LibrarY of Complex ICA Algorithms (LYCIA) Toolbox v1.0.2

Walk-through

Josselin Dea, Sai Ma, and Tülay Adalı
Department of CSEE, University of Maryland, Baltimore County, MD 212150
Updated: November 2, 2011

Introduction

The toolbox LibrarY of Complex ICA Algorithms (LYCIA) includes many of the complex ICA and source separation algorithms that are publicly available, and allows the user to compare their performances using a number of metrics and visualization tools. In this walk-through, we explain the basic functionality of LYCIA and its use.

A - Installing LYCIA

Unzip the file 'LYCIA1.0.zip' and copy the folder 'LYCIA' onto your local machine. Add LYCIA directories to the MATLAB search path and type 'LYCIA' in MATLAB command window or double click 'LYCIA.fig' file to open LYCIA, as shown below.

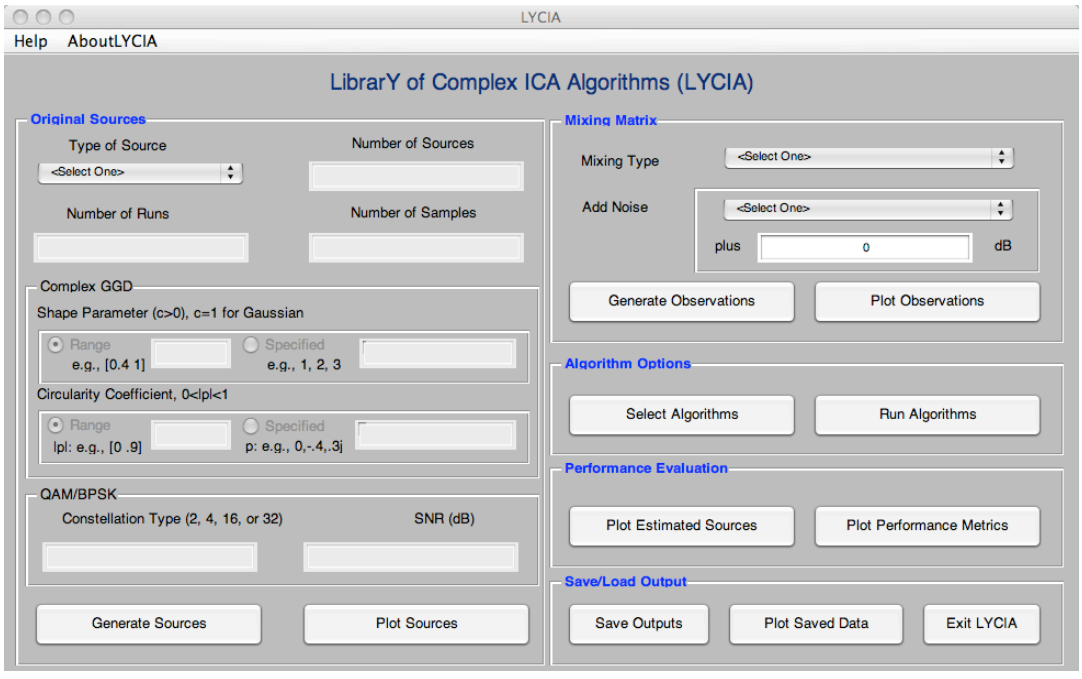


Figure 1. The main interface of LYCIA toolbox

B - Steps involved in LYCIA

A complete execution of LYCIA toolbox consists of three dependent steps that are 1)

source generation, 2) source mixing, and 3) source separation with complex ICA algorithms. An independent step is available for visualizing the result of each of the three steps. This option is treated as a fourth step in this document. After running any of these steps in the pre-cited order, users can save the output data, as discussed in Appendix A.

B.1 – Generating complex sources

Several models of complex sources are available in LYCIA. These are complex generalized Gaussian distribution (GGD), QAM/BPSK, complex Gaussian mixture (MoG) and general complex model. Before clicking the ***Generate Sources*** button in figure 1, users should set subsequent parameters as follows:

a) Choose a model of source in the ***type of source*** menu,

b) Set the number of sources, the number of runs, and the number of samples. This step is common for all models of source; however, it might be different from one model to another. For example, if users choose ***General complex Model*** in step a), the number of sources will automatically depend on specific parameters of this model,

c) Set model specific parameters.

The differences mentioned in b) have been included in the details of model specific parameter setting.

>>>> Specific parameters for GGD sources

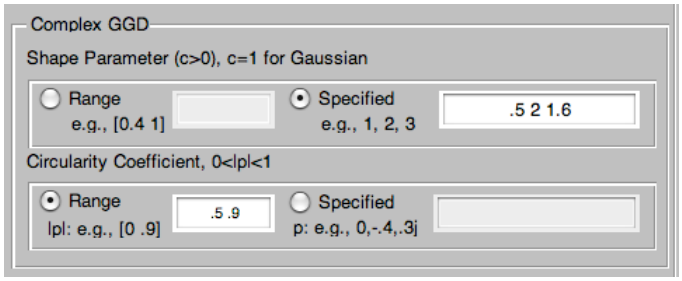


Figure 2. Complex GGD parameter panel

There are two parameters for GGD sources, the distribution shape parameter ***c*** and its circularity coefficient p. These parameters can be entered either as range of values in the field ***Range***, or as fixed values in the field ***Specified.*** If a range is specified, the shape parameter (and/or circularity parameter) will be generated randomly. If the second option is selected, users should enter a set of values with a total number of values equal to the number of sources. Figure 2 shows GGD parameters setting for 3 sources with specified values for ***c*** and a range for $|p|$.

>>>> Specific parameters for QAM/BPSK sources

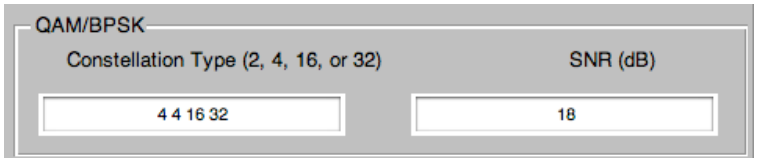


Figure 3. QAM/BPSK parameter panel

In addition to general parameters such as the number of sources, QAM/BPSK source generation required two more parameters. These parameters are the constellation type of the sources, and its signal to noise ratio (SNR). If users generate N sources, N integer values from the set {2, 4, 16, 32} are required for the constellation type.

>>>> Specific parameters for General Complex Model sources

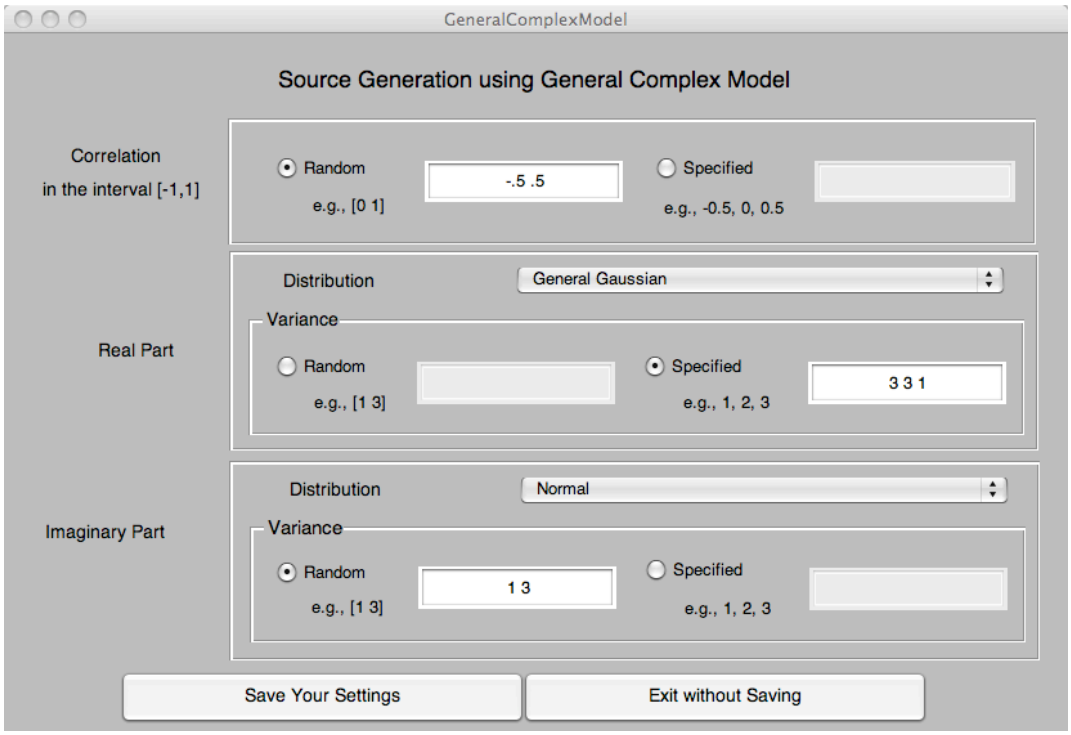


Figure 4. General complex Model parameter setting window

When users select ***General complex model,*** a new window pops up for specific source parameter setting. The setting of these parameters is similar to the one explained for GGD sources – correlation and variance are entered either as a range of values, or as specified values. The correlation value between the real part and the imaginary part is in the interval [-1,1]. Users should also select a distribution for both real part and imaginary part. If more than one parameter among ***correlation, variance of real part and variance of imaginary part*** is set by specifying fixed values, then these

parameter must have the same number of values. In this case the number of sources will be automatically set equal to the number of specified values. An example is shown in figure 4 where a) the correlation will be generated randomly in the interval [-0.5,0.5], b) the distribution of the real part is set to ***General Gaussian,*** c) the variances for the real part of three sources are set to three fixed values. In this case the number of source is automatically set to 3.

>>>> Complex Gaussian Mixture

Besides those general parameters, generating Complex Gaussian Mixture sources does not require any specific parameter.

>>>> Load file

Files loaded by users do not require specific parameters. However, the source data should be saved in a Matlab '.mat' file and the data must be in a variable named ***Sources.*** This variable should also be a $1 \times N$ cell array; each element of the array represents a data matrix from different runs. In addition all the matrices must have the same dimension. Figure 5 shows an example containing three sources for each of five runs.

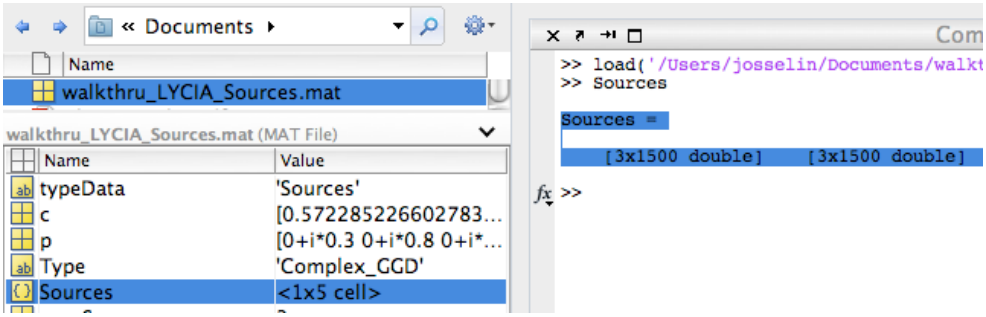


Figure 5. Example of user data file – Sources variable (left panel) and two elements (right panel)

B.2 – Mixing sources

After generating successfully the desired complex sources, the second step is to form the observations. In order to generate mixtures of the independent sources, the user should 1) select type of mixing matrix in the list ***Mixing type***, 2) select a noise power in the list ***Add Noise*** and/or enter the desired noise power in the field ***plus***. When the mixing parameter setting is completed, clicking the ***Generate Observations*** button will create the mixture. Figure 6.1. shows the mixture generation panel that must be set before generating observations.

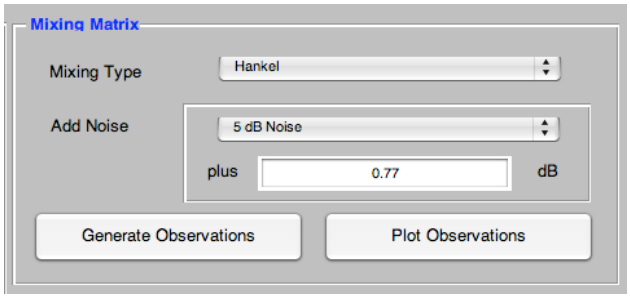


Figure 6.1. Panel for generating observations

>>>> Load file

Users can load their own mixing matrix by selecting ***Load Files*** in the list ***Mixing Type***. The mixing matrix should be saved in a Matlab '.mat' file and must be in a variable named ***A***. This variable should be a 1×*N* cell array where *N* is the number of runs; each element of the array represents a mixing matrix for one run. Figure 6.2. shows an example containing a 4×4 mixing matrix for each of two runs.

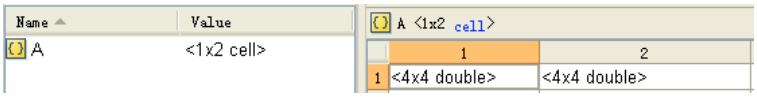


Figure 6.2. Example of user mixing matrix file

B.3 – Source separation with complex ICA algorithms

Once the sources are generated and mixed to form the observations, the users can perform source separation using a set of algorithms available in LYCIA, or test their own algorithms. Source separation involves two steps: a) select ICA algorithms and b) run selected algorithms. Figure 7 shows the corresponding panel.

Figure 7. ICA Algorithm Panel

>>>> *Selecting ICA Algorithms*

When users click the button ***Select Algorithms***, a new window opens with several choices of algorithms (Figure 8.1). Additional choices are available by clicking the ***More Algorithms*** button to display the second panel as in Figure 8.2.

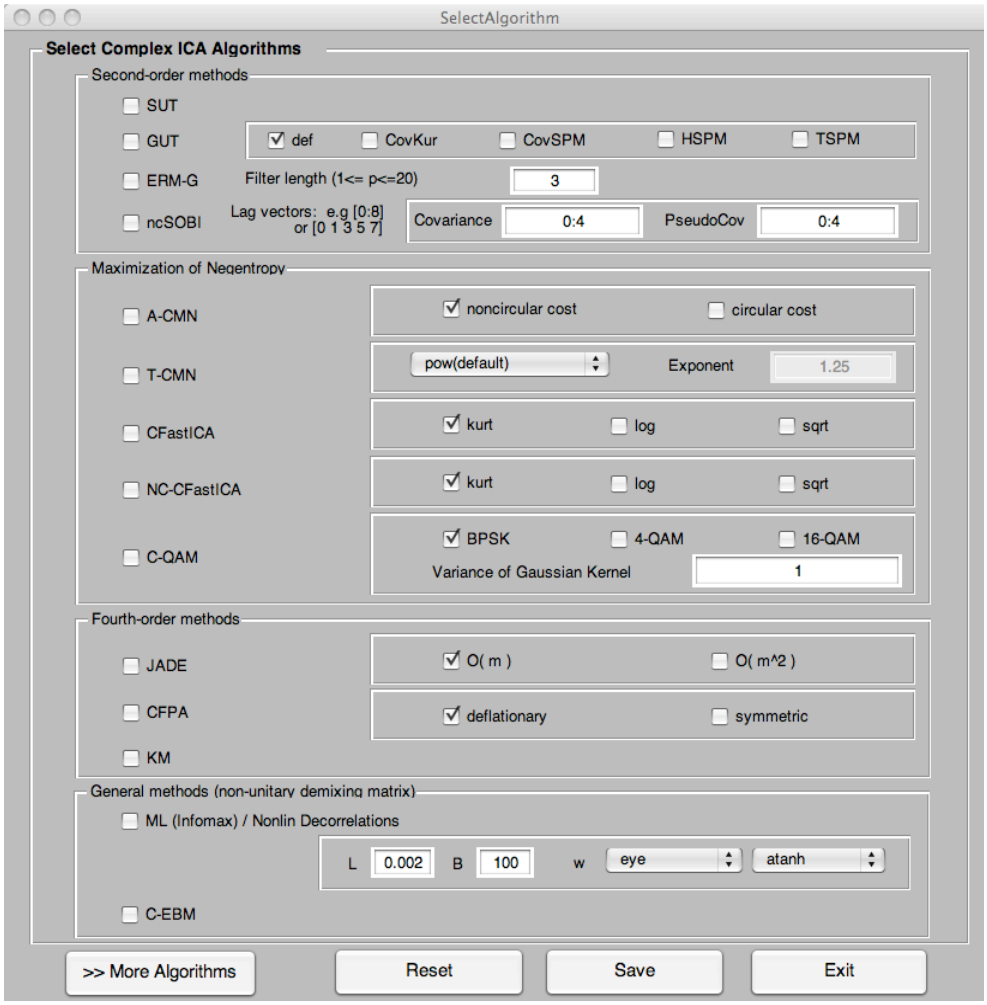


Figure 8.1. Algorithm selection – main panel

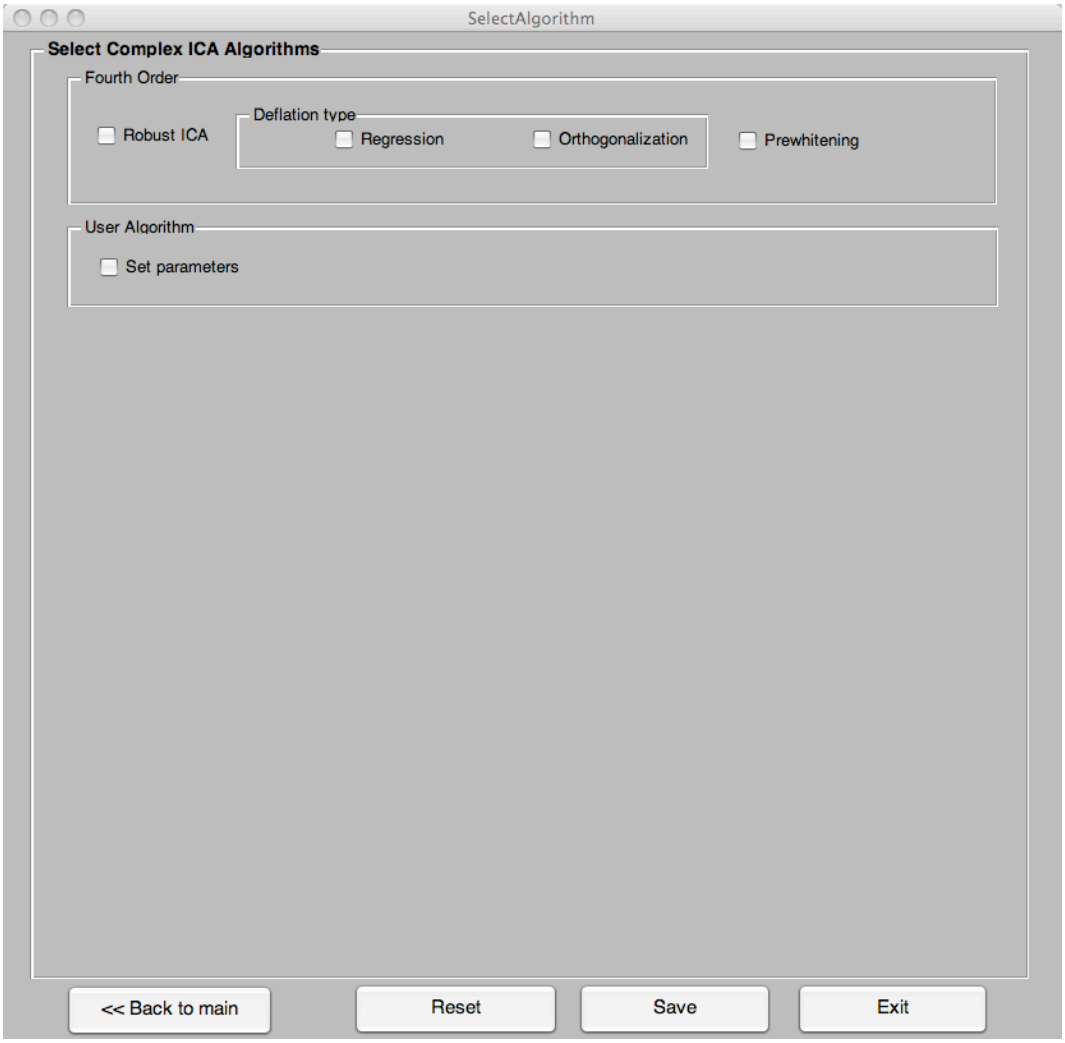


Figure 8.2. Algorithm selection – second panel

Users can find detailed information about these complex ICA algorithms in the references listed at the end of this document and also in the "Algorithms" link on the LYCIA webpage (http://mlsp.umbc.edu/lycia/lycia.html).

In addition, users can add and test their own algorithms.

>>>>> Adding user's own algorithm

Users can run and evaluate the performance of their own algorithms by clicking the ***Set parameters*** option button in figure 8.2, and setting the parameters required for this algorithm. Figure 9 shows an example of adding user algorithm. Note that only one algorithm can be added at a time. In this example, the algorithm ***jadem2.m*** located in the folder specified in ***Function Path*** with the input arguments ***Number of sources*** (provided by LYCIA) is added and one user's own argument set to 2.

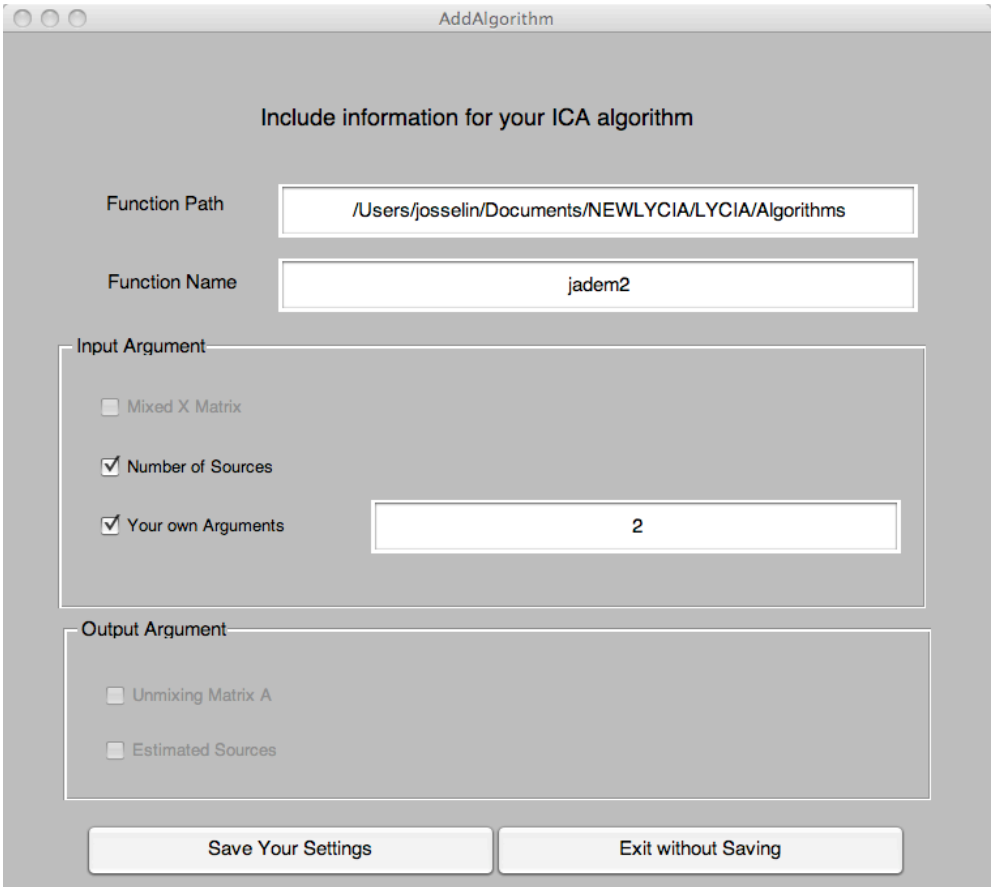


Figure 9. User's own algorithm selection parameter window

In general, user ICA algorithm will have the signature,

[**W**, **S**] = userAlgorithmname (**X**, *N*, userArg1, userArg2,…, userArgn)

where,

- **W** and **S** are the outputs of the algorithm, representing the estimate of the pseudo-inverse of the mixing matrix and the estimates of the independent components, respectively.
- **X** is the mixture matrix to be separated, and *N* is the number of sources

Then the parameters will be set in the window of figure 9, as follows:

Number of sources: check the box,
Function Name: userAlgorithmname,
Your own Argument: userArg1, userArg2, …, userArgn

Note: The mixture **X** is provided by LYCIA and has been generated in previous steps (see B.2 mixing sources). If the order of parameter in the signature above does not match your algorithm, please make subsequent change to your algorithm.

>>>> Running selected algorithm

After selecting the algorithm and setting the required parameters, clicking the ***Run selected Algorithms*** button as shown in figure 9 will initiate the source separation process. An information window will show to notify of the progression of the source separation.

B.4 – Visualization

As noted previously, the visualization option of LYCIA is available for each of the three major steps described in the previous sections; users do not need to complete all three steps before visualizing the results. For example, once the sources are generated, they can be visualized.

>>>> Visualizing Sources and observations

To plot the sources, or the observations, click the appropriate button – either ***Plot Sources,*** or ***Plot Observations*** to open the viewer**.** The windows that open in both cases are quite similar, containing the same commands and information. The difference between them is that the source viewer gives information related to sources, while the observation viewer deals with observations. Figure 10-11 show these windows. The objects in the viewing windows are explained in Table 1.

Objects	Descriptions
Type of plot	LYCIA offers five different presentations of the data. For example, the user can display a ***scatter plot*** of the sources.
Run	Indicates the run for which the user wants to plot the data
Left, Right Index	It is possible to display simultaneously two sources or observations. Then the right (resp. left) index refers to the index of the source that is plotted in the right (resp. left) side axis. (see figure 10)
Load data	This button is used to load and visualize previously saved data.
Statistics	Above each plot, the statistics of the corresponding source are displayed – ***mean, standard deviation, kurtosis, noncircularity***

Table 1. Objects of the visualization window

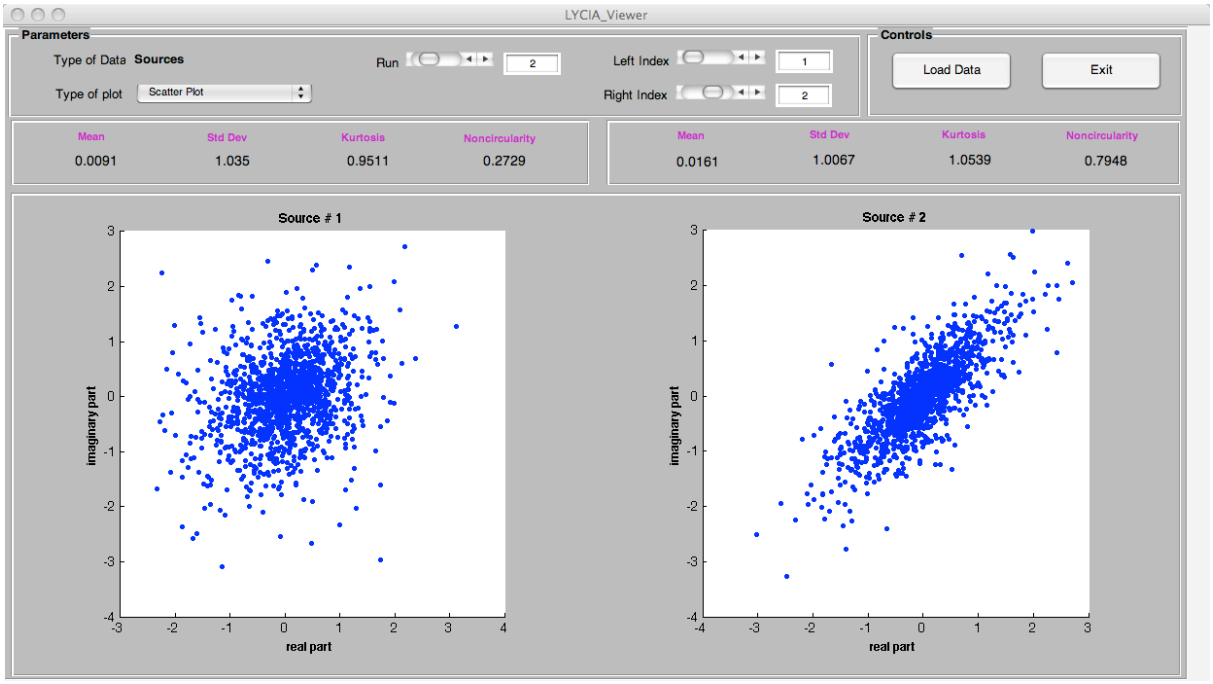


Figure 10. Source visualizing panel – 2 sources (Right index = 2)

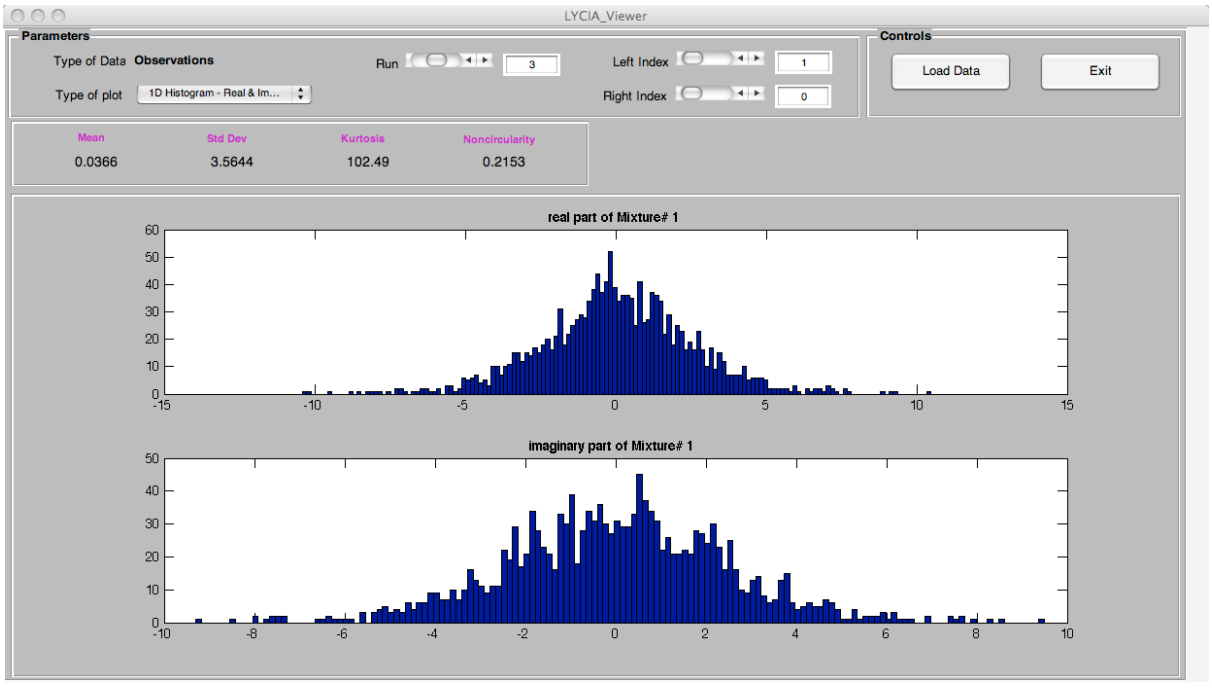


Figure 11. Observation visualizing window – single observation (Right index = 0)

>>>> Different visualization options

Sources and observations can be visualized with five different plotting methods. Figures 12-16 show screenshots of these methods.

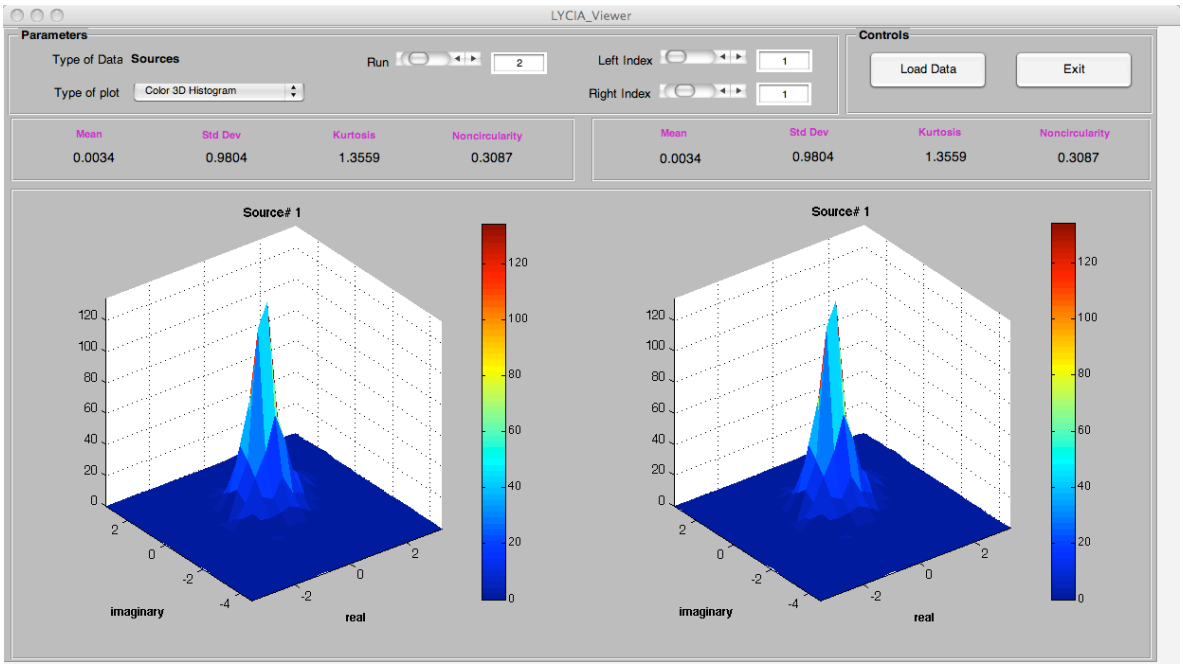


Figure 12. Color 3D Histogram

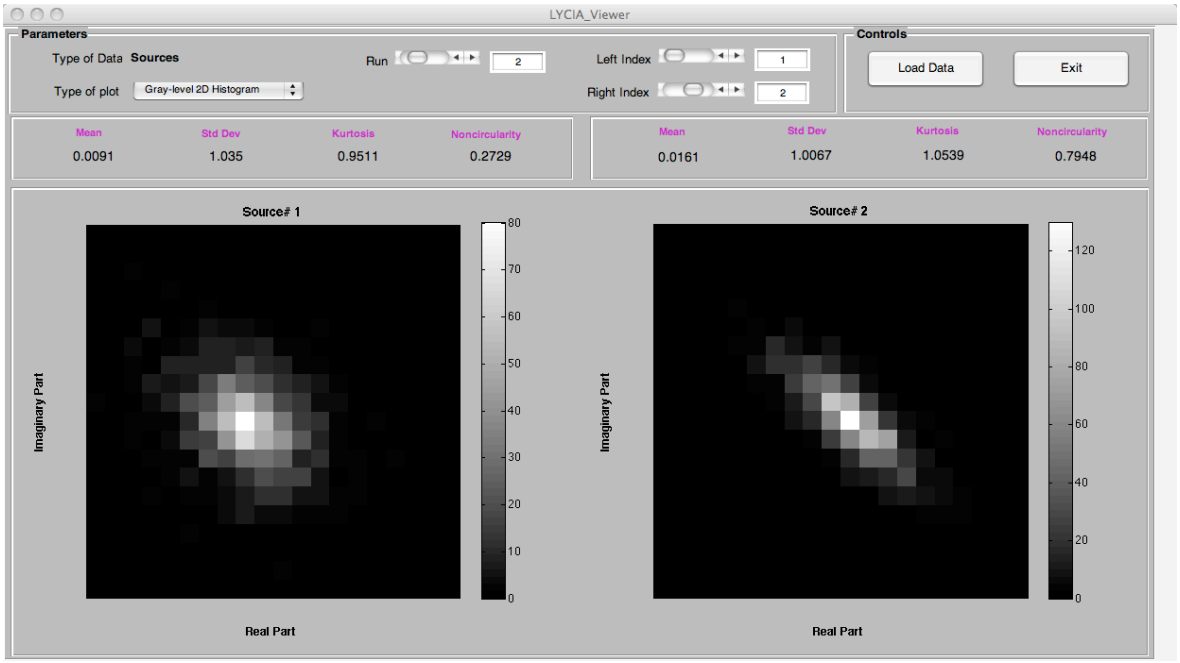


Figure 13. Grayscale 2D Histogram

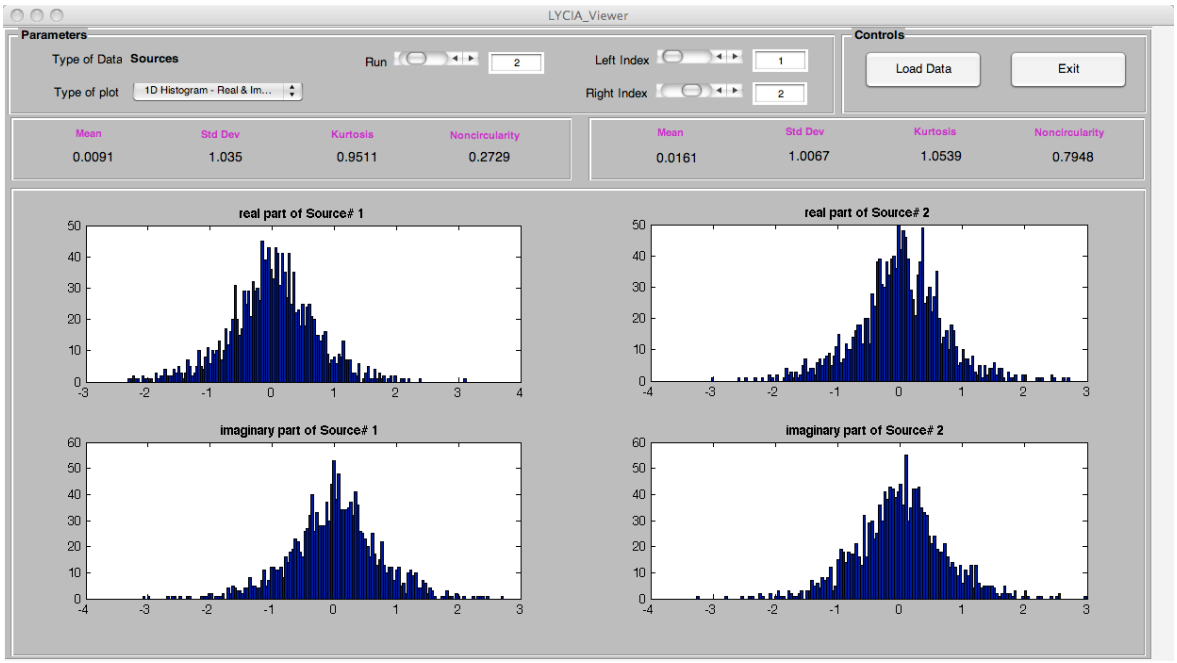


Figure 14. 1D Histogram – Real and Imaginary parts

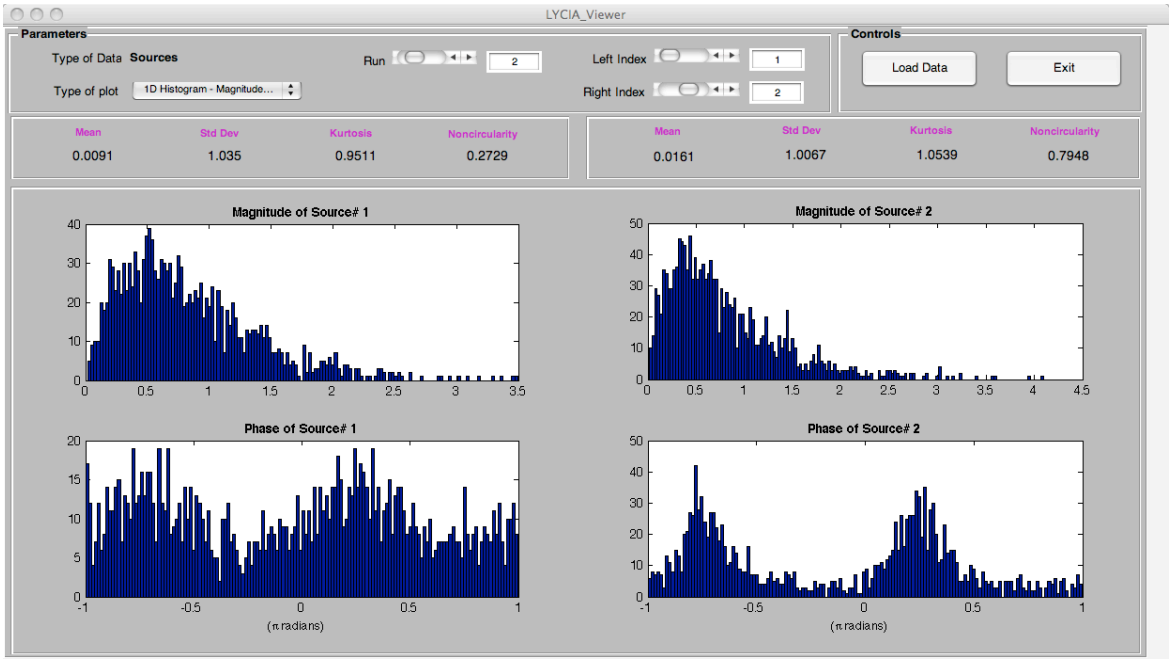


Figure 15. 1D Histogram – Phase and Magnitude

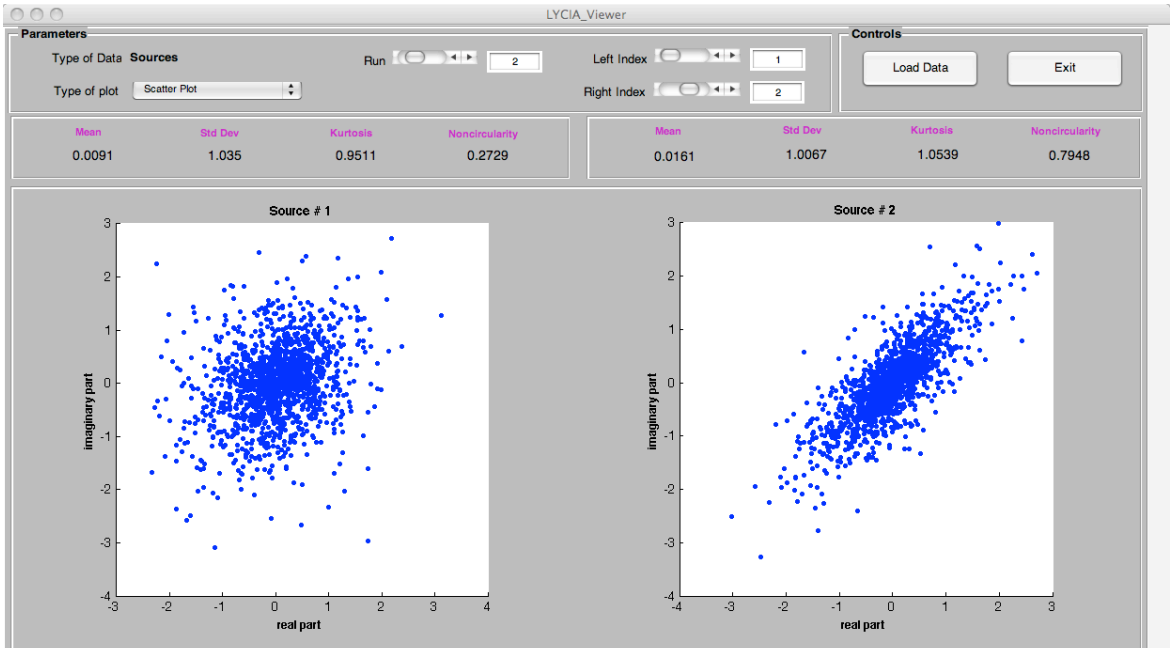


Figure 16. Scatter Plot

>>>> Visualizing the estimates

Visualizing the estimates is almost identical to visualizing sources and observations. The estimates can be plotted using the five methods previously mentioned. The only difference is that each shown with its corresponding source side by side. Therefore users can display only one estimate at a time. Figure 17 shows a display of an estimate. There is one more option to choose the algorithm to display in the menu ***Algorithm.***

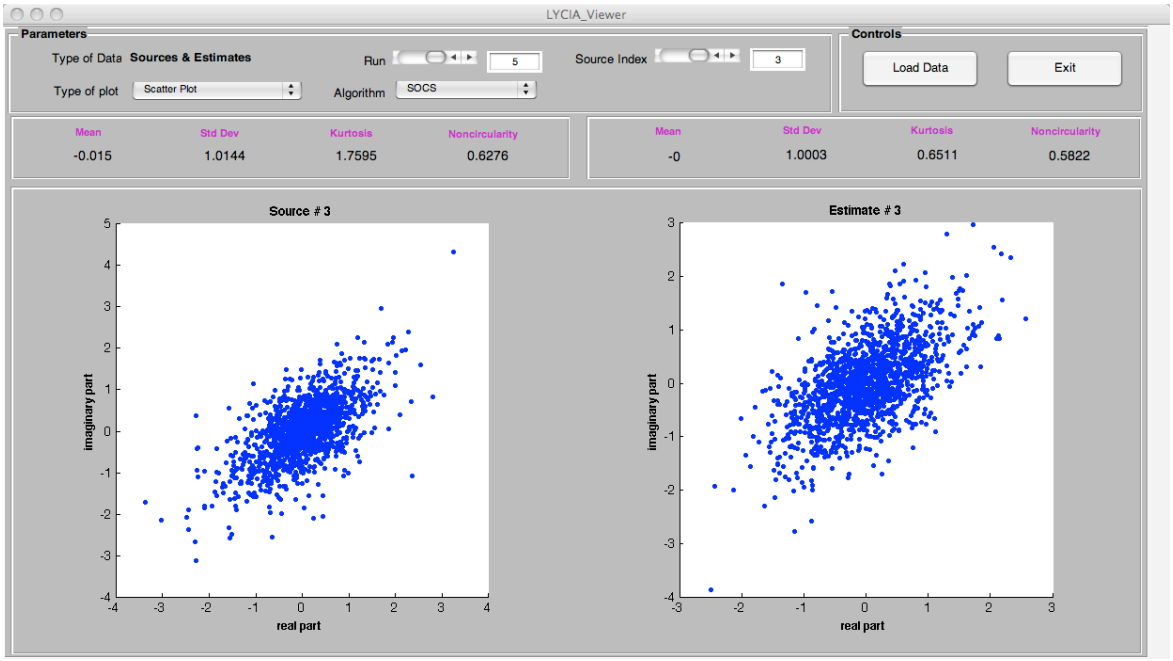


Figure 17. Scatter plot of a source and its estimate

>>>> Visualizing performance indices

An important feature of LYCIA is the evaluation of the ICA algorithms by their performance indices. For each algorithm used to separate the independent components, LYCIA evaluates the *computing time (**CT**), the inter-symbol interference (**ISI**), and the interference to signal ratio (**ISR**)*. The parameter can be visualized by clicking the ***Plot Performance Index*** button in the main interface of LYCIA. This visualization offers two options: 1) ***average performance index*** and 2) ***performance index per run.*** In the first case the average value of each index across all runs is displayed, while the second option provides the performance for each run. Figures 18 and 19 show one plot of each index.

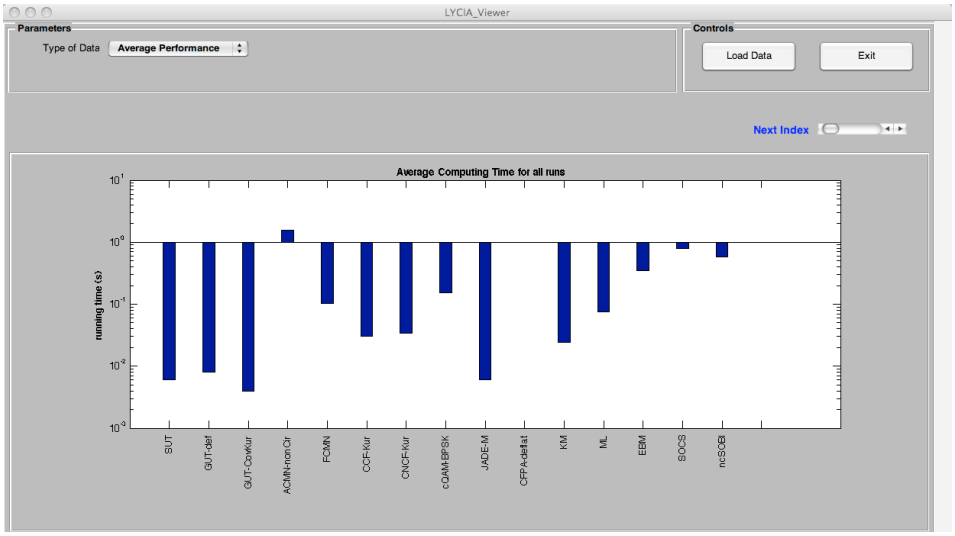


Figure18. Average computing time

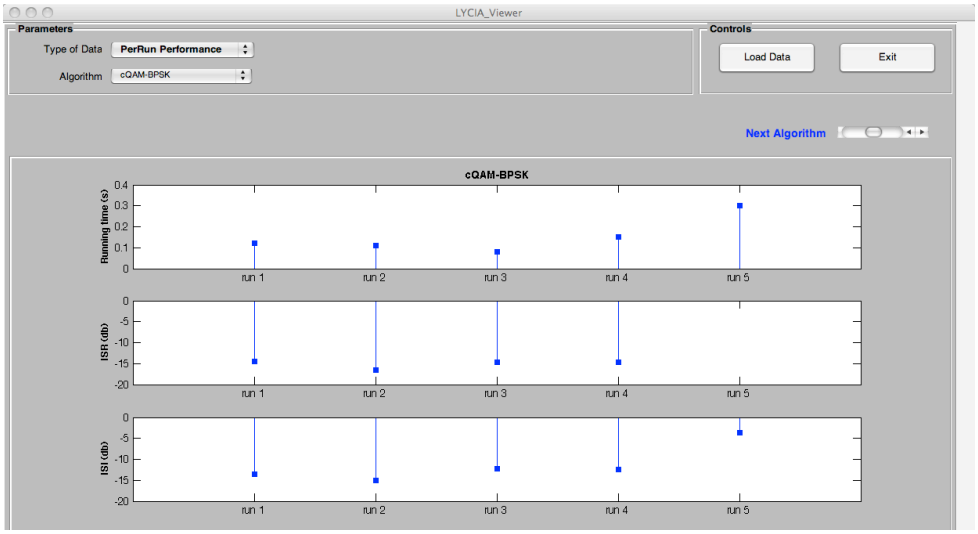


Fig. 19: Per run performance

Appendix A: Saving and loading data

>>>> Saving data

Data generated in LYCIA can be saved for further processing and/or visualization. Since the visualization of estimates is performed relatively to the original sources, they are saved when the original sources are available. It is recommended to save the data after a complete operation – source generation, source mixing and source separation. In this case users save the data by clicking the ***Save Outputs*** button, entering the ***Current Folder*** and the file ***Prefix*** in the dialog box shown in Figure 20.

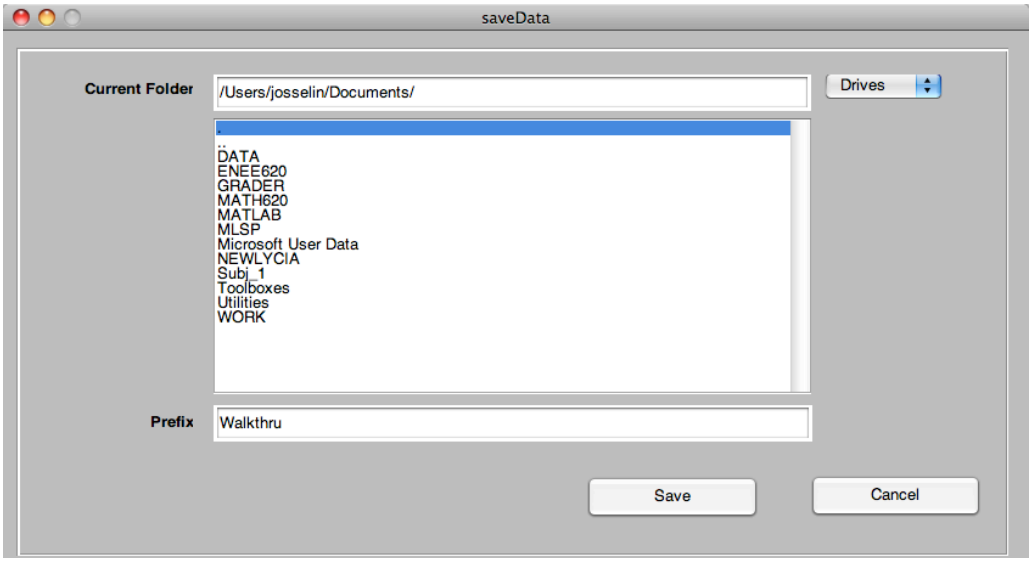


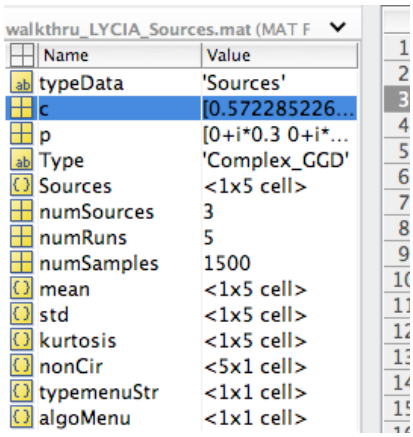
Figure 20. Dialog box for saving data

In figure 20, the prefix is set to *Walkthru.* If sources, observations, and estimates correspond to each other, saving the data will produce the following *.mat files.

File contents	File names
Sources	***Walkthru_LYCIA_Sources***
Observations	***Walkthru_LYCIA_Observations***
Estimates	***Walkthru_LYCIA_Estimates***
Performance indices	***Walkthru_LYCIA_Performances***

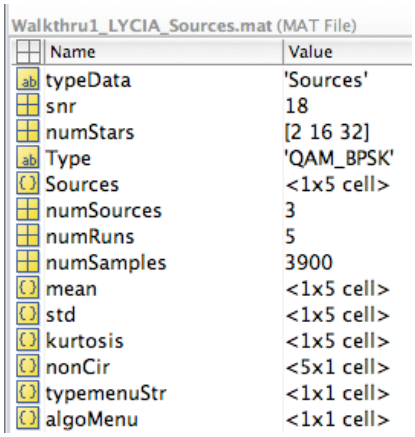
>>>>> Variables in the MAT files

Figure 21-24 show list of variables in each file followed by a brief description of file specific variables. Note that all four types of files have common variable. These are *typeData , numSources, numRuns and numSamples* representing respectively the type of data, the number of sources per run, the number of runs and the number of samples.



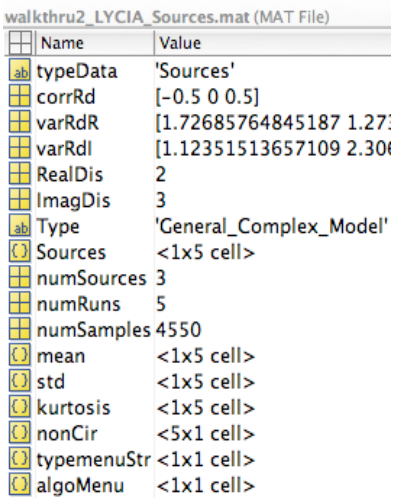
Fields	Descriptions
c	User entered circularity coefficients before generation
p	Shape parameters
type	Model of source distribution
mean	Cell array of mean of the sources
std	Cell array of standard deviation
kurtosis	Cell array of Kurtosis of the sources
nonCir	Computed noncircularity after generation of the sources
Sources	Cell array of source data for all runs

Figure 21. Variables in source file for GGD sources



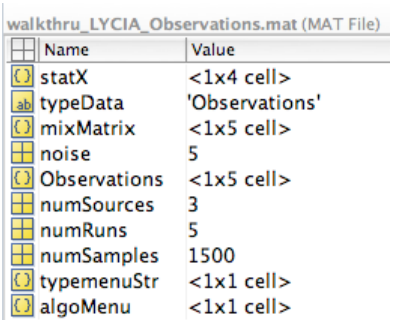
Fields	Descriptions
snr	Signal to noise ration entered by user before generation
numStars	Vector of constellation type for each source
Note: the other parameters are the same as in figure 21	

Figure 22. Variables in source file for QAM/BPSK sources



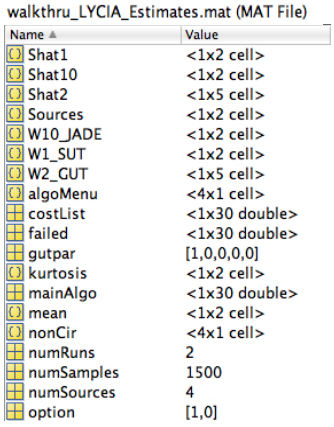
Fields	Descriptions
corrRd	Vector of correlation between real and imaginary parts
varRdR	Variance of the real part
varRdI	Variance of the imaginary part
RealDis	Distribution of the real part
ImagDis	Distribution of the imaginary part

Figure 23. Variables in source file for General Complex Model sources



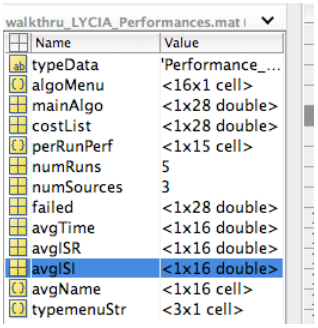
Fields	Descriptions
statX	Statistics of the observation, please load file in viewer
mixMatrix	Mixing matrix used to mix the sources
noise	Noise – added by user before generating mixture
Observations	Cell array of observations for all runs

Figure 24. Variables in observation file



Fields	Descriptions
Shat1, …	Estimates data for the first algorithm, …
W1_SUT	Cell array for demixing matrix of the first algorithm (SUT) The number indicates index for algorithm and the label in the end indicates the name of the algorithm. Each element represents the demixing matrix for one run.
Note: The remaining parameter are options of algorithms used to Separate Sources and depend on user's choices.	

Figure 25. Variables in estimates file



Fields	Descriptions
perRunPerf	Cell array of performance data for individual runs
failed	Failure information – related to the algorithms
avgTime	Average computing time for all runs per algorithm
avgISI	Average ISI for all runs per algorithm
avgISR	Average ISR for all runs per algorithm
Note: the remaining data are used by graphing routines	

Figure 26. Variables in performance index file

>>>> Loading data

There are two possible ways to load saved data for further visualizations – either in the main interface of LYCIA, or in any of the visualizing interface. In the main window of LYCIA, the command button ***Plot saved data*** (see figure 1) allows the user to select one of the previous files for visualization. Once in the Viewer, the user can choose another source of data by clicking the ***Load data*** button (figure 10).

References

Source generation:

Complex GGD
M. Novey, T. Adalı, and A. Roy, "A complex generalized Gaussian distribution-characterization, generation, and estimation," *IEEE Trans. Signal Processing*, vol. 58, no. 3, part. 1, pp. 1427–1433, March 2010.

QAM/BPSK
M. Novey and T. Adalı, "Complex fixed-point ICA algorithm for separation of QAM sources using Gaussian mixture model," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Honolulu, Hawaii, April 2007.

Algorithms:

SUT
J. Eriksson and V. Koivunen, "Complex-valued ICA using second order statistics," in *Proc. IEEE Workshop on Machine Learning for Signal Processing (MLSP)*, Sao Luis, Brazil, pp. 183–192, September 2004.

GUT
E. Ollila and V. Koivunen, "Complex ICA using generalized uncorrelating transform," *IEEE Trans. Signal Processing*, vol. 89, no. 4, pp.365–377, 2006.

SOBI
A. Belouchrani, K. Abed Meraim, J. F. Cardoso, and E. Moulines, "A blind source separation technique based on second order statistics," *IEEE Trans. Signal Processing*, vol. 45, no. 2, pp. 434–444, 1997.

ERM-G and ncSOBI
X.-L. Li and T. Adalı, "Blind Separation of complex noncircular stationary Gaussian sources," *IEEE Trans. Signal Processing*, vol. 59, no. 6, pp. 2969–2975, 2011.

A-CMN
M. Novey and T. Adalı, "Adaptable nonlinearity for complex maximization of nongaussianity and a fixed-point algorithm," in *Proc. IEEE Workshop on Machine Learning for Signal Processing (MLSP)*, Maynooth, Ireland, pp. 79–84, September 2006.

T-CMN
M. Novey and T. Adalı, "Complex ICA by negentropy maximization," *IEEE Trans. Neural Networks*, vol. 19, no. 4, pp. 596–609, April 2008.

Circular complex FastICA
E. Bingham and A. Hyvarinen, "A fast fixed-point algorithm for independent component analysis of complex valued signals," *International Journal of Neural Systems*, vol. 10, no. 1, pp. 1–8, 2000.

Noncircular complex FastICA
M. Novey and T. Adalı, "On extending the complex FastICA algorithm to noncircular sources," *IEEE Trans. Signal Processing*, vol. 56, no. 5, pp. 2148–2154, May 2008.

C-QAM
M. Novey and T. Adalı, "Complex fixed-point ICA algorithm for separation of QAM sources using Gaussian mixture model," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Honolulu, Hawaii, April 2007.

JADE
J. F. Cardoso and A. Souloumiac, "Blind beamforming for nongaussian signals," *IEEE Trans. Signal Processing*, vol. 140, no.6, pp. 362–370, 1993.

CFPA
S. C. Douglas, "Fixed-point algorithms for the blind separation of arbitrary complex-valued non-Gaussian signal mixtures," *EURASIP Journal on Applied Signal Processing*, vol. 2007, no. 1, pp. 83–83, 2007.

Fixed point KM
H. Li and T. Adalı, "A class of complex ICA algorithms based on the kurtosis cost function," *IEEE Trans. Neural Networks*, vol. 19, no. 3, pp. 408-420, March 2008.

Complex likelihood maximization
T. Adalı, H. Li, M. Novey, and J. F. Cardoso, "Complex ICA using nonlinear functions," *IEEE Trans. Signal Processing*, vol. 56, no. 9, pp. 4536–4544, Sept. 2008.

Nonlinear decorrelations
T. Adalı, T. Kim, and V. Calhoun, "Independent component analysis by complex nonlinearities," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, vol. 5, pp. 525–528, Montreal, Canada, May 2004.

Complex ICA-EBM
X.-L. Li and T. Adalı, "Complex independent component analysis by entropy bound minimization," *IEEE Trans. Circuits and Systems I,* vol. 57, no. 7, pp. 1417–1430, July 2010.

Robust ICA

V. Zarzoso and P. Comon, “Robust Independent Component Analysis by Iterative Maximization of the Kurtosis Contrast with Algebraic Optimal Step Size,” *IEEE Trans. Neural Networks*, vol. 21, no. 2, pp. 248–261, February 2010.

A complete reference for complex GGD generation, A-CMN, T-CMN, and C-QAM algorithms:

M. Novey, “Complex ICA using Nonlinear Functions,” Ph.D. Thesis, University of Maryland Graduate School, Baltimore, MD, 2009.