Package: BUSseq (via r-universe)

June 30, 2024

Type Package

Title Batch Effect Correction with Unknow Subtypes for scRNA-seq data

Version 1.11.0

Date 2021-06-01

Description BUSseq R package fits an interpretable Bayesian hierarchical model---the Batch Effects Correction with Unknown Subtypes for scRNA seq Data (BUSseq)---to correct batch effects in the presence of unknown cell types. BUSseq is able to simultaneously correct batch effects, clusters cell types, and takes care of the count data nature, the overdispersion, the dropout events, and the cell-specific sequencing depth of scRNA-seq data. After correcting the batch effects with BUSseq, the corrected value can be used for downstream analysis as if all cells were sequenced in a single batch. BUSseq can integrate read count matrices obtained from different scRNA-seq platforms and allow cell types to be measured in some but not all of the batches as long as the experimental design fulfills the conditions listed in our manuscript.

Depends R (>= 3.6)

License Artistic-2.0

Imports SingleCellExperiment, SummarizedExperiment, S4Vectors, gplots, grDevices, methods, stats, utils

Suggests BiocStyle, knitr, BiocGenerics

LazyData true

Keywords Single-cell RNA-seq experiments; Batch effects; Model-based clustering; Integrative analysis

VignetteBuilder knitr

biocViews ExperimentalDesign, GeneExpression, StatisticalMethod, Bayesian, Clustering, FeatureExtraction, BatchEffect, SingleCell, Sequencing

URL https://github.com/songfd2018/BUSseq

BugReports https://github.com/songfd2018/BUSseq/issues

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

RemoteUrl https://github.com/bioc/BUSseq

RemoteRef HEAD

RemoteSha 976e9048755327512ef7235a51fe6090ce10c47d

Contents

`BUSseq-package` *Batch Effect Correction with Unknow Subtypes for scRNA-seq data*

Description

BUSseq R package fits an interpretable Bayesian hierarchical model—the Batch Effects Correction with Unknown Subtypes for scRNA seq Data (BUSseq)—to correct batch effects in the presence of unknown cell types. BUSseq is able to simultaneously correct batch effects, clusters cell types, and takes care of the count data nature, the overdispersion, the dropout events, and the cell-specific sequencing depth of scRNA-seq data. After correcting the batch effects with BUSseq, the corrected value can be used for downstream analysis as if all cells were sequenced in a single batch. BUSseq can integrate read count matrices obtained from different scRNA-seq platforms and allow cell types to be measured in some but not all of the batches as long as the experimental design fulfills the conditions listed in our manuscript.

Author(s)

Fangda Song [aut, cre] (<https://orcid.org/0000-0001-6007-3517>), Ga Ming Chan [aut], Yingying Wei [aut] (<https://orcid.org/0000-0003-3826-336X>)

Maintainer: Fangda Song <sfd1994895@gmail.com>

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
######################################
# Apply BUSseq to the Simulation Data #
######################################
library(SingleCellExperiment)
RawCountData <- assay(BUSseqfits_example, "counts")
batch_ind <- colData(BUSseqfits_example)
sce <- SingleCellExperiment(assays = list(counts = RawCountData),
                            colData = DataFrame(Batch_ind = batch_ind))
BUSseqfits_res <- BUSseq_MCMC(ObservedData = sce,
                              seed = 1234, n.cores = 2,
                              n.celltypes = 4, n.iterations = 500)

################################################
# Extract Estimates from the BUSseqfits Object #
################################################

#return cell type indicators
w.est <- celltypes(BUSseqfits_res)

#return the intercept and odds ratio of the logistic regression
#for dropout events
gamma.est <- dropout_coefficient_values(BUSseqfits_res)

#return the log-scale baseline expression values
alpha.est <-  baseline_expression_values(BUSseqfits_res)

#return the cell-type effects
beta.est <- celltype_effects(BUSseqfits_res)

#return the mean expression levels
mu.est <- celltype_mean_expression(BUSseqfits_res)

#return the cell-specific global effects
delta.est <- cell_effect_values(BUSseqfits_res)

#return the location batch effects
nu.est <- location_batch_effects(BUSseqfits_res)
```

```
#return the overdispersion parameters
phi.est <- overdispersions(BUSseqfits_res)

#return the intrinsic gene indices
D.est <- intrinsic_genes_BUSseq(BUSseqfits_res)

#return the BIC value
BIC <- BIC_BUSseq(BUSseqfits_res)

#return the raw read count matrix
CountData_raw <- raw_read_counts(BUSseqfits_res)

#return the imputed read count matrix
CountData_imputed <- imputed_read_counts(BUSseqfits_res)

#return the corrected read count matrix
CountData_corrected <- corrected_read_counts(BUSseqfits_res)

#################
# Visualization #
#################
#generate the heatmap of raw read count data
heatmap_data_BUSseq(BUSseqfits_res, project_name="Heatmap_raw")

#generate the heatmap of imputed read count data
heatmap_data_BUSseq(BUSseqfits_res, data_type = "Imputed",
                    project_name="Heatmap_imputed")

#generate the heatmap of corrected read count data
heatmap_data_BUSseq(BUSseqfits_res, data_type = "Corrected",
                    project_name="Heatmap_corrected")
```

`baseline_expression_values` *Obtain the Log-Scale Baseline Expression Levels from the Output of the* `BUSseq_MCMC` *Function*

Description

The function gives the estimated log-scale baseline expression levels from the output `SingleCellExperiment` object.

Usage

```
baseline_expression_values(sce_BUSseqfit)
```

Arguments

`sce_BUSseqfit`	An output `SingleCellExperiment` object by the function `BUSseq_MCMC`.

Value

`alpha.est` The estimated log-scale baseline expression levels, a G-dimensional vector whose g-th element is the estimated log-scale mean gene expression level of gene g in the first cell type.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
alpha.est <- baseline_expression_values(BUSseqfits_example)
```

BIC_BUSseq *Obtain BIC from the Output of the* BUSseq_MCMC *Function*

Description

The function gives the Bayesian Informtion Criterion (BIC) value in the output `SingleCellExperiment` object.

Usage

```
BIC_BUSseq(sce_BUSseqfit)
```

Arguments

`sce_BUSseqfit` An output `SingleCellExperiment` object obtained from the function `BUSseq_MCMC`.

Value

`BIC_val` The BIC value.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
Example_BIC <- BIC_BUSseq(BUSseqfits_example)
```

BUSseqfits_example *An external example of the output of the* BUSseq_MCMC

Description

This data set is a SingleCellExperiment object generated by running the BUSseq_MCMC function on the simulatied data in the "Example" of BUSseq-package.

Usage

```
BUSseqfits_example
```

Format

A SingleCellExperiment object.

Details

The simuated read count data consist of two 150-cell batches. In total, 300 genes are measured. Moreover, all cells come from four cell types.

Source

The "Example" of BUSseq-package

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
library(SingleCellExperiment)
################################
# Set the Synthetic Parameters #
################################
rm(list=ls())
set.seed(1234)
#The number of batches
B<-2

#The number of cells per batch
nb<-c(150,150)

#The total number of cells
N<-sum(nb)

#The number of genes
G<-300

#The number of cell types
K<-4

# the project name
proj <- "demo"

#The first column of gamma.syn denotes the intercept of
#the logistic regression for dropout events
#The second column of gamma.syn denotes the odds ratios
#of the logistic regression for dropout events
gamma.syn<-matrix(0,B,2)
gamma.syn[1,]<-c(-1,-0.5)
gamma.syn[2,]<-c(-1,-0.5)

#the log-scale baseline expression levels
alpha.syn<-rep(0,G)
alpha.syn[1:(G*0.2)]<-2

#the cell-type effects
beta.syn<-matrix(0,G,K)

#the first cell type is regarded as the reference cell type
beta.syn[,1] <- 0

#the effects of the second cell type
beta.syn[1:(G * 0.2),2] <- -2
beta.syn[(G * 0.20 + 1):(G * 0.4),2] <- 2
beta.syn[(G * 0.4 + 1):G,2] <- 0

#the effects of the third cell type
beta.syn[1:(G * 0.2),3]<- -2
beta.syn[(G * 0.2 + 1):(G * 0.4),3] <- 0
beta.syn[(G * 0.4 + 1):(G * 0.6),3] <- 2
```

```
beta.syn[(G * 0.6 + 1):G,3] <- 0

#the effects of the forth cell type
beta.syn[1:(G * 0.2),4]<- -2
beta.syn[(G * 0.2 + 1):(G * 0.6),4] <- 0
beta.syn[(G * 0.6 + 1):(G * 0.8),4] <- 2
beta.syn[(G * 0.8 + 1):G,4] <- 0

#the batch effects
nu.syn<-matrix(NA,G,B)

#the first batch is regarded as the reference batch
nu.syn[,1] <- 0

#the effect of the second batch
nu.syn[1:(G * 0.4),2]<-2
nu.syn[(G * 0.4 + 1):(G * 0.8),2]<-1
nu.syn[(G * 0.8 + 1):G,2]<-2

#the cell-specific size factors
delta.syn <- rep(NA, N)

#The frist cell in each bathc is regarded as the reference cell
delta.syn[1:(nb[1] * 0.5)]<-0
delta.syn[(nb[1] * 0.5 + 1):(nb[1] * 0.9)]<-1
delta.syn[(nb[1] * 0.9 + 1):nb[1]]<-2

#the second batch
delta.syn[1:(nb[2] * 0.5) + nb[1]]<-0
delta.syn[(nb[2] * 0.5 + 1):(nb[2] * 0.7) + nb[1]]<-2
delta.syn[(nb[2] * 0.7 + 1):nb[2] + nb[1]]<--1

#the batch-specific and gene-specific overdispersion parameters
phi.syn<-matrix(10,G,B)

#the cell-type proportions in each batch
pi.syn <- matrix(NA,K,B)

#the frist batch
pi.syn[,1]<-c(0.3,0.3,0.2,0.2)

#the second batch
pi.syn[,2]<-c(0.2,0.24,0.2,0.36)


################################################
# Simulate Latent Varibles and Observed data #
################################################
#the cell-type indicators of each cell
w<- rep(NA, N)

#the first batch
nw<-nb[1] * pi.syn[,1]
```

```
w[1:nb[1]]<- rep(1:4,nw)

#the second batch
nw<-nb[2] * pi.syn[,2]
w[1:nb[2] + nb[1]]<- rep(1:4,nw)

#the indicators for dropout events
z<- matrix(NA, G, N)

#the underlying true expression levels
x <- matrix(NA, G, N)

#the observed expression levels
y <- matrix(NA, G, N)

#the logarithm of mean expreesion level of each gene in each cell
log.mu <- matrix(NA, G, N)

#generate the latent variable and observed data
batch_ind <- rep(1:B, nb)
for(i in 1:N){
  b <- batch_ind[i]
  log.mu[,i] <- alpha.syn + beta.syn[,w[i]]
  log.mu[,i] <- log.mu[,i] + nu.syn[,b]
  log.mu[,i] <- log.mu[,i] + delta.syn[i]

  for(j in 1:G){
    x[j,i]<-rnbinom(1,phi.syn[j,b],
                    mu=exp(log.mu[j,i]))
    logit_pi <- gamma.syn[b,1] + gamma.syn[b,2] * x[j,i]

    z[j,i]<-rbinom(1,1,prob = 1/(1+exp(-logit_pi)))
    if(z[j,i]==0){
      y[j,i]<- x[j,i]
    }else{
      y[j,i]<- 0
    }
  }
}

sce <- SingleCellExperiment(assays = list(counts = y),
                colData = DataFrame(Batch_ind = factor(batch_ind)))
BUSseqfits_example <- BUSseq_MCMC(ObservedData = sce,
                                  seed = 1234, n.cores = 2,
                                  n.celltypes = 4, n.iterations = 500)
```

BUSseq_MCMC *Condcut MCMC sampling and posterior inference for the BUSseq Model*

Description

The function BUSseq_MCMC runs a Markov chain Monte Carlo (MCMC) algorithm to fit the Batch effects correction with Unknown Subtypes for scRNA-seq data (BUSseq) model. BUSseq is an interpretable Bayesian hierarchical model that closely follows the data-generating mechanism of scRNA-seq experiments. BUSseq can simultaneously correct batch effects, cluster cell types, impute missing data caused by dropout events and detect differentially expressed genes without requiring a preliminary normalization step. We adopt the MCMC algorithm to conduct posterior inference for the BUSseq model. Here, we denote the total number of cells as N, the batch number as B, the gene number as G, and the cell-type number as K.

Usage

```
BUSseq_MCMC(ObservedData, n.celltypes, seed = round(runif(1,1,10000)),
            n.cores = 8,
            n.iterations = 2000, n.burnin = floor(n.iterations/2),
            n.unchanged = min(floor(n.iterations * 0.3), 500),
            n.output = n.iterations/10, working_dir = getwd(),
            Drop_ind = rep(TRUE, length(ObservedData)), fdr = 0.05,
            hyper_pi = 2, hyper_gamma0 = 3,
            hyper_gamma1 = c(0.001, 0.01),
            hyper_alpha = 5, tau1sq = 50, hyper_p = c(1, 3),
            hyper_tau0sq = c(2, 0.01), hyper_nu = 5,
            hyper_delta = 5, hyper_phi = c(1, 0.1))
```

Arguments

ObservedData	ObservedData is a list with the length equal to the batch number, or a SingleCellExperiment object. If it is a list, the b-th element of ObservedData should be the raw count data matrix of batch b, where each row corresponds to a gene, and each column corresponds to a cell. If it is a SingleCellExperiment object, the raw count data matrix should have a count matrix in counts, and Batch_ind exists in the colData to indicate the corresponding batch of each cell. Moreover, cells from the same batch should be arranged together.
n.celltypes	n.celltypes is an integer and represents the number of cell types, which needs to be specified by the user.
seed	seed is an integer for the Random Number Generator.
n.cores	n.cores is an integer and denotes the number of cores used for the parallel MCMC algorithm.
n.iterations	n.iterations is the total number of iterations of the MCMC algorithm. The default is 2000.
n.burnin	n.burnin is the number of burn-in iterations of the MCMC sampling. The default is a half of n.iterations
n.unchanged	n.unchanged is the number of iterations where the MCMC does not update the hyperparameter p and tau0 of the slab and spike prior of cell-type effects. The default is the minimum of one third of n.iterations and 500.
n.output	n.output is the number of iterations per hard-disk writing of the posterior sampling. The default is one tenth of n.iterations.

working_dir	working_dir is the directory to store the posterior samples. The default is the current directory.
Drop_ind	Drop_ind is a Boolean vector with length equal to the number of batches and indicates which batch suffers from dropout events.
fdr	The false discovery rate level we want to control in order to identify intrinsic genes. The default is 0.05.
hyper_pi	hyper_pi is a scalar representing the hyperparameter of the Dirichlet prior for cell type proportions. The default is 2.
hyper_gamma0	hyper_gamma0 is a scalar representing the variance of the normal prior for the intercept of the logistic regression for dropout events. The default is 3.
hyper_gamma1	hyper_gamma1 is a vector representing the hyperparameter of the Gamma prior for the log-odds ratio of dropout events. The default is (0.001, 0.01) such that the prior mean of gamma1 is 0.1.
hyper_alpha	hyper_alpha is a scalar representing the variance of the normal prior for the log-scale baseline expression levels. The default is 5.
tau1sq	tau1sq is the slab variance of the spike-and-slab prior for the cell type effects. The default is 50.
hyper_p	hyper_p is a two-dimensional vector representing the two shape parameters of the beta prior for the proportion of intrinsic genes. The default is c(1,3).
hyper_tau0sq	hyper_tau0sq is a two-dimensional vector representing the shape and scale of the inverse gamma prior for the variance of the spike normal prior. The default is c(2,0.01).
hyper_nu	hyper_nu is a scalar representing the variance of the normal prior for the batch effects. The default is 5.
hyper_delta	hyper_delta is a scalar representing the variance of the normal prior for the cell-specific size effects. The default is sqrt(5).
hyper_phi	hyper_phi is a two-dimensional vector representing the shape and rate of the gamma prior for the overdispersion parameters. The default is c(1,0.1).

Value

A SingleCellExperiment object res is outputed with an imputed count data matrix assay(res, "imputed_data") after imputing dropout events. At the same time, the estimated cell type labels and relative size factors of cells are added in the column-level metadata int_colData(res)$BUSseq, while the identified intrinsic genes are stored in the row-level metadata int_elementMetadata(res)$BUSseq.

Morover, the dimension information, the posterior inference of parameters and latent variables are stored as a list in metadata(res)$BUSseq:

n.cell	The total number of cells in all batches, a scalar.
n.gene	The number of genes, a scalar.
n.batch	The number of batches, a scalar.
n.perbatch	The number of cells in each batch, a B-dimensional vector.
n.celltype	The number of cell types specified by user, a scalar.
n.iter	The total number of iterations applied in the MCMC algorithm, a scalar.

`seed` The seed for the MCMC algorithm.

`n.burnin` The number of iterations as burnin, a scalar.

`gamma.est` The estimated intercept and odds ratio of the logistic regression for dropout events, a B by 2 matrix.

`alpha.est` The estimated log-scale baseline expression levels, a G-dimensional vector whose g-th element is the estimated log-scale mean gene expression level of gene g in the first cell type.

`beta.est` The estimated cell-type effects, a G by K matrix, whose [g,k] element is the effects of cell type k on gene g compared with the first cell type. Note that the first column is zero as the first cell type is taken as the baseline cell type.

`nu.est` The estimated location batch effects, a G by B matrix, where [g,b] element is the location batch effect on gene g in the batch b compared with the first batch. Note that the first column is zero as the first batch is taken as the reference batch without batch effects.

`delta.est` The estimated cell-specific global effects, an N-dimensional vector. Note that the first element in each vector is zero as the first cell in each batch is taken as the reference cell.

`phi.est` The estimated overdispersion parameters, a G by B matrix, where [g,b] element is the overdispersion parameter on gene g in batch b.

`pi.est` The estimated cell-type proportions across batches, a B by K matrix, whose [b,k] element is the estimated proportion of cell type k in batch b.

`w.est` The estimated cell-type indicators of each cell, a N-dimensional vector.

`p.est` The estimated proportion of differentially expressed genes compared with the first cell type, a scalar.

`PPI.est` The estimated posterior marginal probability of being differentially expressed genes compared with the first cell type. The return is G by K matrix. Noted that the first column consist of zeros as there is no differentially expressed gene compared with the cell type of itself.

`D.est` The intrinsic gene indicators. The return is an N-dimensional vector.

`BIC` The BIC value when K = `n.celltypes`, which is used to determine the number of cell types by varying the value of K.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
#####################################
# Apply BUSseq to the Simulation Data #
#####################################
library(SingleCellExperiment)
RawCountData <- assay(BUSseqfits_example, "counts")
batch_ind <- colData(BUSseqfits_example)
sce <- SingleCellExperiment(assays = list(counts = RawCountData),
                            colData = DataFrame(Batch_ind = batch_ind))
BUSseqfits_res <- BUSseq_MCMC(ObservedData = sce,
                              seed = 1234, n.cores = 2,
                              n.celltypes = 4, n.iterations = 500)

#################################################
# Extract Estimates from the BUSseqfits Object #
#################################################

#return cell type indicators
w.est <- celltypes(BUSseqfits_res)

#return the intercept and odds ratio of the logistic regression
#for dropout events
gamma.est <- dropout_coefficient_values(BUSseqfits_res)

#return the log-scale baseline expression values
alpha.est <-  baseline_expression_values(BUSseqfits_res)

#return the cell-type effects
beta.est <- celltype_effects(BUSseqfits_res)

#return the mean expression levels
mu.est <- celltype_mean_expression(BUSseqfits_res)

#return the cell-specific global effects
delta.est <- cell_effect_values(BUSseqfits_res)

#return the location batch effects
nu.est <- location_batch_effects(BUSseqfits_res)

#return the overdispersion parameters
phi.est <- overdispersions(BUSseqfits_res)

#return the intrinsic gene indices
D.est <- intrinsic_genes_BUSseq(BUSseqfits_res)

#return the BIC value
BIC <- BIC_BUSseq(BUSseqfits_res)

#return the raw read count matrix
CountData_raw <- raw_read_counts(BUSseqfits_res)

#return the imputed read count matrix
```

```
CountData_imputed <- imputed_read_counts(BUSseqfits_res)

#return the corrected read count matrix
CountData_corrected <- corrected_read_counts(BUSseqfits_res)

#################
# Visualization #
#################
#generate the heatmap of raw read count data
heatmap_data_BUSseq(BUSseqfits_res, project_name="Heatmap_raw")

#generate the heatmap of imputed read count data
heatmap_data_BUSseq(BUSseqfits_res, data_type = "Imputed",
                    project_name="Heatmap_imputed")

#generate the heatmap of corrected read count data
heatmap_data_BUSseq(BUSseqfits_res, data_type = "Corrected",
                    project_name="Heatmap_corrected")
```

`celltypes` *Obtain the Cell-type Indicators from the Output of the* `BUSseq_MCMC` *Function*

Description

The function gives the cell-type indicators of the output `SingleCellExperiment` object.

Usage

```
celltypes(sce_BUSseqfit)
```

Arguments

`sce_BUSseqfit`	An output `SingleCellExperiment` object by the function `BUSseq_MCMC`.

Value

`w.est`	The estimated cell-type indicators, an N-dimensional vector, where N is the total number of cells.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
celltypes_est <- celltypes(BUSseqfits_example)
```

`celltype_effects` *Obtain the Cell-type Effects from the Output of the* `BUSseq_MCMC` *Function*

Description

The function gives the estimated cell-type effects in the output `SingleCellExperiment` object.

Usage

```
celltype_effects(sce_BUSseqfit)
```

Arguments

`sce_BUSseqfit`	An output `SingleCellExperiment` object by the function `BUSseq_MCMC`.

Value

`beta.est`	The estimated cell-type effects, a G by K matrix, whose [g,k] element is the effects of cell type k on gene g compared with the first cell type. Note that the first column is zero as the first cell type is taken as the baseline cell type.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
beta.est <- celltype_effects(BUSseqfits_example)
```

`celltype_mean_expression` *Obtain the Cell-Type-Specific Mean Expression Levels from the Output of the* `BUSseq_MCMC` *Function*

Description

The function gives the estimated cell-type-specific mean expression levels in the output `SingleCellExperiment` object.

Usage

```
celltype_mean_expression(sce_BUSseqfit)
```

Arguments

`sce_BUSseqfit`	An output `SingleCellExperiment` object by the function `BUSseq_MCMC`.

Value

`mu.est`	The estimated cell-type-specific mean expression levels, a G by K matrix, whose [g,k] element is the mean expression levels of cell type k on gene g.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
mu.est <- celltype_mean_expression(BUSseqfits_example)
```

cell_effect_values *Obtain the cell-specific size effects from the Output of the* BUSseq_MCMC *Function*

Description

The function gives the estimated cell-specific size effects in the output SingleCellExperiment object.

Usage

```
cell_effect_values(sce_BUSseqfit)
```

Arguments

sce_BUSseqfit	An output SingleCellExperiment object obtained from the function BUSseq_MCMC.

Value

delta.est	The estimated cell-specific global effects, an N-dimensional vector. Note that the first element in each vector is zero as the first cell in each batch is taken as the reference cell.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
delta.est <- cell_effect_values(BUSseqfits_example)
```

corrected_read_counts *Generate the Corrected Read Count Matrix within the Output of the* BUSseq_MCMC

Description

The function generates a version of count data, for which the batch effects are removed and the biological variabilities are retained. We develop a quantile match approach based on the idea of inverse transform sampling. The users can perform downstream analysis on the corrected read count matrix, such as clustering, differentially expressed gene identification and so on, as if all the data were measured in a single batch.

Usage

```
corrected_read_counts(sce_BUSseqfit)
```

Arguments

sce_BUSseqfit	An output SingleCellExperiment object obtained from the function BUSseq_MCMC.

Value

corrected_data	The corrected read count matrix, which is added to the assays of sce_BUSseqfit and can be loaded by assay(sce_BUSseqfit, "corrected_data").

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
library(SingleCellExperiment)
BUSseqfits_example
BUSseqfits_corrected <- corrected_read_counts(BUSseqfits_example)
corrected_data <- assay(BUSseqfits_corrected, "corrected_data")
```

dropout_coefficient_values *Obtain the Coefficients of the Logistic Regression for the Dropout Events from the Output of the* BUSseq_MCMC *Function*

Description

The function gives the intercept and odds ratio of the logistic regression for dropout events in the output SingleCellExperiment object.

Usage

```
dropout_coefficient_values(sce_BUSseqfit)
```

Arguments

sce_BUSseqfit	An output SingleCellExperiment object by the function BUSseq_MCMC.

Value

gamma.est	The estimated intercept and log ratio of the logistic regression for dropout events, a 2-diminsional vector.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
gamma.est <- dropout_coefficient_values(BUSseqfits_example)
```

heatmap_data_BUSseq *Draw the Heatmap of the Log-scale Read Count Data for the Output of the* BUSseq_MCMC *Function*

Description

Plot the heatmap of the log-scale read count data across multiple batches, and then save the resulting images in the user's directory as "png" format.

Usage

```
heatmap_data_BUSseq(sce_BUSseqfit,
                    data_type = c("Raw","Imputed","Corrected"),
                    gene_set = NULL,
                    project_name= paste0("BUSseq_heatmap_",data_type),
                    image_dir = NULL, color_key_seq = NULL,
                    image_width = 1440, image_height = 1080)
```

Arguments

sce_BUSseqfit	An output SingleCellExperiment object obtained from the function BUSseq_MCMC.
data_type	A string to determine which count data matrix is used to draw the heatmap, "Raw" for the raw count data, "Imputed" for the imputed data, and "Corrected" for the corrected data.
gene_set	A vector of gene indices indicating the gene set of interest to display in the heatmap. The default is all genes. We also recommend displaying the intrinsic genes obtained from intrisic_genes_BUSseq(BUSseqfits_obj).
project_name	A string to name the "png" image. By default, the figure is named as "BUSseq_heatmap_Raw_log1p_data.
image_dir	A directory to store the gnereated heatmap. The default is to create a folder called "image" in the current directory and save there.
color_key_seq	A numeric vector indicating the splitting points for binning log-scale read counts into colors. The default is to space the color key points equally between the minimum and maximum of the log-scale read count data.
image_width	The width in pixels of the graphical device to plot. The default is 1440 px.
image_height	The height in pixels of the graphical device to plot. The default is 1080 px.

Details

To cope with the zeros in the count data, we take the transformation log(1+x) on all count data, which corresponds to the R function log1p() instead of log().

Value

Visualize the gene expression data matrix, where each row represents a gene, and each column represents a sample.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
library(SingleCellExperiment)
# Plot the imputed read count data of the first 100 genes
heatmap_data_BUSseq(BUSseqfits_example, data_type = "Imputed",
                    gene_set = 1:100)
```

---

imputed_read_counts *Obtain the Imputed Read Count Matrix from the Output of the* BUSseq_MCMC *Function*

Description

The function gives the imputed read counts in the output `SingleCellExperiment` object.

Usage

```
imputed_read_counts(sce_BUSseqfit)
```

Arguments

`sce_BUSseqfit`	An output `SingleCellExperiment` object obtained from the function `BUSseq_MCMC`.

Value

`CountData_imputed`

The imputed read counts, a `CountData` object with length equal to the batch number. Each element is a read count matrix for a specific batch, where each row corresponds to a gene and each column represents a cell.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
library(SingleCellExperiment)
BUSseqfits_example
Example_CountData_imputed <- imputed_read_counts(BUSseqfits_example)
```

`intrinsic_genes_BUSseq` *Obtain the Intrinsic Gene Indicators from the Output of the* `BUSseq_MCMC` *Function*

Description

The function gives the estimated intrinsic gene indicators in the output `SingleCellExperiment` object.

Usage

```
intrinsic_genes_BUSseq(sce_BUSseqfit)
```

Arguments

`sce_BUSseqfit`	An output `SingleCellExperiment` object obtained from the function `BUSseq_MCMC`.

Value

`intrinsic_genes`
A vector indicating whether a gene is intrinsic or not.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
intri_genes <- intrinsic_genes_BUSseq(BUSseqfits_example)
```

location_batch_effects *Obtain the Location Batch Effects from the Output of the* BUSseq_MCMC *Function*

Description

The function gives the estimated location batch effects in the output SingleCellExperiment object.

Usage

```
location_batch_effects(sce_BUSseqfit)
```

Arguments

sce_BUSseqfit	An output SingleCellExperiment object obtained from the function BUSseq_MCMC.

Value

nu.est	The estimated location batch effects, a G by B matrix, where [g,b] element is the location batch effect on gene g in the batch b compared with the first batch. Note that the first column is zero as the first batch is taken as the reference batch without batch effects.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
nu.est <- location_batch_effects(BUSseqfits_example)
```

`overdispersions` *Obtain the Overdispersion Parameters from the Output of the* `BUSseq_MCMC` *Function*

Description

The function gives the estimated overdispersion parameters in the output `SingleCellExperiment` object.

Usage

```
overdispersions(sce_BUSseqfit)
```

Arguments

`sce_BUSseqfit`	An output `SingleCellExperiment` object obtained from the function `BUSseq_MCMC`.

Value

`phi.est`	The estimated overdispersion parameters, a G by B matrix, where [g,b] element is the overdispersion parameter of gene g in batch b.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
BUSseqfits_example
phi.est <- overdispersions(BUSseqfits_example)
```

raw_read_counts *Obtain the Raw Read Count Matrix from the Output of the* BUSseq_MCMC *Function*

Description

The function gives the original read count matrix. This function is actually equivalent to `assay(sce_BUSseqfit, "counts")`.

Usage

```
raw_read_counts(sce_BUSseqfit)
```

Arguments

sce_BUSseqfit	An output SingleCellExperiment object obtained from the function BUSseq_MCMC.

Value

CountData_raw	The raw read counts, a matrix with each row for a gene and each column for a cell.

Author(s)

Fangda Song

References

Song, Fangda, Ga Ming Angus Chan, and Yingying Wei. Flexible experimental designs for valid single-cell RNA-sequencing experiments allowing batch effects correction. Nature communications 11, no. 1 (2020): 1-15.

Examples

```
# "BUSseqfits_example" is an example output
library(SingleCellExperiment)
BUSseqfits_example
Example_CountData_raw <- raw_read_counts(BUSseqfits_example)
```

Index