Package ‘peco’
March 26, 2024

Encoding UTF-8
Type Package
Version 1.14.0
Title A Supervised Approach for Predicting Cell Cycle Progression using scRNA-seq data
Description Our approach provides a way to assign continuous cell cycle phase using scRNA-seq data, and consequently, allows to identify cyclic trend of gene expression levels along the cell cycle. This package provides method and training data, which includes scRNA-seq data collected from 6 individual cell lines of induced pluripotent stem cells (iPSCs), and also continuous cell cycle phase derived from FUCCI fluorescence imaging data.
URL https://github.com/jhsiao999/peco
BugReports https://github.com/jhsiao999/peco/issues
License GPL (>= 3)
Depends R (>= 2.10)
Imports assertthat, circular, conicfit, doParallel, foreach, genlasso (>= 1.4), graphics, methods, parallel, scater, SingleCellExperiment, SummarizedExperiment, stats, utils
Suggests knitr, rmarkdown
biocViews Sequencing, RNASeq, GeneExpression, Transcriptomics, SingleCell, Software, StatisticalMethod, Classification, Visualization
VignetteBuilder knitr
RoxygenNote 6.1.1
git_url https://git.bioconductor.org/packages/peco
git_branch RELEASE_3_18
git_last_commit 52a5c2d
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-03-25
cellcyclegenes_whitfield2002

list of cell cycle genes identified in Whitfield et al. 2002.

Description

List of cell cycle genes and their associated cell cycle state as reported in Whitfield et al. 2002.

Usage

data(cellcyclegenes_whitfield2002)
**Format**

A list with the following elements

- **hgnc**: Gene symbol
- **ensembl**: ENSEMBL gene ID
- **phase**: Marker phase identified in Whitfield et al. 2002

---

**circ_dist**

*Pairwise distance between two circular variables*

**Description**

We define distance between two angles: the minimum of the differences in both clockwise and counterclockwise directions.

**Usage**

```r
circ_dist(y1, y2)
```

**Arguments**

- **y1**: A vector of angles.
- **y2**: A vector of angles.

**Value**

A vector of distances between angles.

**Author(s)**

Joyce Hsiao, Matthew Stephens

**Examples**

```r
# a vector of angles
theta_ref <- seq(0,2*pi, length.out=100)

# shift the origin of theta_ref to pi
theta_compare <- shift_origin(theta_ref, origin = pi)
mean(circ_dist(theta_ref, theta_compare))

# after rotation of angles, difference is 0 between the original # and the shifted angles
theta_compare_rotated <- rotation(ref_var=theta_ref, shift_var=theta_compare)
mean(circ_dist(theta_ref, theta_compare_rotated))
```
**cycle_npreg_insample**

*Obtain cyclic trend estimates from the training data*

---

**Description**

Estimates cyclic trends of gene expression levels using training data.

**Usage**

```r
cycle_npreg_insample(Y, theta, ncores = 2, polyorder = 2,
                      method.trend = c("trendfilter", "loess", "bspline"))
```

**Arguments**

- `theta`: A vector of angles.
- `ncores`: We use doParallel package for parallel computing.
- `polyorder`: We estimate cyclic trends of gene expression levels using nonparamtric trend filtering. The default fits second degree polynomials.
- `method.trend`: Various methods that can be applied to estimate cyclic trend of gene expression levels.

**Value**

A list with four elements:

- `Y`: Gene expression marix.
- `theta`: Vector of angles or cell cycle phases.
- `sigma_est`: Estimated standard error of the cyclic trend for each gene.
- `funs_est`: A list of functions for approximating the cyclic trends of gene express levels for each gene.

**Author(s)**

Joyce Hsiao

**See Also**

- `cycle_npreg_mstep` for estimating cyclic functions given inferred phases from `cycle_npreg_loglik`
- `cycle_npreg_outsample` for predicting cell cycle phase using parameters learned from `cycle_npreg_insample`

Other peco classifier functions: `cycle_npreg_loglik, cycle_npreg_mstep, cycle_npreg_outsample, initialize_grids`
**cycle_npreg_loglik**

**Examples**

```r
# see \code{\link{cycle_npreg_insample}}
```

---

**Infer angles or cell cycle phase based on gene expression data**

**Description**

Infer angles or cell cycle phase based on gene expression data

**Usage**

```r
cycle_npreg_loglik(Y, sigma_est, funs_est, grids = 100,
                    method.grid = c("pca", "uniform"))
```

**Arguments**

- **Y**: Gene by sample expression matrix.
- **sigma_est**: A vector of standard errors for each gene from the training data.
- **funs_est**: A vector of cyclic functions estimated for each gene from the training data.
- **grids**: number of bins to be selected along 0 to 2pi.
- **method.grid**: The approach to initialize angles in the computation. uniform creates k equally-spaced bins (grids). pca uses gene expression values to infer angles, and then use these pca-based angles to move the cells to the closest bin (as defined by uniform).

**Value**

A list with the following three elements:

- **cell_times_est**: Inferred angles or cell cycle phases, NOT ordered.
- **loglik_est**: Log-likelihood estimates for each gene.
- **prob_per_cell_by_celltimes**: Probabilities of each cell belong to each bin.

**Author(s)**

Joyce Hsiao

**See Also**

`initialize_grids` for selecting angles in `cycle_npreg_loglik`, `cycle_npreg_mstep` for estimating cyclic functions given inferred phases from `cycle_npreg_loglik`, `cycle_npreg_outsample` for predicting cell cycle phase using parameters learned from `cycle_npreg_insample`

Other peco classifier functions: `cycle_npreg_insample`, `cycle_npreg_mstep`, `cycle_npreg_outsample`, `initialize_grids`
cycle_npreg_mstep  Estimate parameters of the cyclic trends

Description

This is used in both cycle_npreg_insample (training data fitting) and cycle_npreg_outsample (testing data prediction) to estimate cyclic trends of gene expression values. The function outputs for each gene standard error of the cyclic trend, cyclic function, and the estimated expression levels given the cyclic function.

Usage

```r
cycle_npreg_mstep(Y, theta, method.trend = c("trendfilter", "loess", "bspline"), polyorder = 2, ncores = 2)
```

Arguments

- **Y**Gene by sample expression matrix (log2CPM).
- **theta**Observed cell times.
- **method.trend**How to estimate cyclic trend of gene expression values? We offer three options: 'trendfilter' (fit_trendfilter()), 'loess' (fit_loess()) and 'bsplines' (fit_bspline()). 'trendfilter' provided the best fit in our study. But 'trendfilter' uses cross-validation and takes some time. Therefore, we recommend using bspline for quick results.
- **polyorder**We estimate cyclic trends of gene expression levels using nonparametric trend filtering. The default fits second degree polynomials.
- **ncores**How many computing cores to use? We use doParallel package for parallel computing.

Value

A list with the following elements:

- **Y**Input gene expression data.
- **theta**Input angles.
- **mu_est**Estimated expression levels given the cyclic function for each gene.
- **sigma_est**Estimated standard error of the cyclic trends for each gene
- **funs**Estimated cyclic functions

Author(s)

Joyce Hsiao
**cycle_npreg_outsample**

**See Also**

*cycle_npreg_insample* for estimating cyclic functions given known phases from training data, *cycle_npreg_outsample* for predicting cell cycle phase using parameters learned from *cycle_npreg_insample*

Other peco classifier functions: *cycle_npreg_insample, cycle_npreg_loglik, cycle_npreg_outsample, initialize_grids*

---

**Description**

Apply the estimates of cycle_npreg_insample to another gene expression dataset to infer an angle or cell cycle phase for each cell.

**Usage**

```r
cycle_npreg_outsample(Y_test, sigma_est, funs_est,
method.trend = c("trendfilter", "loess", "bspline"), normed = TRUE,
polyorder = 2, method.grid = "uniform", ncores = 2, grids = 100,
get_trend_estimates = FALSE)
```

**Arguments**

- **Y_test**: A SingleCellExperiment object.
- **sigma_est**: Input from training data. A vector of gene-specific standard error of the cyclic trends.
- **funs_est**: Input from training data. A vector of cyclic functions estimating cyclic trends.
- **method.trend**: Various methods that can be applied to estimate cyclic trend of gene expression levels.
- **normed**: Is the data already normalized? TRUE or FALSE.
- **polyorder**: We estimate cyclic trends of gene expression levels using nonparametric trend filtering. The default fits second degree polynomials.
- **method.grid**: Method for defining bins along the circle.
- **ncores**: We use doParallel package for parallel computing.
- **grids**: number of bins to be selected along 0 to 2pi.
- **get_trend_estimates**: To re-estimate the cyclic trend based on the predicted cell cycle phase or not (T or F). Default FALSE. This step calls trendfilter and is computationally intensive.
Value

A list with the following elements:

- **Y**  
The input gene expression matrix.

- **cell_times_est**  
Inferred angles or cell cycle phases, NOT ordered.

- **loglik_est**  
Log-likelihood estimates for each gene.

- **cell_times_reordered**  
The inferred angles reordered (in ascending order).

- **Y_reordered**  
The input gene expression matrix reordered by cell_times_reordered.

- **sigma_reordered**  
Estimated standard error of the cyclic trend for each gene, reordered by cell_times_reordered.

- **funs_reordered**  
A list of functions for approximating the cyclic trends of gene expression levels for each gene, reordered by cell_times_reordered.

- **mu_reordered**  
Estimated cyclic trend of gene expression values for each gene, reordered by cell_times_reordered.

- **prob_per_cell_by_celltimes**  
Probabilities of each cell belong to each bin.

See Also

cycle_npreg_outsample for obtaining parameters for cyclic functions from training data, cycle_npreg_loglik for log-likelihood at angles between 0 to 2π, initialize_grids for selecting angles in cycle_npreg_loglik, cycle_npreg_mstep for estimating cyclic functions given inferred phases from cycle_npreg_loglik

Other pco classifier functions: cycle_npreg_insample, cycle_npreg_loglik, cycle_npreg_mstep, initialize_grids

Examples

```r
# import data
library(SingleCellExperiment)
data(sce_top101genes)

# select top 5 cyclic genes
sce_top5 <- sce_top101genes[order(rowData(sce_top101genes)$pve_fucci, decreasing = TRUE)[1:5],]

# Select samples from NA18511 for our prediction example
coldata <- colData(sce_top5)
which_samples_train <- rownames(coldata)[coldata$chip_id != "NA18511"]
which_samples_predict <- rownames(coldata)[coldata$chip_id == "NA18511"]

# learning cyclic functions of the genes using our training data
sce_top5 <- data_transform_quantile(sce_top5)
expr_quan <- assay(sce_top5, "cpm_quantNormed")
Y_train <- expr_quan[, colnames(expr_quan) %in% which_samples_train]
theta_train <-
  coldata$theta_shifted[rownames(coldata) %in% which_samples_train]
names(theta_train) <-
```
rownames(coldata)[rownames(coldata) %in% which_samples_train]

# obtain cyclic function estimates
model_5genes_train <- cycle_npreg_ins amples(Y = Y_train,
    theta = theta_train,
    polyorder=2,
    ncores=2,
    method.trend="trendfilter")

# predict cell cycle
model_5genes_predict <- cycle_npreg_ ou sample(
    Y_test=sce_top5[,colnames(sce_top5) %in% which_samples_predict],
    sigma_est=model_5genes_train$sigma_est,
    funs_est=model_5genes_train$funs_est,
    method.trend="trendfilter",
    ncores=2,
    get_trend_estimates=FALSE)

# estimate cyclic gene expression levels given cell cycle for each gene
predict_cyclic <-
    fit_cyclical_many(Y=assay(model_5genes_predict$Y,"cpm_quantNormed"),
        theta=colData(model_5genes_predict$Y)$cellcycle_peco)
all.equal(names(predict_cyclic[[2]]), colnames(predict_cyclic[[1]]))

par(mfrow=c(2,3), mar=c(4,4,3,1))
for (g in seq_along(rownames(model_5genes_predict$Y))) {
    plot(assay(model_5genes_predict$Y,"cpm_quantNormed")[
        rownames(model_5genes_predict$Y)[g],],
    x=colData(model_5genes_predict$Y)$cellcycle_peco, axes=FALSE,
    xlab="FUCCI phase",
    ylab="Predicted phase")
    points(y=predict_cyclic$cellcycle_function[ [
        rownames(model_5genes_predict$Y)[g]]( 
        seq(0, 2*pi, length.out = 100)),
    x=seq(0, 2*pi, length.out = 100),
    pch=16, col="royalblue")
    axis(2); axis(1,at=c(0,pi/2, pi, 3*pi/2, 2*pi),
    labels=c(0,expression(pi/2), expression(pi),
    expression(3*pi/2), expression(2*pi)))
    abline(h=0, lty=1, col="black", lwd=.7)
    title(rownames(model_5genes_predict$Y_reordered)[g])
}

title("Predicting cell cycle phase for NA18511", outer=TRUE)
**fit_bspline**

*Use bsplines to cyclic trend of gene expression levels*

**Description**

Use bsplines to cyclic trend of gene expression levels

**Usage**

`fit_bspline(yy, time)`
Arguments

yy A vector of gene expression values for one gene. The expression values are assumed to have been normalized and transformed to standard normal distribution.
time A vector of angels (cell cycle phase).

Value

A list with one element, pred.yy, giving the estimated cyclic trend.

Author(s)

Joyce Hsiao

Examples

```r
library(SingleCellExperiment)
data(sce_top101genes)

# select top 10 cyclic genes
tsce_top10 <- sce_top101genes[order(rowData(sce_top101genes)$pve_fucci, decreasing=TRUE)[1:10],]
coldata <- colData(sce_top10)

# cell cycle phase based on FUCCI scores
theta <- coldata$theta
names(theta) <- rownames(coldata)

# normalize expression counts
sce_top10 <- data_transform_quantile(sce_top10, ncores=2)
exprs_quant <- assay(sce_top10, "cpm_quantNormed")

# order FUCCI phase and expression
theta_ordered <- theta[order(theta)]
yy_ordered <- exprs_quant[1, names(theta_ordered)]

fit <- fit_bspline(yy_ordered, time=theta_ordered)

plot(x=theta_ordered, y=yy_ordered, pch=16, cex=.7, axes=FALSE,
ylab="quantile-normalized expression values", xlab="FUCCI phase", main = "bspline fit")
points(x=theta_ordered, y=fit$pred.yy, col="blue", pch=16, cex=.7)
axis(2, at=c(0,pi/2, pi, 3*pi/2, 2*pi), labels=c("0", expression(pi/2), expression(pi), expression(3*pi/2), expression(2*pi)))
abline(h=0, lty=1, col="black", lwd=.7)
```
Compute proportion of variance explained by cyclic trends in the gene expression levels for each gene.

Description
We applied quadratic (second order) trend filtering using the trendfilter function in the genlasso package (Tibshirani, 2014). The trendfilter function implements a nonparametric smoothing method which chooses the smoothing parameter by cross-validation and fits a piecewise polynomial regression. In more specifics: The trendfilter method determines the folds in cross-validation in a non-random manner. Every k-th data point in the ordered sample is placed in the k-th fold, so the folds contain ordered subsamples. We applied five-fold cross-validation and chose the smoothing penalty using the option lambda.1se: among all possible values of the penalty term, the largest value such that the cross-validation standard error is within one standard error of the minimum. Furthermore, we desired that the estimated expression trend be cyclical. To encourage this, we concatenated the ordered gene expression data three times, with one added after another. The quadratic trend filtering was applied to the concatenated data series of each gene.

Usage

fit_cyclical_many(Y, theta, polyorder = 2, ncores = 2)

Arguments

Y
A matrix (gene by sample) of gene expression values. The expression values are assumed to have been normalized and transformed to standard normal distribution.

theta
A vector of cell cycle phase (angles) for single-cell samples.

polyorder
We estimate cyclic trends of gene expression levels using nonparamtric trend filtering. The default fits second degree polynomials.

ncores
doParallel package is used to perform parallel computing to reduce computational time.

Value
A list containing the following objects

predict.yy
A matrix of predicted expression values at observed cell cycle.

cellcycle_peco_ordered
A vector of predicted cell cycle. The values range between 0 to 2pi

cellcycle_function
A list of predicted cell cycle functions.

pve
A vector of proportion of variance explained in each gene by the predicted cell cycle.
fit_loess

Use loess to estimate cyclic trends of expression values

Description

Use loess to estimate cyclic trends of expression values

Usage

fit_loess(yy, time)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>yy</td>
<td>A vector of gene expression values for one gene. The expression values are assumed to have been normalized and transformed to standard normal distribution.</td>
</tr>
<tr>
<td>time</td>
<td>A vector of angles (cell cycle phase).</td>
</tr>
</tbody>
</table>
**Value**

A list with one element, `pred.yy`, giving the estimated cyclic trend.

**Author(s)**

Joyce Hsiao

**Examples**

```r
library(SingleCellExperiment)
data(sce_top101genes)

# select top 10 cyclic genes
sce_top10 <- sce_top101genes[order(rowData(sce_top101genes)$pve_fucci, decreasing=TRUE)[1:10],]
coldata <- colData(sce_top10)

# cell cycle phase based on FUCCI scores
theta <- coldata$theta
names(theta) <- rownames(coldata)

# normalize expression counts
sce_top10 <- data_transform_quantile(sce_top10, ncores=2)
exprs_quant <- assay(sce_top10, "cpm_quantNormed")

# order FUCCI phase and expression
theta_ordered <- theta[order(theta)]
yy_ordered <- exprs_quant[1, names(theta_ordered)]

fit <- fit_loess(yy_ordered, time=theta_ordered)

plot(x=theta_ordered, y=yy_ordered, pch=16, cex=.7, axes=FALSE,
     ylab="quantile-normalized expression values", xlab="FUCCI phase",
     main = "loess fit")
points(x=theta_ordered, y=fit$pred.yy, col="blue", pch=16, cex=.7)
axis(2)
axis(1,at=c(0,pi/2, pi, 3*pi/2, 2*pi),
     labels=c(0, expression(pi/2), expression(pi), expression(3*pi/2),
             expression(2*pi)))
abline(h=0, lty=1, col="black", lwd=.7)
```

**Description**

We applied quadratic (second order) trend filtering using the trendfilter function in the genlasso package (Tibshirani, 2014). The trendfilter function implements a nonparametric smoothing method using trendfiltering to estimate cyclic trend of gene expression.
fit_trendfilter

which chooses the smoothing parameter by cross-validation and fits a piecewise polynomial regression. In more specifics: The trendfilter method determines the folds in cross-validation in a non-random manner. Every k-th data point in the ordered sample is placed in the k-th fold, so the folds contain ordered subsamples. We applied five-fold cross-validation and chose the smoothing penalty using the option lambda.1se: among all possible values of the penalty term, the largest value such that the cross-validation standard error is within one standard error of the minimum. Furthermore, we desired that the estimated expression trend be cyclical. To encourage this, we concatenated the ordered gene expression data three times, with one added after another. The quadratic trend filtering was applied to the concatenated data series of each gene.

Usage

fit_trendfilter(yy, polyorder = 2)

Arguments

yy
A vector of gene expression values for one gene that are ordered by cell cycle phase. Also, the expression values are normalized and transformed to standard normal distribution.

polyorder
We estimate cyclic trends of gene expression levels using nonparametric trend filtering. The default fits second degree polynomials.

Value

A list with two elements:

trend.yy
The estimated cyclic trend.

pve
Proportion of variance explained by the cyclic trend in the gene expression levels.

Author(s)

Joyce Hsiao

Examples

library(SingleCellExperiment)
data(sce_top101genes)

# select top 10 cyclic genes
sce_top10 <- sce_top101genes[order(rowData(sce_top101genes)$pve_fucci, decreasing=TRUE)[1:10],]
coldata <- colData(sce_top10)

# cell cycle phase based on FUCCI scores
theta <- coldata$theta
names(theta) <- rownames(coldata)

# normalize expression counts
sce_top10 <- data_transform_quantile(sce_top10, ncores=2)
exprs_quant <- assay(sce_top10, "cpm_quantNormed")
# order FUCCI phase and expression
theta_ordered <- theta[order(theta)]
yy_ordered <- exprs_quant[1, names(theta_ordered)]

fit <- fit_trendfilter(yy_ordered)

plot(x=theta_ordered, y=yy_ordered, pch=16, cex=.7, axes=FALSE,
     ylab="quantile-normalized expression values", xlab="FUCCI phase",
     main = "trendfilter fit")
points(x=theta Ordered, y=fit$trend.yy, col="blue", pch=16, cex=.7)
axis(2)
axis(1,at=c(0,pi/2, pi, 3*pi/2, 2*pi),
     labels=c(0,expression(pi/2), expression(pi), expression(3*pi/2),
              expression(2*pi)))
abline(h=0, lty=1, col="black", lwd=.7)

initialize_grids

For prediction, initialize grid points for cell cycle phase on a circle.

Description
For prediction, initialize grid points for cell cycle phase on a circle.

Usage
initialize_grids(Y, grids = 100, method.grid = c("pca", "uniform"))

Arguments

Y  

grids  
number of bins to be selected along 0 to 2pi.

method.grid  
The approach to initialize angles in the computation. uniform creates k equally-spaced bins (grids). pca uses gene expression values to infer angles, and then use these pca-based angles to move the cells to the closest bin (as defined by uniform).

Value
A vector of initialized angles to be used in cycle_npreg_loglik to infer angles.

Author(s)
Joyce Hsiao
intensity2circle

See Also

cycle_npreg_loglik for log-likelihood at angles between 0 to 2pi, cycle_npreg_mstep for estimating cyclic functions given inferred phases from cycle_npreg_loglik, cycle_npreg_outsample for predicting cell cycle phase using parameters learned from cycle_npreg_insample

Other peco classifier functions: cycle_npreg_insample, cycle_npreg_loglik, cycle_npreg_mstep, cycle_npreg_outsample

intensity2circle  Infer angles for each single-cell samples using fluorescence intensities

Description

We use FUCCI intensities to infer the position of the cells in cell cycle progression. The result is a vector of angles on a unit circle corresponding to the positions of the cells in cell cycle progression.

Usage

intensity2circle(mat, plot.it = FALSE, method = c("trig", "algebraic"))

Arguments

mat A matrix of two columns of summarized fluorescence intensity. Rows correspond to samples.
plot.it TRUE or FALSE. Plot the fitted results.
method The method used to fit the circle. trig uses trigonometry to transform intensity measurements from cartesian coordinates to polar coordinates. algebraic uses an algebraic approach for circle fitting, using the conicfit package.

Value

The inferred angles on unit circle based on the input intensity measurements.

Author(s)

Joyce Hsiao

Examples

# use our data
library(SingleCellExperiment)
data(sce_top101genes)

# FUCCI scores - log10 transformed sum of intensities that were # corrected for background noise
ints <- colData(sce_top101genes)[,c("rfp.median.log10sum.adjust", "gfp.median.log10sum.adjust")]
intensity2circle(ints, plot.it=TRUE, method = "trig")
model_5genes_predict

**A SingleCellExperiment object**

**Description**

Pre-computed results. Applied `cycle_npreg_outsample` and results stored in `model_5genes_train` to predict cell cycle phase for single-cell samples of NA19098. The predicted cell cycle is stored as variable `cellcycle_peco`.

**Usage**

```r
data(model_5genes_predict)
```

**Format**

A list with the following elements

- `cellcycle_peco` Predict cell cycle, the values ranged between 0 to 2pi

---

model_5genes_train

**Training model results among samples from 5 individuals.**

**Description**

Pre-computed results. Applied `cycle_npreg_insamp` to obtain gene-specific cyclic trend parameters using samples from 5 individuals.

**Usage**

```r
data(model_5genes_train)
```

**Format**

A list with the following elements

- `Y` a data.frame (gene by sample) of quantile-normalized gene expression values
- `theta` a vector of cell cycl phase values (range between 0 to 2pi)
- `sigma_est` a vector of estimated standard errors
- `funs_est` a list of estimated cyclic functions
rotation

Rotate circular variable shift_var to minimize distance between ref_var and shift_var

Description

Because the origin of the cell cycle phases is arbitrary, we transform the angles prior to computing the distance (rotation and shifting) to minimize the distance between two vectors. After this, one can apply circ_dist to compute the distance between the output value and ref_var.

Usage

rotation(ref_var, shift_var)

Arguments

ref_var A vector of reference angles.
shift_var A vector of angles to be compared to ref_var.

Value

The transformed values of shift_var after rotation and shifting.

Author(s)

Matthew Stephens

Examples

# a vector of angles
theta_ref <- seq(0,2*pi, length.out=100)

# shift the origin of theta_ref to pi
theta_compare <- shift_origin(theta_ref, origin = pi)

# rotate theta_compare in such a way that the distance # between theta_ref and theta_compare is minimized
theta_compare_rotated <- rotation(ref_var=theta_ref, shift_var=theta_compare)

par(mofrow=c(1,2))
plot(x=theta_ref, y = theta_compare)
plot(x=theta_ref, y = theta_compare_rotated)
sce_top101genes  Molecule counts of the 101 significant cyclical genes in the 888 samples analyzed in the study.

Description
A SingleCellExperiment object (require SingleCellExperiment package) including molecule count data after gene and sample filtering. The ‘colData()’ slot contains sample phenotype information and the ‘rowData()’ slot contains gene feature information.

Usage

data(sce_top101genes)

Format
A SingleCellExperiment object with 888 samples and the 101 significant cyclic genes,

theta  Inferred angles of each cell along a circle, also known as FUCCI phase.

shift_origin  Shift origin of the angles

Description
Shift origin of the angles for visualization

Usage

shift_origin(phase, origin)

Arguments

phase  A vector of angles (in radians).
origin  the new origin of the angles.

Value
A vector of angles shifted to the new origin.

Author(s)
Joyce Hsiao
Examples

```r
# make a vector of angles
theta <- seq(0, 2*pi, length.out=100)

# shift the origin of theta to pi
theta_shifted <- shift_origin(theta, origin = pi)

plot(x=theta, y = theta_shifted)
```

---

**training_human**

*Training data from 888 single-cell samples and 101 top cyclic genes*

---

**Description**

Pre-computed results. Applied `fit_cyclic_many` to 888 single-cell samples that have both normalized gene expression values and cell cycle labels to obtain training results that can be used as input for predicting cell cycle phase in other data.

**Usage**

```r
data(training_human)
```

**Format**

A list with the following elements

- `predict.yy` Estimated cyclic expression values in the training data
- `cellcycle_peco_ordered` Training labels ordered from 0 to 2pi
- `cell_cycle.function` Nonparametric function of cyclic gene expression trend obtained by trendfilter function in genlasso
- `pve` Proportion of variance explained in each gene by the cell cycle phase label
Index

* data
  cellcyclegenes_whitfield2002, 2
  model_5genes_predict, 18
  model_5genes_train, 18
  sce_top101genes, 20
  training_human, 21

* peco classifier functions
  cycle_npreg_insamp, 4
  cycle_npreg_loglik, 5
  cycle_npreg_mstep, 6
  cycle_npreg_outsample, 7
  initialize_grids, 16

  cellcyclegenes_whitfield2002, 2
  circ_dist, 3
  cycle_npreg_insamp, 4, 5, 7, 8, 17
  cycle_npreg_loglik, 4, 5, 7, 8, 17
  cycle_npreg_mstep, 4, 5, 6, 8, 17
  cycle_npreg_outsample, 4, 5, 7, 17

  data_transform_quantile, 9

  fit_bspline, 10
  fit_cyclical_many, 12
  fit_loess, 13
  fit_trendfilter, 13, 14

  initialize_grids, 4, 5, 7, 8, 16
  intensity2circle, 17

  model_5genes_predict, 18
  model_5genes_train, 18

  rotation, 19

  sce_top101genes, 20
  shift_origin, 20

  training_human, 21