

# curatedOvarianData

Benjamin Frederick Ganzfried, Markus Riester, Benjamin Haibe-Kains, Thomas Risch, Svitlana Tyekucheva, Ina Jazic, Victoria Xin Wang, Mahnaz Ahmadifar, Michael Birrer, Giovanni Parmigiani, Curtis Huttenhower, Levi Waldron

2013

## Contents

<b>1</b>	<b>curatedOvarianData: Clinically Annotated Data for the Ovarian Cancer Transcriptome</b>	<b>1</b>
<b>2</b>	<b>Load TCGA data</b>	<b>2</b>
<b>3</b>	<b>Load datasets based on rules</b>	<b>2</b>
<b>4</b>	<b>Association of CXCL12 expression with overall survival</b>	<b>5</b>
<b>5</b>	<b>Batch correction with ComBat</b>	<b>9</b>
<b>A</b>	<b>Available Clinical Characteristics</b>	<b>12</b>
<b>B</b>	<b>Summarizing the List of ExpressionSets</b>	<b>12</b>
<b>C</b>	<b>For non-R users</b>	<b>13</b>
<b>D</b>	<b>Session Info</b>	<b>13</b>

## 1 curatedOvarianData: Clinically Annotated Data for the Ovarian Cancer Transcriptome

This package represents a manually curated data collection for gene expression meta-analysis of patients with ovarian cancer. This resource provides uniformly prepared microarray data with curated and documented clinical metadata. It allows a computational user to efficiently identify studies and patient subgroups of interest for analysis and to run such analyses immediately without the challenges posed by harmonizing heterogeneous microarray technologies, study designs, expression data processing methods, and clinical data formats.

Please see <http://bcf.dfc.harvard.edu/ovariancancer> for alternative versions of this package, differing in how redundant probe sets are dealt with.

In this vignette, we give a short tour of the package and will show how to use it efficiently.

## 2 Load TCGA data

Loading a single dataset is very easy. First we load the package:

```
> library(curatedOvarianData)
```

To get a listing of all the datasets, use the `data` function:

```
> data(package="curatedOvarianData")
```

Now to load the TCGA data, we use the `data` function again:

```
> data(TCGA_eset)
> TCGA_eset
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 12981 features, 578 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: TCGA.20.0987 TCGA.23.1031 ...
               TCGA.13.1819 (578 total)
  varLabels: alt_sample_name unique_patient_ID ...
             uncurated_author_metadata (29 total)
  varMetadata: labelDescription
featureData
  featureNames: A1CF A2M ... ZZZ3 (12981 total)
  fvarLabels: probeset gene
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
pubMedIds: 21720365
Annotation: hthgu133a
```

The datasets are provided as Bioconductor `ExpressionSet` objects and we refer to the Bioconductor documentation for users unfamiliar with this data structure.

## 3 Load datasets based on rules

For a meta-analysis, we typically want to filter datasets and patients to get a population of patients we are interested in. We provide a short but powerful R script that does the filtering and provides the data as a list of `ExpressionSet` objects. One can use this script within R by first sourcing a config file which specifies the filters, like the minimum numbers of patients in each dataset. It is also possible to filter samples by annotation, for example to remove early stage and normal samples.

```
> source(system.file("extdata",
+ "patientselection.config", package="curatedOvarianData"))
> ls()
```

```

[1] "TCGA_eset"          "add.surv.y"
[3] "keep.common.only"   "meta.required"
[5] "min.number.of.events" "min.number.of.genes"
[7] "min.sample.size"     "quantile.cutoff"
[9] "rescale"             "rule.1"
[11] "strict.checking"

```

See what the values of these variables we have loaded are. The variable names are fairly descriptive, but note that “rule.1” is a character vector of length 2, where the first entry is the name of a clinical data variable, and the second entry is a Regular Expression providing a requirement for that variable. Any number of rules can be added, with increasing identifiers, e.g. “rule.2”, “rule.3”, etc.

Here `strict.checking` is `FALSE`, meaning that samples not annotated for the variables in these rules are allowed to pass the filter. If `strict.checking == TRUE`, samples missing this annotation will be removed.

```
> sapply(ls(), get)
```

```

$TCGA_eset
ExpressionSet (storageMode: lockedEnvironment)
assayData: 12981 features, 578 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: TCGA.20.0987 TCGA.23.1031 ...
                TCGA.13.1819 (578 total)
  varLabels: alt_sample_name unique_patient_ID ...
                uncured_author_metadata (29 total)
  varMetadata: labelDescription
featureData
  featureNames: A1CF A2M ... ZZZ3 (12981 total)
  fvarLabels: probeset gene
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
  pubMedIds: 21720365
Annotation: hthgu133a

$add.surv.y
function (X)
Surv(X$days_to_death, X$vital_status == "deceased")

$keep.common.only
[1] FALSE

$meta.required
[1] "days_to_death" "vital_status"

$min.number.of.events
[1] 15

$min.number.of.genes
[1] 1000

$min.sample.size

```

```
[1] 40
```

```
$quantile.cutoff  
[1] 0
```

```
$rescale  
[1] TRUE
```

```
$rule.1  
[1] "sample_type" "^tumor$"
```

```
$strict.checking  
[1] FALSE
```

Now that we have defined the sample filter, we create a list of `ExpressionSets` by sourcing the `createEsetList.R` file:

```
> source(system.file("extdata", "createEsetList.R", package =  
+ "curatedOvarianData"))
```

```
INFO [2014-04-12 12:27:27] Inside script createEsetList.R - inputArgs =  
INFO [2014-04-12 12:27:27] Loading curatedOvarianData 1.2.0  
INFO [2014-04-12 12:28:11] Clean up the esets.  
INFO [2014-04-12 12:28:12] including E.MTAB.386_eset  
INFO [2014-04-12 12:28:12] excluding GSE12418_eset (min.number.of.events or min.sample.size)  
INFO [2014-04-12 12:28:12] excluding GSE12470_eset (min.number.of.events or min.sample.size)  
INFO [2014-04-12 12:28:13] including GSE13876_eset  
INFO [2014-04-12 12:28:13] including GSE14764_eset  
INFO [2014-04-12 12:28:14] including GSE17260_eset  
INFO [2014-04-12 12:28:14] including GSE18520_eset  
INFO [2014-04-12 12:28:15] excluding GSE19829.GPL570_eset (min.number.of.events or min.sample.size)  
INFO [2014-04-12 12:28:15] including GSE19829.GPL8300_eset  
INFO [2014-04-12 12:28:15] excluding GSE20565_eset (min.number.of.events or min.sample.size)  
INFO [2014-04-12 12:28:15] excluding GSE2109_eset (min.number.of.events or min.sample.size)  
INFO [2014-04-12 12:28:16] including GSE26712_eset  
INFO [2014-04-12 12:28:16] excluding GSE30009_eset (min.number.of.genes)  
INFO [2014-04-12 12:28:16] including GSE30161_eset  
INFO [2014-04-12 12:28:18] including GSE32062.GPL6480_eset  
INFO [2014-04-12 12:28:18] including GSE32063_eset  
INFO [2014-04-12 12:28:18] excluding GSE6008_eset (min.number.of.events or min.sample.size)  
INFO [2014-04-12 12:28:18] excluding GSE6822_eset (min.number.of.events or min.sample.size)  
INFO [2014-04-12 12:28:20] including GSE9891_eset  
INFO [2014-04-12 12:28:20] excluding PMID15897565_eset (min.number.of.events or min.sample.size)  
INFO [2014-04-12 12:28:20] including PMID17290060_eset  
INFO [2014-04-12 12:28:21] including PMID19318476_eset  
INFO [2014-04-12 12:28:21] excluding TCGA.mirna.8x15kv2_eset (min.number.of.genes)  
INFO [2014-04-12 12:28:23] including TCGA_eset  
INFO [2014-04-12 12:28:23] Ids with missing data:
```

It is also possible to run the script from the command line and then load the R data file within R:

```
R --vanilla "--args patientselection.config ovarian.eset.rda tmp.log" < createEsetList.R
```

Now we have 14 datasets with samples that passed our filter in a list of `ExpressionSets` called `esets`:

```
> names(esets)

[1] "E.MTAB.386_eset"      "GSE13876_eset"
[3] "GSE14764_eset"       "GSE17260_eset"
[5] "GSE18520_eset"       "GSE19829.GPL8300_eset"
[7] "GSE26712_eset"       "GSE30161_eset"
[9] "GSE32062.GPL6480_eset" "GSE32063_eset"
[11] "GSE9891_eset"        "PMID17290060_eset"
[13] "PMID19318476_eset"   "TCGA_eset"
```

## 4 Association of CXCL12 expression with overall survival

Next we use the list of 14 datasets from the previous example and test if the expression of the CXCL12 gene is associated with overall survival. CXCL12/CXCR4 is a chemokine/chemokine receptor axis that has previously been shown to be directly involved in cancer pathogenesis.

We first define a function that will generate a forest plot for a given gene. It needs the overall survival information as `Surv` objects, which the `createEsetList.R` function already added in the `phenoData` slots of the `ExpressionSets`, accessible at the `y` label. The resulting forest plot is shown for the CXCL12 gene in Figure 1.

```
> esets[[1]]$y

[1] 840.9+ 399.9+ 524.1+ 1476.0 144.0 516.9
[7] 405.0 87.0 45.9+ 483.9+ 917.1 1013.1+
[13] 69.9 486.0 369.9 2585.1+ 738.9 362.1
[19] 2031.9+ 477.9 1091.1+ 1062.0+ 720.9 1200.9+
[25] 977.1 537.9 638.1 587.1 1509.0 1619.1+
[31] 1043.1 198.9 1520.1 696.9 1140.9 1862.1+
[37] 1751.1+ 1845.0+ 1197.0 1401.0 399.0 992.1
[43] 927.9+ 1509.0 1914.0+ 591.9 426.0 1374.9+
[49] 546.9 809.1+ 480.9+ 486.0+ 642.9+ 540.9+
[55] 962.1 2025.0 473.1 1140.0 512.1 1002.9+
[61] 1731.9+ 690.0 930.0 1026.9 1193.1+ 720.9
[67] 369.0 1326.9+ 501.9+ 1677.0+ 1773.9+ 251.1
[73] 1338.9+ 35.1 1467.9+ 165.9 981.9 1280.1
[79] 1800.0+ 399.9 422.1 861.9 2010.0+ 660.0
[85] 2138.1+ 516.0+ 1001.1+ 693.9 825.0+ 815.1+
[91] 657.0+ 1013.1+ 426.0 656.1 1356.0 1610.1+
[97] 1068.9+ 1221.9+ 2388.0+ 447.9+ 602.1+ 1875.0+
[103] 920.1+ 959.1 708.0 546.0 1254.9+ 611.1+
[109] 1317.9 1899.0 1886.1 642.0 1763.1 276.0
[115] 1857.0+ 540.0 852.9 498.0+ 3.9+ 836.1
[121] 1452.0 2721.0 450.9 1398.9 1481.1 2724.0+
[127] 2061.9 651.9 2349.0+

> forestplot <- function(esets, y="y", probeset, formula=y~probeset,
+ mlab="Overall", rma.method="FE", at=NULL, xlab="Hazard Ratio",...) {
+   require(metafor)
+   esets <- esets[sapply(esets, function(x) probeset %in% featureNames(x))]
+   coefs <- sapply(1:length(esets), function(i) {
```

```
> res <- forestplot(esets=esets,probeset="CXCL12",at=log(c(0.5,1,2,4)))
```

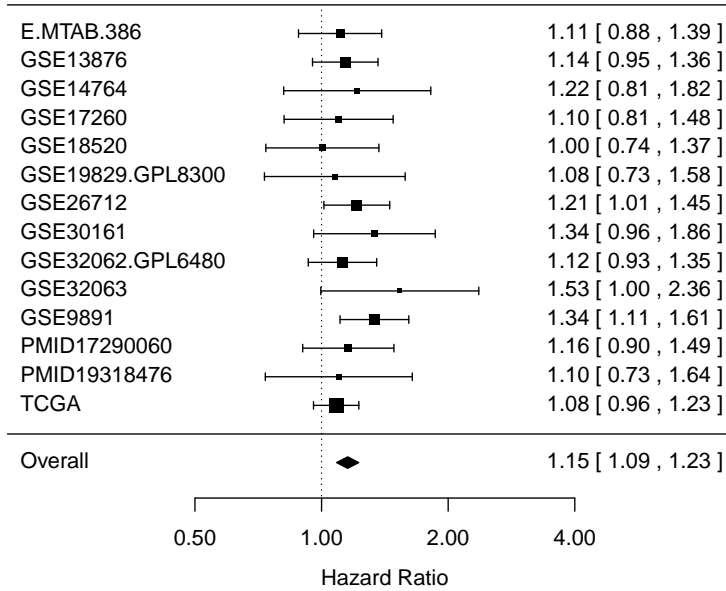


Figure 1: The database confirms CXCL12 as prognostic of overall survival in patients with ovarian cancer. Forest plot of the expression of the chemokine CXCL12 as a univariate predictor of overall survival, using all 14 datasets with applicable expression and survival information. A hazard ratio significantly larger than 1 indicates that patients with high CXCL12 levels had poor outcome. The p-value for the overall HR, found in `res$pval`, is 3.5e-06. This plot is Figure 3 of the `curatedOvarianData` manuscript.

```
+ tmp <- as(phenoData(esets[[i]]), "data.frame")
+ tmp$y <- esets[[i]][[y]]
+ tmp$probeset <- exprs(esets[[i]])[probeset,]
+
+ summary(coxph(formula,data=tmp))$coefficients[1,c(1,3)]
+ })
+
+ res.rma <- metafor::rma(yi = coefs[1,], sei = coefs[2,],
+ method=rma.method)
+
+ if (is.null(at)) at <- log(c(0.25,1,4,20))
+ forest.rma(res.rma, xlab=xlab, slab=gsub("_eset$", "", names(esets)),
+ atranf=exp, at=at, mlab=mlab,...)
+ return(res.rma)
+ }
```

We now test whether CXCL12 is an independent predictor of survival in a multivariate model together with success of debulking surgery, defined as residual tumor smaller

```
> res <- forestplot(esets=esets[idx.debulking & idx.tumorstage],
+   probeset="CXCL12", formula=y~probeset+debulking+tumorstage,
+   at=log(c(0.5,1,2,4)))
```

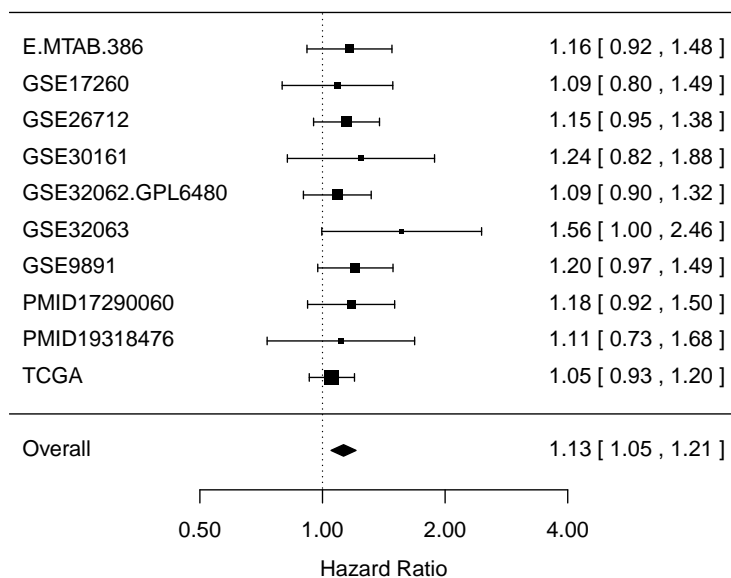


Figure 2: Validation of CXCL12 as an independent predictor of survival. This figure shows a forest plot as in Figure 1, but the CXCL12 expression levels were adjusted for debulking status (optimal versus suboptimal) and tumor stage. The p-value for the overall HR, found in `res$pval`, is 0.00098.

than 1 cm, and Federation of Gynecology and Obstetrics (FIGO) stage. We first filter the datasets without debulking and stage information:

```
> idx.tumorstage <- sapply(esets, function(X)
+   sum(!is.na(X$tumorstage))) > 0
> idx.debulking <- sapply(esets, function(X)
+   sum(X$debulking=="suboptimal",na.rm=TRUE)) > 0
```

In Figure 2, we see that CXCL12 stays significant after adjusting for debulking status and FIGO stage. We repeated this analysis for the CXCR4 receptor and found no significant association with overall survival (Figure 3).

```
> res <- forestplot(esets=esets,probeset="CXCR4",at=log(c(0.5,1,2,4)))
```

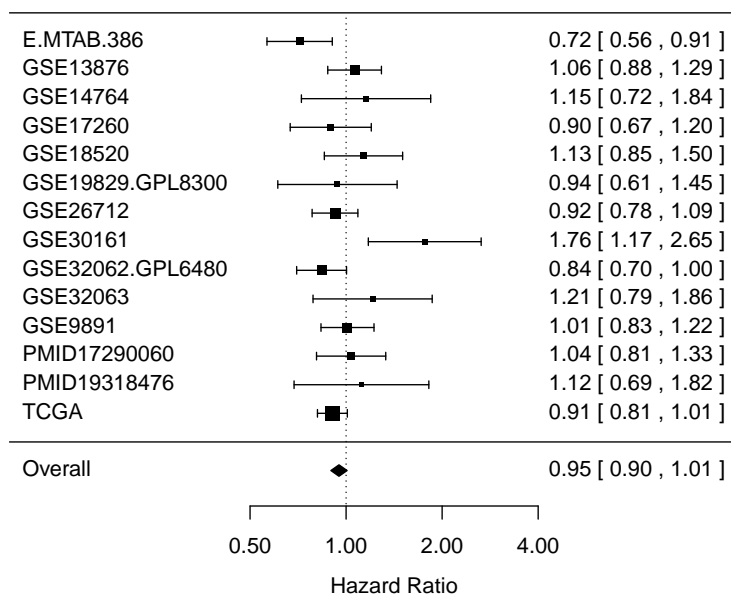


Figure 3: Up-regulation of CXCR4 is not associated with overall survival. This figure shows again a forest plot as in Figure 1, but here the association of mRNA expression levels of the CXCR4 receptor and overall survival is shown. The p-value for the overall HR, found in `res$pval`, is 0.087.



## 5 Batch correction with ComBat

If datasets are merged, it is typically recommended to remove a very likely batch effect. We will use the ComBat [Johnson et al., 2007] method, implemented for example in the SVA Bioconductor package [Leek et al.]. To combine two ExpressionSet objects, we can use the `combine()` function. This function will fail when the two ExpressionSets have conflicting annotation slots, for example `annotation` when the platforms differ. We write a simple `combine2` function which only considers the `exprs` and `phenoData` slots:

```
> combine2 <- function(X1, X2) {
+   fids <- intersect(featureNames(X1), featureNames(X2))
+   X1 <- X1[fids,]
+   X2 <- X2[fids,]
+   ExpressionSet(cbind(exprs(X1), exprs(X2)),
+     AnnotatedDataFrame(rbind(as(phenoData(X1), "data.frame"),
+                               as(phenoData(X2), "data.frame"))))
+ }
+ }
```

In Figure 4, we combined two datasets from different platforms, resulting in a huge batch effect. Now we apply ComBat and adjust for the batch and show the boxplot after batch correction in Figure 5:

```
> mod <- model.matrix(~as.factor(tumorstage), data=X)
> batch <- as.factor(grepl("DFCI", sampleNames(X)))
> combat_edata <- ComBat(dat=exprs(X), batch=batch, mod=mod)
```

```
Found 2 batches
Found 2 categorical covariate(s)
Standardizing Data across genes
Fitting L/S model and finding priors
Finding parametric adjustments
Adjusting the Data
```

```

> data(E.MTAB.386_eset)
> data(GSE30161_eset)
> X <- combine2(E.MTAB.386_eset, GSE30161_eset)
> boxplot(exprs(X))

```

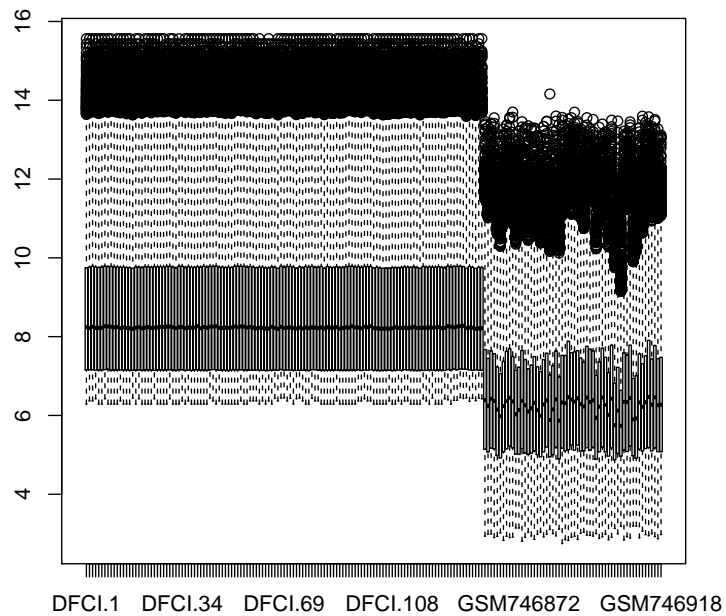


Figure 4: Boxplot showing the expression range for all samples of two merged datasets arrayed on different platforms. This illustrates a huge batch effect.

```
> boxplot(combat_edata)
```

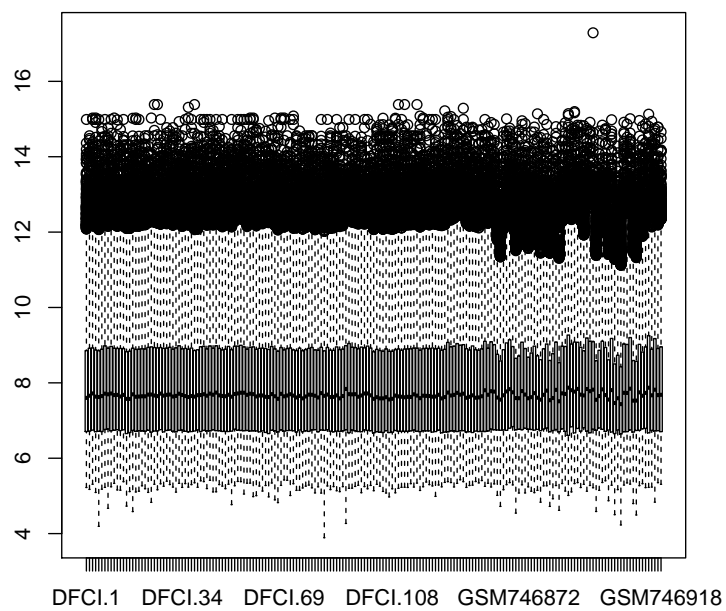


Figure 5: Boxplot showing the expression range for all samples of two merged datasets arrayed on different platforms after batch correction with ComBat.

## References

- W E Johnson, C Li, and A Rabinovic. Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics*, 8(1):118–127, Jan 2007.
- Jeffrey T. Leek, W. Evan Johnson, Hilary S. Parker, Andrew E. Jaffe, and John D. Storey. *sva: Surrogate Variable Analysis*. R package version 3.4.0.

## A Available Clinical Characteristics

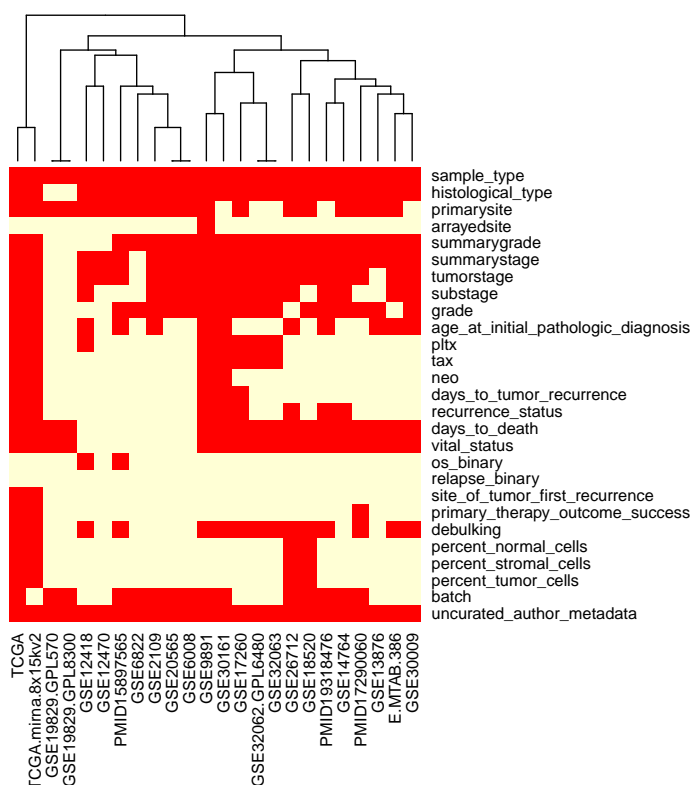


Figure 6: Available clinical annotation. This heatmap visualizes for each curated clinical characteristic (rows) the availability in each dataset (columns). Red indicates that the corresponding characteristic is available for at least one sample in the dataset. This plot is Figure 2 of the curatedOvarianData manuscript.

## B Summarizing the List of ExpressionSets

This example provides a table summarizing the datasets being used, and is useful when publishing analyses based on curatedOvarianData. First, define some useful functions for this purpose:

```
> source(system.file("extdata", "summarizeEsets.R", package =
+   "curatedOvarianData"))
```

Now create the table, used for Table 1 of the curatedOvarianData manuscript:

Optionally write this table to file, for example ( replace `myfile <- tempfile()` with something like `myfile <- "nicetable.csv"` )

```
> (myfile <- tempfile())

[1] "E:\\biocbld\\bbs-2.14-data-experiment\\tmpdir\\Rtmpi8JaGG\\file171c69711c56"

> write.table(summary.table, file=myfile, row.names=FALSE, quote=TRUE, sep=",")
```

## C For non-R users

If you are not doing your analysis in R, and just want to get some data you have identified from the curatedOvarianData manual, here is a simple way to do it. For one dataset:

```
> library(curatedOvarianData)
> library(affy)
> data(GSE30161_eset)
> write.csv(exprs(GSE30161_eset), file="GSE30161_eset_exprs.csv")
> write.csv(pData(GSE30161_eset), file="GSE30161_eset_clindata.csv")
```

Or for several datasets:

```
> data.to.fetch <- c("GSE30161_eset", "E.MTAB.386_eset")
> for (onedata in data.to.fetch){
+   print(paste("Fetching", onedata))
+   data(list=onedata)
+   write.csv(exprs(get(onedata)), file=paste(onedata, "_exprs.csv", sep=""))
+   write.csv(pData(get(onedata)), file=paste(onedata, "_clindata.csv", sep=""))
+ }
```

## D Session Info

- R version 3.1.0 RC (2014-04-02 r65358), i386-w64-mingw32
- Locale: LC\_COLLATE=C, LC\_CTYPE=English\_United States.1252, LC\_MONETARY=English\_United States.1252, LC\_NUMERIC=C, LC\_TIME=English\_United States.1252
- Base packages: base, datasets, grDevices, graphics, methods, parallel, splines, stats, utils
- Other packages: Biobase 2.24.0, BiocGenerics 0.10.0, Formula 1.1-1, affy 1.42.0, corpcor 1.6.6, curatedOvarianData 1.2.0, futile.logger 1.3.7, genefilter 1.46.0, metafor 1.9-2, mgcv 1.7-29, nlme 3.1-117, survival 2.37-7, sva 3.10.0, xtable 1.7-3
- Loaded via a namespace (and not attached): AnnotationDbi 1.26.0, BiocInstaller 1.14.0, DBI 0.2-7, GenomeInfoDb 1.0.0, IRanges 1.21.45, Matrix 1.1-3, RSQLite 0.11.4, XML 3.98-1.1, affyio 1.32.0, annotate 1.42.0, futile.options 1.0.0, grid 3.1.0, lambda.r 1.1.6, lattice 0.20-29, preprocessCore 1.26.0, stats4 3.1.0, tools 3.1.0, zlibbioc 1.10.0

	PMID	N samples	stage	histology	Platform
E.MTAB.386	22348002	129	1/128/0	129/0/0/0/0/0/0	Illumina humanRef-8 v2.0
GSE12418	16996261	54	0/54/0	54/0/0/0/0/0/0	SWEGENE H_v2.1.1_27k
GSE12470	19486012	53	8/35/10	43/0/0/0/0/0/10	Agilent G4110B
GSE13876	19192944	157	0/157/0	157/0/0/0/0/0/0	Operon v3 two-color
GSE14764	19294737	80	9/71/0	68/2/6/0/2/0/2	Affymetrix HG-U133A
GSE17260	20300634	110	0/110/0	110/0/0/0/0/0/0	Agilent G4112A
GSE18520	19962670	63	0/53/10	53/0/0/0/0/0/10	Affymetrix HG-U133Plus2
GSE19829.GPL570	20547991	28	0/0/28	0/0/0/0/0/0/28	Affymetrix HG-U133Plus2
GSE19829.GPL8300	20547991	42	0/0/42	0/0/0/0/0/0/42	Affymetrix HG_U95Av2
GSE20565	20492709	140	27/67/46	71/6/6/7/6/0/44	Affymetrix HG-U133Plus2
GSE2109	PMID unknown	204	37/87/80	85/9/28/11/59/0/12	Affymetrix HG-U133Plus2
GSE26712	18593951	195	0/185/10	185/0/0/0/0/0/10	Affymetrix HG-U133A
GSE30009	22492981	103	0/103/0	102/1/0/0/0/0/0	TaqMan qRT-PCR
GSE30161	22348014	58	0/58/0	47/5/1/1/1/0/3	Affymetrix HG-U133Plus2
GSE32062.GPL6480	22241791	260	0/260/0	260/0/0/0/0/0/0	Agilent G4112F
GSE32063	22241791	40	0/40/0	40/0/0/0/0/0/0	Agilent G4112F
GSE6008	19440550	103	42/53/8	41/8/37/13/0/0/4	Affymetrix HG-U133A
GSE6822	PMID unknown	66	0/0/66	41/11/7/1/0/0/6	Affymetrix Hu6800
GSE9891	18698038	285	42/240/3	264/0/20/0/1/0/0	Affymetrix HG-U133Plus2
PMID15897565	15897565	63	11/52/0	63/0/0/0/0/0/0	Affymetrix HG-U133A
PMID17290060	17290060	117	1/115/1	117/0/0/0/0/0/0	Affymetrix HG-U133A
PMID19318476	19318476	42	2/39/1	42/0/0/0/0/0/0	Affymetrix HG-U133A
TCGA.mirna.8x15kv2	21720365	554	39/511/4	554/0/0/0/0/0/0	Agilent miRNA-8x15k2 G4470B
TCGA	21720365	578	43/520/15	568/0/0/0/0/0/10	Affymetrix HT_HG-U133A

Table 1: Datasets provided by curatedOvarianData. This is an abbreviated version of Table 1 of the manuscript; the full version is written by the write.table command above. Stage column is early/late/unknown, histology column is ser/clearcell/endo/mucinous/other/unknown.