The ASSURANCE project is about developing new state-of-the-art algorithms for sparsity-aware distributed models, mainly for Machine Learning in fMRI data decomposition/analysis and in telecommunications.

fMRI-Sparse toolbox is a minimalistic collection of low-level data handling (matrix) functions for fMRI processing, block-based & event-based test pattern series, as well as "realistic" simulated fMRI data series for algorithm benchmarking, template scripts for various fMRI decomposition methods (GLM, PCA, ICA, BP, CCA, KSVD), analysis of components & activation maps, etc. Since the toolbox can be used as a benchmarking suite, several data generators are included for creating fully-identifiable fMRI-like data series.

Currently, the toolbox implements the low-level functionality for fMRI data structures and uses some standard implementations for commonly used algorithms for analysis/decomposition and blind source separation, including principal component analysis, multivariate linear regression, basis pursuit and independent component analysis (see below: *"Decomposition / analysis algorithms"*). It also includes some extended wrapper functions for fMRI-like data series generation, based on existing similar code packages (e.g. from the MLSP lab , see below: *"fMRI data series generation"*), that can be used in benchmarking experiments. The plan is to provide frequent version updates of the toolbox, extending it with new implementations, performance enhancements and novel algorithms for fMRI data processing with special focus in sparse data modeling.

The main implementation platform is Matlab , currently using toolboxes from versions 8.0+ (releases R2012b and above), e.g. the Basis Pursuit algorithms from Wavelet Toolbox. However, with some small conversions in the source codes, the main functions should be fully runnable with other Matlab-compatible platforms like Octave (GNU) . Also, there are some external codes and toolboxes required to run some methods (e.g. fastICA), which too can be substituted by other similar packages if necessary - see below for details.

Some features are not yet fully implemented. All code is under constant development and in beta version, so it should not be considered a 'stable' release.

What does the fMRI-Sparse toolbox include?

Here is a list of the main features currently included in the toolbox:

fMRI data series manipulation:

- vxVectorize / vxDevectorize : A fMRI data series typically contains a 3-D grid of several 2-D 'slices' of brain fMRI 'images', registered along a 4th dimension of time (temporal evolution). However, most decomposition and analysis algorithms require standard 2-D matrices. These two functions convert a 3-D (slice-based) or 4-D (grid-based) fMRI data series into a standard 'flattened' matrix and vice versa. These are compatible with Matlab's standard 'reshape' function, but they keep the row-wise axis for time and column-wise axis for voxels (while 'reshape' works differently).

fMRI voxel dependencies:

- vxBOLDtwogammaHRF : A standard, fully-parametric two-Gamma implementation of the Haemodynamic Response Function (HRF) for BOLD registration in fMRI. It is typically used as a convolution kernel when generating the components of a design matrix in various decomposition methods, e.g. GLM or OMP. The implementation and default parameter values are based on the model proposed in: Jagath C. Rajapakse, Juan Zhou, *Learning effective brain connectivity with dynamic Bayesian networks*, NeuroImage 37 (2007) 749-760.

- vxARMA : When generating artificial fMRI data series, various temporal and/or spatial dependencies between voxels need to be included. This function can be used as a post-processing stage for 1-D, 2-D or 3-D spatial convolution kernels, as well as in the 4th (time) dimension for temporal convolution kernels. It also includes options for adding noise elements in the final data.

Test-pattern data series generators:

- vxBoxPatternCreate / vxBoxSeriesCreate : One of the most common cases of benchmarking data for fMRI decomposition algorithms is the use of 'box' patterns, with possible

spatial and/or temporal overlapping. These two functions automate the generation procedures of such test-pattern data series.

- vxSeriesBlockCreate / vxSeriesEventCreate : Create block-based and event-based fMRI activation patterns to be used in combination with any test patterns for the generation of corresponding data series.

fMRI post-processing / various:

- vxCalcActivMap : After analyzing a fMRI data series into components, the 'sources' are correlated to known (task-related) activation patterns in order to identify 'activated' voxels in brain areas. This function is currently based on correlation in the temporal domain, since this is much faster, more robust and more generic than any other similar model-based approach, e.g. performing statistical significance tests upon the 'beta' coefficients of a GLM model. However, the corresponding activation maps (based on p-values) right now are much more inaccurate, since this feature is not yet fully implemented here. Instead, the correlation itself, either on the temporal ('fixed' versus 'discovered' ones) or the spatial domain (voxel runs with/without specific 'sources' included) should be used for the illustration of 'active' brain areas.

{rokbox thumb=|../images/fig4.png| thumbsize=|300 227| title=|vxCalcActivMap : Correlation map between full PCA-reconstruction and task-related source (T1).| }../images/fig4.png{/rokbox}
{rokbox thumb=|../images/fig5.png| thumbsize=|300 225| title=|vxCalcActivMap : Typical activation map (p-values) between full PCA-reconstruction and task-related source (T1).| }../images/fig5.png{/rokbox}

- signal_rescale : General-purpose function for rescaling a data series - e.g. used by 'vxBOLDtwogammaHRF
'

- signal_white : General-purpose function for 'whitening' a data series, i.e., normalizing it for mean=0 and std=1.
- calcSparsity : Given a coefficient matrix (after decomposition), this function calculates the number of non-zero coefficients per 'signal'. On sparse coding stages (e.g. using BP), these numbers are the given sparsity constraint (T0).

fMRI data series generation:

- vxCreateSimData : Create a typical block-based test-pattern data series with (optionally) spatial and temporal overlapping between two 'box' activations. This can be modified to create any other type and shape of activation.
- vxCreateRealData : Create a "realistic" fMRI-like data series (real-valued) for use with any type of decomposition algorithm, for a single 'slice' and a total of 8 independent 'sources', one of

which is the main task-related component to be identified. The generator optionally displays the complete run, illustrating the full-mixture and (only) the task-related activation map. The final output is the full-mixture run, i.e., a 2-D image evolving through time, as well as the components (sources) and mixture matrices. The implementation is a rescaling-enabled modular version of a similar script by the MLSP lab - for further details see: N. Correa, Y.-O. Li, T. Adali, and V. Calhoun, *Compariso n of blind source separation algorithms for fMRI using a new Matlab toolbox: GIFT* , in Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP), Philadelphia, PA, vol. 5, pp. 401-404, March 2005.

{rokbox thumb=|../images/fig1.png| thumbsize=|600 257| title=|vxCreateRealData : Task-related activation component and full-mixture fMRI-like image.| }../images/fig1.png{/rokbox}

- vxCreateComplexData : Very similar to 'vxCreateRealData' but with complex-valued elements that include randomized phase shifts. These data series are useful when the decomposition process is linked to blind-source separation and/or low-level compressive sensing, e.g. during the acquisition stage of fMRI (K-space compression). The implementation is a rescaling-enabled modular version of a similar script by the MLSP lab - for further details see: N. Correa, Y.-O. Li, T. Adali, and V. Calhoun, *Comparison of blind source separation algorithms for fMRI using a new Matlab toolbox: GIFT* , in Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP), Philadelphia, PA, vol. 5, pp. 401-404, March 2005.

Decomposition / analysis algorithms

The fMRI-Sparse toolbox currently includes implementation/usage of four main decomposition algorithms for the analysis of fMRI data series:

Fixed-components methods:

- General Linear Model (GLM) : A standard regression-based method for analyzing fMRI data series, employing multivariate linear regression against a pre-defined design matrix that usually contains HRF-convolutions of task-related and transient task-related activation patterns. In fMRI-Sparse toolbox, this method is implemented by using Matlab's mvregress function from the Statistics toolbox

.

- Canonical Correlation Analysis (CCA) : A standard correlation-based method for analyzing fMRI data series by calculating a pair of matrices A and B that maximize the correlation between the fMRI data series and a pre-defined de sign matrix that usually contains HRF-convolutions of task-related and transient task-related activation patterns. In fMRI-Sparse toolbox, this method is implemented by using Matlab's canoncorr function from the Statistics toolbox .

- Matching Pursuit (MP) : One of the most popular algorithms for sparse representation and compressive sensing via redundant dictionaries. In fMRI-Sparse toolbox, this method uses the Orthogonal Matching Pursuit (OMP) variant as it is implemented by Matlab's wmpalg function from the Wavelet toolbox . Since the current version includes only template scripts, instead of an over-complete dictionary, a typical GLM design matrix is used here as well - this means that there is no real "selection" of appropriate components from the dictionary, only an optimal ordering and weighting of them. Furthermore, due to sparsity constraints, the final reconstruction quality in this case is deteriorated (more components are normally used). For real benchmarking tests, the wmpdictionary function can be used to create a proper redundant dictionary, containing an adequate number e.g. of wavelet components.

Nonfixed-components (blind) methods:

- Principal Compoments Analysis (PCA) : The standard SVD-based method for the decomposition of a data series into a (reduced) eigenspace. Instead of providing a pre-defined design matrix, here the activation patterns are "discovered" as well; however, due to its inherent sensitivity to noise, the corresponding spatial components are usually poisoned by "mirror" artifacts (complementary dependencies). In fMRI-Sparse toolbox, this method is implemented by using Matlab's pca function from the Statisti

cs toolbox
.

- Independent Components Analysis (ICA) : Instead of just uncorrelated components, ICA analyzes the data series into *independent* components using higher-order statistical constraints. The two most popular approaches of ICA is FastICA and Infomax. If noise levels are low, the "discovered" components closely match the true "sources" that produce the signal, e.g. the corresponding internal "sources" of the mixture that is produced by
vxCreateRealData
(see above). In fMRI-Sparse toolbox, this method is employed by using the
FastICA toolbox for Matlab
(see below for details) and can be substituted by any other similar implementation.

- Sparse Dictionary Learning-Decomposition (KSVD) : KSVD is one of the most popular and well-studied methods for data (matrix) decomposition under sparsity constraints, using a 'trained' dictionary of coding atoms. The method is a generalization of K-means clustering, essentially employing singular value decomposition (svd) techniques to iteratively pursuit optimal atoms formulation into the dictionary, while at the same time adjusting the corresponding regression coefficients. Thus, it is a 'blind' decomposition method, similar to PCA or ICA, but at the same time similar to classic signal transformations (e.g., DFT, DWT, DCT, ...) but with "learned" data-driven base functions. The KSVD analysis can work by applying constraints either on the final approximation error or on the required sparsity level on the coefficients. If dictionary size and sparsity level are chosen appropriately, the "learned" dictionary atoms closely match the true "sources" that produce the signal, e.g. the corresponding internal "sources" of the mixture that is produced by
vxCreateRealData
(see above). In fMRI-Sparse toolbox, this method is employed by using the
KSVD toolbox for Matlab
(see below for details) and can be substituted by any other similar implementation

Main scripts - Test-runs (demos)

Currently, there are two main scripts that can be used as code templates for similar procedures, i.e., how can fMRI be generated, analyzed and illustrated in various ways:

Test-pattern fMRI data series decomposition scripts:

1. run1_createblockseries : This is a code template for a typical block-based test-pattern series generation, analysis and presentation. The script does not actually include any real or simulated fMRI data series - this is used only for illustration purposes (data handling &

processing) and to verify the proper use of decomposition algorithms, e.g. PCA. It uses the ' vxCreateSimData
' function to generate a full fMRI-like block-pattern series, perform various decomposition procedures (PCA, ICA, GLM, BP), as well as HRF convolution on activation patterns and ARMA-plus-noise post-processing on the input data, and display full runs of generated and decomposed/reconstructed data series.

2. run2_createRealSeries : This is a code template for a typical block-based simulated fMRI data series generation, analysis and presentation.This script uses "realistic" real-valued fMRI data including a single 'slice', i.e., a 2-D image evolving through time - this can be used for illustration purposes (data handling & processing), as well as to verify the proper use of decomposition algorithms, e.g. PCA or KSVD. It can also be used as a template for comparative testing of varius decomposition methods, with regard to the accuracy of identified components, reconstruction error, etc. It uses the ' vxCreateRealData' function to generate a full "realistic" one-slice fMRI data series, perform various decomposition procedures (PCA, ICA, GLM, BP, CCA, KSVD) and display full runs of generated and decomposed/reconstructed data series.

{rokbox thumb=|../images/fig3.png| thumbsize=|350 257| title=|run2_createRealSeries : PCA decomposition and activation components of a realistic fMRI-like data series. The (r) value is the correlation (strength) of each component against the true task-related temporal activation. Since 8 sources were used to generate the data series, components 9 and 10 are dummy.| }../images/fig3.png{/rokbox}

{rokbox thumb=|../images/fig2.png| thumbsize=|300 171| title=|run2_createRealSeries : PCA reconstruction error along time axis for realistic fMRI-like data series.| }../images/fig2.png{/rokbox}

{rokbox thumb=|../images/fig6.png| thumbsize=|300 208| title=|run2_createRealSeries : Plot of the two largest PCA components 4 and 3 (red, green) identified, against the true task-related component (blue).| }../images/fig6.png{/rokbox}

Downloads / Resources

The fMRI-Sparse toolbox is freely available under the Creative Commons Attribution Share-Alike international license 3.0 (CC-BY-SA)
, with no guarantee and no support (at least for now). See source code for individual options and version history details.

Please feel free to send feedback, suggestions and bug reports. However, keep in mind that this is still a work-in-progress, so incomplete features, errors and instabilities are to be expected.

- **fMRI-Sparse toolbox for Matlab** / version 1.11 / release 25-Sept-2013 : **Download** (22

KB)
(MD5: 7c84efb50d667b75a64976127e08a44f)

- FastICA toolbox for Matlab : Developed and published by the ICS team in the Helsinki University of Technology
, Finland. It is used as external package by the fMRI-Sparse toolbox, but any similar ICA implementation of FastICA or Infomax can be used instead after minor modifications in the main (template) scripts. An Octave implementation is also provided, see their main page for details & downloads.

- KSVD toolbox for Matlab : A joint work developed and published by Michael Elad, Michal Aharon and Freddy Bruckstein. It is used as external package by the fMRI-Sparse toolbox, but any similar KSVD implementation can be used instead after minor modifications in the main (template) scripts. The specific tollbox is adopted specifically due to it simple implementation and maximum compatibility with Octave (no .mex compiled files are required, pure Matlab implementation, may be slow with large datasets). See their main page for details & more downloads.
- Kittipat fMRI preprocessing toolbox for Matlab : Currently not required by the current version of fMRI-Sparse toolbox, but highly recommended, as it provides a very lightweight and compact alternative (e.g. compared to SP M8
) for handling
fMRI data files in NIFTI format
. Instead of simulated data series, a real fMRI data file can be imported and used as input in the main (template) scripts, either for a single 'slice' (2-D/3-D) or a full brain scan (3-D/4-D).

- Openfmri.org : One of the most comprehensive and well-documented collection of fMRI datasets, freely available for download, in NIFTI format .