Package: tempted (via r-universe)

October 31, 2024

Type Package

Title Temporal Tensor Decomposition, a Dimensionality Reduction Tool for Longitudinal Multivariate Data

Version 0.1.1

Date 2024-5-8

Depends R (>= 4.2.0), np (>= 0.60-17), ggplot2 (>= 3.4.0), methods (>= 4.2.1)

Author Pixu Shi

Maintainer Pixu Shi <pixu.shi@duke.edu>

Description TEMPoral TEnsor Decomposition (TEMPTED), is a dimension reduction method for multivariate longitudinal data with varying temporal sampling. It formats the data into a temporal tensor and decomposes it into a summation of low-dimensional components, each consisting of a subject loading vector, a feature loading vector, and a continuous temporal loading function. These loadings provide a low-dimensional representation of subjects or samples and can be used to identify features associated with clusters of subjects or samples. TEMPTED provides the flexibility of allowing subjects to have different temporal sampling, so time points do not need to be binned, and missing time points do not need to be imputed.

URL https://github.com/pixushi/tempted

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Repository https://pixushi.r-universe.dev

RemoteUrl https://github.com/pixushi/tempted

RemoteRef HEAD

RemoteSha 8c0f189b33406bcdc217c053745143e3b4acc3c7

Contents

`aggregate_feature` *Aggregate features using feature loadings*

Description

This function aggregate the features into "meta features" by calculating a weighted summation of the features using feature loading of each component as weights. It can also aggregate features by using the combination of multiple components by ranking the features by a linear combination of feature loadings from multiple components.

Usage

```
aggregate_feature(
  res_tempted,
  mean_svd = NULL,
  datlist,
  pct = 1,
  contrast = NULL
)
```

Arguments

`res_tempted`	Output of `tempted`.
`mean_svd`	Output of `svd_centralize`.
`datlist`	Output of `format_tempted`, the original temporal tensor that will be aggregated.

`pct` The percent of features to aggregate, features ranked by absolute value of the feature loading of each component. Default is 1, which means 100% of features are aggregated. Setting `pct=0.01` means top 1% of features is aggregated, where features are ranked in absolute value of feature loading of each component.

`contrast` A matrix choosing how components are combined, each column is a contrast of length r and used to calculate the linear combination of the feature loadings of r components.

Value

A list of results.

metafeature_aggregate The meta feature obtained by aggregating the observed temporal tensor. It is a data.frame with four columns: "value" for the meta feature values, "subID" for the subject ID, "timepoint" for the time points, and "PC" indicating which component was used to construct the meta feature.

metafeature_aggregate_est The meta feature obtained by aggregating the denoised temporal tensor. It has the same structure as `metafeature_aggregate`.

contrast The contrast used to linearly combine the components from input.

toppct A matrix of TRUE/FALSE indicating which features are aggregated in each component and contrast.

References

Shi P, Martino C, Han R, Janssen S, Buck G, Serrano M, Owzar K, Knight R, Shenhav L, Zhang AR. (2023) *Time-Informed Dimensionality Reduction for Longitudinal Microbiome Studies*. bioRxiv. doi: 10.1101/550749. https://www.biorxiv.org/content/10.1101/550749.

Examples

```
# Take a subset of the samples so the example runs faster

# Here we are taking samples from the odd months
sub_sample <- rownames(meta_table)[(meta_table$day_of_life%/%12)%%2==1]
count_table_sub <- count_table[sub_sample,]
processed_table_sub <- processed_table[sub_sample,]
meta_table_sub <- meta_table[sub_sample,]

datlist <- format_tempted(count_table_sub,
                          meta_table_sub$day_of_life,
                          meta_table_sub$studyid,
                          pseudo=0.5,
                          transform="clr")

mean_svd <- svd_centralize(datlist, r=1)

res_tempted <- tempted(mean_svd$datlist, r=2, smooth=1e-5)

contrast <- matrix(c(1/2,1), 2, 1)
```

```
res_aggregate <- aggregate_feature(res_tempted,
                                   mean_svd,
                                   datlist,
                                   pct=1,
                                   contrast=contrast)

# plot the aggregated features


group <- unique(meta_table[, c("studyid", "delivery")])

plot_metafeature(res_aggregate$metafeature_aggregate, group, bws=30)
```

`bernoulli_kernel` *Caculate the Bernoulli kernel*

Description

This function is used to calculate the kernel matrix for the RKHS regression that iteratively updates the temporal loading function.

Usage

```
bernoulli_kernel(x, y)
```

Arguments

`x, y`	Two values between which the Bernoulli kernel is calculated.

Value

The calculated kernel between `x` and `y`.

References

Han, R., Shi, P. and Zhang, A.R. (2023) *Guaranteed functional tensor singular value decomposition*. Journal of the American Statistical Association, pp.1-13. doi: 10.1080/01621459.2022.2153689.

count_table *OTU read count table from the ECAM data*

Description

OTU read count table from the ECAM data

Usage

```
count_table
```

Format

A data.frame with rows representing samples and matching with data.frame `meta_table` and columns representing microbial features (i.e. OTUs). Each entry is a read count.

References

Bokulich, Nicholas A., et al. "Antibiotics, birth mode, and diet shape microbiome maturation during early life." Science translational medicine 8.343 (2016): 343ra82-343ra82.

est_test_subject *Estimate subject loading of testing data*

Description

This function estimates the subject loading of the testing data based on feature and temporal loading from training data, so that both the testing data and training data have the same dimensionality reduction.

Usage

```
est_test_subject(datlist, res_tempted, mean_svd = NULL)
```

Arguments

`datlist`	Testing data formatted into datlist in the same fashion as the training data. The same transformation needs to be used for both training and testing data.
`res_tempted`	Result from `tempted` ran on the training data.
`mean_svd`	Result from `svd_centralize` ran on the training data.

Value

estimated subject loading of testing data

References

Shi P, Martino C, Han R, Janssen S, Buck G, Serrano M, Owzar K, Knight R, Shenhav L, Zhang AR. (2023) *Time-Informed Dimensionality Reduction for Longitudinal Microbiome Studies*. bioRxiv. doi: 10.1101/550749. https://www.biorxiv.org/content/10.1101/550749.

Examples

```
# Take a subset of the samples so the example runs faster

# Here we are taking samples from the odd months
sub_sample <- rownames(meta_table)[(meta_table$day_of_life%/%12)%%2==1]
count_table_sub <- count_table[sub_sample,]
processed_table_sub <- processed_table[sub_sample,]
meta_table_sub <- meta_table[sub_sample,]

# split the example data into training and testing

id_test <- meta_table_sub$studyid=="2"

count_train <- count_table_sub[!id_test,]
meta_train <- meta_table_sub[!id_test,]

count_test <- count_table_sub[id_test,]
meta_test <- meta_table_sub[id_test,]

# run tempted on training data

datlist_train <- format_tempted(count_train,
                                meta_train$day_of_life,
                                meta_train$studyid,
                                threshold=0.95,
                                pseudo=0.5,
                                transform="clr")

mean_svd_train <- svd_centralize(datlist_train, r=1)

res_tempted_train <- tempted(mean_svd_train$datlist,
r=2, smooth=1e-5)

# get the overlapping features

count_test <- count_test[,rownames(datlist_train[[1]])[-1]]

datlist_test <- format_tempted(count_test,
                               meta_test$day_of_life,
                               meta_test$studyid,
                               threshold=1,
                               pseudo=0.5,
                               transform="clr")

# estimate the subject loading of the testing subject
```

```
sub_test <- est_test_subject(datlist_test, res_tempted_train, mean_svd_train)

# train logistic regression classifier on training subjects

metauni <- unique(meta_table_sub[,c("studyid", "delivery")])
rownames(metauni) <- metauni$studyid
Atrain <- as.data.frame(res_tempted_train$A_hat)
Atrain$delivery <- metauni[rownames(Atrain),"delivery"]=="Cesarean"
glm_train <- glm(delivery ~ PC1+PC2,
                 data=Atrain, family=binomial(link="logit"))
summary(glm_train)

# predict the label of testing subject, whose true label is "Cesarean"

predict(glm_train, newdata=as.data.frame(sub_test), type="response")
```

`format_tempted` *Format data table into the input of tempted*

Description

This function applies a variety of transformations to the read counts and format the sample by feature table and meta data into a data list that can be used as the input of `tempted` and `svd_centralize`. For data that are not read counts, or data that are not microbiome data, the user can apply their desired transformation to the data before formatting into list.

Usage

```
format_tempted(
  featuretable,
  timepoint,
  subjectID,
  threshold = 0.95,
  pseudo = NULL,
  transform = "clr"
)
```

Arguments

`featuretable`	A sample by feature matrix.
`timepoint`	The time stamp of each sample, matched with the rows of `featuretable`.
`subjectID`	The subject ID of each sample, matched with the rows of `featuretable`.
`threshold`	A threshold for feature filtering for microbiome data. Features with zero value percentage > threshold will be excluded. Default is 0.95.

pseudo — A small number to add to all the counts before normalizing into proportions and log transformation. Default is 1/2 of the smallest non-zero value that is specific for each sample. This pseudo count is added for `transform=c("logcomp", "clr", "logit")`.

transform — The transformation applied to the data. `"logcomp"` for log of compositions. `"comp"` for compositions. `"ast"` for arcsine squared transformation. `"clr"` for central log ratio transformation. `"lfb"` for log 2 fold change over baseline (first time point) transformation. `"logit"` for logit transformation. `"none"` for no transformation. Default `transform="clr"` is recommended for microbiome data. For data that are already transformed, use `transform="none"`.

Value

A length n list of matrices as the input of `tempted` and `svd_centralize`. Each matrix represents a subject, with columns representing samples from this subject, the first row representing the sampling time points, and the following rows representing the feature values.

See Also

Examples can be found in `tempted`.

---

meta_table *Meta data table from the ECAM data*

Description

Meta data table from the ECAM data

Usage

```
meta_table
```

Format

A data.frame with rows representing samples and matching with data.frame `count_table` and `processed_table` and three columns:

studyid character denoting the subject ID of the infants.

delivery character denoting the delivery mode of the infants.

day_of_life character denoting the age of infants measured in days when microbiome sample was taken.

References

Bokulich, Nicholas A., et al. "Antibiotics, birth mode, and diet shape microbiome maturation during early life." Science translational medicine 8.343 (2016): 343ra82-343ra82.

`plot_feature_summary` *Plot nonparametric smoothed mean and error bands of features versus time*

Description

This is a handy function to plot the smoothed mean and error bands for multiple features.

Usage

```
plot_feature_summary(
  feature_mat,
  time_vec,
  group_vec,
  coverage = 0.95,
  bws = NULL,
  nrow = 1
)
```

Arguments

`feature_mat`	A sample by feature matrix. Each feature will be plotted separately as a facet. The features can be original features, meta features, log ratios, or any variables of interest.
`time_vec`	A vector of time points matched to the rows of `feature_mat`.
`group_vec`	A vector of factor variable indicating the group membership of samples matched to the rows of `feature_mat`.
`coverage`	The coverage rate for the error band. Default is 0.95.
`bws`	The smoothness parameter for the smoothing lines and error bands. A larger value means a smoother line. Default is NULL and calculated by function `np::npreg()`.
`nrow`	The number of rows to plot the features used in function `ggplot2::facet_wrap()`.

Value

A ggplot2 object.

Examples

```
# plot the summary of selected features

feat.names <- c("OTU4447072", "OTU4467447")

proportion_table <- count_table/rowSums(count_table)

plot_feature_summary(proportion_table[,feat.names],
                     meta_table$day_of_life,
```

```
meta_table$delivery,
bws=30)
```

`plot_metafeature` *Plot nonparametric smoothed mesan and error bands of meta features versus time*

Description

This function plot the smoothed mean and error band of meta features grouped by a factor variable provided by the user.

Usage

```
plot_metafeature(metafeature, group, coverage = 0.95, bws = NULL, nrow = 1)
```

Arguments

`metafeature`	`metafeature_ratio` from the output of `ratio_feature` and `tempted_all`, `metafeature_aggregate` from the output of `ratio_feature` and `tempted_all`, or `metafeature_aggregate_est` from the output of `ratio_feature`.
`group`	A subject by 2 data.frame with the first column for subject ID and second column for group membership.
`coverage`	The coverage rate for the error band. Default is 0.95.
`bws`	The smoothness parameter for the smoothing lines and error bands. A larger value means a smoother line. Default is NULL and calculated by function `np::npreg()`.
`nrow`	The number of rows to plot the features used in function `ggplot2::facet_wrap()`.

Value

A ggplot2 object.

See Also

Examples can be found in `tempted_all`, `ratio_feature` and `aggregate_feature`.

plot_time_loading *Plot the temporal loading functions*

Description

This function uses `ggplot2::geom_line()` in ggplot2 to plot the temporal loading functions from `tempted`.

Usage

```
plot_time_loading(res, r = NULL, ...)
```

Arguments

`res`	Output of function `tempted`.
`r`	The number of components to plot. By default all the components estimated by `tempted` will be plotted.
`...`	Arguments to put in `ggplot2::geom_line(aes(...))`.

Value

An ggplot2 object.

See Also

Examples can be found in `tempted_all` and `tempted`.

processed_table *Central-log-ratio (clr) transformed OTU table from the ECAM data*

Description

Central-log-ratio (clr) transformed OTU table from the ECAM data

Usage

```
processed_table
```

Format

A data.frame with rows representing samples and matching with data.frame `meta_table` and columns representing microbial features (i.e. OTUs). Entries do not need to be transformed, and will be directly used by `tempted`. This data.frame is used to illustrate how `tempted` can be used for general form of multivariate longitudinal data already preprocessed by user.

References

Bokulich, Nicholas A., et al. "Antibiotics, birth mode, and diet shape microbiome maturation during early life." Science translational medicine 8.343 (2016): 343ra82-343ra82.

`ratio_feature` *Take log ratio of the abundance of top features over bottom features*

Description

Top and bottom ranking features are picked based on feature loadings (and their contrasts). The log ratio abundance of the top ranking features over the bottom ranking features is produced as the main result. This function and its result is designed for longitudinal microbiome data, and may not be meaningful for other type of temporal data.

Usage

```
ratio_feature(
  res_tempted,
  datlist,
  pct = 0.05,
  absolute = FALSE,
  contrast = NULL
)
```

Arguments

`res_tempted`	Output of `tempted`.
`datlist`	Output of `format_tempted(, transform="none")`, the temporal tensor that include the raw read counts.
`pct`	The percent of features to sum up. Default is 0.05, i.e. 5%.
`absolute`	`absolute = TRUE` means features are ranked by the absolute value of feature loadings, and the top `pct` percent of features are picked. `absolute = FALSE` means features are ranked by the original value of feature loadings, and the top and bottom `pct` percent of features are picked. Then ratio is taken as the abundance of the features with positive loading over the abundance of the features with negative loading.
`contrast`	A matrix choosing how components are combined, each column is a contrast of length r and used to calculate the linear combination of the feature loadings of r components.

Value

A list of results:

metafeature_ratio The log ratio abundance of the top over bottom ranking features. It is a data.frame with five columns: "value" for the log ratio values, "subID" for the subject ID, and "timepoint" for the time points, and "PC" indicating which component was used to construct the meta feature.

contrast The contrast used to linearly combine the components from input.

toppct A matrix of TRUE/FALSE indicating which features are ranked top in each component (and contrast) and used as the numerator of the log ratio.

bottompct A matrix of TRUE/FALSE indicating which features are ranked bottom in each component (and contrast) and used as the denominator of the log ratio.

References

Shi P, Martino C, Han R, Janssen S, Buck G, Serrano M, Owzar K, Knight R, Shenhav L, Zhang AR. (2023) *Time-Informed Dimensionality Reduction for Longitudinal Microbiome Studies*. bioRxiv. doi: 10.1101/550749. https://www.biorxiv.org/content/10.1101/550749.

Examples

```
# Take a subset of the samples so the example runs faster

# Here we are taking samples from the odd months
sub_sample <- rownames(meta_table)[(meta_table$day_of_life%/%12)%%2==1]
count_table_sub <- count_table[sub_sample,]
processed_table_sub <- processed_table[sub_sample,]
meta_table_sub <- meta_table[sub_sample,]

datlist <- format_tempted(count_table_sub,
                          meta_table_sub$day_of_life,
                          meta_table_sub$studyid,
                          pseudo=0.5,
                          transform="clr")

mean_svd <- svd_centralize(datlist, r=1)

res_tempted <- tempted(mean_svd$datlist, r=2, smooth=1e-5)

datalist_raw <- format_tempted(count_table_sub, meta_table_sub$day_of_life, meta_table_sub$studyid,
transform="none")

contrast <- cbind(c(1,1), c(1,-1))

res_ratio <- ratio_feature(res_tempted, datalist_raw, pct=0.1,
absolute=FALSE, contrast=contrast)

group <- unique(meta_table[, c("studyid", "delivery")])

# plot the log ratios

plot_metafeature(res_ratio$metafeature_ratio, group, bws=30)
```

reconstruct *Reconstruct tensor from low dimensional components*

Description

This function reconstructs the temporal tensor from the low dimensional components extracted by tempted and svd_centralize.

Usage

```
reconstruct(res_tempted, res_svd = NULL, datlist = NULL, r_reconstruct = NULL)
```

Arguments

res_tempted	Output of function tempted.
res_svd	Output of function svd_centralize if mean subtraction svd_centralize was performed before tempted. If mean subtraction was not performed, specify res_svd=NULL and provide datlist.
datlist	Output of function format_tempted if mean subtraction was not performed and res_svd=NULL, or left as NULL if mean subtraction was performed and res_svd is provided.
r_reconstruct	The number of components from TEMPTED to be used for the tensor reconstruction.

Value

datlist_est The reconstructed tensor stored in a datlist format same as the output of format_tempted.

Examples

```
# Take a subset of the samples so the example runs faster

# Here we are taking samples from the odd months
sub_sample <- rownames(meta_table)[(meta_table$day_of_life%/%12)%%2==1]
count_table_sub <- count_table[sub_sample,]
processed_table_sub <- processed_table[sub_sample,]
meta_table_sub <- meta_table[sub_sample,]
# reconstruct with mean subtraction
datlist <- format_tempted(processed_table_sub,
                          meta_table_sub$day_of_life,
                          meta_table_sub$studyid,
                          pseudo=NULL,
                          transform="none")

mean_svd <- svd_centralize(datlist, r=1)

res_tempted <- tempted(mean_svd$datlist, r=2, smooth=1e-5)
```

```
datlist_est <- reconstruct(res_tempted, mean_svd, datlist=NULL, r_reconstruct=2)
vec_est <- unlist(sapply(datlist_est, function(x){x[-1,]}))
vec_obs <- unlist(sapply(datlist, function(x){x[-1,]}))
R2 <- 1-sum((vec_est-vec_obs)^2)/sum(vec_obs^2)
R2

# reconstruct without mean subtraction
datlist <- format_tempted(processed_table_sub,
                          meta_table_sub$day_of_life,
                          meta_table_sub$studyid,
                          pseudo=NULL,
                          transform="none")

res_tempted <- tempted(datlist, r=2, smooth=1e-5)

datlist_est <- reconstruct(res_tempted, res_svd=NULL, datlist=datlist, r_reconstruct=2)
vec_est <- unlist(sapply(datlist_est, function(x){x[-1,]}))
vec_obs <- unlist(sapply(datlist, function(x){x[-1,]}))
R2 <- 1-sum((vec_est-vec_obs)^2)/sum(vec_obs^2)
R2
```

`svd_centralize` *Remove the mean structure of the temporal tensor*

Description

This function first average the feature value of all time points for each subject to form a subject by feature matrix. Next, it performs a singular value decomposition of this matrix and construct the matrix's rank-r approximation. Then, it subtracts this rank-r subject by feature matrix from the temporal tensor.

Usage

```
svd_centralize(datlist, r = 1)
```

Arguments

`datlist`	A length n list of matrices. Each matrix represents a subject, with columns representing samples from this subject, the first row representing the sampling time points, and the following rows representing the feature values.
`r`	The number of ranks in the mean structure. Default is 1.

Value

A list of results.

datlist The new temporal tensor after mean structure is removed.

A_tilde The subject singular vector of the mean structure, a subject by r matrix.

B_tilde The feature singular vector of the mean structure, a feature by r matrix.

lambda_tilde The singular value of the mean structure, a length r vector.

References

Shi P, Martino C, Han R, Janssen S, Buck G, Serrano M, Owzar K, Knight R, Shenhav L, Zhang AR. (2023) *Time-Informed Dimensionality Reduction for Longitudinal Microbiome Studies*. bioRxiv. doi: 10.1101/550749. https://www.biorxiv.org/content/10.1101/550749.

See Also

Examples can be found in `tempted`.

`tdenoise` *Calculate the de-noised temporal tensor*

Description

This function constructs a de-noised version of the temporal tensor using the low-rank components obtained by `svd_centralize` `tempted` and uses the loadings to

Usage

```
tdenoise(res_tempted, mean_svd = NULL)
```

Arguments

`res_tempted`	Output of tempted
`mean_svd`	Output of svd_centralize

Value

The de-noised functional tensor

`tempted` *Decomposition of temporal tensor*

Description

This is the main function of tempted.

Usage

```
tempted(
  datlist,
  r = 3,
  smooth = 1e-06,
  interval = NULL,
  resolution = 101,
  maxiter = 20,
  epsilon = 1e-04
)
```

Arguments

`datlist`	A length n list of matrices. Each matrix represents a subject, with columns representing samples from this subject, the first row representing the sampling time points, and the following rows representing the feature values.
`r`	Number of components to decompose into, i.e. rank of the CP type decomposition. Default is set to 3.
`smooth`	Smoothing parameter for RKHS norm. Larger means smoother temporal loading functions. Default is set to be 1e-8. Value can be adjusted depending on the dataset by checking the smoothness of the estimated temporal loading function in plot.
`interval`	The range of time points to ran the decomposition for. Default is set to be the range of all observed time points. User can set it to be a shorter interval than the observed range.
`resolution`	Number of time points to evaluate the value of the temporal loading function. Default is set to 101. It does not affect the subject or feature loadings.
`maxiter`	Maximum number of iteration. Default is 20.
`epsilon`	Convergence criteria for difference between iterations. Default is 1e-4.

Value

The estimations of the loadings.

A_hat Subject loading, a subject by r matrix.

B_hat Feature loading, a feature by r matrix.

Phi_hat Temporal loading function, a resolution by r matrix.

time_Phi The time points where the temporal loading function is evaluated.

Lambda Eigen value, a length r vector.

r_square Variance explained by each component. This is the R-squared of the linear regression of the vectorized temporal tensor against the vectorized low-rank reconstruction using individual components.

accum_r_square Variance explained by the first few components accumulated. This is the R-squared of the linear regression of the vectorized temporal tensor against the vectorized low-rank reconstruction using the first few components.

Examples

```
# Take a subset of the samples so the example runs faster

# Here we are taking samples from the odd months
sub_sample <- rownames(meta_table)[(meta_table$day_of_life%/%12)%%2==1]
count_table_sub <- count_table[sub_sample,]
processed_table_sub <- processed_table[sub_sample,]
meta_table_sub <- meta_table[sub_sample,]

# for count data from longitudinal microbiome studies

datlist <- format_tempted(count_table_sub,
                          meta_table_sub$day_of_life,
                          meta_table_sub$studyid,
                          pseudo=0.5,
                          transform="clr")

mean_svd <- svd_centralize(datlist, r=1)

res_tempted <- tempted(mean_svd$datlist, r=2, smooth=1e-5)

# for preprocessed data that do not need to be transformed

datlist <- format_tempted(processed_table_sub,
                          meta_table_sub$day_of_life,
                          meta_table_sub$studyid,
                          pseudo=NULL,
                          transform="none")

mean_svd <- svd_centralize(datlist, r=1)

res_tempted <- tempted(mean_svd$datlist, r=2, smooth=1e-5)

# plot the temporal loading

plot_time_loading(res_tempted, r=2)
```

tempted_all *Run all major functions of tempted*

Description

This function wraps functions format_tempted, svd_centralize, tempted, ratio_feature, \ and aggregate_feature.

Usage

```
tempted_all(
  featuretable,
  timepoint,
  subjectID,
  threshold = 0.95,
  pseudo = NULL,
  transform = "clr",
  r = 3,
  smooth = 1e-06,
  interval = NULL,
  resolution = 51,
  maxiter = 20,
  epsilon = 1e-04,
  r_svd = 1,
  do_ratio = TRUE,
  pct_ratio = 0.05,
  absolute = FALSE,
  pct_aggregate = 1,
  contrast = NULL
)
```

Arguments

featuretable	A sample by feature matrix. It is an input for format_tempted.
timepoint	The time stamp of each sample, matched with the rows of featuretable. It is an input for format_tempted.
subjectID	The subject ID of each sample, matched with the rows of featuretable. It is an input for format_tempted.
threshold	A threshold for feature filtering for microbiome data. Features with zero value percentage >= threshold will be excluded. Default is 0.95. It is an input for format_tempted.
pseudo	A small number to add to all the counts before normalizing into proportions and log transformation. Default is 1/2 of the smallest non-zero value that is specific for each sample. This pseudo count is added for transform=c("logcomp", "clr", "logit"). It is an input for format_tempted.

transform	The transformation applied to the data. "logcomp" for log of compositions. "comp" for compositions. "ast" for arcsine squared transformation. "clr" for central log ratio transformation. "logit" for logit transformation. "none" for no transformation. Default transform="clr" is recommended for microbiome data. For data that are already transformed, use transform="none". It is an input for format_tempted.
r	Number of components to decompose into, i.e. rank of the CP type decomposition. Default is set to 3. It is an input for tempted.
smooth	Smoothing parameter for RKHS norm. Larger means smoother temporal loading functions. Default is set to be 1e-8. Value can be adjusted depending on the dataset by checking the smoothness of the estimated temporal loading function in plot. It is an input for tempted.
interval	The range of time points to ran the decomposition for. Default is set to be the range of all observed time points. User can set it to be a shorter interval than the observed range. It is an input for tempted.
resolution	Number of time points to evaluate the value of the temporal loading function. Default is set to 101. It does not affect the subject or feature loadings. It is an input for tempted.
maxiter	Maximum number of iteration. Default is 20. It is an input for tempted.
epsilon	Convergence criteria for difference between iterations. Default is 1e-4. It is an input for tempted.
r_svd	The number of ranks in the mean structure. Default is 1. It is an input for svd_centralize.
do_ratio	Whether to calculate the log ratio of features.
pct_ratio	The percent of features to sum up. Default is 0.05, i.e. 5%. It is an input for ratio_feature.
absolute	absolute = TRUE means features are ranked by the absolute value of feature loadings, and the top pct_ratio percent of features are picked. absolute = FALSE means features are ranked by the original value of feature loadings, and the top and bottom pct_ratio percent of features are picked. Then ratio is taken as the abundance of the features with positive loading over the abundance of the features with negative loading. It is an input for ratio_feature.
pct_aggregate	The percent of features to aggregate, features ranked by absolute value of the feature loading of each component. Default is 1, which means 100% of features are aggregated. Setting pct_aggregate=0.01 means top 1% of features is aggregated, where features are ranked in absolute value of feature loading of each component. It is an input for aggregate_feature.
contrast	A matrix choosing how components are combined, each column is a contrast of length r and used to calculate the linear combination of the feature loadings of r components. It is an input for ratio_feature and It is an input for aggregate_feature.

Value

A list including all the input and output of functions format_tempted, svd_centralize, tempted, ratio_feature, and aggregate_feature.

input All the input options of function `tempted_all`.

datalist_raw Output of `format_tempted` with option `transform="none"`.

datlist Output of `format_tempted`.

mean_svd Output of `svd_centralize`.

A_hat Subject loading, a subject by r matrix.

B_hat Feature loading, a feature by r matrix.

Phi_hat Temporal loading function, a resolution by r matrix.

time_Phi The time points where the temporal loading function is evaluated.

Lambda Eigen value, a length r vector.

r_square Variance explained by each component. This is the R-squared of the linear regression of the vectorized temporal tensor against the vectorized low-rank reconstruction using individual components.

accum_r_square Variance explained by the first few components accumulated. This is the R-squared of the linear regression of the vectorized temporal tensor against the vectorized low-rank reconstruction using the first few components.

metafeature_ratio The log ratio abundance of the top over bottom ranking features. It is a data.frame with five columns: "value" for the log ratio values, "subID" for the subject ID, and "timepoint" for the time points, and "PC" indicating which component was used to construct the meta feature.

toppct_ratio A matrix of TRUE/FALSE indicating which features are ranked top in each component (and contrast) and used as the numerator of the log ratio.

bottompct_ratio A matrix of TRUE/FALSE indicating which features are ranked bottom in each component (and contrast) and used as the denominator of the log ratio.

metafeature_aggregate The meta feature obtained by aggregating the observed temporal tensor. It is a data.frame with four columns: "value" for the meta feature values, "subID" for the subject ID, "timepoint" for the time points, and "PC" indicating which component was used to construct the meta feature.

toppct_aggregate A matrix of TRUE/FALSE indicating which features are aggregated in each component and contrast.

contrast The contrast used to linearly combine the components from input.

References

Shi P, Martino C, Han R, Janssen S, Buck G, Serrano M, Owzar K, Knight R, Shenhav L, Zhang AR. (2023) *Time-Informed Dimensionality Reduction for Longitudinal Microbiome Studies*. bioRxiv. doi: 10.1101/550749. https://www.biorxiv.org/content/10.1101/550749.

Examples

```
# Take a subset of the samples so the example runs faster

# Here we are taking samples from the odd months
sub_sample <- rownames(meta_table)[(meta_table$day_of_life%/%12)%%2==1]
count_table_sub <- count_table[sub_sample,]
```

```
processed_table_sub <- processed_table[sub_sample,]
meta_table_sub <- meta_table[sub_sample,]

# for preprocessed data that do not need to be transformed


res.processed <- tempted_all(processed_table_sub,
                             meta_table_sub$day_of_life,
                            meta_table_sub$studyid,
                             threshold=1,
                             transform="none",
                             r=2,
                             smooth=1e-5,
                             do_ratio=FALSE)

# for count data that will have pseudo added and clr transformed

res.count <- tempted_all(count_table_sub,
                         meta_table_sub$day_of_life,
                         meta_table_sub$studyid,
                         threshold=0.95,
                         transform="clr",
                         pseudo=0.5,
                         r=2,
                         smooth=1e-5,
                         pct_ratio=0.1,
                         pct_aggregate=1)

# for proportional data that will have pseudo added and clr transformed

res.proportion <- tempted_all(count_table_sub/rowSums(count_table_sub),
                              meta_table_sub$day_of_life,
                              meta_table_sub$studyid,
                              threshold=0.95,
                              transform="clr",
                              pseudo=NULL,
                              r=2,
                              smooth=1e-5,
                              pct_ratio=0.1,
                              pct_aggregate=1)

# plot the temporal loading and subject trajectories grouped by delivery mode

plot_time_loading(res.proportion, r=2)

group <- unique(meta_table[,c("studyid", "delivery")])

# plot the aggregated features

plot_metafeature(res.proportion$metafeature_aggregate, group, bws=30)
```

Index