Package: cepip (via r-universe)

October 29, 2026

Type Package

Title Cell Explanatory Power with Inflection Point (CEP-IP) with CMRF

Version 0.1.0

Description Implements the cell explanatory power with inflection point (CEP-IP) framework for single-cell co-expression analysis. A generalized additive model (GAM) fitted by restricted maximum likelihood (REML) with a penalized residual sum of squares (PRSS) search over the basis dimension relates a gene of interest to the mean of its partner genes. Cells are ranked by their explanatory power, and the inflection point is the first negative-to-positive zero crossing of a residual GAM. Optional extensions are the Spearman-Kendall cell-level score (SKCS) and the copula monotone recovery filter (CMRF), which recovers cells near the module through a tolerance band and a repeated stratified k-fold cross-validation gate. Downstream helpers give per-donor module fractions, paired donor-level differential expression and donor-level association models. CEP-IP is described in Wong (2026) <doi:10.1016/j.cmpb.2026.109372>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports mgcv, stats, utils, methods, grDevices, grid, parallel, Matrix, ggplot2 (>= 3.4.0), gridExtra, openxlsx

Suggests Hmisc, energy, clinfun, pcaPP, testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.1

URL https://github.com/kahkengwong/cepip

BugReports https://github.com/kahkengwong/cepip/issues

NeedsCompilation no

Author Kah Keng Wong [aut, cre] (ORCID:
<https://orcid.org/0000-0001-7359-6202>)

Maintainer Kah Keng Wong <kahkeng3@gmail.com>

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

Date/Publication 2026-10-29 21:14:50 UTC

RemoteUrl https://github.com/cran/cepip

RemoteRef HEAD

RemoteSha 49283c8c1341a9b015bfe35820d2b1f69fa87f03

Contents

cepip . . . 3
cepip_associate . . . 4
cepip_band . . . 5
cepip_cmrf . . . 7
cepip_composite . . . 9
cepip_deg . . . 9
cepip_donor . . . 11
cepip_ep . . . 11
cepip_example . . . 12
cepip_example_clear . . . 13
cepip_fit . . . 14
cepip_groups . . . 15
cepip_ip . . . 16
cepip_save . . . 17
cepip_screen . . . 19
cepip_skcs . . . 20
cepip_toy . . . 22
cepip_validate . . . 22
plot_associate . . . 23
plot_band . . . 24
plot_copula . . . 24
plot_deg . . . 25
plot_donor . . . 26
plot_fit . . . 27
plot_module . . . 27
plot_pitcher . . . 28
plot_prss . . . 29
plot_residual . . . 29
plot_rskfcv . . . 30
plot_screen . . . 31
plot_validate . . . 31
print.cepip_figure . . . 32

Index **33**

`cepip` *Run CEP-IP, optionally with SKCS and CMRF*

Description

Runs the whole method on one population of cells.

Usage

```
cepip(
  x,
  y,
  donor = NULL,
  id = NULL,
  skcs = FALSE,
  cmrf = FALSE,
  validate = cmrf,
  ip_k = "prss",
  ip_axis = c("y", "x"),
  band = NULL,
  kappa = 0.5,
  bw = 0.1,
  y_spine = NULL,
  workers = 1L,
  repeats = 40L,
  tests = TRUE,
  verbose = TRUE
)
```

Arguments

`x`	Expression of the gene of interest (GOI), one value per cell.
`y`	Composite expression of the partner genes (see [cepip_composite()]).
`donor`	Optional donor label per cell.
`id`	Optional cell identifiers (for example barcodes).
`skcs`	Add the SKCS step.
`cmrf`	Add the CMRF steps (implies ‘skcs = TRUE‘).
`validate`	Run Validation1 after CMRF.
`ip_k`	k of the GAM f: ‘"prss"‘ (default) or a fixed integer. The published SEA-AD analysis used 10.
`ip_axis`	‘"y"‘ or ‘"x"‘, the axis that splits Pre-IP from Post-IP cells.
`band`	Optional ‘cepip_band‘ object to use instead of building the band.
`kappa, bw`	Band settings (see [cepip_band()]).

y_spine	Optional composite values for the spine fit (see [cepip_band()]).
workers	Number of parallel workers (GAM k searches and the RSk-fCV gate).
repeats	Number of RSk-fCV repeats of the gate (see [cepip_cmrf()]).
tests	Run the CMRF class tests.
verbose	Print progress messages.

Details

* Always: the CEP-IP GAM with the PRSS k search ([cepip_fit()]), the EP and TREP cells ([cepip_ep()]) and the inflection point with its IPRS ([cepip_ip()]). With ‘skcs = FALSE‘ and ‘cmrf = FALSE‘ this is the published CEP-IP, except that cells are labelled Pre-IP or Post-IP on the response axis (‘ip_axis = "y"‘); set ‘ip_axis = "x"‘ for the published split on the GOI axis. * ‘skcs = TRUE‘: the Spearman-Kendall cell-level scores, the SKCS-core cells and the SKCS-core fit ([cepip_skcs()]). * ‘cmrf = TRUE‘: also SKCS, the CMRF band ([cepip_band()]), the RSk-fCV gate ([cepip_cmrf()]) and, by default, Validation1 ([cepip_validate()]). CMRF always uses the CEP-IP inflection point.

When no IP is found, the run stops after the IP step and the object reports why; no error is raised.

Value

An object of class ‘cepip‘.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, ip_k = 6)
res

# With SKCS, CMRF and Validation1 (a short gate of 5 repeats)
res2 <- cepip(d$x, d$y, donor = d$donor, id = d$id, cmrf = TRUE, repeats = 5,
              tests = FALSE)
res2
```

---

cepip_associate *Donor-level association with outcomes*

Description

For each outcome, fits ‘outcome ~ predictor + covariates‘ and ‘outcome ~ covariates‘ by ordinary least squares on the donors with complete data, after z-scoring every variable. It reports the standardized beta of the predictor, its p value, and delta R2, the R2 of the full model minus that of the covariate-only model. An outcome is never used as its own covariate. The BH false discovery rate is applied across all outcomes passed.

Usage

```
cepip_associate(
  data,
  predictor,
  outcomes,
  covariates = character(0),
  min_donors = 20L
)
```

Arguments

data	Donor table (one row per donor).
predictor	Column name of the predictor.
outcomes	Column names of the outcomes.
covariates	Column names of the covariates.
min_donors	Minimum number of complete donors for a model.

Details

In the SEA-AD analysis the predictor was the Module Post-IP percentage per donor, the covariates were age at death, sex, PMI and APOE e4 dose, and the BH family was 15 outcomes (nine clinical and pathology outcomes plus six control variables). Outcomes and covariates depend on the data available.

Value

A data frame with one row per outcome.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
dt <- cepip_donor(res)
set.seed(1)
dt$age <- rnorm(nrow(dt), 80, 5)
dt$score <- 0.05 * dt$pct_PostIP + rnorm(nrow(dt))
cepip_associate(dt, "pct_PostIP", outcomes = "score", covariates = "age",
                min_donors = 10)
```

---

cepip_band *CMRF tolerance band (Stage 1)*

Description

Builds the CMRF Stage 1 band in copula space. This step is fast, so it can be run several times to choose ‘kappa‘ and ‘bw‘ (see [plot_band()]) before running the slower Stage 2 gate in [cepip_cmrf()]. It is optional: [cepip_cmrf()] builds the band itself with ‘kappa = 0.5‘ and ‘bw = 0.1‘ when no band is given.

Usage

```
cepip_band(
  skcs,
  ip,
  kappa = 0.5,
  bw = 0.1,
  coverage = 0.95,
  spine_k = "prss",
  min_spine_cells = 50L,
  min_donors = 3L,
  y_spine = NULL,
  workers = 1L,
  verbose = FALSE
)
```

Arguments

skcs	A ‘cepip_skcs‘ object (with its SKCS-core fit).
ip	A ‘cepip_ip‘ object of the same cells.
kappa	Height of the widening at IP_u, as a fraction of delta_base.
bw	Bandwidth (standard deviation) of the Gaussian kernel, in u units.
coverage	Quantile of |v - g(u)| that sets delta_base.
spine_k	‘"prss"‘ or a fixed k for the spine.
min_spine_cells	Minimum number of SKCS-core TREP cells for the spine.
min_donors	Minimum number of donors for the donor random intercept.
y_spine	Optional composite values (one per cell, same order) whose ranks give the v coordinates of the spine fit. By default the composite of ‘skcs‘ is used. The published SEA-AD analysis fitted the spine on the composite column of its data export, which is stored in single precision; passing that column reproduces the published spine exactly.
workers	Number of parallel workers for the spine k search.
verbose	Print progress messages.

Details

* Copula coordinates over all cells: u = rank(x) / (n + 1) and v = rank(y) / (n + 1), with average ranks for ties. * IP_u = (number of cells with x <= IPx) / (n + 1) and IP_v = (number of cells with y <= IPy) / (n + 1). * The spine g(u) is a GAM, v ~ s(u, bs = "tp", k) (with the donor random intercept and inverse-donor weights when donors are known), fitted by REML (‘select = TRUE‘, ‘gamma = 1.5‘) on the SKCS-core TREP cells, with k chosen by PRSS as in [cepip_fit()]. The donor term is excluded when predicting. * delta_base is the 95 |v - g(u)| over all SKCS-core cells. * delta_adapt(u) = delta_base x (1 + kappa x K(u)), with the Gaussian kernel K(u) = exp(-(u - IP_u)^2 / (2 bw^2)), which is 1 at IP_u. The band is therefore widest at the IP, and at IP_u +- bw the widening term has decayed to exp(-0.5) = 60.65 * A cell passes Stage 1 when |v - g(u)| <= delta_adapt(u). SKCS-core cells that pass are the reference set (Original + Refined, OriRef); non-SKCS cells that pass are the Stage 2 candidates; SKCS-core cells that fail are "Original only" and leave the Module.

Value

An object of class ‘cepip_band‘.

Examples

```
d <- cepip_toy
s <- cepip_skcs(d$x, d$y, donor = d$donor, id = d$id)
ip <- cepip_ip(d$x, d$y, donor = d$donor, id = d$id)
b <- cepip_band(s, ip)
b$delta_base
table(b$cells$is_candidate)
```

---

cepip_cmrf *Copula monotone recovery filter (CMRF)*

Description

Runs CMRF Stage 2, the repeated stratified k-fold cross-validation (RSk-fCV) gate, on the Stage 1 band of [cepip_band()].

Usage

```
cepip_cmrf(
  skcs = NULL,
  ip = NULL,
  band = NULL,
  kappa = 0.5,
  bw = 0.1,
  folds = 5L,
  repeats = 40L,
  alpha = 0.05,
  seed = 42L,
  workers = 1L,
  tests = TRUE,
  nperm_jt = 10000L,
  nperm_hd = 9999L,
  R_dcor = 999L,
  verbose = FALSE
)
```

Arguments

skcs, ip	‘cepip_skcs‘ and ‘cepip_ip‘ objects of the same cells. Not needed when ‘band‘ is given.
band	Optional ‘cepip_band‘ object. When it is missing, the band is built with ‘kappa‘ and ‘bw‘.

kappa, bw	Band settings used when ‘band‘ is missing.
folds, repeats	Number of folds and of repeats of RSk-fCV.
alpha	Significance level of the gate.
seed	Base seed of the fold assignment.
workers	Number of parallel workers for the repeats.
tests	Run the class tests (needs the Suggests packages clinfun, Hmisc and energy).
nperm_jt, nperm_hd, R_dcor	Permutations of the Jonckheere-Terpstra, Hoeffding D and distance correlation tests.
verbose	Print progress messages.

Details

* Curve A is an isotonic (PAVA) fit of v on u for the reference cells (OriRef) alone; curve B is the same fit for OriRef plus the active candidates. Each curve is cross-validated with ‘folds‘ folds stratified into deciles of u, repeated ‘repeats‘ times, which gives ‘folds‘ x ‘repeats‘ test RMSE values per curve. * The gate compares B with A by a one-sided paired Wilcoxon test (B greater than A) and by dRMSE = median(B - A) / median(A) against the background variability BV = IQR(A) / median(A). * While p < ‘alpha‘ and dRMSE > BV, the active candidate with the largest |v - PAVA(u)| (PAVA fitted once on OriRef plus all candidates) is removed and curve B is refitted. The candidates left when the gate closes are the Recovered cells.

Classes: Original + Refined (OriRef; SKCS-core, inside the band), Recovered (non-SKCS, inside the band, passed the gate), Original only (SKCS-core, outside the band) and Excluded (all other cells, including candidates removed by the gate). The Module is OriRef plus Recovered.

The fold assignment of repeat r uses the seed ‘seed‘ + r x 1000 + stratum, so results do not depend on ‘workers‘.

Value

An object of class ‘cepip_cmrf‘.

Examples

```
d <- cepip_toy
s <- cepip_skcs(d$x, d$y, donor = d$donor, id = d$id)
ip <- cepip_ip(d$x, d$y, donor = d$donor, id = d$id)
b <- cepip_band(s, ip)
# a short gate of 5 repeats (the default is 40)
m <- cepip_cmrf(band = b, repeats = 5, tests = FALSE)
m$counts
```

cepip_composite *Composite expression of partner genes*

Description

The composite is the arithmetic mean of the partner genes per cell, on the scale of the input values.

Usage

```
cepip_composite(expr, partners)
```

Arguments

expr	Matrix or data frame of expression values (cells x genes).
partners	Names (or column indices) of the partner genes.

Value

A numeric vector, one value per cell.

Examples

```
d <- cepip_toy
y <- cepip_composite(d, paste0("P", 1:8))
head(y)
```

cepip_deg *Differential expression between cell groups, paired by donor*

Description

Compares two of the four groups of [cepip_groups()] (foreground against background). These four contrasts are built in: 'PostIP_vs_PreIP', 'NMPost_vs_NMPre', 'PostIP_vs_NMPost' and 'PreIP_vs_NMPre'. Other pairs of the four groups can be given as '"A_vs_B"'.

Usage

```
cepip_deg(
  x,
  expr,
  contrast = c("PostIP_vs_PreIP", "NMPost_vs_NMPre", "PostIP_vs_NMPost",
    "PreIP_vs_NMPre"),
  unit = c("donor", "cell"),
  min_pct = 0.05,
  min_cells_donor = 3L,
  min_donors = 5L,
```

```
  fdr = 0.05,
  lfc = 0.25,
  min_cells = 10L,
  verbose = FALSE
)
```

Arguments

x	A ‘cepip‘, ‘cepip_cmrf‘ or ‘cepip_ip‘ object.
expr	Log1p-normalized expression, genes x cells, with cell ids as column names (the ids used in the analysis) and gene names as row names.
contrast	One or more contrasts.
unit	‘"donor"‘ or ‘"cell"‘.
min_pct, min_cells_donor, min_donors, fdr, lfc	See Details.
min_cells	Minimum number of cells per group.
verbose	Print progress messages.

Details

Donor unit (default): 1. Genes expressed (value > 0) in at least ‘min_pct‘ of the foreground or of the background cells are kept. 2. Donors with at least ‘min_cells_donor‘ cells in both groups are paired; at least ‘min_donors‘ paired donors are needed. 3. For each donor and gene, log2FC = log2(mean(expm1(x_fg)) + 1) - log2(mean(expm1(x_bg)) + 1), on log1p-normalized values. 4. Genes that change in fewer than 5 gene is tested by a Wilcoxon signed-rank test of its non-zero donor log2FCs against 0 (at least 3 donors); BH is applied across genes. avg_log2FC is the mean over all paired donors and r_rb the matched-pairs rank-biserial correlation.

The cell unit compares the cells directly with a Wilcoxon rank-sum test. A gene is significant when its BH-adjusted p < ‘fdr‘ and |avg_log2FC| >= ‘lfc‘. Which groups to compare, and the thresholds, depend on the question and the data; the defaults are those of the SEA-AD analysis.

Value

A named list with one data frame per contrast (‘NULL‘ when a contrast cannot be tested), with the attribute ‘info‘ giving the cell and donor counts.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
expr <- t(as.matrix(d[, c("x", paste0("P", 1:8))]))
colnames(expr) <- d$id
deg <- cepip_deg(res, expr, contrast = "PostIP_vs_PreIP")
deg$PostIP_vs_PreIP
```

cepip_donor *Per-donor share of cells in each group*

Description

Counts each donor's cells per group and divides by the donor's total cell count. The total should include every cell of the donor in the population, also cells left out of the analysis (for example cells with GOI = 0), so 'n_total' can be given separately. Donors in 'n_total' without any cell in a group get 0.

Usage

```
cepip_donor(x, n_total = NULL)
```

Arguments

x	A 'cepip', 'cepip_cmrf' or 'cepip_ip' object.
n_total	Optional named vector of total cells per donor. By default the number of analysed cells per donor.

Value

A data frame with one row per donor: total cells, counts and percentages of PostIP, PreIP, NMPost and NMPre, and the Module percentage.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
head(cepip_donor(res))
```

cepip_ep *Cell explanatory power and TREP cells*

Description

Computes the explanatory power of each cell, EP_i = 1 - (y_i - f(x_i))^2 / (y_i - mean(y))^2, where f is the population-level curve (donor term excluded) and the null is the plain mean of y. The top round(n x DE) cells by EP are the top-ranked explanatory power (TREP) cells. DE here is the weighted population-level DE, 1 - RSS_w / TSS_w, from the same curve; without donors it equals the usual deviance explained.

Usage

```
cepip_ep(fit)
```

Arguments

fit	A ‘cepip_fit‘ object.

Value

A data frame with one row per cell: id, x, y, donor, fitted value, EP, EP rank and TREP flag. The attribute ‘de_pop‘ holds the DE used for the TREP count.

Examples

```
d <- cepip_toy
f <- cepip_fit(d$x, d$y, donor = d$donor, id = d$id)
ep <- cepip_ep(f)
head(ep)
sum(ep$is_TREP)
```

---

cepip_example *Download the SEA-AD example data*

Description

Downloads the example data used in the paper from Hugging Face once and keeps them in a local cache. The data come from the Seattle Alzheimer's Disease Brain Cell Atlas (SEA-AD) and are subject to the Allen Institute Terms of Use (noncommercial research use, with citation); see <https://alleninstitute.org/legal/terms-of-use> and <https://alleninstitute.org/legal/citation-policy>.

Usage

```
cepip_example(
  which = c("pvm3", "hkg8", "donors", "deg_NeoC", "deg_NonNeoC"),
  cache = tools::R_user_dir("cepip", "cache"),
  repo = "kahkengwong/cepip_example",
  quiet = FALSE
)
```

Arguments

which	Which file to load.
cache	Cache folder.
repo	Hugging Face dataset repository.
quiet	Suppress the download progress.

Details

Files: * ‘"pvm3"‘: the 24,463 Micro-PVM_3-SEAAD cells with MT-CO3 > 0 (NeoC 13,331, NonNeoC 11,132): barcode, region set, region, donor, MT-CO3, the eight mitochondrial partner genes, the composite of the data export (‘composite_part8‘), ACTB and the eight other housekeeping genes, all as ln(CP10K + 1). * ‘"hkg8"‘: the 2,511 cells of the housekeeping-gene control arm of the paper (ACTB > 0 cells of the M4 arm): ACTB, the eight housekeeping genes and their export composite. * ‘"donors"‘: donor metadata (covariates and outcomes) and the total number of Micro-PVM_3-SEAAD cells per donor and region set. * ‘"deg_NeoC"‘, ‘"deg_NonNeoC"‘: sparse ln(CP10K + 1) matrices (genes x cells) of the PVM_3 cells for [cepip_deg()].

The files are kept in ‘cache‘ (by default the user cache folder of R, see [tools::R_user_dir()]), so they are downloaded only once. [cepip_example_clear()] deletes them. When the download fails (for example without an internet connection), the function gives a message and returns ‘NULL‘.

Value

A data frame (or a sparse matrix for the DEG files), or ‘NULL‘ when the download fails.

Examples

```
hk <- cepip_example("hkg8", cache = file.path(tempdir(), "cepip_cache"))
if (!is.null(hk)) dim(hk)
```

---

cepip_example_clear *Delete the cached example data*

Description

Deletes the files that [cepip_example()] saved in the cache folder, and the folder itself when it is then empty.

Usage

```
cepip_example_clear(cache = tools::R_user_dir("cepip", "cache"))
```

Arguments

cache	Cache folder (the same as in [cepip_example()]).

Value

The paths of the deleted files, invisibly.

Examples

```
cepip_example_clear(cache = file.path(tempdir(), "cepip_cache"))
```

`cepip_fit` *Fit the CEP-IP GAM*

Description

Fits ‘y ~ x + s(x, bs = "tp", k)‘ by REML with ‘select = TRUE‘ and ‘gamma = 1.5‘, as in the published CEP-IP. When ‘donor‘ is given, a donor random intercept ‘s(donor, bs = "re")‘ is added (if there are at least ‘min_donors‘ donors) and each cell is weighted by 1 / (cells of its donor), so that every donor carries the same total weight.

Usage

```
cepip_fit(
  x,
  y,
  donor = NULL,
  id = NULL,
  k = "prss",
  k_max = 10L,
  gamma = 1.5,
  select = TRUE,
  min_donors = 3L,
  workers = 1L,
  verbose = FALSE
)
```

Arguments

`x`	Expression of the gene of interest (GOI), one value per cell.
`y`	Composite expression of the partner genes (see [cepip_composite()]).
`donor`	Optional donor label per cell.
`id`	Optional cell identifiers (for example barcodes).
`k`	‘"prss"‘ for the PRSS search, or a fixed integer k.
`k_max`	Largest k in the PRSS search.
`gamma, select`	Passed to [mgcv::gam()].
`min_donors`	Minimum number of donors for the donor random intercept.
`workers`	Number of parallel workers for the k search.
`verbose`	Print progress messages.

Details

The basis dimension k is chosen by the lowest penalized residual sum of squares (PRSS) over k = 3 to min(‘k_max‘, unique x - 1), with ties going to the smaller k. The published pipeline ran a second phase of up to five iterations as a sanity check of this search; it cannot change the selected k because the first phase already fits every k in the range and each fit is deterministic, so it is not repeated here.

The package does not transform the data. The published CEP-IP analyses used log2(value + 1) of Seurat log-normalized data; the SEA-AD analysis used ln(CP10K + 1) directly.

Value

An object of class 'cepip_fit': the model, the selected k, the PRSS table, the deviance explained (DE) and the fitting frame.

Examples

```
d <- cepip_toy
f <- cepip_fit(d$x, d$y, donor = d$donor, id = d$id)
f$k
f$de
```

---

cepip_groups *Cell groups for downstream analyses*

Description

Returns the four groups used by [cepip_donor()] and [cepip_deg()]: Module Post-IP ('PostIP'), Module Pre-IP ('PreIP'), non-Module Post-IP ('NMPost') and non-Module Pre-IP ('NMPre'). With CMRF, the Module is OriRef plus Recovered; without CMRF (the published CEP-IP), the TREP cells take the place of the Module. The IP side follows the IP axis of the analysis (by default y < IPy is Pre-IP).

Usage

```
cepip_groups(x)
```

Arguments

x	A 'cepip' object, a 'cepip_cmrf' object, or a 'cepip_ip' object.

Value

A data frame with the cell id, donor, module flag, IP side and group.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
table(cepip_groups(res)$group)
```

`cepip_ip` *Inflection point (IP) and IP reliability score (IPRS)*

Description

Finds the CEP-IP inflection point.

Usage

```
cepip_ip(
  x,
  y,
  donor = NULL,
  id = NULL,
  ip_k = "prss",
  resid_k = 20L,
  grid = 2000L,
  z = 1.96,
  ip_axis = c("y", "x"),
  min_cells = 10L,
  gamma = 1.5,
  select = TRUE,
  min_donors = 3L,
  fit = NULL,
  verbose = FALSE
)
```

Arguments

`x`	Expression of the gene of interest (GOI), one value per cell.
`y`	Composite expression of the partner genes (see [cepip_composite()]).
`donor`	Optional donor label per cell.
`id`	Optional cell identifiers (for example barcodes).
`ip_k`	k of the GAM f: ‘"prss"‘ (default) or a fixed integer. The published SEA-AD analysis used 10.
`resid_k`	Largest k of the residual GAM R(x).
`grid`	Number of grid points for R(x).
`z`	Multiplier of the standard error for the R(x) bands.
`ip_axis`	‘"y"‘ or ‘"x"‘, the axis that splits Pre-IP from Post-IP cells.
`min_cells`	Minimum number of top-EP cells for the residual GAM.
`gamma, select`	Passed to [mgcv::gam()].
`min_donors`	Minimum number of donors for the donor random intercept.
`fit`	Optional ‘cepip_fit‘ of the same cells, to reuse instead of refitting.
`verbose`	Print progress messages.

Details

1. The CEP-IP GAM f is fitted on all cells (‘ip_k‘), and the top round(n x DE) cells by explanatory power are kept (see [cepip_ep()]). 2. The residuals y - f(x) of these cells are modelled by a residual GAM R(x) = s(x, bs = "tp", k = min(‘resid_k‘, unique x - 1)), fitted by REML (with the donor random intercept and inverse-donor weights when ‘donor‘ is given; the donor term is excluded when predicting). 3. R(x) is evaluated on a grid of ‘grid‘ points. IPx is the first negative-to-positive zero crossing, located by linear interpolation between the two grid points. 4. The 95 R(x) - 1.96 se(x) (upper), found by Brent’s method ([stats::uniroot()]) from the band sign change nearest the IP. When a band has no sign change, the grid point where the band is closest to zero on that side is used. 5. IPy = f(IPx), the population-level curve at IPx. 6. IPRS = (C1 + C2 + C3 + C4 + C5) / 5: CI precision, a one-sided test of post-IP against pre-IP residuals, the slope of R(x) at the IP, the ribbon width of R(x), and the smoothness of R(x) combined with the share of post-IP residuals that are positive.

Cells are labelled Pre-IP or Post-IP on the axis given by ‘ip_axis‘. The default ‘"y"‘ splits on the response, y < IPy. The published CEP-IP analyses split on x (x < IPx), which ‘ip_axis = "x"‘ reproduces.

Value

An object of class ‘cepip_ip‘. ‘has_ip‘ is ‘FALSE‘ when R(x) has no negative-to-positive crossing; the Pre-IP and Post-IP labels are then ‘NA‘.

Examples

```
d <- cepip_toy
ip <- cepip_ip(d$x, d$y, donor = d$donor, id = d$id)
ip$ip_x
ip$iprs
```

---

cepip_save *Save the tables and plots of one arm*

Description

Writes all tables and plots of one arm into one folder, with no subfolders. Several region sets (or other subsets) of the same arm go into the same folder, each with its name in the file and sheet names.

Usage

```
cepip_save(
  results,
  dir,
  arm = basename(dir),
  extras = NULL,
  plots = TRUE,
  formats = c("png", "pdf"),
```

```
  log = NULL,
  max_cell_rows = 20000L
)
```

Arguments

results	A ‘cepip‘ object, or a named list of ‘cepip‘ objects (one per region set, for example ‘list(NeoC = a, NonNeoC = b)‘).
dir	Output folder (created if needed).
arm	Arm name used in the workbook names. Defaults to the folder name.
extras	Optional named list (same names as ‘results‘) of lists with any of ‘screen‘ ([cepip_screen()]), ‘donor‘ ([cepip_donor()]), ‘deg‘ ([cepip_deg()]), ‘associate‘ ([cepip_associate()]) and ‘outcomes‘ (a donor table for [plot_donor()]) for that region set.
plots	Write the plots.
formats	Plot file formats.
log	Optional character vector written to ‘00_log.txt‘.
max_cell_rows	Row limit for keeping cell-level sheets in the main workbook.

Details

* ‘00_settings.txt‘: settings, package versions and the summary of each result. ‘00_log.txt‘: the text passed in ‘log‘ (for example the console output of the run, captured with [utils::capture.output()] or [sink()]). * ‘<arm>_cepip_tables.xlsx‘: one sheet per table, numbered in pipeline order. When the cell-level tables are large (more than ‘max_cell_rows‘ rows in total), they go into a second workbook, ‘<arm>_cepip_cells.xlsx‘. * Plots as .png and .pdf, numbered in pipeline order.

Value

The paths of the files written, invisibly.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
out <- file.path(tempdir(), "toy")
files <- cepip_save(res, out, plots = FALSE)
basename(files)
unlink(out, recursive = TRUE)
```

`cepip_screen` *Spearman-Kendall dual-filter screen for partner genes*

Description

Screens candidate genes of interest (GOIs) for partner genes (dual-filter genes, DFGs) across the cells of one population, as in the published CEP-IP and the SEA-AD analysis.

Usage

```
cepip_screen(
  expr,
  goi = NULL,
  donor = NULL,
  strata = NULL,
  min_pct = 0.25,
  min_mean = 0.3,
  min_partner_cells = 10L,
  min_donor_cells = 5L,
  rs = 0.6,
  tau = 0.5,
  negative = TRUE,
  max_cells = NULL,
  seed = 42L,
  goi_block = 500L,
  partner_block = 2000L,
  verbose = FALSE
)
```

Arguments

`expr`	Expression matrix, genes x cells (dense or sparse), with gene names as row names.
`goi`	Candidate GOIs. 'NULL' screens every gene.
`donor`	Optional donor label per cell.
`strata`	Optional stratum per cell (for example region) for 'max_cells'.
`min_pct`, `min_mean`	GOI expression gate.
`min_partner_cells`	Minimum number of expressing cells of a partner.
`min_donor_cells`	Minimum number of cells per donor.
`rs`, `tau`	Spearman and Kendall thresholds.
`negative`	Keep negative DFGs.
`max_cells`	Optional number of cells to screen.

seed	Seed of the subsample.
goi_block, partner_block	Block sizes for the Spearman step (memory).
verbose	Print progress messages.

Details

1. Donors with fewer than ‘min_donor_cells‘ cells are dropped. 2. A GOI is screened if it is expressed (value > 0) in at least ‘min_pct‘ of the cells and its mean over the expressing cells is at least ‘min_mean‘. In the SEA-AD data (ln(CP10K + 1)) these defaults kept 2,500 to 4,000 genes per population. Adjust them for other data and normalizations. 3. Partner genes must be expressed in at least ‘min_partner_cells‘ cells. 4. Spearman rs is computed for every GOI-partner pair (average ranks, zeros included). Kendall tau-b is computed for the pairs with |rs| >= ‘rs‘. 5. A partner is a DFG when rs >= ‘rs‘ and tau-b >= ‘tau‘ (positive), or, with ‘negative = TRUE‘, rs <= -‘rs‘ and tau-b <= -‘tau‘ (negative).

GOIs are ranked by the number of DFGs, then by the largest rs. The choice of the GOI and of the DFG panel for [cepip_composite()] is left to the user.

Run time grows with cells x GOIs x partners. For large populations, ‘max_cells‘ screens a random subsample (stratified by ‘strata‘ when given). Kendall tau-b uses ‘pcaPP::cor.fk()‘ when pcaPP is installed (fast, same result), else [stats::cor()].

Value

An object of class ‘cepip_screen‘ with ‘ranking‘ (one row per screened GOI with at least one DFG), ‘partners‘ (one row per GOI-DFG pair), ‘gate‘ (the GOI gate of every candidate) and ‘cells‘ (the cells screened).

Examples

```
d <- cepip_toy
expr <- t(as.matrix(d[, c("x", paste0("P", 1:8))]))
colnames(expr) <- d$id
sc <- cepip_screen(expr, goi = "x", donor = d$donor)
sc$ranking
head(sc$partners)
```

---

cepip_skcs *Spearman-Kendall cell-level scores (SKCS)*

Description

Splits the global Spearman and Kendall correlations between x and y into per-cell contributions.

Usage

```
cepip_skcs(
  x,
  y,
  donor = NULL,
  id = NULL,
  s_min = 0.05,
  k_min = 0.5,
  fit = TRUE,
  verbose = FALSE,
  ...
)
```

Arguments

x	Expression of the gene of interest (GOI), one value per cell.
y	Composite expression of the partner genes (see [cepip_composite()]).
donor	Optional donor label per cell.
id	Optional cell identifiers (for example barcodes).
s_min, k_min	SKCS-core gates.
fit	Refit the GAM on the SKCS-core cells and define TREP cells there.
verbose	Print progress messages.
...	Passed to [cepip_fit()] for the SKCS-core fit (for example ‘k‘).

Details

* S_i = (r_x,i - mean r_x)(r_y,i - mean r_y) / (n - 1), from average ranks; S_i,norm = S_i / ((n - 1) / 4). * K_i = (C_i - D_i) / sqrt((n - 1 - t_x,i)(n - 1 - t_y,i)), where C_i and D_i are the concordant and discordant pairs of cell i and t_x,i and t_y,i its ties in x and in y (tau-b form).

SKCS-core cells pass both gates, S_i,norm > ‘s_min‘ and K_i > ‘k_min‘. The CEP-IP GAM is then refitted on the SKCS-core cells (with the same settings as [cepip_fit()]) and the TREP cells are defined within the core.

The gates S_i,norm > 0.05 and K_i > 0.5 were introduced in the SEA-AD analysis; the published CEP-IP did not use cell-level scores.

Value

An object of class ‘cepip_skcs‘: a per-cell table (‘cells‘) with S_i, S_i,norm, K_i and the SKCS-core flag, the global rho and tau, and, when ‘fit = TRUE‘, the SKCS-core fit and its EP and TREP table.

Examples

```
d <- cepip_toy
s <- cepip_skcs(d$x, d$y, donor = d$donor, id = d$id)
s$n_core
```

cepip_toy *Simulated example data*

Description

1,200 simulated cells from 12 donors with a gene of interest ‘x‘, eight partner genes ‘P1‘ to ‘P8‘ and their composite ‘y‘ (the row mean of the partners). The data are simulated (see ‘data-raw/make_toy.R‘) and run through the whole method in under a minute.

Usage

```
cepip_toy
```

Format

A data frame with 1,200 rows and columns ‘id‘, ‘donor‘, ‘region‘, ‘x‘, ‘P1‘ to ‘P8‘ and ‘y‘.

cepip_validate *Validate the CMRF Module by refitting the CEP-IP GAM (Validation1)*

Description

Refits the CEP-IP GAM (same settings as [cepip_fit()], k by PRSS) on the Module cells (OriRef plus Recovered), recomputes EP and the TREP cells, and labels every Module cell Pre-IP or Post-IP on the IP axis. This gives the Recovered cells their own four-group labels. The agreement between the refitted and the original four-group labels of the OriRef cells is reported.

Usage

```
cepip_validate(cmrf, min_donors = 3L, workers = 1L, verbose = FALSE)
```

Arguments

cmrf	A ‘cepip_cmrf‘ object.
min_donors	Minimum number of donors for the donor random intercept.
workers	Number of parallel workers for the k search.
verbose	Print progress messages.

Details

The refitted labels are kept in ‘group_ref‘. Plots of the CMRF classes use the original labels (‘group_orig‘), because the Original-only cells have no refitted label.

Value

An object of class 'cepip_validate': the refit, the per-cell table of the Module cells, the label agreement and the cross-tabulation.

Examples

```
d <- cepip_toy
# CMRF with a short gate of 5 repeats (the default is 40)
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, cmrf = TRUE, repeats = 5,
             tests = FALSE, verbose = FALSE)
v <- cepip_validate(res$cmrf)
v$de
v$agreement
```

---

plot_associate *Forest plot of donor-level associations*

Description

The standardized beta of the predictor with its 95 [cepip_associate()], in the order given. Point size shows delta R2; filled points have FDR < 0.05. delta R2 and the FDR are written beside each row.

Usage

```
plot_associate(assoc, title = NULL)
```

Arguments

assoc	Output of [cepip_associate()].
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
dt <- cepip_donor(res)
set.seed(1)
dt$score <- 0.05 * dt$pct_PostIP + rnorm(nrow(dt))
a <- cepip_associate(dt, "pct_PostIP", outcomes = "score", min_donors = 10)
plot_associate(a)
```

plot_band *Plot the formation of the CMRF band (Figure 6)*

Description

Panels: (1) the spine fitted on the SKCS-core TREP cells, (2) all SKCS-core cells in their four groups, (3) the flat delta_base band, (4) the adaptive delta_adapt band over delta_base, a QC panel zoomed to IP_u +- 0.25 (QC1 the width at IP_u, QC2 the widths at IP_u +- bw, where 60.65 percent of the widening is left, and QC3 the tangents of the band edges at IP_u +- bw), (5) the Stage 1 candidates, (6) the SKCS-core cells outside the band (Original only) and, when ‘cmrf‘ is given, (7) the cells recovered by the gate and those it removed and (8) the final Module. This plot is useful to choose ‘kappa‘ and ‘bw‘ with [cepip_band()] before running [cepip_cmrf()].

Usage

```
plot_band(band, cmrf = NULL, title = NULL)
```

Arguments

band	A ‘cepip_band‘ object, or a ‘cepip_cmrf‘ or ‘cepip‘ object.
cmrf	Optional ‘cepip_cmrf‘ object; adds panels 7 and 8.
title	Optional overall title.

Value

A gtable of class ‘cepip_figure‘ (printing it draws it; it can also be drawn with [grid::grid.draw()]). The attribute ‘cepip_size‘ gives a suitable width and height in inches.

Examples

```
d <- cepip_toy
# CMRF with a short gate of 5 repeats (the default is 40)
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, cmrf = TRUE, repeats = 5,
             tests = FALSE, verbose = FALSE)
plot_band(res)
```

plot_copula *Plot the cells and the CMRF band in copula space*

Description

(u, v) of all cells, the SKCS-core cells coloured by their four groups, the spine g(u), the flat delta_base band, the adaptive delta_adapt band and the IP_u and IP_v lines.

Usage

```
plot_copula(band, title = NULL)
```

Arguments

band	A ‘cepip_band‘ object, or a ‘cepip_cmrf‘ or ‘cepip‘ object.
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
# CMRF with a short gate of 5 repeats (the default is 40)
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, cmrf = TRUE, repeats = 5,
             tests = FALSE, verbose = FALSE)
plot_copula(res)
```

---

plot_deg *Volcano plot of one differential expression contrast*

Description

avg_log2FC (x) against -log10 of the BH-adjusted p (y) for one contrast of [cepip_deg()]. Significant genes are coloured by direction and the top 10 are labelled; dashed lines mark the ‘lfc‘ and ‘fdr‘ thresholds.

Usage

```
plot_deg(deg, contrast = NULL, title = NULL)
```

Arguments

deg	A ‘cepip_deg‘ object.
contrast	Name of the contrast. By default the first one that was tested.
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
expr <- t(as.matrix(d[, c("x", paste0("P", 1:8))]))
colnames(expr) <- d$id
deg <- cepip_deg(res, expr, contrast = "PostIP_vs_PreIP")
plot_deg(deg)
```

`plot_donor` *Plot the per-donor Module Post-IP percentage*

Description

With only the table of [cepip_donor()], a sorted bar chart of ‘pct_PostIP‘ per donor. With ‘data‘ and ‘outcome‘, a scatter of ‘pct_PostIP‘ against the outcome with a linear fit.

Usage

```
plot_donor(donor_table, outcome = NULL, data = NULL, title = NULL)
```

Arguments

`donor_table`	Output of [cepip_donor()], or a ‘cepip‘, ‘cepip_cmrf‘ or ‘cepip_ip‘ object (then [cepip_donor()] is run on it).
`outcome`	Optional column name of ‘data‘.
`data`	Optional donor table with a ‘donor‘ column and the outcome.
`title`	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
plot_donor(res)
```

plot_fit *Plot the CEP-IP GAM with the inflection point*

Description

GOI (x) against the composite (y). Cells are coloured by the four groups (Pre-IP or Post-IP, TREP or non-TREP) when an IP was found; otherwise TREP cells are coloured and non-TREP cells are gray. The population curve (donor term excluded) is drawn with its 95 as a shaded strip, and IPy as a horizontal line.

Usage

```
plot_fit(x, title = NULL)
```

Arguments

x	A ‘cepip‘ object (uses ‘x$fit‘, ‘x$ep‘ and ‘x$ip‘), a ‘cepip_ip‘ object, or a ‘cepip_fit‘ object (no IP lines).
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
plot_fit(res)
```

plot_module *Pitcher plot of the CMRF classes in copula space (Figure 8A)*

Description

u_SK = F(S_i,norm) (x) against v_SK = F(K_i) (y). Module cells are the OriRef cells in their four groups and the Recovered cells (purple diamonds); non-Module cells are the Excluded cells (gray) and the Original only cells (tan triangles). The S_i,norm and K_i gates are drawn in copula coordinates.

Usage

```
plot_module(cmrf, title = NULL)
```

Arguments

cmrf	A ‘cepip_cmrf‘ object, or a ‘cepip‘ object.
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
# CMRF with a short gate of 5 repeats (the default is 40)
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, cmrf = TRUE, repeats = 5,
             tests = FALSE, verbose = FALSE)
plot_module(res)
```

---

plot_pitcher *Pitcher plot of the SKCS scores*

Description

K_i (y) against S_i,norm (x) for every cell, with the S_i,norm and K_i gates. SKCS-core cells are coloured by the four groups when ‘ip‘ is given, otherwise in one colour.

Usage

```
plot_pitcher(skcs, ip = NULL, title = NULL)
```

Arguments

skcs	A ‘cepip_skcs‘ object, or a ‘cepip‘ object (its ‘skcs‘ is used, and its band or IP gives the groups).
ip	Optional ‘cepip_band‘, ‘cepip_cmrf‘ or ‘cepip_ip‘ object of the same cells, for the four-group colours.
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, skcs = TRUE, verbose = FALSE)
plot_pitcher(res)
```

plot_prss *Plot the PRSS search over the basis dimension k*

Description

Plot the PRSS search over the basis dimension k

Usage

```
plot_prss(fit, title = NULL)
```

Arguments

fit	A ‘cepip_fit‘ object, or a ‘cepip‘ or ‘cepip_ip‘ object (its fit is used).
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
plot_prss(res)
```

plot_residual *Plot the residual GAM R(x) and the IP*

Description

R(x) with its 95 (faint points), the zero line and the IPx with its CI. The subtitle gives the IPRS, its tier and the components C1 to C5.

Usage

```
plot_residual(ip, title = NULL)
```

Arguments

ip	A ‘cepip_ip‘ object, or a ‘cepip‘ object (its ‘ip‘ is used).
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, verbose = FALSE)
plot_residual(res)
```

`plot_rskfcv` *Plot the RSk-fCV gate of CMRF Stage 2 (Figure 7)*

Description

Panels for the first or the last gate iteration: (A) the fold layout of repeat 1, (B) the u strata of the pool, (C) curve A (isotonic fit on the OriRef cells) and curve B (on the pool) with the candidates' residual sticks, (D) the per-fold RMSE of repeat 1, (E) the RMSE_A and RMSE_B heatmaps (repeat x fold), (F) the RMSE distributions with the Wilcoxon p and the paired differences against BV, (G) the candidates ranked by |v - PAVA(u)|, (I) the gate convergence over all iterations and (Q) the class counts by IP side. When the class tests were run, (P) |v - g(u)| by class with the Jonckheere-Terpstra p and (O) the Recovered cells with the Hoeffding D and distance correlation p are added.

Usage

```
plot_rskfcv(cmrf, iteration = c("first", "last"), title = NULL)
```

Arguments

`cmrf`	A ‘cepip_cmrf‘ object, or a ‘cepip‘ object.
`iteration`	‘"first"‘ or ‘"last"‘.
`title`	Optional overall title.

Details

The pool of the first iteration is the OriRef cells plus all candidates; the pool of the last iteration is the OriRef cells plus the candidates still active (the Recovered cells).

Value

A gtable of class ‘cepip_figure‘ (printing it draws it; it can also be drawn with [grid::grid.draw()]). The attribute ‘cepip_size‘ gives a suitable width and height in inches.

Examples

```
d <- cepip_toy
# CMRF with a short gate of 5 repeats (the default is 40)
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, cmrf = TRUE, repeats = 5,
             tests = FALSE, verbose = FALSE)
plot_rskfcv(res, "first")
plot_rskfcv(res, "last")
```

plot_screen *Plot the partner-gene screen of one GOI*

Description

Scatter of Spearman rs (x) against Kendall tau-b (y) for the partner genes of one GOI that passed the Spearman step of [cepip_screen()]. The dual-filter genes (DFGs) are highlighted and labelled; dashed lines mark the rs and tau thresholds.

Usage

```
plot_screen(screen, goi = NULL, title = NULL)
```

Arguments

screen	A ‘cepip_screen‘ object.
goi	The GOI to show. By default the top-ranked GOI.
title	Optional plot title.

Value

A ggplot object.

Examples

```
d <- cepip_toy
expr <- t(as.matrix(d[, c("x", paste0("P", 1:8))]))
colnames(expr) <- d$id
sc <- cepip_screen(expr, goi = "x", donor = d$donor)
plot_screen(sc)
```

plot_validate *Plot the Validation1 refit of the CMRF Module*

Description

Left: the Module cells (x against y) in their refitted four groups (Recovered cells as diamonds) with the refitted population curve and its 95 OriRef cells, with the agreement in the title.

Usage

```
plot_validate(validate, cmrf = NULL, title = NULL)
```

Arguments

validate	A ‘cepip_validate‘ object, or a ‘cepip‘ object (its ‘validate‘ and ‘cmrf‘ are used).
cmrf	Optional ‘cepip_cmrf‘ object, for the IPx and IPy lines.
title	Optional overall title.

Value

A gtable of class ‘cepip_figure‘ (printing it draws it; it can also be drawn with [grid::grid.draw()]). The attribute ‘cepip_size‘ gives a suitable width and height in inches.

Examples

```
d <- cepip_toy
# CMRF with a short gate of 5 repeats (the default is 40)
res <- cepip(d$x, d$y, donor = d$donor, id = d$id, cmrf = TRUE, repeats = 5,
             tests = FALSE, verbose = FALSE)
plot_validate(res)
```

---

print.cepip_figure *Draw a multi-panel cepip figure*

Description

The multi-panel plots ([plot_band()], [plot_rskfcv()], [plot_validate()]) return a gtable with the extra class ‘cepip_figure‘, so that printing it (at the console or in a report) draws it on a new page.

Usage

```
## S3 method for class 'cepip_figure'
print(x, ...)
```

Arguments

x	A ‘cepip_figure‘ object.
...	Not used.

Value

‘x‘, invisibly.

Index

∗ **datasets**
cepip_toy, 22

cepip, 3
cepip_associate, 4
cepip_band, 5
cepip_cmrf, 7
cepip_composite, 9
cepip_deg, 9
cepip_donor, 11
cepip_ep, 11
cepip_example, 12
cepip_example_clear, 13
cepip_fit, 14
cepip_groups, 15
cepip_ip, 16
cepip_save, 17
cepip_screen, 19
cepip_skcs, 20
cepip_toy, 22
cepip_validate, 22

plot_associate, 23
plot_band, 24
plot_copula, 24
plot_deg, 25
plot_donor, 26
plot_fit, 27
plot_module, 27
plot_pitcher, 28
plot_prss, 29
plot_residual, 29
plot_rskfcv, 30
plot_screen, 31
plot_validate, 31
print.cepip_figure, 32