pp-form
%   data     - many (real) outcomes of the random process
%              data out of range are not used, range is  bs(1) <= data(i) <= bs(N+1)
%   L        - number of points to use in the range, the pdf is first estimated
%              at these L points evenly distributed in range, this is
%              done based on 'histogram' of data.
%   bs       - break sequence, N+1 points, bs(1) < bs(2) < ... < bs(N+1)
%   deg      - degree of each polynomial, deg is a vector of length N.
%              0 <= deg(n) <= maxdeg, where maxdeg is the maximal degree of
%              the polynomials. (order is degree+1)
%   cont     - continuity properties at the break points, a vector of length (N-1)
%              if 0 - no continuity demands for this break point
%              1 - continuity, 2 also continuious first derivate,
%              and 3 for continuious second derivate.
%              if this variable is omitted no continuity properties is assumed
%   fixval   - a table of points where the function should have special values
%              fixval is of size Kx3 where fixval(k,1) is the x-value,
%              the x-value must be within range of breaks.
%              fixval(k,2) the y-value (or the value of the derivate), and
%              fixval(k,3) is the type or derivate degree, i.e. 0 for value
%              1 for first derivate, 2 for second derivate.
%              if this variable is omitted no fixed values is assumed
% ------------------------------------------------------------------------------
%-------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  17.01.2002  KS, the first version made
% Ver. 1.1  05.02.2002  KS, y(end) divided by 2 in line 107
% Ver. 1.2  18.12.2002  KS: moved from ..\Frames\ to ..\FrameTools
%-------------------------------------------------------------------
```

PlotF.m	8678 bytes	27-Nov-2002 13:50:18

```
% PlotF     Plot the column vectors of the one-dimensional frame F
%          The frame may be overlapping or block-oriented.
% The calling program should decide the figure number to use and clear it
% ex.  figure(1);clf;PlotF(F);
% or   figure(1);clf;subplot(211);PlotF(Fv);subplot(212);PlotF(Fh);
%
% PlotF(F);                    % first argument is a matrix, size NxK or NxKxP
% PlotF(FrameFile);            % or first argument is a character array
% PlotF(BuildFg(F,G));         % if F is a vector, Fg is given by F and G
% PlotF(F,NoText,Scale,UseK,PlotType);   % all possible arguments given
% ------------------------------------------------------------------------------
% Arguments:
%  F         - This is the frame (size is NxKxP, or if P=1 NxK)
%  FrameFile - the name of the mat-file used to store the frame
%  NoText    - If 1 a more clean plot without text is made
%              If 2 a clean plot with some text is made
%              If 0 Some text is written as well
%  Scale     - How much to scale each vector to avoid vectors to plot into
%              each other, 0 (or  omitted) for automatic scaling.
%  UseK      - an array for the k-values to plot, if omitted (or 0) all are plotted
%  PlotType  - different plots are possible (default is 1):
%              1 : each frame vector a vertical line (suitable for few frame vectors)
%              2 : each frame vector as a horizontal line (ok plot with many vectors)
%              3 : time discrete signal plot
% ------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  29.09.2000  KS  made function
% Ver. 1.1  12.12.2000  KS  some changes
% Ver. 1.2  29.03.2001  KS  added the option to give FrameFile
% Ver. 1.3  04.10.2001  KS  a different plot is made if number of vectors to
%                       plot is large (>15, line 98).
% Ver. 1.4  23.11.2001  KS  the second argument is changed from SizeF to NoText
%                       and arg 5 is added as PlotType
% Ver. 1.5  27.11.2002  KS: moved from ..\Frames to ..\FrameTools
%--------------------------------------------------------------------
```

PlotF2D.m	5630 bytes	27-Nov-2002 15:45:19

```
% PlotF2D    Plot the image blocks in frame F, F is size NxK
%            and N is 4, 16, 64 or 256, TransMet must correspond to N
% BlkSize is then given by N (or TransMet), blocks are square.
% Image is made by takeing a sparse W with some few seperated ones, and
% do the reconstruction process on this W. Each block of this image is
% the the result of only one of the vectors in F.
% Before display these blocks may be seperated by "BlkDist" pixels,
% and blocks corresponding to a certain vector in F may be numbered.
% PlotF2D(F,TransMet,BlkDist,FontSize);
% Ir=PlotF2D(F,TransMet,BlkDist,FontSize);
% PlotF2D(eye(64),7,4,8);   % plot the 8x8 DCT basis images
%--------------------------------------------------------------------
% arguments:
%  F         The representation vectors or dictionary, size NxK.
%            if F is scalar and have values 1,2 or 3 we use the initial frames
%            made by SetF01, SetF02 and SetF03 respectivly.
%  TransMet  An integer for the decomposition method that will be done.
```

```
%            Note that this also give the blocksize.
%            This is almost like used in Decom2D
%                1- 4 : NxN identity,    N=[2,4,8,16]
%                5- 8 : NxN DCT,         N=[2,4,8,16]
%                9-12 : 2N*N LOT,        N=[2,4,8,16]
%               13-16 : 4N*N ELT,        N=[2,4,8,16]
%  BlkDist   Distance between each block (in pixels)
%  FontSize  >0 : Display numbers below each block using FontSize
%  Ir        The image that is plotted, (no distance between blocks)
%-------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  18.08.2000  KS: made function
% Ver. 1.1  29.08.2000  KS: changed input (and added output) arguments
% Ver. 1.2  28.10.2001  KS: changed name to PlotF2D, and some minor changes
% Ver. 1.3  27.11.2002  KS: moved from ..\Frames2D to ..\FrameTools
%-------------------------------------------------------------------------
```

PlotFt.m	3404 bytes	27-Nov-2002 15:42:13

```
% PlotFt     Plot the image blocks in frame F, F is size NxK
%            and N is prod(BlkSize)
% PlotFt(F,BlkSize,BlkDist,FontSize);
% Ir=PlotFt(F,BlkSize,BlkDist,FontSize);
%-------------------------------------------------------------------------
% arguments:
%  F         The representation vectors or dictionary, size NxK.
%  BlkSize   The block size to use, BlockSize=[7,7]; for a 7x7 block
%  BlkDist   Distance between each block (in pixels)
%  FontSize  >0 : Display numbers below each block using FontSize
%  Ir        The image that is plotted, (no distance between blocks)
%-------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  17.01.2002  KS: made function, based on PlotF2D
% Ver. 1.1  27.11.2002  KS: moved from ..\Textures to ..\FrameTools
%-------------------------------------------------------------------------
```

PlotSNR.m	3369 bytes	03-Jan-2003 14:55:40

```
% PlotSNR   Plot development in SNR after each iteration during training
% PlotSNR(SNRtot,History,M);
% ------------------------------------------------------------------------------
% Arguments:
%  SNRtot    - an array of size 1xItNo or (Mdat+1)xItNo which stores the SNR or PSNR
%              after each time Vector Selection is done. ItNo is
%              number of iterations done
%  History   - a character array telling when the frame was made and trained
%  M         - which of the signal to plot SNR for, 0 (or omitted) is total for all
% ------------------------------------------------------------------------------
```

```
% --------------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  22.08.2000  KS  made function
% Ver. 1.1  12.12.2000  KS  some changes
% Ver. 1.2  28.11.2002  KS: moved from ..\Frames to ..\FrameTools
% Ver. 1.3  03.01.2003  KS: do not need History
% --------------------------------------------------------------------------------
```

PolyInterPol.m	9005 bytes	18-Dec-2002 17:52:49

```
% PolyInterPol Polynomial interpolation of a function given as points.
%              The function will be approximated by N polynomials, given in
% the pp-form (piecewise polynomial). ex: fnplt(pp); % plot function
% Note the lengths of bs, deg and cont which are (N+1), N and (N-1) respectively.
%
% pp=PolyInterPol(x,y,bs,deg);
% pp=PolyInterPol(x,y,bs,deg,cont,fixval);
%-------------------------------------------------------------------------------
% arguments:
%   x        - the x values of which the function is given, size Lx1
%              x should be in ascending order
%   y        - the function value for the corresponding x, size Lx1
%   bs       - break sequence, the start and end points for the N polynomials.
%              bs is a vector of length (N+1). Values of x should be within
%              range of bs: bs(1) <= x(i) <= bs(N+1)
%              bs should be in ascending order
%   deg      - degree of each polynomial, deg is a vector of length N.
%              0 <= deg(n) <= maxdeg, where maxdeg is the maximal degree of
%              the polynomials. (order is degree+1)
%   cont     - continuity properties at the break points, a vector of length (N-1)
%              if 0 - no continuity demands for this break point
%              1 - continuity, 2 also continuious first derivate,
%              and 3 for continuous second derivate.
%              if this variable is omitted no continuity properties is assumed
%   fixval   - a table of points where the function should have special values
%              fixval is of size Kx3 where fixval(k,1) is the x-value,
%              the x-value must be within range of bs.
%              fixval(k,2) the y-value (or the value of the derivate), and
%              fixval(k,3) is the type or derivate degree, i.e. 0 for value
%              1 for first derivate, 2 for second derivate.
%              if this variable is omitted no fixed values is assumed
%-------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  15.01.2002  KS: function made
% Ver. 1.1  18.12.2002  KS: moved from ..\Frames\ to ..\FrameTools
%-------------------------------------------------------------------------------
```

Reorder.m	4654 bytes	27-Nov-2002 18:59:13

```
% Reorder    Reorder distinct image blocks into columns, or vice versa
%            Y and X will be of the same type (uint8, uint16 or double)
% If Ord>0 first argument is assumed to be the image, and columns are returned
% for this case Reorder is similar (but not identical) to im2col.m
% If Ord<0 first argument is assumed to be the columns, and image is returned
% for this case Reorder is similar (but not identical) to col2im.m
% Examples:
% X=Reorder(Y,ImSize,BlkSize,Ord);
% X=Reorder(Y,ImSize,BlkSize,1);    % take image to columns (im2col)
% Y=Reorder(X,ImSize,BlkSize,-1);   % take columns to image (col2im)
% --------------------------------------------------------------------------
% arguments:
%  Y        the image (or coefficients of the image), size MxN
%  X        the vectors of the blocks, size (Mb*Nb)xL,  where L=(M*N)/(Mb*Nb)
%  ImSize   [M,N] should be the size of the image (coefficients)
%  BlkSize  [Mb,Nb] size of the image blocks
%  Ord      an integer that gives the direction and type
%             +- 1  : columnwise ordering of elements
%             +- 2  : zig-zag ordering of elements (like JPEG)
%             +- 3  : tree-like ordering of elements (like wavelets)
% --------------------------------------------------------------------------
% --------------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  15.08.2000 Karl Skretting, function made
% Ver. 1.1  27.11.2002  KS: moved from ..\Frames to ..\FrameTools
% --------------------------------------------------------------------------
```

SetF03.m	9443 bytes	03-Dec-2002 15:35:18

```
% SetF03    Set F using initial values from DCT/Haar/LOT/ELT/eye/randn
%           The size of F is given by the SizeF variable.
% This file makes SetF01 and SetF02 (for most cases) redundant.
% The FrameFile is created, or overwritten, with the new initial values.
% ErrCode=SetF03(FrameFile,SizeF,Use);
% ErrCode=SetF03(FrameFile,SizeF,bin2dec('01100111')); % LP+BP from DCT, BP from Haar
% ------------------------------------------------------------------------------------
% Arguments:
%  ErrCode   - 0 is returned if the function execute without error
%              1 an 'unexpected' error occurred, lasterr may explain it
%              2 or larger, another error occurred,
%  FrameFile - the name of the mat-file used to store the frame
%  SizeF     - This should be the size of F, ex 1D signal: NxKxP
%              (if P is not given, P=1).
%              We should have N/4 as an integer
%  Use       - an integer specifying which synthesis vectors to use,
%              The N synthesis vectors from each transform are divided into
%              4 groups, from low-pass to high-pass. The bits that are set
%              in Use specify which to use, one bit selects N/4 vectors.
%              Note that the bits are numbered from the right to the left!
%              bit 1-4   DCT vectors (1 is LP group, 2 and 3 BP, and 4 is HP)
%              bit 5-8   Haar vectors,
%              bit 9-12  LOT vectors, only if P>=2
%              bit 13-16 ELT vectors, only if P>=4
%              bit 17-20 eye-matrix vectors or unit-vectors
%              If total is fewer than K then the frame is fillud up with randn vectors
%              If Use is 0, only random synthesis vectors are used
```

```
%  Use       - an example signal (of size NKPx1) which gives the initial vectors
% ------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  19.12.2000  KS: function made
% Ver. 1.1  03.12.2002  KS: moved from ..\Frames to ..\FrameTools
%--------------------------------------------------------------------
```

SetF05.m	7437 bytes	06-Dec-2002 18:25:55

```
% SetF05    Set F as a general filter bank, using initial values from FIR filters
%           The FrameFile will be of Type 'g'
% ErrCode=SetF05(FrameFile,FIRspec);
% ErrCode=SetF05(FrameFile,FIRspec,a);
% ------------------------------------------------------------------------------
% Arguments:
%  ErrCode   - 0 is returned if the function execute without error
%              1 an 'unexpected' error occurred, lasterr may explain it
%              2 or larger, another error occurred,
%  FrameFile - the name of the mat-file used to store the frame
%  FIRspec   - the specifications for the filters, a matrix of size Jx4
%              where J is number of different filters to use.
%              The filters are designed using the fir1 function
%              FIRspec(j,1) is Lj, length (=order+1) of filter j
%              FIRspec(j,2) is nj, the upsampling factor for filter j
%              FIRspec(j,3) is w1, the lower frequency for the passband of
%                 filter j, if w1 is 0 we have a low-pass filter
%              FIRspec(j,4) is w2, the higher frequency for the passband of
%                 filter j, if w1 is 1 we have a high-pass filter
%              If w1=0 and w2=1 the filter coefficients are just filled with
%              random values.
%  a         - indicate the kind of structure imposed on the filter bank
%              this affect the G matrix
%              0  - all non-zero variables are free, but zeros are fixed
%              a  - a is a real number and 0<a<1, then [F,G]=BuildG(Fg,a)
%                   a should be small!
%              1  - all filter coefficients are free, but equal filters are
%                   kept equal.
%              a  - a is a real number and 1<a<2, then BuildG(Fg,a-1) is used
%                   but filters are kept equal, (a-1) should be small!
% ------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2001.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  11.01.2001  KS: function made
% Ver. 1.1  03.12.2002  KS: moved from ..\Frames to ..\FrameTools
%--------------------------------------------------------------------
```

SetF06.m	5362 bytes	03-Dec-2002 19:34:07

```
% SetF06    Set F as a general filter bank, using initial values from a wavelet
```

```
%          tree structured filter bank. The wavelet is made by GetWave.m
% The FrameFile will be of Type 'g'
% Also we will always have K==N (corresponds to critically sampled filter bank)
% ErrCode=SetF06(FrameFile,wname,Level,a);
% ------------------------------------------------------------------------------
% Arguments:
%  ErrCode   - 0 is returned if the function execute without error
%              1 an 'unexpected' error occurred, lasterr may explain it
%              2 or larger, another error occurred,
%  FrameFile - the name of the mat-file used to store the frame
%  wname     - name of the wavelet filter, as used in wfilters.m
%              also 'db79' for Daubechies 7-9 biorthogonal filter is possible
%              Used as argument in GetWave.m
%  Level     - number of levels to use, use Level=1 to return just
%              the synthesis filters
%              Used as argument in GetWave.m
%  a         - indicate the kind of structure imposed on the filter bank
%              this affect the G matrix
%              0  - all non-zero variables are free, but zeros are fixed
%              a  - a is a real number and 0<a<1, then [F,G]=BuildG(Fg,a)
%                   a should be small!
%              1  - all filter coefficients are free, mut equal filters are
%                   kept equal.
%              a  - a is a real number and 1<a<2, then BuildG(Fg,a-1) is used
%                   but filters are kept equal, (a-1) should be small!
% ------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2001.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  10.01.2001  KS: function made
% Ver. 1.1  03.12.2002  KS: moved from ..\Frames to ..\FrameTools
%--------------------------------------------------------------------
```

SetF07.m	5157 bytes	04-Dec-2002 19:41:58

```
% SetF07    Set F as a general filter bank with N=8, K=16, P=6 and Q=208
%           The FrameFile will be of Type 'g'
% Some values here are chosen to fit the hump in an ecg signal.
% ErrCode=SetF07(FrameFile);
% ------------------------------------------------------------------------------
% Arguments:
%  ErrCode   - 0 is returned if the function execute without error
%              1 an 'unexpected' error occurred, lasterr may explain it
%              2 or larger, another error occurred,
%  FrameFile - the name of the mat-file used to store the frame
% ------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2001.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  07.05.2001  KS: function made
% Ver. 1.1  04.12.2002  KS: moved from ..\Frames to ..\FrameTools
%--------------------------------------------------------------------
```

SetFfromImage.m	2530 bytes	25-Nov-2002 17:48:54

```
% SetFfromImage    Set F as a frame of randomly selected blocks from image B.
%                  K blocks from the image are extracted, each block is reordered into
% a column vector and normalized before stored as a column in the frame F.
% Normalized: norm(f)==1 and sum(f)>0 where f is a column of F
% Also a set of training vectors may be extracted using this function.
% [F,History]=SetFfromImage(B,K,BlockSize);
% ---------------------------------------------------------------------------
% Arguments:
%  F         - the frame of vectors, size NxK, N=prod(BlockSize)
%  History   - one or more strings describing how F has been made
%  B         - the image/texture a 2D matrix of numbers
%  K         - Number of frame vectors to make
%  BlockSize - The block size to use, BlockSize=[7,7]; for a 7x7 block
% ---------------------------------------------------------------------------
%---------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  25.12.2002  KS  made function based on ..\Textures\SetFt01.m
%---------------------------------------------------------------------
```

SignalExpansion.m	20025 bytes	03-Dec-2002 19:11:34

```
% SignalExpansion   Here we test many different methods for signal expansion.
%                   A signal expansion is the representation of a signal as
% a linear combination of some basic synthesis signals, which depend on the
% transform, filter bank or frame used.
% The functions tested in this file are:
%   dct function in Matlab (for comparision) in TestNo 1
%   Ttimes function in TestNo 2, 3, and 4
%   Decom1D function in TestNo 5
%   Decom2D function in TestNo 6
% Y=SignalExpansion(TestNo,Signal,N,p1,p2);
%---------------------------------------------------------------------------
% arguments:
%  Y          - the expansion coefficients
%               for 1D signal the size of Y depends on the expansion
%               for 2D signals (and orthogonal expansions) the size of Y is as size of X
%  TestNo     - which test to do
%                1 - use NxN DCT (Matlab functions only)
%                    Y=SignalExpansion(1,1,8);  % 8x8 DCT, 1D AR(1) signal
%                    Y=SignalExpansion(1,2,8);  % 8x8 DCT (dct function), 2D image
%                    Y=SignalExpansion(1,2,8,'dct2');  % 8x8 DCT (dct2 function), 2D image
%                2 - use NxN KLT (Karhunen Loewe Transform) and Ttimes function (case a)
%                    Y=SignalExpansion(2,1,8);  % 8x8 KLT, 1D AR(1) signal
%                    Y=SignalExpansion(2,2,8);  % 8x8 KLT, two KLTs (rows and columns)
%                    Y=SignalExpansion(2,2,8,1);  % 8x8 KLT, the same KLT for rows and columns
%                3 - use 4NxN ELT (Extended Lapped Transform) and Ttimes function (case b)
%                    Y=SignalExpansion(3,1,8);  % 32x8 ELT, 1D AR(1) signal
%                    Y=SignalExpansion(3,2,8);  % 32x8 ELT, image
%                4 - use a defined method in Ttimes function (case c), note that argument N
%                    is not used and argument p1 gives the method used in Ttimes
%                    Y=SignalExpansion(4,1,0,1);  % 2x2 Haar wavelet
%                    Y=SignalExpansion(4,1,0,7);  % 8x8 DCT
%                    Y=SignalExpansion(4,2,0,12);  % 32x16 LOT
%                5 - use the methods in Decom1D, only for 1D signal,
%                    note the third argument, N, now is Method used in Decom1D,
%                    Y=SignalExpansion(5,1,8);  % Decom1D, 8x8 DCT
```

```
%                   Y=SignalExpansion(5,1,203);  % Decom1D, IIR filter bank
%                   Y=SignalExpansion(5,1,213);  % Decom1D, Daubechies 7-9 wavelet
%                   Y=SignalExpansion(5,1,218,'db3',3);  % Decom1D, filter from wfilters
%                   Y=SignalExpansion(5,1,235);  % Decom1D, 64x16 ELT
%                   Y=SignalExpansion(5,1,255,'FrameEx2s20',0.25);  % Decom1D, frame
%               6 - use the methods in Decom2D, only for 2D signal, Decom2D is quite similar to
%                   Ttimes and it uses the Ttimes function to do the work.
%                   Y=SignalExpansion(6,2,8);   % Decom2D, 16x16 DCT
%                   Y=SignalExpansion(6,2,12);  % Decom2D, 32x16 LOT
%                   Y=SignalExpansion(6,2,2,'db4',3);   % Decom2D, a wavelet
%                   Y=SignalExpansion(6,2,2,'db79',4);   % Decom2D, a wavelet
%  Signal     -  1 - a simple one-dimensional signal, AR(1)
%                2 - a two-dimensional signal, the image Lena (or Barbara)
%                X - the signal X, 1D or 2D
%  N          - the transform (filter bank) size parameter N, default is 8
%  p1         - an extra argument used by some tests, default 0
%-------------------------------------------------------------------------------
%-----------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  28.11.2002  KS: function made
%-----------------------------------------------------------------------
```

TestGenLloyd.m	1784 bytes	18-Dec-2002 15:19:26

```
% TestGenLloyd  Test of the function GenLloyd
%-----------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  26.11.2002  KS: function made
%-----------------------------------------------------------------------
```

TestPDFpoly.m	4929 bytes	18-Dec-2002 17:52:43

```
% TestPDFpoly  Test of the function PDFpoly and PolyInterPol
%              Most functions can be approximated by piecewise polynomials
% Here a 1D function y=f(x) is approximated by polynomials
% the function is given in different ways in PDFpoly and PolyInterPol
% see help text for piecewise polynomials in Matlab, ex: help mkpp,
% and PDFpoly and PolyInterPol
%-----------------------------------------------------------------------
% Copyright (c) 2002.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  18.12.2002  KS: function made
%-----------------------------------------------------------------------
```

TestVS.m	5578 bytes	19-Sep-2003 14:39:04

```
% TestVS  test of Vector Selection algorithms
%--------------------------------------------------------------------
% Copyright (c) 2002-2003.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  19.09.2003  KS: m-file made
%--------------------------------------------------------------------
```

TestVSblockC.m	4313 bytes	22-Sep-2003 13:55:44

```
% TestVSblockC     Test of VSblockC, vector selection done by program
%                  written in C an compiled to VSblockC.exe
% This m-file gives an example on how to use the fast vector selection
% program. We also use VSblock for comparison.
%--------------------------------------------------------------------
% Copyright (c) 2003.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  15.05.2003  KS: function made
% Ver. 1.1  22.09.2003  KS: some changes
%--------------------------------------------------------------------
```

Ttimes.m	5772 bytes	02-Dec-2002 20:12:58

```
% Ttimes    Do the operation Y=T*X, where T represent a filterbank or a transform.
%           For examples (and tests) of this function, see SignalExpansion.m
% using tests 2, 3 and 4.
% Y=Ttimes(T,X);
% [Y,T]=Ttimes(T,X,a3);
% ------------------------------------------------------------------------
% arguments:
%  X     A NMx1 column vector or a NMxL matrix.
%        If X is a matrix, the operation is applied to each column.
%  Y     the result, the size of Y will be KMxL (when K==N, same as X)
%  T   a If T is of size KxN, this is the transform y=Tx, where x and y
%        are parts of X and Y, with size Nx1 and Kx1
%      b If T is of size KxNxP, with P>1, T represent a filterbank which is
%        applied on each column of X to give a column of Y.
%        Delay: Note that if you give T as a NxNxP matrix (and K=N), and then use the
%        inverse of T (if it exist, U) again of size NxNxP (like LOT and ELT)
%        the resulting signal will be delayed N*(P-1) samples.
%        That is: Y=Ttimes(T,X); Xr=Ttimes(U,Y); then XR(n)==X(n-N*(P-1))
%        This delay will automatically be compensated if
%        a3 is given as 1 (true or non-zero). If P==1 delay is no problem.
%      c If T is a single integer (size 1x1), we select transform or filterbank
%        using the following table. A Negative number is used for the inverse (synthesis).
%          1    : 2x2 Haar wavelet, LP component is split until it has less than
%                 32 elements. (32 elements in LP is split again, 31 is not)
%                 this transform also handle any length in columns of X.
%                 Use like: Y=Ttimes(1,X);Xr=Ttimes(-1,Y,0.5);
%                 it is an integer transform, X is integer => Y is integer
%          2- 4 : may be used for other special transforms
%          5- 8 : NxN DCT transform, N=K=[2,4,8,16], P=1
%          9-12 : 2N*N LOT,          N=K=[2,4,8,16], P=2
```

```
%         13-16 : 4N*N ELT,          N=K=[2,4,8,16], P=4
%  a3    a third argument may be given. It may be used to indicate
%        - compensation for delay, when T is given as a matrix
%        - multiply result by a3, only for abs(T)==1, 2*2 Haar wavelet
%          (may be used as 1/sqrt(2) to keep transform unitary)
% --------------------------------------------------------------------------
% --------------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:
% Ver. 1.0  11.08.2000 Karl Skretting, function made
% Ver. 1.1  27.11.2002  KS: moved from ..\Frames to ..\FrameTools
% --------------------------------------------------------------------------
```

VSab2.m	6322 bytes	19-Sep-2003 14:18:50

```
% VSab2     Vector Selection algorithm that Always returns a Better (or the same) w
% This was made to ensure that the weights selected are at least as good as the
% previous selecte weights.
% Essentially this use the same algorithm as VSmp2
% w=VSab2(x,w);
% w=VSab2(x,S);
%-------------------------------------------------------------------
% arguments:
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%          note that w is both input and output argument
%   x    - the vector to approximate, size Nx1
%   S    - if second argument is not the weight vector
%          it is number of vectors to select or non-zero weights
%          (do not mix with S in FindW function, which has a different meaning)
%          S (which is scalar) may be used instead of w as second input argument,
%          then w is found using VSmp2 or VSfomp2
%-------------------------------------------------------------------------------
% global variables:
%   F    - the dictionary (or F matrix), size NxK
%          the vectors of F must be normalized, F(:,k)'*F(:,k)==1 for all k
%          normalization may be done by: F=F*diag(1./sqrt(diag(F'*F)));
%   FF   - the inner products F'*F, size KxK
%-------------------------------------------------------------------------------
% Note that we here use a simple complete F matrix, and do not make any assumtions
% regarding structure of F.
```

VSblock.m	11518 bytes	22-Sep-2003 13:35:51

```
% VSblock   Vector Selection for block-oriented frame,
%           several (=B) blocks may be done at each step (each call to VSalg)
% At each step we select vectors for B signal blocks, the actual frame used is
% a block-diagonal matrix formed by repeating input frame B times.
% The program works for all block-oriented frames, the signal may be 1D, 2D or
% multi-dimensional, but it must be organized as a matrix of size NxL.
% For overlapping frames (1D signal) use the function VSolap1 or BlockVS
% W=VSblock(X,F,S,VSalg,B);
% W=VSblock(X,F,S,VSalg,B,el,Bf);
%-------------------------------------------------------------------------------
% arguments:
%  W     - The weight matrix, W is a sparse matrix of size KxL
```

```
%  X     - The signal reshaped into blocks of length N, size NxL
%  F     - The frame of synthesis vectors (dictionary), size NxKx1  (P==1)
%          the vectors of F must be normalized, F(:,k)'*F(:,k)==1 for all k
%  S     - the third input argument may have different meaning depending on its size.
%    1x1   S is scalar, then it is assumed to be sparsness factor, 0<S<1,
%          a total of S*N*L weights will be selected, evenly distributed or by GMP.
%    1xL   number of vectors to select for each block of length N
%          S(l) is used for vector X(:,l) to find W(:,l)
%          Note: only when B==1, if B>1 the distribution of weights may be changed
%    KxL   and the third argument should be the previous weights, W.
%          Now S=full(sum(W~=0)); and used as previous case (size 1xL).
%          If VSalg='VSab2' previous weights are used as input when VSalg is called
%  VSalg - Which vector selection algorithm to use, it must be a function
%          called like 'w=VSfomp2(x,S);' and have F and FF as as global variables.
%          Recommended (=tested) are 'VSfomp2', 'VSmp2' or 'VSab2'.
%  B     - number of blocks to use at each call to VSalg, the actual frame will
%          be a block-diagonal matrix of size NBxKB, where the input frame (size NxK)
%          is used in the blocks on the diagonal.
%  el    - extra loops to do, vector selection will be done (1+el) times
%          for each block,   default is el=0
%  Bf    - The increment of l when going from one block to the next
%          We should have (1 <= Bf <= B), and Bf a factor of L, default is Bf=B
%-------------------------------------------------------------------------------
%-------------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  04.10.2001  KS: made function based on BlockVS (based on FindW)
% Ver. 1.1  26.10.2001  KS: some minor changes, el and Bf arguments added
% Ver. 1.2  25.11.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------------------
```

VSblockC.m	4399 bytes	22-Sep-2003 12:03:58

```
% VSblockC  Vector Selection for block-oriented frame, (special version)
%           which use a compiled C++ program to do the actual work.
% The interface of this variant is much simpler than the VSblock function in FrameTools
% with fewer possibilities to specify how vector selection is to be done.
% If this function does not work as needed, or if it gives any errors,
% you should use VSblock in FrameTools or for overlapping frames the
% function VSolap1.
% W=VSblockC(X,F,S,VSalg);
% W=VSblockC(X,F,S,'VSps',M);       % algorithm may be gived as a string
%-------------------------------------------------------------------------------
% arguments:
%  W     - The weight matrix, W is a sparse matrix of size KxL
%  X     - The signal reshaped into blocks of length N, size NxL
%  F     - The frame of synthesis vectors (dictionary), size NxKx1  (P==1)
%          the vectors of F must be normalized, F(:,k)'*F(:,k)==1 for all k
%  S     - the third input argument is here DIFFERENT from in the FrameTools function
%          S is number of vectors to select for each coulmn vector in X
%  M     - number of combinations to search in VSps
%  VSalg - A number (or string) indicating which vector selection algorithm to use in VSblock.exe
%          1 - Basic Matching Pursuit
%          2 - Matching Pursuit (main loop is done more times)
%          3 - Matching Pursuit (main loop is done even more times)
%          4 - Orthogonal Matching Pursuit
%          5 - Order Recursive Matching Pursuit
```

```
%          6 - Partial Search
%          7 - Always Better.  The previous weights must be written
%          to file filW before this function is called.
%-------------------------------------------------------------------------------
%-------------------------------------------------------------------------------
% Copyright (c) 2000.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  15.05.2003  KS: VSblockC.m based on ..\FrameTools\VSblock
% Ver. 1.1  06.06.2003  KS: also VSps works!
% Ver. 1.2  11.06.2003  KS: added VSab
%-------------------------------------------------------------------------------
```

VSfomp2.m	3814 bytes	14-May-2003 14:21:08

```
% VSfomp2   Fast Orthogonal Matching Pursuit 2, Vector Selection algorithm
%           as implemented by A. Barkhald 1999, (Matlab optimized version of VSfomp)
% Compared to VSfomp the differences are
%   VSfomp2 always select two or more vectors
%   VSfomp2 does not return when error is small, r'*r < Arr
%   VSfomp2 depends on that the vectors in F are normalized!
%
% w=VSfomp2(x,S);
%-------------------------------------------------------------------------------
% arguments:
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%   x    - the vector to approximate, size Nx1
%   S    - number of vectors to select or non-zero weights
%-------------------------------------------------------------------------------
% global variables:
%   F    - the normalized dictionary (or F matrix), size NxK
%          (this matrix is called U in some papers describing OMP/ORMP/FOMP)
%   FF   - the inner products F'*F, size KxK
%-------------------------------------------------------------------------------
%-------------------------------------------------------------------
% Copyright (c) 1999.  Andre' Barkhald.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing
% Mail:  barkhald@online.no
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  15.02.1999  AB: function made as part of diploma 99,
% Ver. 1.1  04.11.1999  KS: function adjusted to a "stand alone" version
% Ver. 2.0  23.03.2000  KS: changed the arguments used
% Ver. 2.1  07.04.2000  KS: added ConditionLimit, and maybe call VSmp2
% Ver. 2.2  11.12.2000  KS: use some global variables
% Ver. 2.3  25.11.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------------------
```

VSfs.m	4205 bytes	26-Jun-2003 16:24:49

```
% VSfs      Vector Selection algorithm doing Full Search
% w=VSfs(F,x,S);
%-------------------------------------------------------------------------------
% arguments:
%   F    - the normalized dictionary (or F matrix), size NxK
%   x    - the vector to approximate, size Nx1
```

```
%   S    - number of vectors to select or non-zero weights, 1<=S<=N
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%-------------------------------------------------------------------------------
% Note that F is not global in this function as in many other VSxxx functions
```

VSmp2.m	2461 bytes	19-Sep-2003 14:06:16

```
% VSmp2     Matching Pursuit variant 2, Vector Selection algorithm
% This is the same algorithm that sometimes is called Basic Matching Pursuit (BMP).
% VSmp2 (this function) do 'it' iterations selecting new weights or improving
% already selected weights while
% w=VSmp2(x,S,it);
% w=VSmp2(x,S);            % default value of 'it' used, it=2*S
%-------------------------------------------------------------------
% arguments:
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%   x    - the vector to approximate, size Nx1
%   S    - number of vectors to select or non-zero weights
%   it   - number of iterations to do
%-------------------------------------------------------------------------------
% global variables:
%   F    - the dictionary (or F matrix), size NxK
%          the vectors of F must be normalized, F(:,k)'*F(:,k)==1 for all k
%          normalization may be done by: F=F*diag(1./sqrt(diag(F'*F)));
%-------------------------------------------------------------------------------
% Note that we here use a simple complete F matrix, and do not make any assumtions
% regarding structure of F.
```

VSolap1.m	11638 bytes	03-Dec-2002 19:54:25

```
% VSolap1   Vector Selection for overlapping frame and 1D signal
%           a larger frame is build by repeating the input frame several
% times, i.e actual frame is input frame repeated B times.
% Then, for each corresponding signal block
% a number of vectors are selected using a block-oriented vector selection
% algorithm, VSxxx, given by VSalg (char array).
% The program works for overlapping frames and 1D dignal.
% W=VSolap1(X,F,S,VSalg,B);
% W=VSolap1(X,F,W,VSalg,B,el);
%-------------------------------------------------------------------------------
% arguments:
%  W     - The weight matrix, W is a sparse matrix of size KxL
%  X     - The signal reshaped into blocks of length N, size NxL
%  F     - The frame of synthesis vectors (dictionary), size NxKxP  (P>1)
%          the vectors of F must be normalized, [F,nf]=NormalizeF(F);
%  S     - the third input argument may have different meaning depending on its size.
%    1x1   S is scalar, then it is assumed to be sparsness factor, 0<S<1,
%          a total of S*N*L weights will be selected, evenly distributed.
%    1xL   number of vectors to select for each block of length N
%          S(l) is number of non-zero weights in W(:,l)
%          Note: only when B==1, if B>1 the distribution of weights may be changed
%    KxL   and the third argument should be the previous weights, W.
%          Now S=full(sum(W~=0)); and used as previous case (size 1xL).
%          If VSalg='VSab2' previous weights are used as input when VSalg is called
%  VSalg - Which vector selection algorithm to use, it must be a function
%          called like 'w=VSfomp2(x,S);' and have F and FF as as global variables.
%          Recommended (=tested) are 'VSfomp2', 'VSmp2' or 'VSab2'.
%  B     - number of blocks to use at each call to VSalg, the actual frame will
```

```
%          be a block-diagonal matrix of size N(B+P-1)xKB, where the input frame,
%          reshaped into size NPxK, is used in the blocks on the diagonal.
%          We should have B larger than or equal to (P-1) to keep overlapping
%          within the adjacent (large/expanded/actual) frame.
%  el    - extra loops to do, vector selection will be done (1+el) times
%          for each extended block,   default is el=0
%-------------------------------------------------------------------------------
% Note: Arguments for this function is almost like for VSblock
%-------------------------------------------------------------------------------
% Copyright (c) 2001.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  04.10.2001  KS: made function based on BlockVS (based on FindW)
% Ver. 1.1  26.10.2001  KS: some minor changes
% Ver. 1.2  03.12.2002  KS: moved from ..\Frames to ..\FrameTools
%-------------------------------------------------------------------------------
% The main difference from BlockVS is that we do not have extra overlap (Be)
% instead we do vector selection twice for every block, first time when the
% neighbor weights are zero, second time when they are set.
% This is not done when previous weights are given as input argument.
% Also the blocks may be different each time the function is called
```

VSolap2.m	12797 bytes	04-Dec-2002 19:04:30

```
% VSolap2    Vector Selection for overlapping frame and 2D signal
%            a larger frame is build by repeating the input frame several
% times, i.e actual frame is input frame repeated B times.
% Then, for each corresponding signal block
% a number of vectors are selected using a block-oriented vector selection
% algorithm, VSxxx, given by VSalg (char array).
% The program works for overlapping frames and 2D signal.
% Remarks on size of variables:
% In this version N,K,P and B are given (by the size of) as input arguments
% All these numbers must be square numbers since
% the program use N1=N2=sqrt(N) P1=P2=sqrt(P) and B1=B2=sqrt(B)
% In a later version we could have X 4D, Fin 5D and [B1, B2] instead of B and
% [N1,N2,L1,L2]=size(X), [N1,N2,K,P1,P2]=size(Fin) to give N=N1*N2, P=P1*P2, B=B1*B2
% W=VSolap2(X,F,S,VSalg,B);
% W=VSolap2(X,F,W,VSalg,B,el);
%-------------------------------------------------------------------------------
% arguments:
%  W     - The weight matrix, W is a sparse matrix of size KxL = Kx(L1*L2)
%  X     - The signal reshaped into blocks of length N, size NxL = (N1*N2)x(L1*L2)
%  F     - The frame of synthesis vectors (dictionary),
%          size NxKxP = (N1*N2)xKx(P1*P2) (P>1, and P a square number)
%          the vectors of F must be normalized, [F,nf]=NormalizeF(F);
%  S     - the third input argument may have different meaning depending on its size.
%    1x1   S is scalar, then it is assumed to be sparsness factor, 0<S<1,
%          a total of S*N*L weights will be selected by a GMP method.
%    1xL   number of vectors to select for each block of length N
%          S(l) is number of non-zero weights in W(:,l)
%          Note: only when B==1, if B>1 the distribution of weights may be changed
%    KxL   and the third argument should be the previous weights, W.
%          Now S=full(sum(W~=0)); and used as previous case (size 1xL).
%          If VSalg='VSab2' previous weights are used as input when VSalg is called
%  VSalg - Which vector selection algorithm to use, it must be a function
%          called like 'w=VSfomp2(x,S);' and have F and FF as as global variables.
%          Recommended (=tested) are 'VSfomp2', 'VSmp2' or 'VSab2'.
```

```
%  B     - number of blocks to use at each call to VSalg, the actual frame will
%          be a block-diagonal matrix of size N(B+P-1)xKB, where the input frame,
%          reshaped into size NPxK, is used in the blocks on the diagonal.
%          We should have B larger than or equal to (P-1) to keep overlapping
%          within the adjacent (large/expanded/actual) frame.
%  el    - extra loops to do, vector selection will be done (1+el) times
%          for each extended block,   default is el=0
%-------------------------------------------------------------------------------
% Note: Arguments for this function is almost like for VSblock, and VSolap1
%-------------------------------------------------------------------------------
% Copyright (c) 2001.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  04.10.2001  KS: made function based on BlockVS (based on FindW)
% Ver. 1.1  26.10.2001  KS: some minor changes
% Ver. 1.2  26.10.2001  KS: VSolap2 made based on VSolap1
%           05.11.2001  KS: VSolap2 seems to work (including the GMP part)
% Ver. 1.3  05.12.2002  KS: moved from ..\Frames2D\ to ..\FrameTools
%-------------------------------------------------------------------------------
```

VSomp.m 2810 bytes 19-Sep-2003 09:53:58

```
% VSomp     Orthogonal Matching Pursuit with QR decomposition, Vector Selection algorithm
%
% w=VSomp(x,S);
%-------------------------------------------------------------------------------
% arguments:
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%   x    - the vector to approximate, size Nx1
%   S    - number of vectors to select or non-zero weights
%-------------------------------------------------------------------------------
% global variables:
%   F    - the normalized dictionary (or F matrix), size NxK
%          (this matrix is called U in some papers describing OMP/ORMP/FOMP)
%   FF   - the inner products F'*F, size KxK
%-------------------------------------------------------------------------------
%-------------------------------------------------------------------------
% Copyright (c) 2003.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  12.06.2003  KS: function made
%-------------------------------------------------------------------------------
```

VSompJHH.m 2996 bytes 19-Sep-2003 10:43:44

```
% VSompJHH  Orthogonal Matching Pursuit, Vector Selection algorithm
% as John Hkon Husy presented in lectures at HiS 1999.
% This algorithm orthogonalize each selected vector to the previous selected vectors.
% w=VSompJHH(x,S);
% w=VSompJHH(x,S,Arr);
%-------------------------------------------------------------------------------
% arguments:
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%   x    - the vector to approximate, size Nx1
```

```
%   S    - number of vectors to select or non-zero weights
%          (do not mix with S in FindW function, which has a different meaning)
%   Arr  - acceptable energy of residual, (r=x-Fw; rr=r'*r;), if residual has lower
%          energy than Arr then function returns even if fewer than S vectors
%          are selected, default is x'*x*1e-6
%---------------------------------------------------------------------------------
% global variables:
%   F    - the normalized dictionary (or F matrix), size NxK
%          (this matrix is called U in some papers describing OMP)
%---------------------------------------------------------------------------------
%---------------------------------------------------------------------------------
% Copyright (c) 1999.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing Group
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  05.11.1999  KS: function made
% Ver. 2.0  20.03.2000  KS: changed the arguments used
% Ver. 2.1  18.12.2000  KS: use some global variables
%---------------------------------------------------------------------------------
```

VSormp.m	2814 bytes	19-Sep-2003 09:57:28

```
% VSormp    Order Recursive Matching Pursuit with QR decomposition, Vector Selection algorithm
%
% w=VSormp(x,S);
%---------------------------------------------------------------------------------
% arguments:
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%   x    - the vector to approximate, size Nx1
%   S    - number of vectors to select or non-zero weights
%---------------------------------------------------------------------------------
% global variables:
%   F    - the normalized dictionary (or F matrix), size NxK
%          (this matrix is called U in some papers describing OMP/ORMP/FOMP)
%   FF   - the inner products F'*F, size KxK
%---------------------------------------------------------------------------------
%---------------------------------------------------------------------
% Copyright (c) 2003.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  12.06.2003  KS: function made
%---------------------------------------------------------------------------------
```

VSormp2.m	2916 bytes	19-Sep-2003 10:31:26

```
% VSormp2   Order Recursive Matching Pursuit with QR decomposition, Vector Selection algorithm
%           This function is more "Matlab-oriented" that VSormp and VSomp,
% here we use fewer loops and more of the Matlab capabilities, but even so I suppose
% that the VSormp and VSomp functions are the better ones (at least with the newer versions
% of Matlab that execute loops quite fast.) Also this function does not ned
% the global variable FF, the inner products of the frame vectors.
%
% w=VSormp2(x,S);
%---------------------------------------------------------------------------------
% arguments:
```

```
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%   x    - the vector to approximate, size Nx1
%   S    - number of vectors to select or non-zero weights
%--------------------------------------------------------------------------------
% global variables:
%   F    - the normalized dictionary (or F matrix), size NxK
%          (this matrix is called U in some papers describing OMP/ORMP/FOMP)
%--------------------------------------------------------------------------------
%--------------------------------------------------------------------
% Copyright (c) 2003.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  19.06.2003  KS: function made
%--------------------------------------------------------------------------------
```

VSps.m	4812 bytes	06-Jun-2003 16:37:37

```
% VSps      Vector Selection algorithm doing Partial Search
% w=VSps(F,x,S,P);
%--------------------------------------------------------------------------------
% arguments:
%   F    - the normalized dictionary (or F matrix), size NxK
%   x    - the vector to approximate, size Nx1
%   S    - number of vectors to select or non-zero weights, 1<=S<=N
%   P    - how many branches to consider at each level, size 1xS
%          total number of combinations examined is prod(P)
%          If P=K:(-1):(K+1-S); full search is done, VSfs will usually work faster
%          since combinations that only differ in order is only done once.
%          If P=ones(1,S); this function should be equal to ORMP (VSfomp2)
%          When last vector is found all is investigated, so P(S)=K-S+1;
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%--------------------------------------------------------------------------------
% Note that F is not global in this function as in many other VSxxx functions
```

VSps2.m	10216 bytes	19-Sep-2003 12:00:09

```
% VSps2     Partial Search 2, Vector Selection algorithm
%
% w=VSps2(x,S,NoCombinations);
% w=VSps2(x,S,NoCombinations,g1,g2,g3);
%--------------------------------------------------------------------------------
% arguments:
%   w    - the weights where at most S of the elements are non-zero, size Kx1
%   x    - the vector to approximate, size Nx1
%   S    - number of vectors to select or non-zero weights
%   NoCombinations    - number of combinations to search
%        Some special parameters for testing (default values are usually ok)
%   g1   - only d-values above this limit (relative to max(d)) is considered
%   g2   - and this will be given g2*(max number of comb. on this level, for max(d))
%   g3   - what is added before the floor function
%--------------------------------------------------------------------------------
% global variables:
%   F    - the normalized dictionary (or F matrix), size NxK
%          (this matrix is called U in some papers describing OMP/ORMP/FOMP)
%   FF   - the inner products F'*F, size KxK
%--------------------------------------------------------------------------------
```

```
%-------------------------------------------------------------------
% Copyright (c) 2003.  Karl Skretting.  All rights reserved.
% Hogskolen in Stavanger (Stavanger University), Signal Processing
% Mail:  karl.skretting@tn.his.no   Homepage:  http://www.ux.his.no/~karlsk/
%
% HISTORY:  dd.mm.yyyy
% Ver. 1.0  12.06.2003  KS: function made
% Ver. 1.1  23.06.2003  KS: some improvements
%-------------------------------------------------------------------------------
```