Package 'mvGPS'

October 17, 2021

Title Causal Inference using Multivariate Generalized Propensity Score

Version 1.2.1

Description Methods for estimating and utilizing the multivariate generalized propensity score (mvGPS) for multiple continuous exposures described in Williams, J.R, and Crespi, C.M. (2020) <arxiv:2008.13767>. The methods allow estimation of a dose-response surface relating the joint distribution of multiple continuous exposure variables to an outcome. Weights are constructed assuming a multivariate normal density for the marginal and conditional distribution of exposures given a set of confounders. Confounders can be different for different exposure variables. The weights are designed to achieve balance across all exposure dimensions and can be used to estimate dose-response surfaces.

License MIT + file LICENSE

Encoding UTF-8

LazyData false

RoxygenNote 7.1.0

RdMacros Rdpack

VignetteBuilder knitr

Depends R (>= 3.6)

Imports Rdpack, MASS, WeightIt, cobalt, matrixNormal, geometry, sp, gbm, CBPS

BugReports https://github.com/williazo/mvGPS/issues

URL https://github.com/williazo/mvGPS

Suggests testthat, knitr, dagitty, ggdag, dplyr, rmarkdown, ggplot2

NeedsCompilation no

Author Justin Williams [aut, cre] (<https://orcid.org/0000-0002-5045-2764>)

Maintainer Justin Williams <williazo@ucla.edu>

Repository CRAN

Date/Publication 2021-10-16 23:30:10 UTC

R topics documented:

`bal` *Construct Covariate Balance Statistics for Models with Multivariate Exposure*

Description

Assessing balance between exposure(s) and confounders is key when performing causal analysis using propensity scores. We provide a list of several models to generate weights to use in causal inference for multivariate exposures, and test the balancing property of these weights using weighted Pearson correlations. In addition, returns the effective sample size.

Usage

```
bal(
  model_list,
  D,
  C,
  common = FALSE,
  trim_w = FALSE,
  trim_quantile = 0.99,
  all_uni = TRUE,
  ...
)
```

Arguments

`model_list`	character string identifying which methods to use when constructing weights. See details for a list of available models
`D`	numeric matrix of dimension n by m designating values of the exposures
`C`	either a list of numeric matrices of length m of dimension n by p_j designating values of the confounders for each exposure value or if `common` is TRUE a single matrix of of dimension n by p that represents common confounders for all exposures.
`common`	logical indicator for whether C is a single matrix of common confounders for all exposures. default is FALSE meaning C must be specified as list of confounders of length m.
`trim_w`	logical indicator for whether to trim weights. default is FALSE

trim_quantile	numeric scalar used to specify the upper quantile to trim weights if applicable. default is 0.99
all_uni	logical indicator. If TRUE then all univariate models specified in model_list will be estimated for each exposure. If FALSE will only estimate weights for the first exposure
...	additional arguments to pass to weightit function if specifying one of these models in the model_list

Details

When using propensity score methods for causal inference it is crucial to check the balancing property of the covariates and exposure(s). To do this in the multivariate case we first use a weight generating method from the available list shown below.

Methods Available:

- "mvGPS": Multivariate generalized propensity score using Gaussian densities
- "entropy": Estimating weights using entropy loss function without specifying propensity score (Tübbicke 2020)
- "CBPS": Covariate balancing propensity score for continuous treatments which adds balance penalty while solving for propensity score parameters (Fong et al. 2018)
- "PS": Generalized propensity score estimated using univariate Gaussian densities
- "GBM": Gradient boosting to estimate the mean function of the propensity score, but still maintains Gaussian distributional assumptions (Zhu et al. 2015)

Note that only the mvGPS method is multivariate and all others are strictly univariate. For univariate methods weights are estimated for each exposure separately using the weightit function given the confounders for that exposure in C when all_uni=TRUE. To estimate weights for only the first exposure set all_uni=FALSE.

It is also important to note that the weights for each method can be trimmed at the desired quantile by setting trim_w=TRUE and setting trim_quantile in \[0.5, 1\]. Trimming is done at both the upper and lower bounds. For further details see mvGPS on how trimming is performed.

Balance Metrics:

In this package we include three key balancing metrics to summarize balance across all of the exposures.

- Euclidean distance
- Maximum absolute correlation
- Average absolute correlation

Euclidean distance is calculated using the origin point as reference, e.g. for m=2 exposures the reference point is \[0, 0\]. In this way we are calculating how far the observed set of correlation points are from perfect balance.

Maximum absolute correlation reports the largest single imbalance between the exposures and the set of confounders. It is often a key diagnostic as even a single confounder that is sufficiently out of balance can reduce performance.

Average absolute correlation is the sum of the exposure-confounder correlations. This metric summarizes how well, on average, the entire set of exposures is balanced.

Effective Sample Size:

Effective sample size, ESS, is defined as

$$ESS = (\Sigma_i w_i)^2 / \Sigma_i w_i^2,$$

where w_i are the estimated weights for a particular method (Kish 1965). Note that when $w = 1$ for all units that the ESS is equal to the sample size n. ESS decreases when there are extreme weights or high variability in the weights.

Value

- `W`: list of weights generated for each model
- `cor_list`: list of weighted Pearson correlation coefficients for all confounders specified
- `bal_metrics`: data.frame with the Euclidean distance, maximum absolute correlation, and average absolute correlation by method
- `ess`: effective sample size for each of the methods used to generate weights
- `models`: vector of models used

References

Fong C, Hazlett C, Imai K (2018). "Covariate balancing propensity score for a continuous treatment: application to the efficacy of political advertisements." *Annals of Applied Statistics*, **In-Press**.

Kish L (1965). *Survey Sampling*. John Wiley \& Sons, New York.

Tübbicke S (2020). "Entropy Balancing for Continuous Treatments." *arXiv e-prints*. 2001.06281.

Zhu Y, Coffman DL, Ghosh D (2015). "A boosting algorithm for estimating generalized propensity scores with continuous treatments." *Journal of Causal Inference*, **3**(1), 25-40.

Examples

```
#simulating data
sim_dt <- gen_D(method="u", n=150, rho_cond=0.2, s_d1_cond=2, s_d2_cond=2,
k=3, C_mu=rep(0, 3), C_cov=0.1, C_var=1, d1_beta=c(0.5, 1, 0),
d2_beta=c(0, 0.3, 0.75), seed=06112020)
D <- sim_dt$D
C <- sim_dt$C

#generating weights using mvGPS and potential univariate alternatives
require(WeightIt)
bal_sim <- bal(model_list=c("mvGPS", "entropy", "CBPS", "PS", "GBM"), D,
C=list(C[, 1:2], C[, 2:3]))

#overall summary statistics
bal_sim$bal_metrics

#effective sample sizes
bal_sim$ess
```

```
#we can also trim weights for all methods
bal_sim_trim <- bal(model_list=c("mvGPS", "entropy", "CBPS", "PS", "GBM"), D,
C=list(C[, 1:2], C[, 2:3]), trim_w=TRUE, trim_quantile=0.9, p.mean=0.5)
#note that in this case we can also pass additional arguments using in
#WeighIt package for entropy, CBPS, PS, and GBM such as specifying the p.mean

#can check to ensure all the weights have been properly trimmed at upper and
#lower bound
all.equal(unname(unlist(lapply(bal_sim$W, quantile, 0.99))),
unname(unlist(lapply(bal_sim_trim$W, max))))
all.equal(unname(unlist(lapply(bal_sim$W, quantile, 1-0.99))),
unname(unlist(lapply(bal_sim_trim$W, min))))
```

---

gen_D *Generate Bivariate Multivariate Exposure*

Description

Generate exposure from a bivariate normal distribution confounded by a set of variables C=\(C1, C2).

Usage

```
gen_D(
  method,
  n,
  rho_cond,
  s_d1_cond,
  s_d2_cond,
  k,
  C_mu,
  C_cov,
  C_var,
  C_sigma = NULL,
  d1_beta,
  d2_beta,
  seed = NULL
)
```

Arguments

method	character value identifying which method to use when generating bivariate exposure. Options include "matrix_normal", "uni_cond", and "vector_normal". See details for a brief explanation of each method. uni_cond is fastest
n	integer value total number of units
rho_cond	scalar value identifying conditional correlation of exposures given covariates between \[0, 1\]

s_d1_cond	scalar value for conditional standard deviation of D1
s_d2_cond	scalar value for conditional standard deviation of D2
k	integer value determining number of covariates to generate in C.
C_mu	numeric vector of mean values for covariates. Must be same length as k
C_cov	scalar value representing constant correlation between covariates
C_var	scalar value representing constant variance of covariates
C_sigma	numeric matrix representing the covariance matrix of covariates. Default is NULL and will use C_var and C_var otherwise.
d1_beta	numeric vector of length k defining the mean of D1 with respect to the covariates
d2_beta	numeric vector of length k defining the mean of D2 with respect to the covariates
seed	integer value setting the seed of random generator to produce repeatable results. set to NULL by default

Details

Generating Confounders:

We assume that there are a total of k confounders that are generated from a multivariate normal distribution with equicorrelation covariance, i.e.,

$$\Sigma_C = \phi(\mathbf{1}\mathbf{1}^T - \mathbf{I}) + \mathbf{I}\sigma_C^2,$$

where $\mathbf{1}$ is the column vector with all entries equal to 1, $\mathbf{I}$ is the identity matrix, σ_C^2 is a constant standard deviation for all confounders, and ϕ is the covariance of any two confounders. Therefore, our random confounders C follow the distribution

$$\mathbf{C} \sim N_k(\boldsymbol{\mu}_C, \Sigma_C).$$

We draw a total of n samples from this multivariate normal distribution using mvrnorm.

Generating Bivariate Exposure:

The first step when generating the bivariate exposure is to specify the effects of the confounders C. We control this for each exposure value using the arguments d1_beta and d2_beta such that

$$E[D_1 \mid \mathbf{C}] = \boldsymbol{\beta}_{D1}^T \mathbf{C}$$

and

$$E[D_2 \mid \mathbf{C}] = \boldsymbol{\beta}_{D2}^T \mathbf{C}$$

.

Note that by specifying d1_beta and d2_beta separately that the user can control the amount of overlap in the confounders for each exposure, and how many of the variables in C are truly related to the exposures. For instance to have the exposure have identical confounding effects d1_beta=d2_beta, and they have separate confounding if there are zero non-zero elements in common between d1_beta and d2_beta.

To generate the bivariate conditional distribution of exposures given the set of confounders C we have the following three methods:

- "matrix_normal"

- "uni_cond"
- "vector_normal"

"matrix_normal" uses the function rmatnorm to generate all n samples as

$$\mathbf{D} \mid \mathbf{C} \sim N_{n\times 2}(\boldsymbol{\beta}\mathbf{C}, \mathbf{I}_n, \Omega)$$

where $\boldsymbol{\beta}$ is a column vector containing $\boldsymbol{\beta}_{D1}^T$ and $\boldsymbol{\beta}_{D2}^T$, and Ω is the conditional covariance matrix.

"vector_normal" simply vectorizes the matrix_normal method above to generate a vector of length $n \times 2$.

"uni_cond" specifies the bivariate exposure using univariate conditional factorization, which in the case of bivariate normal results in two univariate normal expressions.

In general, we suggest using the univariate conditional, "uni_cond", method when generating exposures as it is substantially faster than both the matrix normal and vector normal approaches.

Note that the options use regular expression matching and can be specified uniquely using either "m", "u", or "v".

Marginal Covariance of Exposures:

As described above the exposures are drawn conditional on the set C, so the marginal covariance of exposures is defined as

$$\Sigma_D = \boldsymbol{\beta}\Sigma_C\boldsymbol{\beta}^T + \Omega.$$

In our function we return the true marginal covariance Σ_D as well as the true marginal correlation ρ_D.

Value

- D: nx2 numeric matrix of the sample values for the exposures given the set C
- C: nxk numeric matrix of the sampled values for the confounding set C
- D_Sigma: 2x2 numeric matrix of the true marginal covariance of exposures
- rho: numeric scalar representing the true marginal correlation of exposures

Examples

```
#generate bivariate exposures. D1 confounded by C1 and C2. D2 by C2 and C3
#uses univariate conditional normal to draw samples
sim_dt <- gen_D(method="u", n=200, rho_cond=0.2, s_d1_cond=2, s_d2_cond=2, k=3,
C_mu=rep(0, 3), C_cov=0.1, C_var=1, d1_beta=c(0.5, 1, 0), d2_beta=c(0, 0.3, 0.75), seed=06112020)
D <- sim_dt$D
C <- sim_dt$C

#observed correlation should be close to true marginal value
cor(D); sim_dt$rho


#Use vector normal method instead of univariate method to draw samples
sim_dt <- gen_D(method="v", n=200, rho_cond=0.2, s_d1_cond=2, s_d2_cond=2, k=3,
C_mu=rep(0, 3), C_cov=0.1, C_var=1, d1_beta=c(0.5, 1, 0), d2_beta=c(0, 0.3, 0.75), seed=06112020)
```

`hull_sample` *Sample Points Along a Convex Hull*

Description

To define a proper estimable region with multivariate exposure we construct a convex hull of the data in order to maintain the positivity identifying assumption. We also provide options to create trimmed versions of the convex hull to further restrict to high density regions in multidimensional space.

Usage

```
hull_sample(
  X,
  num_grid_pts = 500,
  grid_type = "regular",
  trim_hull = FALSE,
  trim_quantile = NULL
)
```

Arguments

`X`	numeric matrix of n by m dimensions. Each row corresponds to a point in m-dimensional space.
`num_grid_pts`	integer scalar denoting the number of parameters to search for over the convex hull. Default is 500.
`grid_type`	character value indicating the type of grid to sample from the convex hull from `spsample`
`trim_hull`	logical indicator of whether to restrict convex hull. Default is FALSE
`trim_quantile`	numeric scalar between \[0.5, 1\] representing the quantile value to trim the convex hull. Only used if trim_hull is set to TRUE.

Details

Assume that X is an $n \times m$ matrix representing the multivariate exposure of interest. We can define the convex hull of these observations as $\mathbf{H}$. There are two distinct processes for defining $\mathbf{H}$ depending on whether $m = 2$ or $m > 2$.

If $m = 2$, we use the `chull` function to define the vertices of the convex hull. The algorithm implemented is described in Eddy (1977).

If $m > 2$, we use the `convhulln` function. This algorithm for obtaining the convex hull in m-dimensional space uses Qhull described in Barber et al. (1996). Currently this function returns only the vertex set `hpts_vs` without the grid sample points. There are options to visualize the convex hull when $m = 3$ using triangular facets, but there are no implementable solutions to sample along convex hulls in higher dimensions.

To restrict the convex hull to higher density regions of the exposure we can apply trimming. To apply trimming set `trim_hull=TRUE` and specify `trim_quantile=q` where `q` must be in \[0.5, 1\]. Along each exposure dimension we then calculate the upper and lower bounds using the `quantile` function, i.e., `quantile(q)` and `quantile(1-q)`. Any observations that have a value above or below these sample quantiles is excluded. The remaining observations that fall completely within the sample quantiles across all dimensions are used to estimate the convex hull. We return `X` that represents the observations used. If `trim_hull=FALSE`, then `X` is unchanged. However, if trimming is applied then `X` contains only the remaining observations after trimming.

Value

- `hpts_vs`: vertices of the convex hull in m-dimensional space
- `grid_pts`: values of grid points sampled over the corresponding convex hull
- `X`: data used to generate convex hull which may be trimmed

References

Barber CB, Dobkin DP, Huhdanpaa H (1996). "The quickhull algorithm for convex hulls." *ACM Transactions on Mathematical Software (TOMS)*, **22**(4), 469-483.

Eddy WF (1977). "A new convex hull algorithm for planar sets." *ACM Transactions on Mathematical Software (TOMS)*, **3**(4), 398-403.

Examples

```
#generating exposure with m=3
D <- matrix(unlist(lapply(seq_len(3), function(m) rnorm(100))), nrow=100)

#first using only the first two variables we can return hpts_vs and grid_pts
D_hull <- hull_sample(D[, 1:2])

#when m>2 we only return hpts_vs and grid_pts is NULL
D_hull_large <- hull_sample(D)
is.null(D_hull_large$grid_pts)

#we can also apply trimming to the convex hull and return this reduced matrix
D_hull_trim <- hull_sample(D[, 1:2], trim_hull=TRUE, trim_quantile=0.95)
dim(D_hull$X)
dim(D_hull_trim$X)

#alternatively, we can also define the number of points to sample from for grid_pts
small_grid <- hull_sample(D[, 1:2], num_grid_pts=100)
length(D_hull$grid_pts)
length(small_grid$grid_pts)
```

mvGPS *Multivariate Generalized Propensity Score*

Description

Estimate propensity scores for multivariate continuous exposure by assuming joint normal conditional densities. Simultaneously estimate stabilized inverse probability of treatment weights (IPTW) using joint normal density for marginal distribution of exposure.

Usage

```
mvGPS(D, C, common = FALSE, trim_w = FALSE, trim_quantile = 0.99)
```

Arguments

D	numeric matrix of dimension n by m designating values of the exposures
C	either a list of numeric matrices of length m of dimension n by p_j designating values of the confounders for each exposure value or if common is TRUE a single matrix of of dimension n by p that represents common confounders for all exposures.
common	logical indicator for whether C is a single matrix of common confounders for all exposures. default is FALSE meaning C must be specified as list of confounders of length m.
trim_w	logical indicator for whether to trim weights. default is FALSE
trim_quantile	numeric scalar used to specify the upper quantile to trim weights if applicable. default is 0.99

Details

Generalized propensity scores (GPS) were proposed by Hirano and Imbens (2004) and Imai and Van Dyk (2004) to extend propensity scores to handle continuous exposures. The GPS is constructed using the conditional density of the exposure given a set of confounders. These original methods and the subsequent literature have focused on the case of a single continuous exposure where the GPS could be estimated using normal densities, kernel smoothing (Kennedy et al. 2017), gradient boosting (Zhu et al. 2015), and empirical likelihoods (Fong et al. 2018). In this package we provide an extension to this literature to allow for multivariate continuous exposures to be estimated.

Notation:

Assume that we have a set of continuous exposures, D, of length m, i.e., $\mathbf{D} = D_1, \ldots, D_m$ collected on n units. Further, we assume that there exists a set of confounders $\mathbf{C}_1, \ldots, \mathbf{C}_m$ for each exposure of length p_j for $j = 1, \ldots, m$. The confounders are related to both the exposures and the outcome of interest such. Note that the confounders need not be identical for all exposures.

In our function we therefore expect that the argument D is a numeric matrix of dimension $n \times m$, and that C is a list of length m where each element is a matrix of dimension $n \times p_j$. For the case where we assume that all exposures have common confounders we set common=TRUE and C must be a matrix of dimension $n \times p$.

mvGPS:

We define the multivariate generalized propensity score, mvGPS, as

$$mvGPS = f_{\mathbf{D}|\mathbf{C}_1,\ldots,\mathbf{C}_m}$$

where f represents a joint multivariate conditional density function. For our current development we specify f as multivariate normal, i.e.,

$$\mathbf{D} \mid \mathbf{C}_1, \ldots, \mathbf{C}_m \sim N_m(\boldsymbol{\mu}, \Sigma).$$

Factorizing this joint density we can rewrite the mvGPS as the product of univariate conditional densities, i.e.,

$$mvGPS = f_{D_m|\mathbf{C}_m,D_{m-1},\ldots,D_1} \times \cdots \times f_{D_1|\mathbf{C}_1}.$$

We use this factorized version in our implementation, with parameters for each distribution estimated through least squares.

Constructing Weights:

Following Robins et al. (2000), we use the mvGPS to construct stabilized inverse probability of treatment (IPTW) weights. These have been shown to balance confounders and return unbiased estimated of the dose-response. Weights are constructed as

$$w = \frac{f_{\mathbf{D}}}{f_{\mathbf{D}|\mathbf{C}_1,\ldots,\mathbf{C}_m}},$$

where the marginal density $f_{\mathbf{D}}$ of the exposures is assumed to be multivariate normal as well. Writing the weights using completely factorized densities we have

$$w = \frac{f_{D_m|D_{m-1},\ldots,D_1} \times \cdots \times f_{D_1}}{f_{D_m|\mathbf{C}_m,D_{m-1},\ldots,D_1} \times \cdots \times f_{D_1|\mathbf{C}_1}}.$$

Trimming:

Often when using weights based on the propensity score, practitioners are concerned about the effect of extreme weights. It has been shown that an effective way to protect extreme weights is to trim them at a particular percentile (Crump et al. 2009; Lee et al. 2011). We allow users to specify whether trimmed weights should be returned and at which threshold. To trim weights set `trim_w=TRUE` and specify the desired percentile as `trim_quantile=q`. Note that trimming is applied at both the upper and lower percentile thresholds, i.e.,

$$w^* = w1_{\{Q(w,1-q)\leq w\leq Q(w,q)\}} + Q(w,1-q)1_{\{w<Q(w,1-q)\}} + Q(w,q)1_{\{w>Q(w,q)\}}$$

Value

list of score and wts, where score is the mvGPS score values and wts are the corresponding stabilized inverse probability of treatment weights

References

Crump RK, Hotz VJ, Imbens GW, Mitnik OA (2009). "Dealing with limited overlap in estimation of average treatment effects." *Biometrika*, **96**(1), 187-199.

Fong C, Hazlett C, Imai K (2018). "Covariate balancing propensity score for a continuous treatment: application to the efficacy of political advertisements." *Annals of Applied Statistics*, **In-Press**.

Hirano K, Imbens GW (2004). "The propensity score with continuous treatments." In Gelman A, Meng X (eds.), *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives*, 73-84.

Imai K, Van Dyk DA (2004). "Causal inference with general treatment regimes: generalizing the propensity score." *Journal of the American Statistical Association*, **99**(467), 854-866.

Kennedy EH, Ma Z, McHugh MD, Small DS (2017). "Non-parametric methods for doubly robust estimation of continuous treatment effects." *Journal of the Royal Statistical Society: Series B*, **79**(4), 1229-1245.

Lee BK, Lessler J, Stuart EA (2011). "Weight trimming and propensity score weighting." *PloS One*, **6**(3).

Robins JM, Hernan MA, Brumback B (2000). "Marginal structural models and causal inference in epidemiology." *Epidemiology*, **11**(5), 550-560.

Zhu Y, Coffman DL, Ghosh D (2015). "A boosting algorithm for estimating generalized propensity scores with continuous treatments." *Journal of Causal Inference*, **3**(1), 25-40.

Examples

```
#generating confounded bivariate exposure
sim_dt <- gen_D(method="u", n=200, rho_cond=0.2, s_d1_cond=2, s_d2_cond=2, k=3,
C_mu=rep(0, 3), C_cov=0.1, C_var=1, d1_beta=c(0.5, 1, 0), d2_beta=c(0, 0.3, 0.75), seed=06112020)
D <- sim_dt$D
C <- sim_dt$C

#generating weights and mvGPS
out_mvGPS <- mvGPS(D=D, C=list(C[, 1:2], C[, 2:3]))

# can apply trimming with default 99th percentile
out_mvGPS_trim <- mvGPS(D=D, C=list(C[, 1:2], C[, 2:3]), trim_w=TRUE)

#or assume all exposures have equivalent confounders
out_mvGPS_common <- mvGPS(D=D, C=C, common=TRUE)
```

Index