Information Prevalence Inference using the i-th order statistic: *i*-test

Overview

i-test is the MATLAB implementatation of the second-level (group-level) statistical test for the decoding accuracy proposed by Hirose (Under review), which is an extension of "Permutation-based prevalence inference using the minimum statistic", proposed by Allefeld et al., 2016.

Carsten Allefeld, Kai Görgen and John-Dylan Haynes, 'Valid population inference for information-based imaging: From the second-level t-test to prevalence inference', NeuroImage 2016, https://doi.org/10.1016/j.neuroimage.2016.07.040. https://github.com/allefeld/prevalence-permutation/).

Feedback and bug report

Any feedback and bug reports are welcome. Please mail to satoshi.hirose [at] nict.go.jp (please replace the [at] with the '@' symbol).

Copyright

i-test is free but copyrighted software, distributed under the terms of the GNU General Public Licence. Further details on copyleft can be found at http://www.gnu.org/copyleft/. No formal support or maintenance is provided or implied.

Installation

To install *i*-test, download files and add directory "`itest`" to MATLAB path.

Quickest how to

i-test requires Experimental results (Decoding accuracy), null distribution of the decoding accuracy (decoding accuracy when brain activity do not contain the label information) and three predetermined parameters (γ_0, α, i).

1. Decide the two threshold parameters, e.g. $\gamma_0 = 0.5, \alpha = 0.05$.
2. Decide the parameter for order statistic (i) with `decide_i.m`.
3. Perform decoding analysis and calculate decoding accuracy.
   *Try MVPC toolbox if you are a beginner to decoding analysis.
4. Do (4-1) or (4-2) to estimate null distribution of the decoding accuracy.
   (4-1) Empirical test (e.g. permutation test) to estimate null distribution
   (4-2) Assume the null distribution
      (e.g. binomial distribution with success probability of chance level).
5. Perform *i*-test with `itest.m`

Example

```
load Replication_of_study/Empirical_Experiment_Results.mat
%you will find SD and PD in your workspace.
%
%% Decide the two threshold parameters
g_0 = 0.5;
alpha = 0.05;

%% Decide the parameter for order statistic
% get number of participants
N = size(SD,1);
i_max = check_imax(N,g_0,alpha)

N_trial = 150; % number of trials
P_correct_minus = 0.5; % chance level
i=1:i_max; %define the range of i
gamma = g_0:0.01:1; % define the range of true prevalence
P_correct_plus = P_correct_minus:0.01:1; % define the range of true
expectation of decoding accuracy when label information is represented in
the Brian activity
P_sig =
decide_i(N,N_trial,g_0,alpha,P_correct_minus,gamma,P_correct_plus,i);
%you will find the 3d-plot figure, in which statistical power is plotted.
% From this result shown in the Figure, I select i = 3.
i = 3;
[H, L, stat] = itest(SD,PD,g_0,i,alpha)
% L is 0.0239, which is smaller than alpha!
% Congratulations. the null hypothesis is rejected !  (H=1)
```

Instruction for each codes

```
[H, prob, stat] = itest(SD,PD,g_0,i,alpha,identical)
```

Implementation of i-test, statistical test for 2nd level (group) statistics. i-test evaluates "in population, whether proportion of people with higher-than-chance decoding accuracy is higher than the prevalence threshold."

INPUT

`SD`: Sample Decoding Accuracies from experiment (N x 1 matrix)

`PD`: (see below for detail) estimation of null distribution

`g_0`: Prevalence threshold, γ_0 (Real number between 0 and 1 default:0.5)

`i`: Index of order statistics, i (Positive integer, default: 1)

`alpha`: statistical threshold, α (Real number between 0 and 1 default:0.05)

`identical`: (see below; boolean) if you assume the identical distribution of decoding accuracy distribution among participants.

OUTPUT

`H`: 1 if the result is significant (Prob < alpha)

`Prob`: Probability (Real number between 0 and 1)

`stat`: (structure)

   .`prob_min`: Theoretical lower bound of `Prob` computed from predetermined parameters (`g_0` and `i`) and number of participants.

   .`param`: predetermined parameters (`g_0`,`i` and `alpha`) and

  .`order_stat`: i-th order statistic of SD (real number)

  .`P_0`: probability of `PD` >= `order_stat`

NOTE

`identical`: In the original version of i-test, the identical distribution of the decoding accuracies among participants is assumed (`identical = 1`). But, I also proposed an extension of i-test to relax this assumption (`identical = 0`). In the extension version, the decoding accuracy distribution is not necessarily identical among participants.

    Note that the computational cost of this extension version (`identical = 0`) is much, much higher ($O(N^i)$: N: Number of participnats) than the original version (`identical = 1`) ($O(N)$) and therefore, should not be use when $N > 40$ or so.

    Without assumption of the identical distribution, we have to estimate the null distribution (`PD`) for each participant. Therefore, input format of (`PD`) is different. If not specified, format of `PD` is reffered (see below)

`PD`: Estimation of the decoding accuracy distribution under the null hypothesis "there is no label information in the brain." This is empirically estimated by

e.g. permutation test, or assumed by a statistical model, e.g. binomial distribution.

The input format depends on whether the identical distribution is assumed or not. In both cases, the distribution can be specified with empirical estimation style (list of empirical estimation) or histogram style (similar to output of MATLAB function "hist").

There are 2x2=4 input formats.

1) empirical estimation style.
1-1) `identical = 1`
Format: 1 x [Number of repetition] matrix
Example: After permutation test with 1,000 repetition for each of 10 participants,
     PD is a 1 x 10,000 (10*1,000) horizontal vector.

1-2) `identical = 0`
Format: N x [Number of repetition] matrix
Example: After permutation test with 1,000 repetition for each of 10 participants,
PD is 10 x 1,000 matrix.

2) Histogram style.
2-1) `identical = 1`
Format: 2 x [number of possible decoding accuracies] matrix,
Example: If there's 2 trials of binary decoding for each participants and we assume that decoding accuracy follows binomial distribution, then,

```
PD    =   [0        50        100
           0.25     0.5       0.25]
```

2-2) `identical = 0`
Format: N x 1 cell array. Each cell contains 2 x [number of possible decoding accuracies] for each participant.
Example: If Subject #1 performed 2 trials and Subject #2 performed 4 trials and we assume that decoding accuracy follows binomial distribution,

```
PD{1}    =   [0        50        100
              0.25     0.5       0.25]
PD{2}    =   [0        25        50        75        100
              0.0625   0.25      0.375     0.25      0.0625]
```

```
[i_max,prob_min] = check_imax(N,g_0,alpha)
```

The script calculate the maximum possible i from the number of subjects (`N`), and the predetermined threshold parameters (`g_0` and `alpha`)

INPUT

`N`: Number of participant (integer)

`g_0`: Prevalence threshold, γ_0 (Real number between 0 and 1 default:0.5)

`alpha`: statistical threshold, α (Real number between 0 and 1 default:0.05)

OUTPUT

`i_max`: The maximum available i

`prob_min`: Theoretical lower bound of `Prob` computed from predetermined parameters (`g_0` and `i`) and number of participants.

```
[P_sig,T,p_below_T_individual,p_below_T_plus,p_below_T_minus,L]=
decide_i(N,N_trial,g_0,alpha,P_correct_minus,
gamma,P_correct_plus,i_all)
```

This function estimates the probability of significant result of i-test (`P_sig`) from inputs below. The calculation of `P_sig` is implemented in function `numeric_binomial` (next page). Main role of this function is the visualization to aid the selection of i.

INPUT

`N`: Number of subjects (positive integer)

`N_trial`: Number of trials (positive integer)

`g_0`: Prevalence threshold, γ_0(Real number between 0 and 1 default:0.5)

`alpha`: statistical threshold, α (Real number between 0 and 1 default:0.05)

`P_correct_minus`: chance level decoding accuracy, e.g. 0.5 for binary classification. (Real number between 0 and 1)

`gamma`: true information prevalence: i.e. true proportion of people in population with higher-than-chance decoding accuracy. (Real number between 0 and 1 or vector)

`P_correct_plus`: true expectation of decoding accuracy of people with higher-than-chance decoding accuracy (Real number between 0 and 1 or vector)

`i_all`: candidate values of i (positive integer or vector)

OUTPUT

`P_sig`: probability of achieving significant result of i-test. 3-dimensional matrix with size of [length of `gamma`] × [length of `P_correct_plus`] × [length of `i`]

* The other outputs are variables appear during the calculation of `P_sig`. Please see the original paper.

`Figure`: plot of `P_sig` against `gamma, P_correct_plus` or both.

NOTE

For the population parameters (`gamma` and `P_correct_plus`) that cannot be known, single value of multiple values (e.g. 0:0.01:1) can be used. The output figure changes depending of the set of input formats.
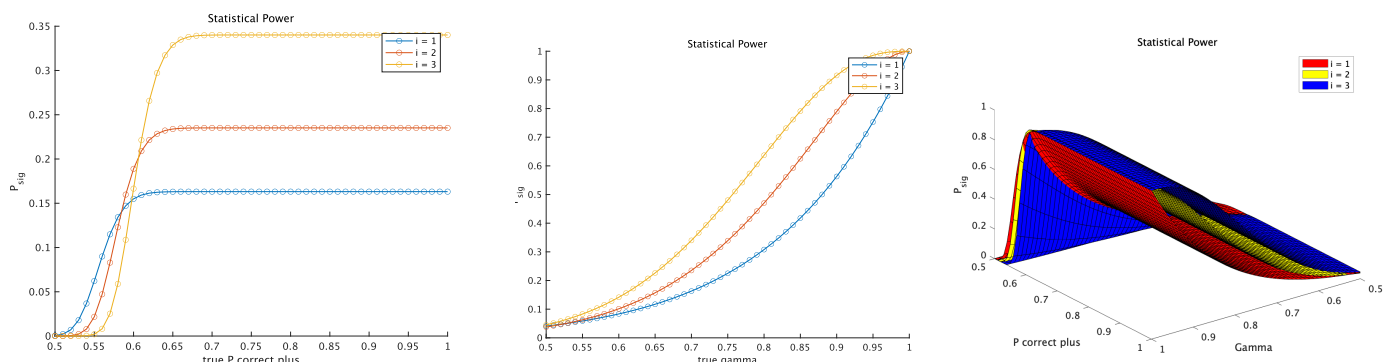
```
[P_sig,T,p_below_T_individual,p_below_T_plus,p_below_T_minus,L]
= numeric_binomial(N_trial,N,g_0,alpha,gamma,
P_correct_minus,P_correct_plus,i)
```

Numerical calculation of the i-test's statistical power (or False alarm ratio). The probability that i-test provide significant results is calculated from the input parameters.

INPUT

`N`: Number of subjects (positive integer)

`N_trial`: Number of trials (positive integer)

`g_0`: Prevalence threshold, γ_0(Real number between 0 and 1 default:0.5)

`alpha`: statistical threshold, α (Real number between 0 and 1 default:0.05)

`P_correct_minus`: chance level decoding accuracy, e.g. 0.5 for binary classification. (Real number between 0 and 1)

`gamma`: true information prevalence: i.e. true proportion of people in population with higher-than-chance decoding accuracy. (Real number between 0 and 1)

`P_correct_plus`: true expectation of decoding accuracy of people with higher-than-chance decoding accuracy (Real number between 0 and 1)

`i`: candidate values of i (positive integer)

OUTPUT

`P_sig`: Estimated probability of achieving significant result (Real number between 0 and 1)

* The other outputs are variables appear during the calculation of `P_sig`. Please see the original paper.