cummeRbund: Visualization and Exploration of Cufflinks High-throughput Sequencing Data

Loyal A. Goff, Cole Trapnell

1 April, 2011

Contents

1 Introduction

cummeRbund is a visualization package for Cufflinks high-throughput sequencing data. The base class, *cuffSet* extends the Biobase *eSet* class and is modeled after the *ExpressionSet* class. In its current iteration a *cuffSet* object can store either gene-level or transcript-level FPKM tracking output from a cufflinks analysis. This will be expanded in future development to enable concurrent analysis of both gene-level and transcript-level expression data.

2 Reading cufflinks output

```
> curdir <- getwd()
> cuffFile <- file.path("../../extdata", "genes.fpkm_tracking")
> cuff <- readCufflinks(cuffFile)
> cuff

cuffSet (storageMode: lockedEnvironment)
assayData: 500 features, 3 samples
  element names: conf_hi, conf_lo, fpkm
protocolData: none
phenoData
  sampleNames: H1.hESC Fibroblasts iPS
  varLabels: sample
  varMetadata: labelDescription
featureData
  featureNames: XLOC_000001 XLOC_000002 ... XLOC_000500
    (500 total)
  fvarLabels: class_code nearest_ref_id ... symbol (10
    total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

2.1 Reading additional annotation files

Additional feature and sample information can be used during the creation of the cuffSet object. For example, a 'pData.txt' file can be created where rows correspond to samples in the cufflinks output and an arbitrary number of columns correspond to parameterizations of the sample data. (By default, there is a header row that corresponds to the name of the parameter).

```
> phenoDataFile <- file.path("../../extdata", "pData.txt")
> cuff <- readCufflinks(cuffFile, phenoDataFile = phenoDataFile)
> head(pData(cuff))

            selection cell_line      sample
H1.hESC         polyA        H1     H1.hESC
Fibroblasts     polyA     IMR90 Fibroblasts
iPS             polyA      iPSC         iPS
```

A 'feature data' file can be created and added in a similar manner in which case the rows correspond to features in the cufflinks output and columns are again feature-level parameterizations of the data.

3 The cuffSet object

3.1 Subsetting

cuffSet objects are subsettable in a manner similar to *eSet* objects. *cuffSet* classes can be subset just as one would subset a standard matrix in R. The first argument subsets the rows (features) and the second argument subsets the columns (samples). Below are a few examples:

Create a *cuffSet* object using only features from indices 100–200:

```
> cuff[100:200, ]
```

```
cuffSet (storageMode: lockedEnvironment)
assayData: 101 features, 3 samples
  element names: conf_hi, conf_lo, fpkm
protocolData: none
phenoData
  sampleNames: H1.hESC Fibroblasts iPS
  varLabels: selection cell_line sample
  varMetadata: labelDescription
featureData
  featureNames: XLOC_000100 XLOC_000101 ... XLOC_000200
    (101 total)
  fvarLabels: class_code nearest_ref_id ... symbol (10
    total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

Only use samples with indices 1 & 4.

```
> cuff[, c(1, 3)]
```

```
cuffSet (storageMode: lockedEnvironment)
assayData: 500 features, 2 samples
  element names: conf_hi, conf_lo, fpkm
protocolData: none
phenoData
  sampleNames: H1.hESC iPS
  varLabels: selection cell_line sample
  varMetadata: labelDescription
featureData
  featureNames: XLOC_000001 XLOC_000002 ... XLOC_000500
    (500 total)
  fvarLabels: class_code nearest_ref_id ... symbol (10
    total)
  fvarMetadata: labelDescription
```

```
experimentData: use 'experimentData(object)'
Annotation:
```

cuffSet objects can even be subset by using feature or sample names. Here is an example of a *cuffSet* subset that returns all features for one particular sample:

```
> cuff[, "H1.hESC"]

cuffSet (storageMode: lockedEnvironment)
assayData: 500 features, 1 samples
  element names: conf_hi, conf_lo, fpkm
protocolData: none
phenoData
  sampleNames: H1.hESC
  varLabels: selection cell_line sample
  varMetadata: labelDescription
featureData
  featureNames: XLOC_000001 XLOC_000002 ... XLOC_000500
    (500 total)
  fvarLabels: class_code nearest_ref_id ... symbol (10
    total)
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
Annotation:
```

3.2 Assay Data

Currently, a *cuffSet* object stores three independent matrices in the *Assay Data* slot. The *fpkm* slot stores the FPKM values. *conf_lo* and *conf_hi* store the lower and upper bounds of the confidence intervals for each FPKM in *fpkm*. FPKM values can be directly accessed (get and set) by using the *fpkm* method.

```
> head(fpkm(cuff))

               H1.hESC Fibroblasts       iPS
XLOC_000001 8.87703000  0.11969500 1.9170900
XLOC_000002 0.02992840  0.00475221 0.6511980
XLOC_000003 6.22087000  0.06915000 1.1499800
XLOC_000004 0.16056100  0.03659030 0.0230601
XLOC_000005 0.01766200  0.03862350 0.2127890
XLOC_000006 0.00322352  0.00426650 0.0000000
```

Similar accessor methods are available for *conf_lo* and *conf_hi*.

3.3 Phenotypic (Sample) Data

The *phenoData* slot is an *AnnotatedDataFrame* object. While the *phenoData* slot is directly accessible, the recommended method for getting and setting *phenoData* values is through the *pData* method. This slot is only populated with *sample* names by default (from cufflinks header).

```
> pData(cuff)
```

```
            selection cell_line      sample
H1.hESC         polyA        H1     H1.hESC
Fibroblasts     polyA     IMR90 Fibroblasts
iPS             polyA      iPSC         iPS
```

```
> pData(cuff)$selection == "polyA"
```

```
[1] TRUE TRUE TRUE
```

The *sampleNames* method can be used to retrieve the sample names from the *cuffSet* object.

```
> sampleNames(cuff)
```

```
[1] "H1.hESC"     "Fibroblasts" "iPS"
```

3.4 Feature Data

The *FeatureData* slot is also an *AnnotatedDataFrame* object. By default, the *featureData* slot is populated with the first few annotation rows from the cufflinks output file. Similar to *phenoData*, *featureData* values can be accessed via the *fData* method.

```
> head(fData(cuff))
```

```
            class_code nearest_ref_id     gene_id
XLOC_000001         NA             NA XLOC_000001
XLOC_000002         NA             NA XLOC_000002
XLOC_000003         NA             NA XLOC_000003
XLOC_000004         NA             NA XLOC_000004
XLOC_000005         NA             NA XLOC_000005
XLOC_000006         NA             NA XLOC_000006
            gene_short_name         tss_id              locus
XLOC_000001   linc-OR4F16-4 TSS1,TSS2,TSS3  chr1:77369-328580
XLOC_000002   linc-OR4F16-3           TSS4  chr1:77369-328580
XLOC_000003   linc-OR4F16-2           TSS5  chr1:77369-328580
XLOC_000004   linc-OR4F16-1      TSS6,TSS7 chr1:329783-565234
XLOC_000005   linc-SAMD11-9      TSS8,TSS9 chr1:329783-565234
XLOC_000006   linc-SAMD11-8          TSS10 chr1:329783-565234
            length coverage status      symbol
XLOC_000001     NA       NA     OK XLOC_000001
XLOC_000002     NA       NA     OK XLOC_000002
XLOC_000003     NA       NA     OK XLOC_000003
```

```
XLOC_000004      NA        NA      OK XLOC_000004
XLOC_000005      NA        NA      OK XLOC_000005
XLOC_000006      NA        NA      OK XLOC_000006
```

The *featureNames* method can be used to retrieve the feature names from the *cuffSet* object.

```
> head(featureNames(cuff))
```

```
[1] "XLOC_000001" "XLOC_000002" "XLOC_000003" "XLOC_000004"
[5] "XLOC_000005" "XLOC_000006"
```

3.5 Experiment Data

4 Plotting

All plotting for the cummeRbund package is done through ggplot2. As such, the returned object for most plots is a ggplot plot object. Using standard ggplot2 syntax, one can add options, themes, or facet plots in any fashion. All *featureData* and *phenoData* parameters are included within the plot data, and are accessible to the returned object.

4.1 Global statistics

Several plotting methods are available that allow for quality-control or global analysis of cufflinks data. For example, to assess the distributions of FPKM scores across samples, you can use the *csDensity* plot (Figure 1).

```
> dens <- csDensity(cuff)
```

Boxplots can be visualized using the *csBoxplot* method (Figure 2).

```
> b <- csBoxplot(cuff)
```

Pairwise comparisons can be made by using *csScatter* (Figure 3). You must specify the samples to use for the x and y axes:

```
> s <- csScatter(cuff, 1, 3, smooth = T)
```

```
[1] "H1.hESC" "iPS"
```

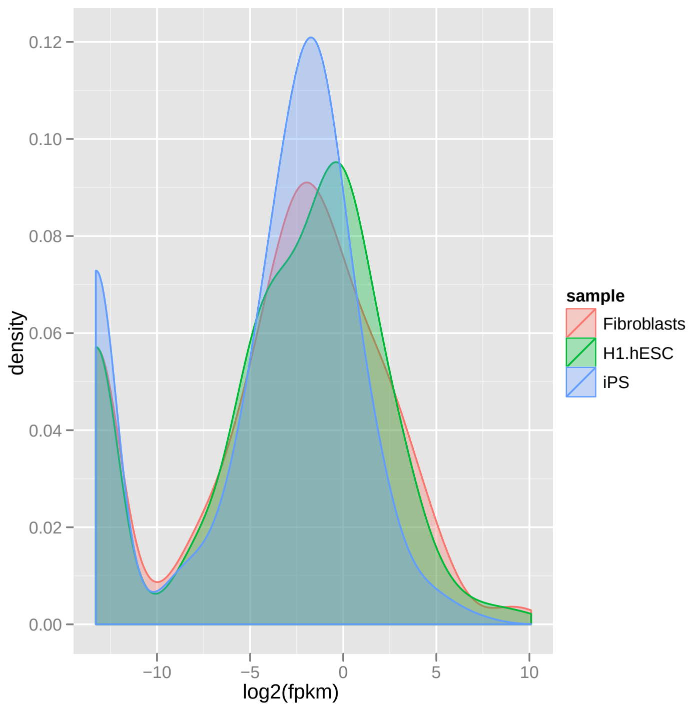

Figure 1: Density plot per sample of cufflinks output FPKM values.

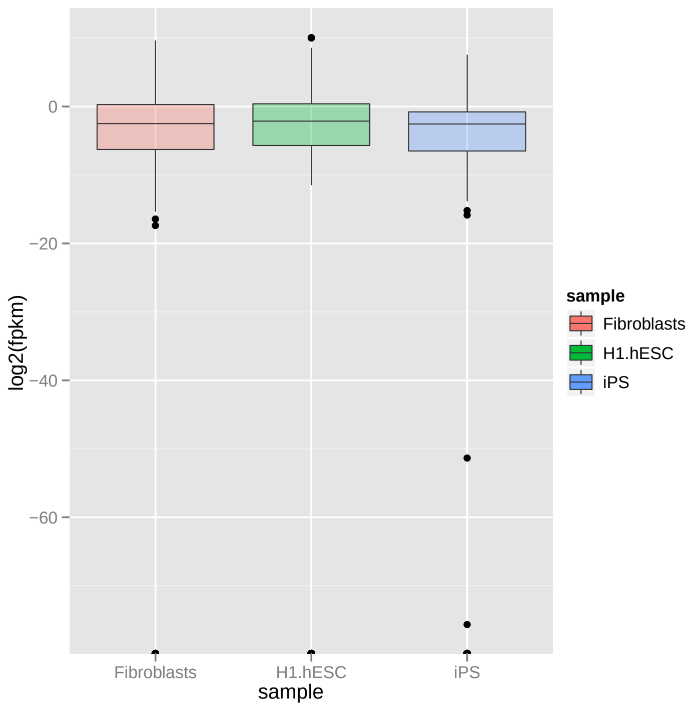

Figure 2: Box plot of FPKM values from cufflinks output.

```
[1] "H1.hESC" "iPS"
```

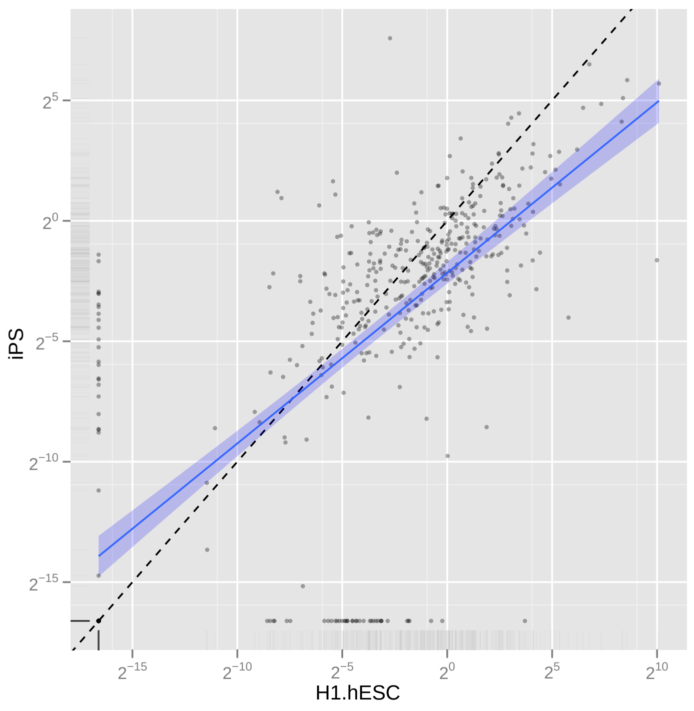

Figure 3: Scatter plot comparing the FPKM values of two samples from cufflinks output.

4.2 Feature-level plots

Several methods are available for plotting feature-level data from cufflinks output. The *expressionPlot* method will produce line plots for all features contained in the *cuffSet* object:

```
> e1 <- expressionPlot(cuff[50:80, ], drawSummary = T)
```

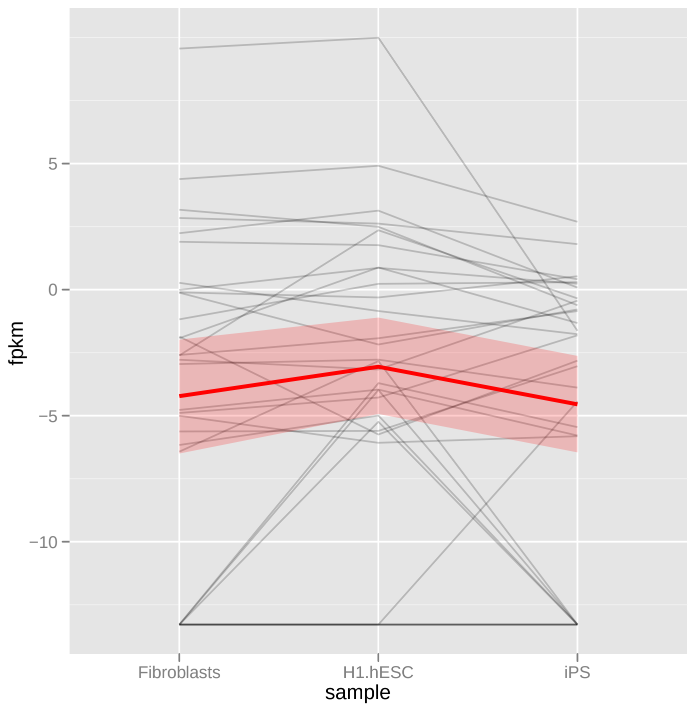

Figure 4: Line plot of select features from cufflinks output.

Alternatively you can use *expressionBarplot* to make a bar plot with error bars for a given set of features.

```
> e2 <- expressionBarplot(cuff[c(24, 77, 493), ])
```

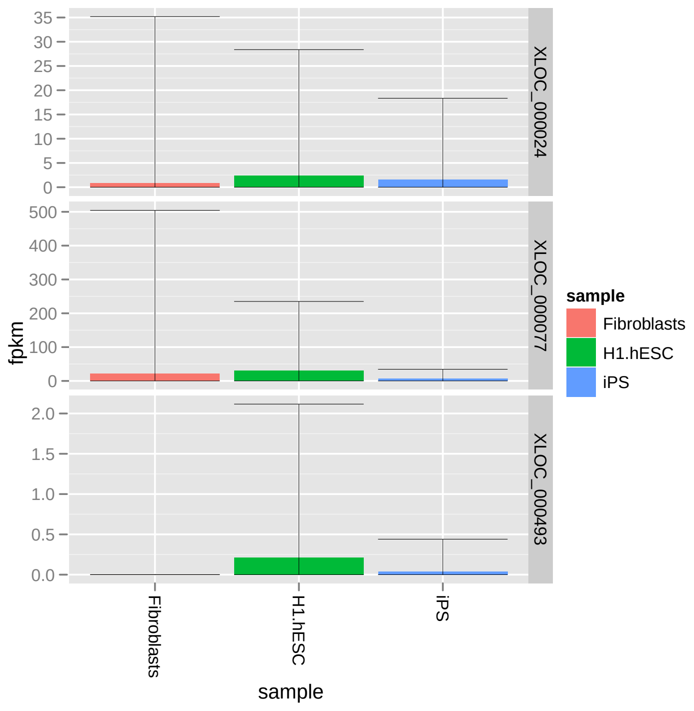

Figure 5: Bar plot of FPKM values with error bars.

5 Session info

```
> sessionInfo()

R version 2.12.1 (2010-12-16)
Platform: x86_64-apple-darwin9.8.0/x86_64 (64-bit)

locale:
[1] C/en_US.utf-8/C/C/en_US.utf-8/en_US.utf-8

attached base packages:
[1] splines   grid      stats     graphics  grDevices utils
[7] datasets  methods   base

other attached packages:
[1] Hmisc_3.8-3      survival_2.36-2  cummeRbund_0.1.1
[4] ggplot2_0.8.9    proto_0.3-8      reshape_0.8.3
[7] plyr_1.4         Biobase_2.10.0

loaded via a namespace (and not attached):
[1] cluster_1.13.2  digest_0.4.2    lattice_0.19-17
[4] tools_2.12.1
```