Package ‘cvcrand’

April 13, 2020

Type Package

Title Efficient Design and Analysis of Cluster Randomized Trials

Version 0.1.0

Date 2020-04-13

Maintainer Hengshi Yu <hengshi@umich.edu>

Description Constrained randomization by Raab and Butcher (2001) <doi:10.1002/1097-0258(20010215)20:3%3C351::AID-SIM797%3E3.0.CO;2-C>
is suitable for cluster randomized trials (CRTs) with a
small number of clusters (e.g., 20 or fewer). The procedure of
constrained randomization is based on the baseline values of some
cluster-level covariates specified. The intervention effect on
the individual outcome can then be analyzed through
clustered permutation test introduced by Gail, et al. (1996) <doi:10.1002/(SICI)1097-0258(19960615)15:11%3C1069::AID-SIM220%3E3.0.CO;2-Q>.
Motivated from Li, et al. (2016) <doi:10.1002/sim.7410>, the package performs constrained randomization on the baseline
values of cluster-level covariates and clustered permutation test on the individual-level outcomes for cluster randomized trials.

License GPL (>= 2)

LazyData TRUE

Depends R (>= 3.4.0)

Imports tableone

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.1.0

Encoding UTF-8

NeedsCompilation no

Author Hengshi Yu [aut, cre],
Fan Li [aut],
John A. Gallis [aut],
Elizabeth L. Turner [aut]

Repository CRAN

Date/Publication 2020-04-13 19:00:02 UTC

R topics documented:

`cptest` *Clustered permutation test for cluster randomized trials*

Description

cptest performs a clustered permutation test on the individual-level outcome data for cluster randomized trials (CRTs). The type of the outcome can be specified by the user to be `"continuous"` or `"binary"`.

Linear regression (for outcome type `"continuous"`) or logistic regression (for outcome type `"binary"`) is applied to the outcome regressed on covariates specified. Cluster residual means are computed. Within the constrained space, the contrast statistic between the treatment and control arms is created from the randomization schemes and the cluster residual means. The permutation test is then conducted by comparing the contrast statistic for the scheme actually utilized to all other schemes in the constrained space.

Usage

```
cptest(
  outcome,
  clustername,
  z = NULL,
  cspacedatname,
  outcometype,
  categorical = NULL
)
```

Arguments

`outcome`	a vector specifying the individual-level outcome.
`clustername`	a vector specifying the identification variable of the cluster.
`z`	a data frame of covariates to be adjusted for in the permutation analysis.

`cspacedatname` gives the path of the csv dataset containing the saved randomization space. This dataset contains the permutation matrix, as well as an indicator variable in the first column indicating which row of the permutation matrix was selected as the final scheme to be implemented in practice.

`outcometype` the type of regression model that should be run. Options are `"continuous"` for linear regression and `"binary"` for logistic regression.

`categorical` a vector specifying categorical (including binary) variables. This can be names of the columns or number indexes of columns, but cannot be both. Suppose there are `p` categories for a categorical variable, `cptest` function creates `(p-1)` dummy variables and drops the reference level if the variable is specified as a factor. Otherwise, the first level in the alphanumerical order will be dropped. If the user wants to specify a different level to drop for a `p`-level categorical variable, the user can create `p-1` dummy variables and these can instead be supplied as covariates to the `cptest` function. Then, the user needs to specify the dummy variables created themselves to be `categorical` when running `cptest`. In addition, the user could also set the variable as a factor with the specific reference level. The user must ensure that the same level of the categorical variable is excluded as was excluded when running `cvrall`, by coding the variables the same way as in the design phase. This is the only optional argument of the `cptest` function. All others are required.

Value

`FinalScheme` the final scheme in the permutation matrix

`pvalue` the p-value of the intervention effect from the clustered permutation test

`pvalue_statement` the statement about the p-value of the intervention effect from the clustered permutation test

Author(s)

Hengshi Yu <hengshi@umich.edu>, Fan Li <fan.f.li@yale.edu>, John A. Gallis <john.gallis@duke.edu>, Elizabeth L. Turner <liz.turner@duke.edu>

References

Gail, M.H., Mark, S.D., Carroll, R.J., Green, S.B. and Pee, D., 1996. On design considerations and randomization based inference for community intervention trials. Statistics in medicine, 15(11), pp.1069-1092.

Li, F., Lokhnygina, Y., Murray, D.M., Heagerty, P.J. and DeLong, E.R., 2016. An evaluation of constrained randomization for the design and analysis of group randomized trials. Statistics in medicine, 35(10), pp.1565-1579.

Li, F., Turner, E. L., Heagerty, P. J., Murray, D. M., Vollmer, W. M., & DeLong, E. R. (2017). An evaluation of constrained randomization for the design and analysis of group randomized trials with binary outcomes. Statistics in medicine, 36(24), 3791-3806.

Gallis, J. A., Li, F., Yu, H., Turner, E. L. (In Press). cvcrand and cptest: Efficient design and analysis of cluster randomized trials. Stata Journal.

Gallis, J. A., Li, Fl. Yu, H., Turner, E. L. (2017). cvcrand and cptest: Efficient design and analysis of cluster randomized trials. Stata Conference. https://www.stata.com/meeting/baltimore17/slides/Baltimore17_Gallis.pdf.

Dickinson, L. M., Beaty, B., Fox, C., Pace, W., Dickinson, W. P., Emsermann, C., & Kempe, A. (2015). Pragmatic cluster randomized trials using covariate constrained randomization: A method for practice-based research networks (PBRNs). The Journal of the American Board of Family Medicine, 28(5), 663-672.

Examples

```
## Not run:
Analysis_result <- cptest(outcome = Dickinson_outcome$outcome,
                          clustername = Dickinson_outcome$county,
                          z = data.frame(Dickinson_outcome[ , c("location", "inciis",
                              "uptodateonimmunizations", "hispanic", "incomecat")]),
                          cspacedatname = "dickinson_constrained.csv",
                          outcometype = "binary",
                          categorical = c("location","incomecat"))

## End(Not run)
```

cvcrand *cvcrand: a package for efficient design and analysis of cluster randomized trials*

Description

cvcrand: A Package for Covariate-constrained Randomization and the Clustered Permutation Test for Cluster Randomized Trials

cvcrand functions

The cvrall function performs constrained randomization on cluster randomized trials (CRTs) The cvrcov function performs covariate-constrained randomization on cluster randomized trials (CRTs) The cptest function performs permutation test on the outcome of cluster randomized trials (CRTs)

cvrall *Covariate-constrained randomization for cluster randomized trials*

Description

`cvrall` performs constrained randomization for cluster randomized trials (CRTs), especially suited for CRTs with a small number of clusters. In constrained randomization, a randomization scheme is randomly sampled from a subset of all possible randomization schemes based on the value of a balancing criterion called a balance score. The `cvrall` function has two choices of "l1" and "l2" metrics for balance score.

The `cvrall` function enumerates all randomization schemes or chooses the unique ones among some simulated randomization schemes as specified by the user. Some cluster-level continuous or categorical covariates are then used to calculate the balance scores for the unique schemes. A subset of the randomization schemes is chosen based on a user-specified cutoff at a certain quantile of the distribution of the balance scores or based on a fixed number of schemes with the smallest balance scores. The `cvrall` function treats the subset as the constrained space of randomization schemes and samples one scheme from the constrained space as the final chosen scheme.

Usage

```
cvrall(
  clustername = NULL,
  x,
  categorical = NULL,
  weights = NULL,
  ntotal_cluster,
  ntrt_cluster,
  cutoff = 0.1,
  numschemes = NULL,
  size = 50000,
  stratify = NULL,
  seed = NULL,
  balancemetric = "l2",
  nosim = FALSE,
  savedata = NULL,
  bhist = TRUE,
  check_validity = FALSE,
  samearmhi = 0.75,
  samearmlo = 0.25
)
```

Arguments

`clustername` a vector specifying the identification variable of the cluster. If no cluster identification variable is specified, the default is to label the clusters based on the order in which they appear.

`x` a data frame specifying the values of cluster-level covariates to balance. With K covariates and n clusters, it will be dimension of `n` by `K`.

`categorical` a vector specifying categorical (including binary) variables. This can be names of the columns or number indexes of columns, but cannot be both. Suppose there are `p` categories for a categorical variable, `cvcrand` function creates `p-1` dummy variables and drops the reference level if the variable is specified as a factor. Otherwise, the first level in the alphanumerical order will be dropped. The results are sensitive to which level is excluded. If the user wants to specify a different level to drop for a `p`-level categorical variable, the user can create `p-1` dummy variables and these can instead be supplied as covariates to the `cvcrand` function. Then, the user needs to specify the dummy variables created to be `categorical` when running `cvcrand`. In addition, the user could also set the variable as a factor with the specific reference level. If the `weights` option is used, the weights for a categorical variable will be replicated on all the dummy variables created.

`weights` a vector of user-specified weights for the covariates to calculate the balance score. The weight for a categorical variable will be replicated for the dummy variables created. Note that the `weights` option can be used to conduct stratification on variables. For example, a variable with a relatively large weight like `1000` and all other variables with a weight of `1` will cause the randomization scheme chosen to be stratified by the variable with the large weight, assuming a low `cutoff` value is specified.

`ntotal_cluster` the total number of clusters to be randomized. It must be a positive integer and equal to the number of rows of the data.

`ntrt_cluster` the number of clusters that the researcher wants to assign to the treatment arm. It must be a positive integer less than the total number of clusters.

`cutoff` quantile cutoff of the distribution of balance score below which a randomization scheme is sampled. Its default is `0.1`, and it must be between 0 and 1. The `cutoff` option is overridden by the `numschemes` option.

`numschemes` number of randomization schemes to form the constrained space for the final randomization scheme to be selected. If specified, it overrides the option `cutoff` and the program will randomly sample the final randomization scheme from the constrained space of randomization schemes with the `numschemes` smallest balance scores. It must be a positive integer.

`size` number of randomization schemes to simulate if the number of all possible randomization schemes is over `size`. Its default is `50,000`, and must be a positive integer. It can be overridden by the `nosim` option.

`stratify` categorical variables on which to stratify the randomization. It overrides the option `weights` when specified. This list of categorical variables should be a subset of the `categorical` option if specified.

`seed` seed for simulation and random sampling. It is needed so that the randomization can be replicated. Its default is `12345`.

`balancemetric` balance metric to use. Its choices are `"l1"` and `"l2"`. The default is `"l2"`.

`nosim` if `TRUE`, it overrides the default procedure of simulating when the number of all possible randomization schemes is over the `size`, and the program enumerates all randomization schemes. Note: this may consume a lot of memory and cause R to crash

`savedata` saves the data set of the constrained randomization space in a csv file if specified by `savedata`. The first column of the csv file is an indicator variable of the final randomization scheme in the constrained space. The constrained randomization space will be needed for analysis after the cluster randomized trial is completed if the clustered permutation test is used.

`bhist` if `TRUE` of the default value, it produces the histogram of all balance scores with a red line on the graph indicating the selected cutoff.

`check_validity` boolean argument to check the randomization validity or not

`samearmhi` clusters assigned to the same arm as least this often are displayed. The default is `0.75`.

`samearmlo` clusters assigned to the same arm at most this often are displayed. The default is `0.25`.

Value

`balancemetric` the balance metric used

`allocation` the allocation scheme from constrained randomization

`bscores` the histogram of the balance score with respect to the balance metric

`assignment_message` the statement about how many clusters to be randomized to the intervention and the control arms respectively

`scheme_message` the statement about how to get the whole randomization space to use in constrained randomization

`cutoff_message` the statement about the cutoff in the constrained space

`choice_message` the statement about the selected scheme from constrained randomization

`data_CR` the data frame containing the allocation scheme, the `clustername`, and the original data frame of covariates

`baseline_table` the descriptive statistics for all the variables by the two arms from the selected scheme

`cluster_coincidence` cluster coincidence matrix

`cluster_coin_des` cluster coincidence descriptive

`clusters_always_pair` pairs of clusters always allocated to the same arm.

`clusters_always_not_pair` pairs of clusters always allocated to different arms.

`clusters_high_pair` pairs of clusters randomized to the same arm at least `samearmhi` of the time.

`clusters_low_pair` pairs of clusters randomized to the same arm at most `samearmlo` of the time.

`overall_allocations` frequency of acceptable overall allocations.

Author(s)

Hengshi Yu <hengshi@umich.edu>, Fan Li <fan.f.li@yale.edu>, John A. Gallis <john.gallis@duke.edu>, Elizabeth L. Turner <liz.turner@duke.edu>

References

Raab, G.M. and Butcher, I., 2001. Balance in cluster randomized trials. Statistics in medicine, 20(3), pp.351-365.

Li, F., Lokhnygina, Y., Murray, D.M., Heagerty, P.J. and DeLong, E.R., 2016. An evaluation of constrained randomization for the design and analysis of group randomized trials. Statistics in medicine, 35(10), pp.1565-1579.

Li, F., Turner, E. L., Heagerty, P. J., Murray, D. M., Vollmer, W. M., & DeLong, E. R. (2017). An evaluation of constrained randomization for the design and analysis of group randomized trials with binary outcomes. Statistics in medicine, 36(24), 3791-3806.

Gallis, J.A., Li, F., Yu, H. and Turner, E.L., 2018. cvcrand and cptest: Commands for efficient design and analysis of cluster randomized trials using constrained randomization and permutation tests. The Stata Journal, 18(2), pp.357-378.

Dickinson, L. M., Beaty, B., Fox, C., Pace, W., Dickinson, W. P., Emsermann, C., & Kempe, A. (2015). Pragmatic cluster randomized trials using covariate constrained randomization: A method for practice-based research networks (PBRNs). The Journal of the American Board of Family Medicine, 28(5), 663-672.

Bailey, R.A. and Rowley, C.A., 1987. Valid randomization. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 410(1838), pp.105-124.

Examples

```
# cvrall examples

Design_result <- cvrall(clustername = Dickinson_design$county,
                         balancemetric = "l2",
                         x = data.frame(Dickinson_design[ , c("location", "inciis",
                              "uptodateonimmunizations", "hispanic", "incomecat")]),
                         ntotal_cluster = 16,
                         ntrt_cluster = 8,
                         categorical = c("location", "incomecat"),
                         ###### Option to save the constrained space ######
                         # savedata = "dickinson_constrained.csv",
                         bhist = TRUE,
                         cutoff = 0.1,
                         seed = 12345,
                         check_validity = TRUE)

# cvrall example with weights specified

Design_result <- cvrall(clustername = Dickinson_design$county,
                         balancemetric = "l2",
                         x = data.frame(Dickinson_design[ , c("location", "inciis",
                             "uptodateonimmunizations", "hispanic", "incomecat")]),
                         ntotal_cluster = 16,
                         ntrt_cluster = 8,
                         categorical = c("location", "incomecat"),
                         weights = c(1, 1, 1, 1, 1),
                         cutoff = 0.1,
                         seed = 12345,
                         check_validity = TRUE)

# Stratification on location, with constrained
# randomization on other specified covariates

 Design_stratified_result <- cvrall(clustername = Dickinson_design$county,
                                     balancemetric = "l2",
                               x = data.frame(Dickinson_design[ , c("location", "inciis",
                                  "uptodateonimmunizations", "hispanic", "incomecat")]),
                                     ntotal_cluster = 16,
                                     ntrt_cluster = 8,
                                     categorical = c("location", "incomecat"),
                                     weights = c(1000, 1, 1, 1, 1),
                                     cutoff = 0.1,
                                     seed = 12345)

 # An alternative and equivalent way to stratify on location

 Design_stratified_result <- cvrall(clustername = Dickinson_design$county,
                                     balancemetric = "l2",
                               x = data.frame(Dickinson_design[ , c("location", "inciis",
                                  "uptodateonimmunizations", "hispanic", "incomecat")]),
                                     ntotal_cluster = 16,
                                     ntrt_cluster = 8,
                                     categorical = c("location", "incomecat"),
                                     stratify = "location",
                                     cutoff = 0.1,
                                     seed = 12345)

 # Stratification on income category
 # Two of the income categories contain an odd number of clusters
 # Stratification is not strictly possible

 Design_stratified_inc_result <- cvrall(clustername = Dickinson_design$county,
                                         balancemetric = "l2",
                               x = data.frame(Dickinson_design[ , c("location", "inciis",
                                  "uptodateonimmunizations", "hispanic", "incomecat")]),
                                         ntotal_cluster = 16,
                                         ntrt_cluster = 8,
                                         categorical = c("location", "incomecat"),
                                         stratify = "incomecat",
                                         cutoff = 0.1,
                                         seed = 12345)
```

cvrcov *Covariate-by-covariate constrained randomization for cluster randomized trials*

Description

cvrcov performs covariate-by-covariate constrained randomization for cluster randomized trials (CRTs), especially suited for CRTs with a small number of clusters. In constrained randomization, a randomization scheme is randomly sampled from a subset of all possible randomization schemes based on the constraints on each covariate.

The cvrcov function enumerates all randomization schemes or simulates a fixed size of unique randomization schemes as specified by the user. A subset of the randomization schemes is chosen based on user-specified covariate-by-covariate constraints. cvrcov treats the subset as the constrained space of randomization schemes and samples one scheme from the constrained space as the final chosen scheme.

Usage

```
cvrcov(
  clustername = NULL,
  x,
  categorical = NULL,
  constraints,
  ntotal_cluster,
  ntrt_cluster,
  size = 50000,
  seed = NULL,
  nosim = FALSE,
  savedata = NULL,
  check_validity = FALSE,
  samearmhi = 0.75,
  samearmlo = 0.25
)
```

Arguments

clustername — a vector specifying the identification variable of the cluster. If no cluster identification variable is specified, the default is to label the clusters based on the order in which they appear.

x — a data frame specifying the values of cluster-level covariates to balance. With K covariates and n clusters, it will be dimension of n by K.

categorical — a vector specifying categorical (including binary) variables. This can be names of the columns or number indexes of columns, but cannot be both. Suppose there are p categories for a categorical variable, cvcrand function creates p-1 dummy variables and drops the reference level if the variable is specified as a factor. Otherwise, the first level in the alphanumerical order will be dropped. The results are sensitive to which level is excluded. If the user wants to specify a different level to drop for a p-level categorical variable, the user can create p-1 dummy variables and these can instead be supplied as covariates to the cvcrand function. Then, the user needs to specify the dummy variables created to be categorical when running cvcrand. In addition, the user could also set the variable as a factor with the specific reference level. If the weights option is used, the weights for a categorical variable will be replicated on all the dummy variables created.

`constraints` a vector of user-specified constraints for all covariates. `"any"` means no constraints. If not `"any"`, the first character letter of `"m"` denotes absolute mean difference, and `"s"` means absolute sum difference. If the second character is `"f"`, the previous metric is constrained to be smaller or equal to the fraction with the number followed of the overall mean for `"m"` or mean arm total for `"s"`. If not `"f"` at the second character, the metric is just constrained to be smaller or equal to the value following letter(s).

`ntotal_cluster` the total number of clusters to be randomized. It must be a positive integer and equal to the number of rows of the data.

`ntrt_cluster` the number of clusters that the researcher wants to assign to the treatment arm. It must be a positive integer less than the total number of clusters.

`size` number of randomization schemes to simulate if the number of all possible randomization schemes is over `size`. Its default is `50,000`, and must be a positive integer. It can be overriden by the `nosim` option.

`seed` seed for simulation and random sampling. It is needed so that the randomization can be replicated. Its default is `12345`.

`nosim` if TRUE, it overrides the default procedure of simulating when the number of all possible randomization schemes is over `size`, and the program enumerates all randomization schemes. Note: this may consume a lot of memory and cause R to crash

`savedata` saves the data set of the constrained randomization space in a csv file if specified by `savedata`. The first column of the csv file is an indicator variable of the final randomization scheme in the constrained space. The constrained randomization space will be needed for analysis after the cluster randomized trial is completed if the clustered permutation test is used.

`check_validity` boolean argument to check the randomization validity or not

`samearmhi` clusters assigned to the same arm as least this often are displayed. The default is `0.75`.

`samearmlo` clusters assigned to the same arm at most this often are displayed. The default is `0.25`.

Value

`allocation` the allocation scheme from constrained randomization

`assignment_message` the statement about how many clusters to be randomized to the intervention and the control arms respectively

`scheme_message` the statement about how to get the whole randomization space to use in constrained randomization

`data_CR` the data frame containing the allocation scheme, the `clustername`, and the original data frame of covariates

`baseline_table` the descriptive statistics for all the variables by the two arms from the selected scheme `cluster_coincidence` cluster coincidence matrix

`cluster_coin_des` cluster coincidence descriptive

`clusters_always_pair` pairs of clusters always allocated to the same arm.

`clusters_always_not_pair` pairs of clusters always allocated to different arms.

`clusters_high_pair` pairs of clusters randomized to the same arm at least `samearmhi` of the time.

`clusters_low_pair` pairs of clusters randomized to the same arm at most `samearmlo` of the time.

`overall_allocations` frequency of acceptable overall allocations.

`overall_summary` summary of covariates with constraints in the constrained space

Author(s)

Hengshi Yu <hengshi@umich.edu>, Fan Li <fan.f.li@yale.edu>, John A. Gallis <john.gallis@duke.edu>, Elizabeth L. Turner <liz.turner@duke.edu>

References

Raab, G.M. and Butcher, I., 2001. Balance in cluster randomized trials. Statistics in medicine, 20(3), pp.351-365.

Li, F., Lokhnygina, Y., Murray, D.M., Heagerty, P.J. and DeLong, E.R., 2016. An evaluation of constrained randomization for the design and analysis of group randomized trials. Statistics in medicine, 35(10), pp.1565-1579.

Li, F., Turner, E. L., Heagerty, P. J., Murray, D. M., Vollmer, W. M., & DeLong, E. R. (2017). An evaluation of constrained randomization for the design and analysis of group randomized trials with binary outcomes. Statistics in medicine, 36(24), 3791-3806.

Gallis, J.A., Li, F., Yu, H. and Turner, E.L., 2018. cvcrand and cptest: Commands for efficient design and analysis of cluster randomized trials using constrained randomization and permutation tests. The Stata Journal, 18(2), pp.357-378.

Dickinson, L. M., Beaty, B., Fox, C., Pace, W., Dickinson, W. P., Emsermann, C., & Kempe, A. (2015). Pragmatic cluster randomized trials using covariate constrained randomization: A method for practice-based research networks (PBRNs). The Journal of the American Board of Family Medicine, 28(5), 663-672.

Bailey, R.A. and Rowley, C.A., 1987. Valid randomization. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 410(1838), pp.105-124.

Greene, E.J., 2017. A SAS macro for covariate-constrained randomization of general cluster-randomized and unstratified designs. Journal of statistical software, 77(CS1).

Examples

```
# cvrcov example

Dickinson_design_numeric <- Dickinson_design
Dickinson_design_numeric$location = (Dickinson_design$location == "Rural") * 1
Design_cov_result <- cvrcov(clustername = Dickinson_design_numeric$county,
                        x = data.frame(Dickinson_design_numeric[ , c("location", "inciis",
                                "uptodateonimmunizations", "hispanic", "income")]),
                            ntotal_cluster = 16,
                            ntrt_cluster = 8,
                            constraints = c("s5", "mf.5", "any", "mf0.2", "mf0.2"),
                            categorical = c("location"),
                            ###### Option to save the constrained space ######
                            # savedata = "dickinson_cov_constrained.csv",
                            seed = 12345,
                            check_validity = TRUE)
```

`Dickinson_design` *Raw county-level variables for study 1 in Dickinson et al (2015)*

Description

Two approaches (interventions) were compared for increasing the "up-to-date" immunization rate in 19- to 35-month-old children. 16 counties in Colorado 1:1 were randomized 1:1 to either a population-based approach or a practice-based approach. Ahead of randomization, several county-level variables were collected, and a subset of them are used for covariate constrained randomization. The continuous variable of average income is categorized into tertiles to illustrate the use of cvcrand on multi-category variables. And the percentage in CIIS variable is truncated at 100

Format

A data frame with 16 rows and 11 variables:

county the identification for the county

location urban or rural

inciis percentage of children aged 19-35 months in the Colorado Immunization Information System (CIIS)

numberofchildrenages1935months number of children aged 19-35 months

uptodateonimmunizations percentage of children already up-to-date on their immunization

africanamerican percentage of population that is African American

hispanic percentage of population that is Hispanic

income average income

incomecat average income categorized into tertiles

pediatricpracticetofamilymedicin pediatric practice-to-family medicine practice ratio

communityhealthcenters number of community health centers

Source

https://www.jabfm.org/content/28/5/663/tab-figures-data

References

Dickinson, L. M., B. Beaty, C. Fox, W. Pace, W. P. Dickinson, C. Emsermann, and A. Kempe (2015): Pragmatic cluster randomized trials using covariate constrained randomization: A method for practice-based research networks (PBRNs). The Journal of the American Board of Family Medicine 28(5): 663-672

`Dickinson_outcome` *Simulated individual-level binary outcome and baseline variables for study 1 in Dickinson et al (2015)*

Description

At the end of the study, the researchers will have ascertained the outcome in the 16 clusters. Suppose that the researchers were able to assess 300 children in each cluster. We simulated correlated outcome data at the individual level using a generalized linear mixed model (GLMM) to induce correlation by including a random effect. The intracluster correlation (ICC) was set to be 0.01, using the latent response definition provided in Eldrige et al. (2009). This is a reasonable value of the ICC for population health studies (Hannan et al. 1994). We simulated one data set, with the outcome data dependent on the county-level covariates used in the constrained randomization design and a positive treatment effect so that the practice-based intervention increases up-to-date immunization rates more than the community-based intervention. For each individual child, the outcome is equal to 1 if he or she is up-to-date on immunizations and 0 otherwise.

Note that we still categorize the continuous variable of average income to illustrate the use of cvcrand on multi-category variables, and we truncated the percentage in CIIS variable at 100

Format

A data frame with 4800 rows and 7 variables:

county the identification for the county

location urban or rural

inciis percentage of children ages 19-35 months in the Colorado Immunization Information System (CIIS)

uptodateonimmunizations percentage of children already up-to-date on their immunization

hispanic percentage of population that is Hispanic

incomecat average income categorized into tertiles

outcome the status of being up-to-date on immunizations

References

Dickinson, L. M., B. Beaty, C. Fox, W. Pace, W. P. Dickinson, C. Emsermann, and A. Kempe (2015): Pragmatic cluster randomized trials using covariate constrained randomization: A method for practice-based research networks (PBRNs). The Journal of the American Board of Family Medicine 28(5): 663-672

Eldridge, S. M., Ukoumunne, O. C., & Carlin, J. B. (2009). The Intra Cluster Correlation Coefficient in Cluster Randomized Trials: A Review of Definitions. International Statistical Review, 77(3), 378-394.

Hannan, P. J., Murray, D. M., Jacobs Jr, D. R., & McGovern, P. G. (1994). Parameters to aid in the design and analysis of community trials: intraclass correlations from the Minnesota Heart Health Program. Epidemiology, 88-95. ISO 690

Index