An introduction to the SingleCellExperiment class

Davide Risso and Aaron Lun

4 January 2019

Package

SingleCellExperiment 1.4.1

1 The `SingleCellExperiment` class definition

The `SingleCellExperiment` class is a light-weight container for single-cell genomics data. It extends the `RangedSummarizedExperiment` class and follows similar conventions, i.e., rows should represent features (genes, transcripts, genomic regions) and columns should represent cells. In addition to the slots already present in the `RangedSummarizedExperiment`, the `SingleCellExperiment` class contains:

- `int_elementMetadata`, to hold internal metadata for each row.
- `int_colData`, to hold internal metadata for each column.
- `int_metadata`, to hold internal metadata for the entire

object.

- `reducedDims` , to hold dimensionality reduction results.

The `int_` prefix describes *internal* slots that are **not** meant for direct manipulation by the user. Instead, they are set by other user-visible methods, which will be discussed in more detail below. Package developers should see the other vignette for how to define methods to access these internal fields.

2 Creating SingleCellExperiment instances

`SingleCellExperiment` objects can be created via the constructor of the same name:

```
library(SingleCellExperiment)
counts <- matrix(rpois(100, lambda = 10), ncol=10, nrow=10)
sce <- SingleCellExperiment(assays = list(counts = counts))
sce
```

```
## class: SingleCellExperiment
## dim: 10 10
## metadata(0):
## assays(1): counts
## rownames: NULL
## rowData names(0):
## colnames: NULL
## colData names(0):
## reducedDimNames(0):
## spikeNames(0):
```

An alternative approach is via coercion from `SummarizedExperiment` objects.

```
se <- SummarizedExperiment(list(counts=counts))
as(se, "SingleCellExperiment")
```

```
## class: SingleCellExperiment
## dim: 10 10
## metadata(0):
## assays(1): counts
## rownames: NULL
## rowData names(0):
## colnames: NULL
## colData names(0):
## reducedDimNames(0):
## spikeNames(0):
```

To demonstrate the use of the class, we will use a subset of the `allen` data set from the *scRNAseq (https://bioconductor.org/packages/3.8/scRNAseq)* package. These data are stored as a `SummarizedExperiment` , so we will first coerce it into a

`SingleCellExperiment`.

```
library(scRNAseq)
data(allen)
allen
```

```
## class: SummarizedExperiment
## dim: 20908 379
## metadata(2): SuppInfo which_qc
## assays(4): tophat_counts cufflinks_fpkm rsem_counts rsem_
tpm
## rownames(20908): 0610007P14Rik 0610009B22Rik ... Zzef1 Zz
z3
## rowData names(0):
## colnames(379): SRR2140028 SRR2140022 ... SRR2139341 SRR21
39336
## colData names(22): NREADS NALIGNED ... Animal.ID passes_q
c_checks_s
```

```
sce <- as(allen, "SingleCellExperiment")
sce
```

```
## class: SingleCellExperiment
## dim: 20908 379
## metadata(2): SuppInfo which_qc
## assays(4): tophat_counts cufflinks_fpkm rsem_counts rsem_
tpm
## rownames(20908): 0610007P14Rik 0610009B22Rik ... Zzef1 Zz
z3
## rowData names(0):
## colnames(379): SRR2140028 SRR2140022 ... SRR2139341 SRR21
39336
## colData names(22): NREADS NALIGNED ... Animal.ID passes_q
c_checks_s
## reducedDimNames(0):
## spikeNames(0):
```

3 Adding spike-in information

One of the main additions to `SummarizedExperiment` is the ability for the user to specify the rows corresponding to spike-in transcripts. This is done with the method `isSpike`, using an appropriate name for the spike-in set.

```
isSpike(sce, "ERCC") <- grepl("^ERCC-", rownames(sce))
sce
```

```
## class: SingleCellExperiment
## dim: 20908 379
## metadata(2): SuppInfo which_qc
## assays(4): tophat_counts cufflinks_fpkm rsem_counts rsem_
tpm
## rownames(20908): 0610007P14Rik 0610009B22Rik ... Zzef1 Zz
z3
## rowData names(0):
## colnames(379): SRR2140028 SRR2140022 ... SRR2139341 SRR21
39336
## colData names(22): NREADS NALIGNED ... Animal.ID passes_q
c_checks_s
## reducedDimNames(0):
## spikeNames(1): ERCC
```

The identities of the spike-in rows can be easily retrieved using the name of the spike-in set, as shown below. The names of currently available spike-in sets can also be returned with the `spikeNames` method.

```
table(isSpike(sce, "ERCC"))
```

```
##
## FALSE  TRUE
## 20816    92
```

```
spikeNames(sce)
```

```
## [1] "ERCC"
```

While most experimental designs use a single set of spike-ins, the class has the flexibility of including more than one set of spikes. Let us pretend that the members of the *Adam* gene family have been spiked-in as external genes in these data.

```
isSpike(sce, "Adam") <- grepl("^Adam[0-9]", rownames(sce))
sce
```

```
## class: SingleCellExperiment
## dim: 20908 379
## metadata(2): SuppInfo which_qc
## assays(4): tophat_counts cufflinks_fpkm rsem_counts rsem_
tpm
## rownames(20908): 0610007P14Rik 0610009B22Rik ... Zzef1 Zz
z3
## rowData names(0):
## colnames(379): SRR2140028 SRR2140022 ... SRR2139341 SRR21
39336
## colData names(22): NREADS NALIGNED ... Animal.ID passes_q
c_checks_s
## reducedDimNames(0):
## spikeNames(2): ERCC Adam
```

```
table(isSpike(sce, "Adam"))
```

```
##
## FALSE  TRUE
## 20875    33
```

```
spikeNames(sce)
```

```
## [1] "ERCC" "Adam"
```

If `isSpike` is used without specifying any name, it will return the union of all spike-in sets.

```
table(isSpike(sce))
```

```
##
## FALSE  TRUE
## 20783   125
```

Running `isSpike<-` with `NULL` will clear the specified set from the `SingleCellExperiment`, while running `clearSpikes` will delete all spike-in information.

```
temp <- sce
isSpike(temp, "Adam") <- NULL
spikeNames(temp)
```

```
## [1] "ERCC"
```

```
temp <- clearSpikes(temp)
spikeNames(temp)
```

```
## character(0)
```

Note that the `isSpike` and `isSpike<-` methods get and set columns in `int_elementMetadata` and `int_metadata`. This information is only relevant to package developers and not necessary for routine use of this class.

4 Adding size factors

One can also store size factors in the `SingleCellExperiment` object. For illustration, we simply compute the total number of reads as size factors here. Note that more sophisticated methods for computing size factors are available (see, e.g., *scran (https://bioconductor.org/packages/3.8/scran)*).

```
sizeFactors(sce) <- colSums(assay(sce))
head(sizeFactors(sce))
```

```
## SRR2140028 SRR2140022 SRR2140055 SRR2140083 SRR2139991 SR
R2140067
##    5173863    6445002    2343379    5438526    4757468
2364851
```

We can compute multiple size factors and store them in the object, by providing a name to `sizeFactors`. This does *not* affect the values of the unnamed size factors.

```
sizeFactors(sce, "ERCC") <- colSums(assay(sce)[isSpike(sce,
"ERCC"),])
head(sizeFactors(sce, "ERCC"))
```

```
## SRR2140028 SRR2140022 SRR2140055 SRR2140083 SRR2139991 SR
R2140067
##     224648     186208     162370     512991     278034
64975
```

```
head(sizeFactors(sce)) # same as before
```

```
## SRR2140028 SRR2140022 SRR2140055 SRR2140083 SRR2139991 SR
R2140067
##    5173863    6445002    2343379    5438526    4757468
2364851
```

5 Extracting `colData` and `rowData`

The `colData` and `rowData` methods can be used to retrieve the stored sample- and gene-level metadata. By default, this will only return the user-visible metadata fields, i.e., not including the fields stored in the `int_*` slots.

```
colData(sce)
```

```
## DataFrame with 379 rows and 22 columns
##               NREADS  NALIGNED    RALIGN TOTAL_DUP     PR
IMER
##            <numeric> <numeric> <numeric> <numeric>  <nume
ric>
## SRR2140028  13743900  13011100   94.6681     51.11  0.014
8148
## SRR2140022  14078700  13521900   96.0454   55.9157 0.0085
3083
## SRR2140055   5842930   5135980   87.9008   59.1126  0.035
6112
## SRR2140083  16784400  15585800   92.8587   55.3076  0.020
9695
## SRR2139991  11558600  10864300   93.9929   50.2258   0.01
6408
## ...              ...       ...       ...       ...
...
## SRR2139325  12875700  11307000   87.8172   70.3564  0.045
3119
## SRR2139373   9699400   8964140   92.4196   45.5249  0.021
6694
## SRR2139379   6175660   5728080   92.7526   45.2652  0.021
7132
## SRR2139341  28038500  26320000   93.8711   65.1959  0.027
0482
## SRR2139336   7878700   7467200   94.7772   56.9675  0.019
0784
##            PCT_RIBOSOMAL_BASES PCT_CODING_BASES PCT_UTR_B
ASES
##                      <numeric>        <numeric>     <nume
ric>
## SRR2140028               1e-06         0.216848      0.26
5609
## SRR2140022                   0         0.263052      0.31
0332
## SRR2140055               3e-06         0.207086      0.32
7241
## SRR2140083               1e-06         0.129243      0.25
3681
## SRR2139991                   0         0.257729      0.27
6831
## ...                        ...              ...
...
## SRR2139325             1.2e-05         0.211253      0.26
9041
## SRR2139373               1e-06         0.220541      0.25
4625
## SRR2139379                   0         0.253996      0.28
9924
## SRR2139341                   0         0.297193      0.36
7713
## SRR2139336               1e-06         0.369537      0.40
7216
##            PCT_INTRONIC_BASES PCT_INTERGENIC_BASES PCT_MR
```

```
NA_BASES
##                     <numeric>           <numeric>
<numeric>
## SRR2140028          0.369509            0.148033
0.482457
## SRR2140022          0.290329            0.136288
0.573384
## SRR2140055          0.291128            0.174542
0.534327
## SRR2140083          0.444594            0.172481
0.382924
## SRR2139991          0.323493            0.141946
0.53456
## ...                      ...                 ...
...
## SRR2139325           0.34636            0.173333
0.480295
## SRR2139373          0.385409            0.139423
0.475167
## SRR2139379            0.3254             0.13068
0.54392
## SRR2139341          0.190673            0.144421
0.664906
## SRR2139336          0.077377             0.14587
0.776752
##            MEDIAN_CV_COVERAGE MEDIAN_5PRIME_BIAS MEDIAN_3
PRIME_BIAS
##                     <numeric>         <numeric>
<numeric>
## SRR2140028          0.507749           0.14181
0.409045
## SRR2140022          0.488182          0.145024
0.41916
## SRR2140055          0.729874          0.069846
0.54856
## SRR2140083          0.781878                 0
0.697916
## SRR2139991           0.48292          0.160644
0.413018
## ...                      ...               ...
...
## SRR2139325          0.754565          0.049809
0.45433
## SRR2139373          0.504545          0.162197
0.518013
## SRR2139379          0.521732          0.136669
0.458149
## SRR2139341          0.491965          0.155719
0.551382
## SRR2139336          0.517592          0.148329
0.445874
##            MEDIAN_5PRIME_TO_3PRIME_BIAS      driver_1_s d
issection_s
##                                <numeric>     <character>
<character>
```

```
## SRR2140028                     0.425234  Scnn1a-Tg3-Cre
L4
## SRR2140022                      0.41926  Scnn1a-Tg3-Cre
L4
## SRR2140055                     0.257657  Scnn1a-Tg3-Cre
All
## SRR2140083                      0.01825  Scnn1a-Tg3-Cre
L4
## SRR2139991                     0.462171  Scnn1a-Tg3-Cre
L4
## ...                                 ...             ...
...
## SRR2139325                     0.413165 Ntsr1-Cre_GN220
L6a
## SRR2139373                     0.356451 Ntsr1-Cre_GN220
L6a
## SRR2139379                     0.367889 Ntsr1-Cre_GN220
L6a
## SRR2139341                     0.330835 Ntsr1-Cre_GN220
L6a
## SRR2139336                     0.390592 Ntsr1-Cre_GN220
L6a
##               Core.Type Primary.Type Secondary.Type Anima
l.ID
##             <character>  <character>    <character> <inte
ger>
## SRR2140028 Intermediate    L4 Scnn1a       L4 Ctxn3    13
3632
## SRR2140022         Core    L4 Scnn1a                   13
3632
## SRR2140055 Intermediate  L5a Tcerg1l      L5a Batf3    15
1560
## SRR2140083           NA           NA             NA
NA
## SRR2139991 Intermediate    L4 Scnn1a       L4 Ctxn3    12
6846
## ...                 ...          ...            ...
...
## SRR2139325 Intermediate      L6a Sla        L6a Mgp    17
5613
## SRR2139373         Core      L6a Sla                   13
2856
## SRR2139379         Core      L6a Sla                   13
2856
## SRR2139341         Core      L6a Sla                   13
2856
## SRR2139336         Core      L6a Sla                   13
2856
##            passes_qc_checks_s
##                   <character>
## SRR2140028                 Y
## SRR2140022                 Y
## SRR2140055                 Y
## SRR2140083                 N
## SRR2139991                 Y
```

```
## ...                         ...
## SRR2139325                   Y
## SRR2139373                   Y
## SRR2139379                   Y
## SRR2139341                   Y
## SRR2139336                   Y
```

```
rowData(sce)
```

```
## DataFrame with 20908 rows and 0 columns
```

However, it is sometimes useful to retrieve a `DataFrame` with the internal fields, i.e., spike-in and size factor information. This can be achieved by specifying `internal=TRUE`.

```
colData(sce, internal=TRUE)
```

```
## DataFrame with 379 rows and 24 columns
##               NREADS  NALIGNED    RALIGN TOTAL_DUP     PR
IMER
##            <numeric> <numeric> <numeric> <numeric>  <nume
ric>
## SRR2140028  13743900  13011100   94.6681     51.11  0.014
8148
## SRR2140022  14078700  13521900   96.0454   55.9157 0.0085
3083
## SRR2140055   5842930   5135980   87.9008   59.1126  0.035
6112
## SRR2140083  16784400  15585800   92.8587   55.3076  0.020
9695
## SRR2139991  11558600  10864300   93.9929   50.2258   0.01
6408
## ...              ...       ...       ...       ...
...
## SRR2139325  12875700  11307000   87.8172   70.3564  0.045
3119
## SRR2139373   9699400   8964140   92.4196   45.5249  0.021
6694
## SRR2139379   6175660   5728080   92.7526   45.2652  0.021
7132
## SRR2139341  28038500  26320000   93.8711   65.1959  0.027
0482
## SRR2139336   7878700   7467200   94.7772   56.9675  0.019
0784
##            PCT_RIBOSOMAL_BASES PCT_CODING_BASES PCT_UTR_B
ASES
##                      <numeric>        <numeric>     <nume
ric>
## SRR2140028               1e-06         0.216848      0.26
5609
## SRR2140022                   0         0.263052      0.31
0332
## SRR2140055               3e-06         0.207086      0.32
7241
## SRR2140083               1e-06         0.129243      0.25
3681
## SRR2139991                   0         0.257729      0.27
6831
## ...                        ...              ...
...
## SRR2139325             1.2e-05         0.211253      0.26
9041
## SRR2139373               1e-06         0.220541      0.25
4625
## SRR2139379                   0         0.253996      0.28
9924
## SRR2139341                   0         0.297193      0.36
7713
## SRR2139336               1e-06         0.369537      0.40
7216
##            PCT_INTRONIC_BASES PCT_INTERGENIC_BASES PCT_MR
```

```
NA_BASES
##                     <numeric>           <numeric>
<numeric>
## SRR2140028          0.369509            0.148033
0.482457
## SRR2140022          0.290329            0.136288
0.573384
## SRR2140055          0.291128            0.174542
0.534327
## SRR2140083          0.444594            0.172481
0.382924
## SRR2139991          0.323493            0.141946
0.53456
## ...                      ...                 ...
...
## SRR2139325           0.34636            0.173333
0.480295
## SRR2139373          0.385409            0.139423
0.475167
## SRR2139379            0.3254             0.13068
0.54392
## SRR2139341          0.190673            0.144421
0.664906
## SRR2139336          0.077377             0.14587
0.776752
##            MEDIAN_CV_COVERAGE MEDIAN_5PRIME_BIAS MEDIAN_3
PRIME_BIAS
##                     <numeric>         <numeric>
<numeric>
## SRR2140028          0.507749           0.14181
0.409045
## SRR2140022          0.488182          0.145024
0.41916
## SRR2140055          0.729874          0.069846
0.54856
## SRR2140083          0.781878                 0
0.697916
## SRR2139991           0.48292          0.160644
0.413018
## ...                      ...               ...
...
## SRR2139325          0.754565          0.049809
0.45433
## SRR2139373          0.504545          0.162197
0.518013
## SRR2139379          0.521732          0.136669
0.458149
## SRR2139341          0.491965          0.155719
0.551382
## SRR2139336          0.517592          0.148329
0.445874
##            MEDIAN_5PRIME_TO_3PRIME_BIAS      driver_1_s d
issection_s
##                                <numeric>     <character>
<character>
```

```
## SRR2140028                    0.425234  Scnn1a-Tg3-Cre
L4
## SRR2140022                     0.41926  Scnn1a-Tg3-Cre
L4
## SRR2140055                    0.257657  Scnn1a-Tg3-Cre
All
## SRR2140083                     0.01825  Scnn1a-Tg3-Cre
L4
## SRR2139991                    0.462171  Scnn1a-Tg3-Cre
L4
## ...                                ...             ...
...
## SRR2139325                    0.413165 Ntsr1-Cre_GN220
L6a
## SRR2139373                    0.356451 Ntsr1-Cre_GN220
L6a
## SRR2139379                    0.367889 Ntsr1-Cre_GN220
L6a
## SRR2139341                    0.330835 Ntsr1-Cre_GN220
L6a
## SRR2139336                    0.390592 Ntsr1-Cre_GN220
L6a
##               Core.Type Primary.Type Secondary.Type Anima
l.ID
##             <character>  <character>    <character> <inte
ger>
## SRR2140028 Intermediate    L4 Scnn1a       L4 Ctxn3    13
3632
## SRR2140022         Core    L4 Scnn1a                   13
3632
## SRR2140055 Intermediate  L5a Tcerg1l      L5a Batf3    15
1560
## SRR2140083           NA           NA             NA
NA
## SRR2139991 Intermediate    L4 Scnn1a       L4 Ctxn3    12
6846
## ...                 ...          ...            ...
...
## SRR2139325 Intermediate      L6a Sla        L6a Mgp    17
5613
## SRR2139373         Core      L6a Sla                   13
2856
## SRR2139379         Core      L6a Sla                   13
2856
## SRR2139341         Core      L6a Sla                   13
2856
## SRR2139336         Core      L6a Sla                   13
2856
##            passes_qc_checks_s size_factor size_factor_ERC
C
##                   <character>   <numeric>        <numeric
>
## SRR2140028                   Y     5173863           22464
8
## SRR2140022                   Y     6445002           18620
```

```
8
## SRR2140055                  Y     2343379           16237
0
## SRR2140083                  N     5438526           51299
1
## SRR2139991                  Y     4757468           27803
4
## ...                       ...         ...
...
## SRR2139325                  Y     4377966           33195
5
## SRR2139373                  Y     3393227            7952
4
## SRR2139379                  Y     2529501            3702
1
## SRR2139341                  Y    13972642           43958
0
## SRR2139336                  Y     4838243           16341
4
```

```
rowData(sce, internal=TRUE)
```

```
## DataFrame with 20908 rows and 3 columns
##                is_spike_ERCC  is_spike is_spike_Adam
##                    <logical> <logical>     <logical>
## 0610007P14Rik          FALSE     FALSE         FALSE
## 0610009B22Rik          FALSE     FALSE         FALSE
## 0610009L18Rik          FALSE     FALSE         FALSE
## 0610009O20Rik          FALSE     FALSE         FALSE
## 0610010F05Rik          FALSE     FALSE         FALSE
## ...                      ...       ...           ...
## Zyg11a                 FALSE     FALSE         FALSE
## Zyg11b                 FALSE     FALSE         FALSE
## Zyx                    FALSE     FALSE         FALSE
## Zzef1                  FALSE     FALSE         FALSE
## Zzz3                   FALSE     FALSE         FALSE
```

See below for some discussion of why an internal storage mechanism is used here.

6 Adding low-dimensional representations

For simplicity and speed, we work on a subset of 100 genes. To avoid ending up with only uninteresting genes, we extract the 100 genes with maximal variance in the log-transformed counts.

```
library(magrittr)
assay(sce) %>% log1p %>% rowVars -> vars
names(vars) <- rownames(sce)
vars <- sort(vars, decreasing = TRUE)

sce_sub <- sce[names(vars[1:100]),]
sce_sub
```

```
## class: SingleCellExperiment
## dim: 100 379
## metadata(2): SuppInfo which_qc
## assays(4): tophat_counts cufflinks_fpkm rsem_counts rsem_
tpm
## rownames(100): Lamp5 Fam19a1 ... Rnf2 Zfp35
## rowData names(0):
## colnames(379): SRR2140028 SRR2140022 ... SRR2139341 SRR21
39336
## colData names(22): NREADS NALIGNED ... Animal.ID passes_q
c_checks_s
## reducedDimNames(0):
## spikeNames(2): ERCC Adam
```

We obtain the PCA and t-SNE representations of the data and add them to the object with the `reducedDims` method.

```
library(Rtsne)
set.seed(5252)

pca_data <- prcomp(t(log1p(assay(sce_sub))))
tsne_data <- Rtsne(pca_data$x[,1:50], pca = FALSE)

reducedDims(sce_sub) <- SimpleList(PCA=pca_data$x, TSNE=tsne
_data$Y)
sce_sub
```

```
## class: SingleCellExperiment
## dim: 100 379
## metadata(2): SuppInfo which_qc
## assays(4): tophat_counts cufflinks_fpkm rsem_counts rsem_
tpm
## rownames(100): Lamp5 Fam19a1 ... Rnf2 Zfp35
## rowData names(0):
## colnames(379): SRR2140028 SRR2140022 ... SRR2139341 SRR21
39336
## colData names(22): NREADS NALIGNED ... Animal.ID passes_q
c_checks_s
## reducedDimNames(2): PCA TSNE
## spikeNames(2): ERCC Adam
```

The stored coordinates can be retrieved by name or by numerical index. Each row of the coordinate matrix is assumed to correspond to a cell, while each column represents a dimension.

```
reducedDims(sce_sub)
```

```
## List of length 2
## names(2): PCA TSNE
```

```
reducedDimNames(sce_sub)
```

```
## [1] "PCA"  "TSNE"
```

```
head(reducedDim(sce_sub, "PCA")[,1:2])
```

```
##                   PC1        PC2
## SRR2140028 17.557295  -7.717162
## SRR2140022 21.468975  -1.198212
## SRR2140055  4.303756 -11.360330
## SRR2140083 21.440479  -9.435868
## SRR2139991 15.592089 -11.043989
## SRR2140067 16.539336  -9.831779
```

```
head(reducedDim(sce_sub, "TSNE")[,1:2])
```

```
##                 [,1]      [,2]
## SRR2140028 -7.305172 -13.21667
## SRR2140022 -5.571360 -12.47031
## SRR2140055  1.730797 -14.54116
## SRR2140083 -3.774831 -14.21703
## SRR2139991 -2.702975 -13.59155
## SRR2140067 -2.529361 -15.59027
```

Any subsetting by column of `sce_sub` will also lead to subsetting of the dimensionality reduction results by cell.

```
dim(reducedDim(sce_sub, "PCA"))
```

```
## [1] 379 100
```

```
dim(reducedDim(sce_sub[,1:10], "PCA"))
```

```
## [1]  10 100
```

7 Convenient access to named assays

In the `SingleCellExperiment`, users can assign arbitrary names to entries of `assays`. To assist interoperability between packages, we provide some suggestions for what the names should be for particular types of data:

- `counts` : Raw count data, e.g., number of reads or transcripts for a particular gene.
- `normcounts` : Normalized values on the same scale as the original counts. For example, counts divided by cell-specific size factors that are centred at unity.
- `logcounts` : Log-transformed counts or count-like values. In most cases, this will be defined as log-transformed `normcounts` , e.g., using log base 2 and a pseudo-count of 1.
- `cpm` : Counts-per-million. This is the read count for each gene in each cell, divided by the library size of each cell in millions.
- `tpm` : Transcripts-per-million. This is the number of transcripts for each gene in each cell, divided by the total number of transcripts in that cell (in millions).

Each of these suggested names has an appropriate getter/setter method for convenient manipulation of the `SingleCellExperiment` . For example, we can take the (very specifically named) `tophat_counts` name and assign it to `counts` instead:

```
counts(sce) <- assay(sce, "tophat_counts")
sce
```

```
## class: SingleCellExperiment
## dim: 20908 379
## metadata(2): SuppInfo which_qc
## assays(5): tophat_counts cufflinks_fpkm rsem_counts rsem_
tpm counts
## rownames(20908): 0610007P14Rik 0610009B22Rik ... Zzef1 Zz
z3
## rowData names(0):
## colnames(379): SRR2140028 SRR2140022 ... SRR2139341 SRR21
39336
## colData names(22): NREADS NALIGNED ... Animal.ID passes_q
c_checks_s
## reducedDimNames(0):
## spikeNames(2): ERCC Adam
```

```
dim(counts(sce))
```

```
## [1] 20908   379
```

This means that functions expecting count data can simply call `counts()` without worrying about package-specific naming conventions.

8 Session Info

```
sessionInfo()
```

```
## R version 3.5.2 (2018-12-20)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.5 LTS
##
## Matrix products: default
## BLAS: /home/biocbuild/bbs-3.8-bioc/R/lib/libRblas.so
## LAPACK: /home/biocbuild/bbs-3.8-bioc/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices uti
ls     datasets
## [8] methods   base
##
## other attached packages:
##  [1] Rtsne_0.15                  magrittr_1.5
##  [3] scRNAseq_1.8.0              SingleCellExperiment_1.
4.1
##  [5] SummarizedExperiment_1.12.0 DelayedArray_0.8.0
##  [7] BiocParallel_1.16.5         matrixStats_0.54.0
##  [9] Biobase_2.42.0              GenomicRanges_1.34.0
## [11] GenomeInfoDb_1.18.1         IRanges_2.16.0
## [13] S4Vectors_0.20.1            BiocGenerics_0.28.0
## [15] BiocStyle_2.10.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.0             knitr_1.21             XVecto
r_0.22.0
##  [4] zlibbioc_1.28.0        lattice_0.20-38        string
r_1.3.1
##  [7] tools_3.5.2            grid_3.5.2             xfun_
0.4
## [10] htmltools_0.3.6        yaml_2.2.0             digest
_0.6.18
## [13] bookdown_0.9           Matrix_1.2-15          Genome
InfoDbData_1.2.0
## [16] BiocManager_1.30.4     bitops_1.0-6           RCurl_
1.95-4.11
## [19] evaluate_0.12          rmarkdown_1.11         string
i_1.2.4
## [22] compiler_3.5.2
```