InterregionalTransfer

Demo to show how to infer interregional effective connections using collective transfer entropy.

examples

Updated Today (moments ago) by joseph.lizier

Demos > Interregional Transfer demo

Interregional Transfer demo

This is a higher-level example, which demonstrates computing information transfer between two regions of variables (e.g. brain regions in fMRI data), using multivariate extensions to the transfer entropy and mutual information to infer effective connections between the regions. This method is distinguished in using asymmetric, multivariate, information-theoretical analysis, which captures not only directional and non-linear relationships, but also collective interactions between the variables.

The software implements the method originally described in:

J.T. Lizier, J. Heinzle, A. Horstmann, J.-D. Haynes, M. Prokopenko, "Multivariate information-theoretic measures reveal directed information structure and task relevant changes in fMRI connectivity", Journal of Computational Neuroscience, vol. 30, pp. 85-107, 2011; doi: 10.1007/s10827-010-0271-2.

A presentation on this paper and method is also available.

This demo is executed purely in Java.

It is found at `demos/java/interregionalTransfer/` in the svn or full distribution.

Contents

This demo includes the following files:

- `interregionalCalc.properties` - a sample properties file to feed to the Java code (parameters are described in comments in the file itself, and some are highlighted below);
- `runSample.sh` - shell script to run the code on a sample data set;
- `region1-numCoupled10-coupling0.30-past0.70.txt` - sample data set (destination) found in the `demos/data` folder;
- `region2-numCoupled10-coupling0.30-past0.70.txt` - sample data set (source) found in the `demos/data` folder;
- `sampleOutput1to2.txt` - sample output when running the code with sample region 1 as the source;
- `sampleOutput2to1.txt` - sample output when running the code with sample region 2 as the source.

Running the code

The code can be run on source and destination data files by using the `runSample.sh` script and altering relevant properties in the `interregionalCalc.properties` file.

We will illustrate how to use these on the sample data set -- this corresponds to the data used for Figure 7(a), with parameters C=10, \chi=0.3, in the above paper, with Y as the region 2 data file and X as region 1. *Note* that the results may be slightly different to what was obtained in that paper, since we are now using a more accurate Kraskov TE estimator (using a single conditional MI estimator instead of two MI estimators).

1. Run the `runSample.sh` script - this will compute the interregional transfer from our sample data set 2 to sample data set 1. Check that you get the standard output (we interpret this below):

   ```
   0.0459,0.0674,-0.0059,0.0051,10.2104,100
   ```

2. Swap the values of props.interregionalChannel.directRun.file1 and props.interregionalChannel.directRun.file2 in the properties file (this sets up the code to look at the transfer from sample data set 1 to 2).
3. Run the `runSample.sh` script - this will compute the interregional transfer from our sample data set 1 to sample data set 2. Check that you get the standard output:

   ```
   0.0041,0.0361,0.0006,0.0053,0.6690,100
   ```

4. You can also alter the property `props.interregionalChannel.writeResultsToStdOut` in the properties file to be `true` (this causes extra data to be dumped to the screen). Then run the `runSample.sh` script again (for each direction) and check that the standard output you get matches `sampleOutput1to2.txt` and `sampleOutput2to1.txt`.

Interpreting the results

The six numbers that are output above are as follows:

1. the mean and
2. standard deviation of the interregional transfer value across the S subsets of v variables (where v is specified by `props.interregionalChannel.jointVars1`, `props.interregionalChannel.jointVars2` and S by `props.interregionalChannel.maxNumSets` -- see definition in paper above);
3. Then the mean and
4. standard deviation of the P surrogate transfer values (P is specified by `props.interregionalChannel.directRun.numReorderingsForSignificance`), followed by
5. the t-value of the actual interregional transfer value in comparison to these, and
6. a confirmation of the number of surrogates used.

You may be simply interested in the raw average transfer values. In this case you should set `props.interregionalChannel.directRun.getSignificance` to `false` since this takes P times more time, and you will only see the first two numbers output.

More likely you are interested in whether this interregional transfer value is *statistically significant*. In this case, you are interested in the later values, including the second last value, the t-value of the actual interregional transfer value in comparison to the surrogate values. As described in the above paper (section 2.2), you can perform a one-sided t-test on this t-value with P-1 degrees of freedom (or simply perform a z-test if P is high enough). You are best served by using a larger value of P than is demonstrated here, preferably in the 1000's. As an alternative to computing a z/t-value, you could simply count the proportion of surrogate values that the actual value was greater than. Indeed, this stays with a model-free view of the distribution of surrogates. However with relatively small numbers of surrogates P, and relatively small \alpha values (particularly if you are correcting for multiple comparisons), this may be quite sensitive to statistical fluctuations and a z/t-test approach is more sensible. Don't forget to **correct for multiple comparisons** if you have many region pairs to evaluate! (As outlined in the paper).

Conclusion -- returning to our example above, you will note that the transfer value for region2 -> region 1 is statistically significant using \alpha=0.05 and region 1 -> region 2 is not, revealing the underlying region2 -> region 1 relationship that was used to construct the data set.

Finally, if you set `props.interregionalChannel.writeResultsToStdOut` to `true` then the more verbose output simply helps you track the progress of the calculation firstly through the S subsets to compute the interregional transfer, then over the P surrogate calculations to compare for the significance calculation. The six main output numbers defined above are then separated -- the first two occur at line 102 of the sample output, after all S subsets are examined, while the last four occur at line 203 of the sample output.

Parameter file settings

You can alter the parameters in the sample parameter file in order to run the code on your own data. The parameters are all described inside `interregionalCalc.properties`; the important ones to note are:

- `props.interregionalChannel.jointVars1` and `props.interregionalChannel.jointVars2` - the number of joint variables v to consider in each region;
- `props.interregionalChannel.maxNumSets` - the number of subsets S of v variables from each region to select to compute the multivariate measure on;
- `props.interregionalChannel.directRun.getSignificance` - whether to run the statistical significance computation;
- `props.interregionalChannel.directRun.numReorderingsForSignificance` - number of surrogate pairs P to use to compute the significance of the actual interregional transfer value;
- `props.interregionalChannel.calculator` - which underlying transfer entropy or mutual information implementation to use for the calculation. I.e. you use this parameter to select whether to compute TE or MI, and also which particular method to use for the computation. Several recommendations are made in the file.
- Several properties then follow, which are relevant to different calculators/methods. Important ones include:
  - `props.interregionalChannel.te.k` - k history value for TE
  - `props.interregionalChannel.calculatorProperties.EPSILON` - kernel width if you use a kernel.* class
  - `props.interregionalChannel.calculatorProperties.k` - window size if you use a kraskov.* class
  - `props.interregionalChannel.calculatorProperties.TIME_DIFF` - which can be used to switch the mutual information calculator into a time-differenced mutual information mode.
  - `props.interregionalChannel.directRun.file1` and `props.interregionalChannel.directRun.file2` - the source data files. See the sample files for the currently accepted format.