Instructions for use of C code

This code designs error spending versions of the delayed response group sequential tests proposed in Section 4.1.1 of Hampson & Jennison (2012). We hope there aren’t any bugs in this code but if you find any, please report them to l.v.hampson@lancaster.ac.uk.

File Descriptions

|  |  |
| --- | --- |
| Function | Purpose |
| driver.c | Main file which searches for the constant ρ indexing the error spending functions for the DR GST terminates properly at stage K under maximum sample size R\*nfix (which is stipulated by the user). This function outputs a summary of the design properties. |
| inflate.c | For a given pair of error spending functions, finds the target maximum sample size for which the DR GST terminates properly at Stage K. |
| findconstants.c | For a given pair of error spending functions and target maximum sample size, finds the boundaries of an error spending DR GST assuming analyses are scheduled following pattern (10) in Hampson & Jennison. This function calls *bisection.c* and *symmetry.c* to find stopping boundaries and decision constants, respectively. |
| bisection.c | Finds stopping boundaries and according to the improved version of Method 1 discussed in Section 4.1.1 of Hampson & Jennison (2012). This approach accounts for the type II error rate actually spent by stage (k-1) to avoid conservatism in controlling the overall type II error rate of the test. |
| symmetry.c | Finds decision constant balancing Stage k reversal probabilities under θ=0. |
|  |  |
| type1error.c | Calculates P(Continue to Stage k and stop with Zk ≥ c; θ). |
| type2error.c | Calculates P(Continue to Stage k and stop with Zk ≤ c; θ). |
| type1errorstar.c | Calculates P(Continue to Stage k and stop to Reject H0; θ). |
| type2errorstar.c | Calculates P(Continue to Stage k and stop to Accept H0; θ). |
| reversal1.c | Calculates P(Continue to Stage k, ; θ). |
| reversal2.c | Calculates P(Continue to Stage k, ; θ). |
|  |  |
| expectation.c | Evaluates objective functions of expected sample on termination of tests. |
|  |  |
| input.txt | Contains data to be read into driver.c |
| header.h | Header file for the C programs. |
| makefile | Makefile including rules needed for compilation of the program. |

Running the program to design error spending Delayed Response GSTs

The C code calls two Nag Fortran 77 routines to calculate probabilities and quantiles of a standard normal random variable. Calls to these external routines can be omitted if they can be substituted by calls to user written C functions instead. The path to the nag routines currently given in the makefile may need to be altered for correct compilation on the user’s own Unix system.

The header.h file contains several global constants defining the properties of the GST to be designed and specification of the numerical integration routines used. The program should be recompiled each time changes are made to the header or C files. This is done by typing

>> make

to the command prompt of the Unix session. This produces an executable called errorspend. When the program is run, it expects to read in three constants from a file *input.txt* which further define the design of the test to be found. Further definitions of the constants are given in the file *driver.c*. To run the program interactively, type

>>./errorspend<input.txt

to the command prompt of the Unix session.

Program output

Error spending tests are designed assuming tests will be conducted for information sequence (10) in Hampson & Jennison (2012) assuming responses are normally distributed with known variance. On completion of the program, a summary of the error spending design will be written to a file called Design\_summary.txt. This output includes the value of the constant (referred to as rho) indexing the error spending functions for which the delayed response GST terminates properly at stage K spending cumulative type I and type II error probabilities α and β under the specified information sequence. Program output also includes the critical values which would define the GST under the analysis schedule (10).

In practice, observed information levels at each interim and decision analysis will be unpredictable. When conducting the trial, code similar to that given in *findconstants.c* would be used to find the critical values defining the stopping rule at each interim analysis and, once recruitment has stopped, the critical value defining the decision analysis.

References

Hampson LV and Jennison C. (2012). Group sequential tests for delayed responses (with discussion). *To appear in J. R. Statist. Soc. B*