

Stefano Mangiola

Maria Doyle

Tidy Transcriptomics for Single-cell RNA Sequencing Analyses



Peter Mac
Peter MacCallum Cancer Centre
Victoria Australia

Resources for #tidytranscriptomics



The blog about tidy transcriptomics

[Home](#) [Archives](#) [Tags](#) [Categories](#) [About](#)

Tidy-transcriptomics manifesto

2021-10-13-tidytranscriptomics

[Read more...](#)



Powered by [Hugo](#) | Theme - [Even](#)

© 2017 - 2021 ❤ Stefano Mangiola, Maria Doyle

[No Title]

Resources for #tidytranscriptomics



The blog about tidy transcriptomics

[Home](#) [Archives](#) [Tags](#) [Categories](#) [About](#)

Tidy-transcriptomics manifesto

2021-10-13-tidytranscriptomics

[Read more...](#)



Powered by [Hugo](#) | Theme - [Even](#)

© 2017 - 2021 ❤ Stefano Mangiola, Maria Doyle

[No Title]

tidybulk: an R tidy framework for modular transcriptomic data analysis

[Stefano Mangiola](#), [Ramyar Molania](#), [Ruining Dong](#), [Maria A. Doyle](#) & [Anthony T. Papenfuss](#)

[Genome Biology](#) **22**, Article number: 42 (2021) | [Cite this article](#)

Interfacing Seurat with the R tidy universe

[Stefano Mangiola](#) , [Maria A Doyle](#), [Anthony T Papenfuss](#)

[Bioinformatics](#), btab404, <https://doi.org/10.1093/bioinformatics/btab404>

Tidy R tools

There are four basic principles to a tidy API:

- Reuse existing data structures.
- Compose simple functions with the pipe.
- Embrace functional programming.
- Design for humans.

```
# A tibble: 100 x 8
  observation      variable_1 variable_2
  <glue>          <chr>       <chr>
1 observation 1   ...         ...
2 observation 2   ...         ...
3 observation 3   ...         ...
4 observation 4   ...         ...
5 observation 5   ...         ...
6 observation 6   ...         ...
7 observation 7   ...         ...
8 observation 8   ...         ...
9 observation 9   ...         ...
10 observation 10  ...         ...
# ... with 90 more rows
```

Tidy R tools

There are four basic principles to a tidy API:

- Reuse existing data structures.
- Compose simple functions with the pipe.
- Embrace functional programming.
- Design for humans.

```
# A tibble: 100 x 8
  observation      variable_1 variable_2 variable_3
  <glue>          <chr>     <chr>     <list>
1 observation 1    ...        ...        <gg>
2 observation 2    ...        ...        <gg>
3 observation 3    ...        ...        <gg>
4 observation 4    ...        ...        <gg>
5 observation 5    ...        ...        <gg>
6 observation 6    ...        ...        <gg>
7 observation 7    ...        ...        <gg>
8 observation 8    ...        ...        <gg>
9 observation 9    ...        ...        <gg>
10 observation 10   ...        ...        <gg>
# ... with 90 more rows
```



Tidy R tools

There are four basic principles to a tidy API:

- Reuse existing data structures.
- Compose simple functions with the pipe.
- Embrace functional programming.
- Design for humans.

```
# A tibble: 100 x 8
  observation      variable_1 variable_2 variable_3 variable_4
  <glue>        <chr>       <chr>      <list>      <list>
1 observation 1 ...        ...        <gg>        <tibble [10 × 2]>
2 observation 2 ...        ...        <gg>        <tibble [10 × 2]>
3 observation 3 ...        ...        <gg>        <tibble [10 × 2]>
4 observation 4 ...        ...        <gg>        <tibble [10 × 2]>
5 observation 5 ...        ...        <gg>        <tibble [10 × 2]>
6 observation 6 ...        ...        <gg>        <tibble [10 × 2]>
7 observation 7 ...        ...        <gg>        <tibble [10 × 2]>
8 observation 8 ...        ...        <gg>        <tibble [10 × 2]>
9 observation 9 ...        ...        <gg>        <tibble [10 × 2]>
10 observation 10 ...       ...        <gg>        <tibble [10 × 2]>
# ... with 90 more rows
```



Tidy R tools

There are four basic principles to a tidy API:

- Reuse existing data structures.
- Compose simple functions with the pipe.
- Embrace functional programming.
- Design for humans.

```
# A tibble: 100 x 8
  observation      variable_1 variable_2 variable_3 variable_4      variable_5
  <glue>        <chr>     <chr>     <list>     <list>     <list>
1 observation 1 ...       ...       <gg>       <tibble [10 × 2]> <lm>
2 observation 2 ...       ...       <gg>       <tibble [10 × 2]> <lm>
3 observation 3 ...       ...       <gg>       <tibble [10 × 2]> <lm>
4 observation 4 ...       ...       <gg>       <tibble [10 × 2]> <lm>
5 observation 5 ...       ...       <gg>       <tibble [10 × 2]> <lm>
6 observation 6 ...       ...       <gg>       <tibble [10 × 2]> <lm>
7 observation 7 ...       ...       <gg>       <tibble [10 × 2]> <lm>
8 observation 8 ...       ...       <gg>       <tibble [10 × 2]> <lm>
9 observation 9 ...       ...       <gg>       <tibble [10 × 2]> <lm>
10 observation 10 ...      ...       <gg>       <tibble [10 × 2]> <lm>
# ... with 90 more rows
```

Tidy R tools

There are four basic principles to a tidy API:

- Reuse existing data structures.
- Compose simple functions with the pipe.
- Embrace functional programming.
- Design for humans.

```
# A tibble: 100 x 8
  observation     variable_1 variable_2 variable_3 variable_4      variable_5 variable_6 variable_7
  <glue>        <chr>      <chr>      <list>      <list>      <list>      <list>      <list>
1 observation 1 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
2 observation 2 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
3 observation 3 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
4 observation 4 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
5 observation 5 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
6 observation 6 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
7 observation 7 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
8 observation 8 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
9 observation 9 ...       ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
10 observation 10 ...      ...       <gg>       <tibble [10 × 2]> <lm>       <Seurat[,80]> <SnglCllE[,80...
# ... with 90 more rows
```

Tidy R tools

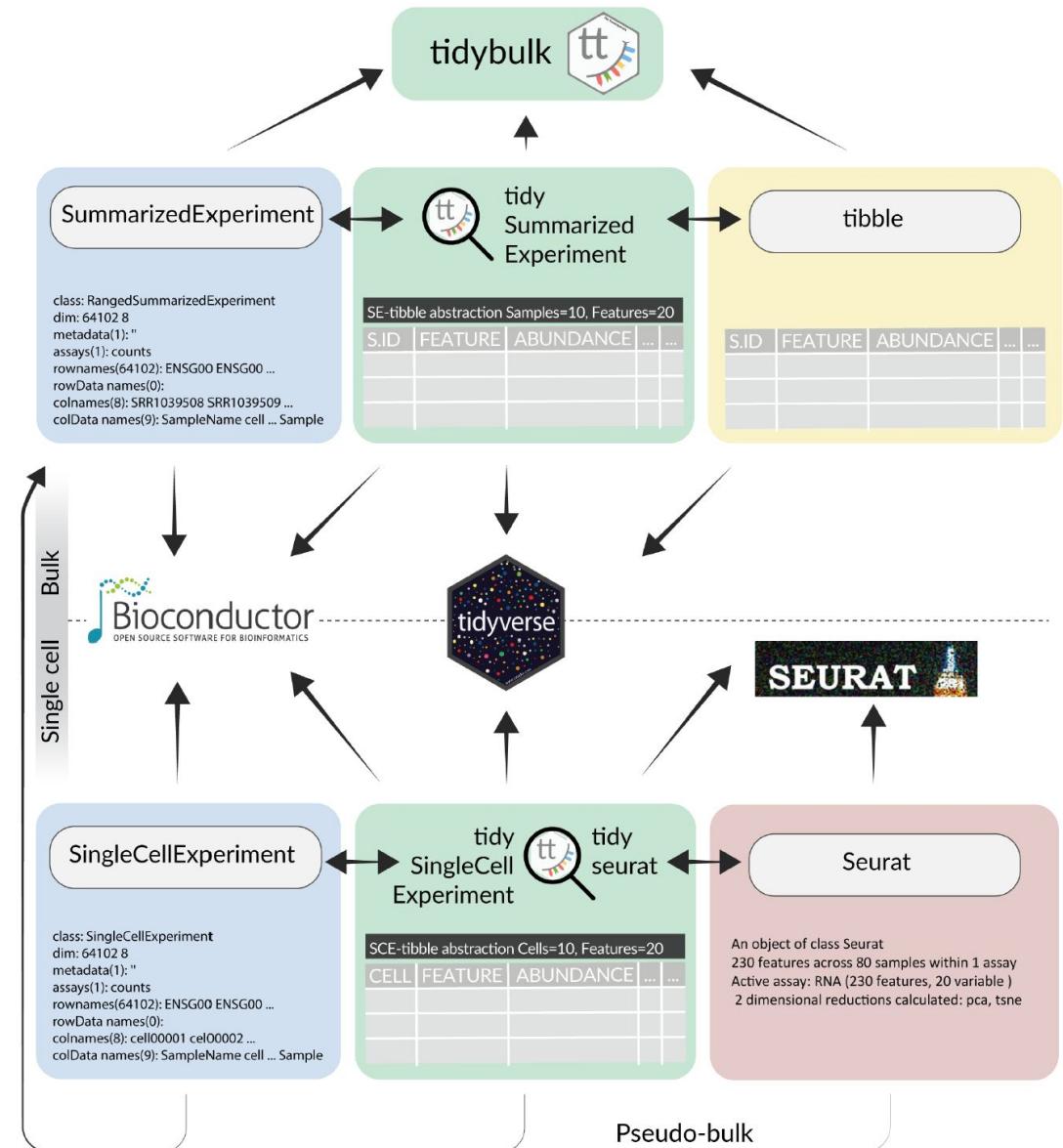
Base R

```
# Filter rows for A class  
data_frame = data_frame[data_frame$class == "A",]  
  
# Create column  
data_frame$x = data_frame$a * data_frame$b  
  
# Plot  
plot(data_frame$x, data_frame$y)
```

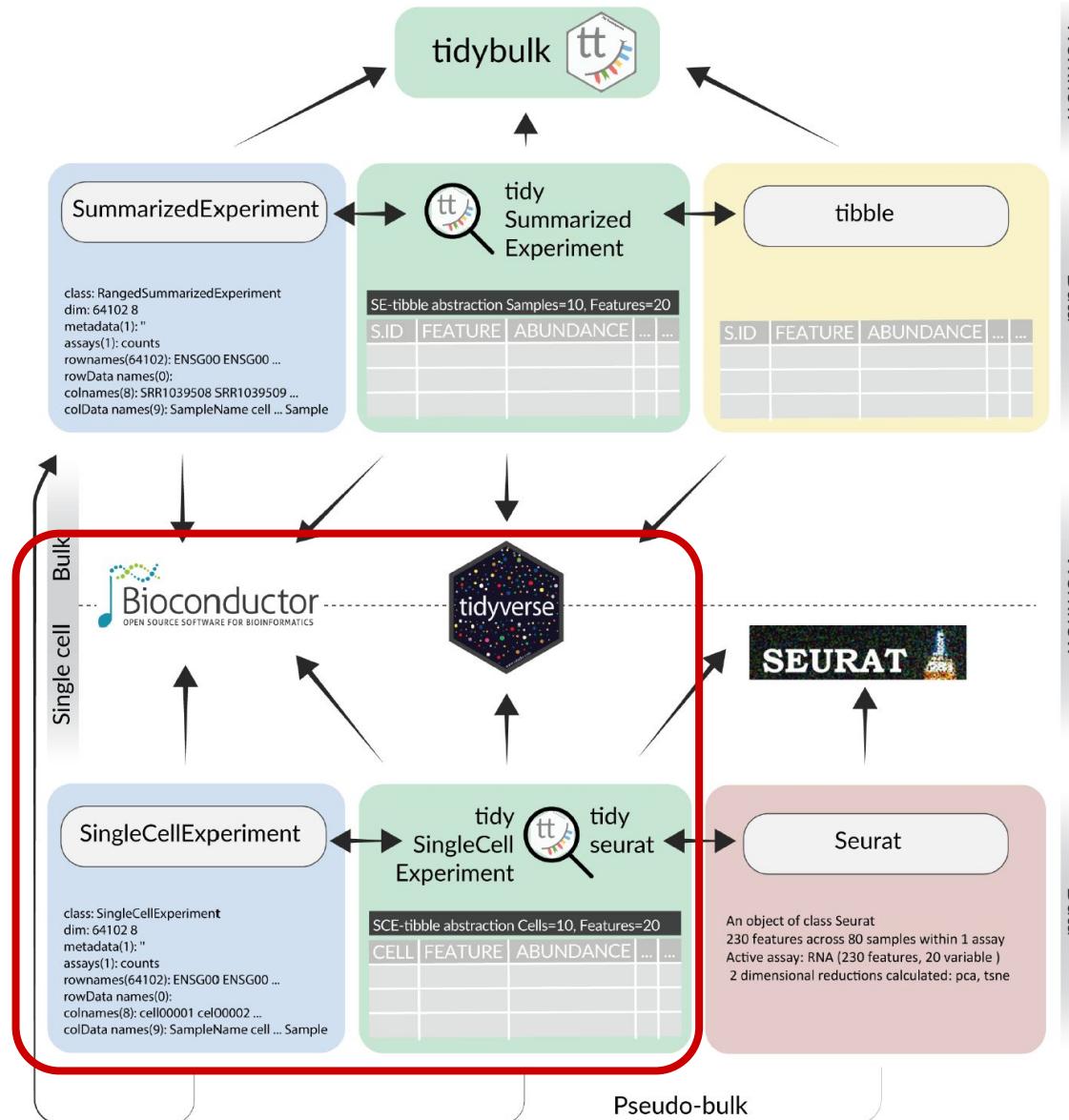
Tidy R

```
data_frame |>  
  
# Filter rows for A class  
filter(class == "A") |>  
  
# Create column  
mutate(x = a * b) |>  
  
# Plot  
ggplot2.scatterplot(x, y)
```

The big picture



The big picture





Analysis infrastructure for single-cell data

Data container



```
class: SingleCellExperiment
dim: 51958 3000
metadata(0):
assays(2): counts logcounts
rownames(51958): DDX11L1 WASH7P ... RP11-141O19.1 RP11-341E12.1
rowData names(0):
colnames(3000): CCAGTCACACTGGT-1 ATGAGCACATCTTC-1 ...
colData names(7): file orig.ident ... G2M.Score ident
reducedDimNames(0):
mainExpName: NULL
altExpNames(0):
```

Analysis infrastructure for single-cell data



Data container

```
class: SingleCellExperiment
dim: 51958 3000
metadata(0):
assays(2): counts logcounts
rownames(51958): DDX11L1 WASH7P ...
rowData names(0):
colnames(3000): CCAGTCACACTGGT-1 ATGAGCACATCTTC-1 ...
colData names(7): file orig.ident ...
reducedDimNames(0):
mainExpName: NULL
altExpNames(0):
```

Analysis

Bioconductor
community

scran/scater

Manipulation

```
colData(data)
reducedDims(data, "umap")
subset(data, , class=="A")
data$info = info

data = cbind( data, cohort_info[
  match(data$sample, cohort_info$sample)
])
subset(data, , !is.na(sample_id))
```

Tidy data representation

Data container

```
# A SingleCellExperiment-tibble abstraction: 80 x 15
# Features=230 | Assays=counts, logcounts
  cell      orig.ident    nCount_RNA nFeature_RNA RNA_snn_res.0.8 letter.idents groups RNA_snn_res.1    PC_1    PC_2    PC_3    PC_4    PC_5    tSNE_1    tSNE_2
  <chr>     <fct>        <dbl>     <int> <fct>           <fct>     <chr>   <fct>        <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
 1 ATGCCAGAACGACT SeuratProject    70         47 0             A          g2      0       -0.774  -0.900  -0.249  0.559  0.465  0.868  -8.10
 2 CAT66CCTGTGCAT SeuratProject    85         52 0             A          g1      0       -0.0268 -0.347  0.665  0.418  0.585  -7.39   -8.77
 3 GAACCTGATGAACC SeuratProject    87         50 1             B          g2      0       -0.457  0.180  1.32   2.01   -0.482  -28.2   0.241
 4 TGACTGGATTCTCA SeuratProject   127        56 0             A          g2      0       -0.812  -1.38   -1.00   0.139  -1.60   16.3   -11.2
 5 AGTCAGACTGCACA SeuratProject   173        53 0             A          g2      0       -0.774  -0.900  -0.249  0.559  0.465  1.91   -11.2
 6 TCTGATACACGTGT SeuratProject   70         48 0             A          g1      0       -0.774  -0.900  -0.249  0.559  0.465  3.15   -9.94
 7 TGGTATCTAACAG SeuratProject   64         36 0             A          g1      0       -0.460  -1.19   -0.312  0.716  -1.65   17.9   -9.90
 8 GCAGCTCTGTTTCT SeuratProject   72         45 0             A          g1      0       -0.900  -0.388  0.693  0.404  0.536  -6.49   -8.39
 9 GATATAACACGCAT SeuratProject   52         36 0             A          g1      0       -0.774  -0.900  -0.249  0.559  0.465  1.33   -9.68
10 AATGTTGACAGTCA SeuratProject  100        41 0             A          g1      0       -0.488  -1.16   -0.306  0.702  -1.47   17.0   -9.43
# ... with 70 more rows
```

Tidy analysis infrastructure



Data container

```
# A SingleCellExperiment-tibble abstraction: 80 x 15
# Features=230 | Assays=counts, logcounts
# ... with 70 more rows
```

cell	orig.ident	nCount_RNA	nFeature_RNA	RNA_snn_res.0.8
<chr>	<fct>	<dbl>	<int>	<fct>
1 ATGCCAGAACGACT	SeuratProject	70	47	0
2 CATGGCTGTGCAT	SeuratProject	85	52	0
3 GAACCTGTGAACC	SeuratProject	87	50	1
4 TGACTGGATTCTCA	SeuratProject	127	56	0
5 AGTCAGACTGCACA	SeuratProject	173	53	0
6 TCTGATACACCGTGT	SeuratProject	70	48	0
7 TGGTATCTAACACAG	SeuratProject	64	36	0
8 GCAGCTCTGTTCT	SeuratProject	72	45	0
9 GATATAAACACGCAT	SeuratProject	52	36	0
10 AATGTTGACAGTCA	SeuratProject	100	41	0

Analysis

Bioconductor community

Manipulation

```
data  
data |> select(contains("UMAP"))  
data |> filter(class=="A")  
data |> mutate(info = info)  
  
data |> inner_join(cohort_info, by="sample")
```

Tidyseurat and tidySingleCellExperiment



Data container

```
# A SingleCellExperiment-tibble abstraction: 80 x 15
# Features=230 | Assays=counts, logcounts
# ... with 70 more rows
```

cell	orig.ident	nCount_RNA	nFeature_RNA	RNA_snn_res.0.8
<chr>	<fct>	<dbl>	<int>	<fct>
1 ATGCCAGAACGACT	SeuratProject	70	47	0
2 CATGGCTGTGCAT	SeuratProject	85	52	0
3 GAACCTGTGAACC	SeuratProject	87	50	1
4 TGACTGGATTCTCA	SeuratProject	127	56	0
5 AGTCAGACTGCACA	SeuratProject	173	53	0
6 TCTGATACACGTGT	SeuratProject	70	48	0
7 TGGTATCTAACAG	SeuratProject	64	36	0
8 GCAGCTCTTTCT	SeuratProject	72	45	0
9 GATATAAACACGCAT	SeuratProject	52	36	0
10 AATGTTGACAGTCA	SeuratProject	100	41	0



```
# A Seurat-tibble abstraction: 80 x 15
# Features=230 | Active assay=RNA | Assays=RNA
  cell      orig.ident nCount_RNA nFeature_RNA RNA_snn_res.0.8
  <chr>     <fct>        <dbl>       <int>      <fct>
1 ATGCCAGAACGACT SeuratProject    70          47  0
2 CATGGCCTGTGCAT SeuratProject    85          52  0
3 GAACCTGTGAACC SeuratProject    87          50  1
4 TGAATTGGATTCTCA SeuratProject   127         56  0
5 AGTCAGACTGCACA SeuratProject   173         53  0
6 TCTGATAACACGTGT SeuratProject   70          48  0
7 TGGTATCTAACAGAG SeuratProject   64          36  0
8 GCAGCTCTGTTCT SeuratProject    72          45  0
9 GATATAAACACGCAT SeuratProject   52          36  0
10 AATGTTGACAGTCA SeuratProject   100         41  0
# ... with 70 more rows
```

Analysis

Bioconductor community

Manipulation

```
data  
data |> select(contains("UMAP"))  
data |> filter(class=="A")  
data |> mutate(info = info)  
  
data |> inner_join(cohort_info, by="sample")
```

Seurat
SeuratWrappers
community

```
data  
data |> select(contains("UMAP"))  
data |> filter(class=="A")  
data |> mutate(info = info)  
  
data |> inner_join(cohort_info, by="sample")
```

Tidy operators available

as_tibble()
mutate()
bind_rows()
left_join() inner_join() *_join()
select() **distinct()**
count() add_count() **summarise()**
pull() slice()
filter() sample_n() sample_frac()
rename()
separate() unite() extract()
nest() unnest() map_()
pivot_longer()
join_features()
ggplot()
plotly()



What tidy data frameworks are and what are not

NO: data containers

NO: analysis tools

YES: data interface

YES: manipulation, integration, visualisation tools

Therefore, the question “**can we go from tidySingleCellExperiment to SingleCellExperiment and vice versa**” is not relevant, as we never leave **SingleCellExperiment**.