# Exploiting the Depmap cancer dependency data using the depmap R package

**Theo Killian[1] and Laurent Gatto[1]**

[1]Computational Biology and Bioinformatics Unit, de Duve Institute, UCLouvain, Brussels, Belgium

## Abstract

The `depmap` package facilitates access in the R environment to the data from the Depmap project, a multi-year collaborative effort by the Broad Institute and Wellcome Sanger Institute, mapping genetic and chemical dependencies and other molecular biological measurements of over 1700 cancer cell lines. The `depmap` package formats this data for use of popular R data analysis and visualizing tools such as `dplyr` and `ggplot2`. In addition, the `depmap` package utilizes `ExperimentHub`, storing versions of the Depmap data accessible from the Cloud, which may be selectively downloaded, providing a reproducible research framework to support exploiting this data. This paper describes a workflow demonstrating how to access and visualize the Depmap data in R using this package.

## Keywords

cancer, cancer dependency, Depmap, ExperimentHub, data mining, reproducible research, Bioconductor

## Introduction

The consequences of genomic alterations of cancer cells on the molecular biological landscape of the cell may result in differential vulnerabilities, or "dependencies" compared to those of healthy cells. An example may be a gene not necessary for the survival in healthy cells, but essential for the vitality of particular cancer cell line. The exact nature of many of these dependencies in cancer cell lines is not completely understood [1]. A map illustrating the relationships between the genetic features of cancer and those of cancer dependencies is desirable. The Cancer Dependency Map or "Depmap", a collaborative initiative between the Broad Institute and the Wellcome Sanger Institute, aims to map such dependencies in a broad range cancer cell lines, intended to mirror the distribution of various cancer diseases in the general population, with the intention of exploiting this knowledge to develop new therapies in precision cancer medicine [2].

The Depmap initative is, as of the date of this publication, an ongoing project, with new data releases of select datasets every 90 days. As of the most current `20Q1` Depmap release, 1775 human cancer cell lines have been mapped for dependencies [2]. The primary method utilized in the Depmap project to map genomic dependencies is gene knockout performed by CRISPR [2, 3, 4, 5]. Genetic dependency is calculated from the observed log fold change in the amount of shRNA detected after gene knockout [6, 7]. To correct for potential off-target effects of gene knockout in overestimating dependency with CRISPR, the Depmap iniative utilized the CERES algorithm to moderate the final dependency estimation [3]. It should be noted that due to advancements in the CERES algorithm to account for seed effects, the RNAi dependency has been rendered redundant, and further data releases for this dependency measurement have been discontinued as of `19Q3` [2, 4]. In addition genomic dependency measurements of cancer cell lines, chemical dependencies were taken via Depmap PRISM viability screens that as of the `20Q1` release, tested 4,518 compounds against 578 cancer cell lines [8, 2]. The Depmap project has also compiled additional datasets detailing molecular biological characterization of cancer cell lines, such as genomic copy number, Reverse Phase Protein Array (RPPA) data, TPM gene expression data for protein coding genes and genomic mutation data. These datasets are updated quarterly on a release schedule and are publically available under CC BY 4.0 licence [2].

The `depmap` Bioconductor package was created in order to maximally exploit these rich datasets and to aide reproducible research, by importing the data into the R environment. The Depmap datasets were cleaned by converting all datasets to the long format, as well as adding the unique key `depmap_id` for all data tables, in order to make features more comparable, facilitating the use of common R packages such as `dplyr` [9] and `ggplot2` [10].

As new Depmap datasets are released on a quarterly basis, it is not feasible to include all dataset files in binary directly within the directory of the `depmap` R package. To keep the package lightweight, the `depmap` package utilizes and fully depends on the `ExperimentHub` package [11] to store and retrieve all versions of the Depmap data (starting from `19Q1` through `20Q1`) in the Cloud using AWS. The `depmap` package contains accessor functions to directly download and cache the most current datasets from the Cloud into the local R environment. Specific datasets, such as older datasets, which have been used in prior research can also be downloaded, if desired. This feature has the added advantage of enhancing reproducible research, such that specific versions of Depmap data can be selected, in addition to having access to the most current datasets. The `depmap` R package is available as part of Bioconductor at: `https://bioconductor.org/packages/depmap`.

## Use cases

The features of primary interest from the Depmap Project are the measurements of cancer dependency scores, found in datasets `crispr` and `rnai`, which illustrate genetic dependency and the dataset `drug_sensitivity`, which illustrates chemical dependency. In the case of genetic dependency, the dependency score is an expression of how vital a particular gene for a given cancer cell line is in terms of the lethality resulting from the knockout or knockdown of that gene. For example, a highly negative dependency score is derived from a large negative log fold change in the population of cancer cells after gene knockout or knockdown, implying that a given cell line is highly dependent on that gene. Genes that possess such highly negative dependency scores may be interesting targets for research in cancer medicine.

Below, we start by loading the packages need to run this workflow.

```
library("depmap")
library("ExperimentHub")
library("dplyr")
library("ggplot2")
library("stringr")
```

The `depmap` datasets are too large to be included into a typical package, therefore these data are stored in the Cloud. There are two ways to access the `depmap` datasets. The first such way calls on dedicated accessor functions that download, cache and load the latest available dataset into the R workspace. Examples for all available data are shown below:

```
rnai <- depmap_rnai()
crispr <- depmap_crispr()
copyNumber <- depmap_copyNumber()
TPM <- depmap_RPPA()
RPPA <- depmap_TPM()
metadata <- depmap_metadata()
mutationCalls <- depmap_mutationCalls()
drug_sensitivity <- depmap_drug_sensitivity()
```

Alternatively, specific dataset (from any available release) can be accessed through Bioconductor's `ExperimentHub`. The `ExperimentHub()` function creates an `ExperimentHub` object, which can be queried for specific terms. The list of datasets available that correspond to the query, `depmap` are shown below:

```
## create ExperimentHub query object
eh <- ExperimentHub()
query(eh, "depmap")
```

```
## ExperimentHub with 32 records
## # snapshotDate(): 2020-04-27
## # $dataprovider: Broad Institute
## # $species: Homo sapiens
## # $rdataclass: tibble
## # additional mcols(): taxonomyid, genome, description,
## #   coordinate_1_based, maintainer, rdatadateadded, preparerclass, tags,
## #   rdatapath, sourceurl, sourcetype
## # retrieve records with, e.g., ’object[["EH2260"]]’
##
##           title
##   EH2260 | rnai_19Q1
##   EH2261 | crispr_19Q1
##   EH2262 | copyNumber_19Q1
##   EH2263 | RPPA_19Q1
##   EH2264 | TPM_19Q1
##   ...      ...
##   EH3290 | crispr_20Q1
##   EH3291 | copyNumber_20Q1
##   EH3292 | TPM_20Q1
##   EH3293 | mutationCalls_20Q1
##   EH3294 | metadata_20Q1
```

Specific datasets are downloaded, cached and loaded into the workspace by selecting them by their unique EH numbers. Shown below, datasets of interest from the `19_Q3` release are downloaded using the unique EH numbers.

```
## download and cache required datasets
metadata <- eh[["EH3086"]]
crispr <- eh[["EH3081"]]
TPM <- eh[["EH3084"]]
mutationCalls <- eh[["EH3085"]]
copyNumber <- eh[["EH3082"]]
```

By importing the depmap data into the R environment, the data can be mined more effectively. For example, if one was interested in researching soft tissue sarcomas and wanted to search all such cancer cell lines for the gene with the greatest dependency, one could accomplish this task by using functions from the dplyr package. Below, the crispr dataset is selected for cell lines with *"SOFT_TISSUE"* in the CCLE name, and displaying a list of the highest dependency scores.

```
## list of dependency scores
crispr %>%
  dplyr::select(cell_line, gene_name, dependency) %>%
  dplyr::filter(stringr::str_detect(cell_line, "SOFT_TISSUE")) %>%
  dplyr::arrange(dependency)
```

```
## # A tibble: 586,656 x 3
##    cell_line          gene_name dependency
##    <chr>              <chr>          <dbl>
##  1 RH30_SOFT_TISSUE   RAN            -3.19
##  2 SCS214_SOFT_TISSUE RPL37          -2.85
##  3 RH30_SOFT_TISSUE   BUB3           -2.83
##  4 RH30_SOFT_TISSUE   C1orf109       -2.82
##  5 SCS214_SOFT_TISSUE POLR2J         -2.79
##  6 RH30_SOFT_TISSUE   PSMD7          -2.75
##  7 SCS214_SOFT_TISSUE SOD1           -2.73
##  8 RH30_SOFT_TISSUE   SS18L2         -2.69
##  9 SCS214_SOFT_TISSUE RNPC3          -2.69
## 10 RH30_SOFT_TISSUE   CHAF1B         -2.68
## # ... with 586,646 more rows
```

The gene RPL14 appears several times in the top dependencies scores, and may make an interesting candidate target. Figure 1 a plot of the crispr data is displayed as a histogram showing the distribution of dependency scores for gene RPL14. The red dotted line signifies the mean dependency score for that gene, while the blue dotted line signifies the global mean dependency score for all crispr measurements.

```
mean_crispr_dep <- crispr %>%
  dplyr::select(gene_name, dependency) %>%
  dplyr::filter(gene_name == "RPL14")

crispr %>%
  dplyr::select(gene, gene_name, dependency) %>%
  dplyr::filter(gene_name == "RPL14") %>%
  ggplot(aes(x = dependency)) + geom_histogram() +
  geom_vline(xintercept = mean(mean_crispr_dep$dependency, na.rm = TRUE),
             linetype = "dotted", color = "red") +
  geom_vline(xintercept = mean(crispr$dependency, na.rm = TRUE),
             linetype = "dotted", color = "blue")
```

A more complex plot of the crispr data, as shown below involves plotting the distribution of dependency scores for gene RPL14 for each major type of cancer, while highlighting the nature of mutations of this gene in such cancer cell lines (e.g. if such mutations are damaging, etc.). Notice that the plot above reflects the same overall distribution in two dimensions.

```
meta_crispr <- metadata %>%
  dplyr::select(depmap_id, lineage) %>%
  dplyr::full_join(crispr, by = "depmap_id") %>%
  dplyr::filter(gene_name == "RPL14") %>%
  dplyr::full_join((mutationCalls %>%
                      dplyr::select(depmap_id, entrez_id,
```

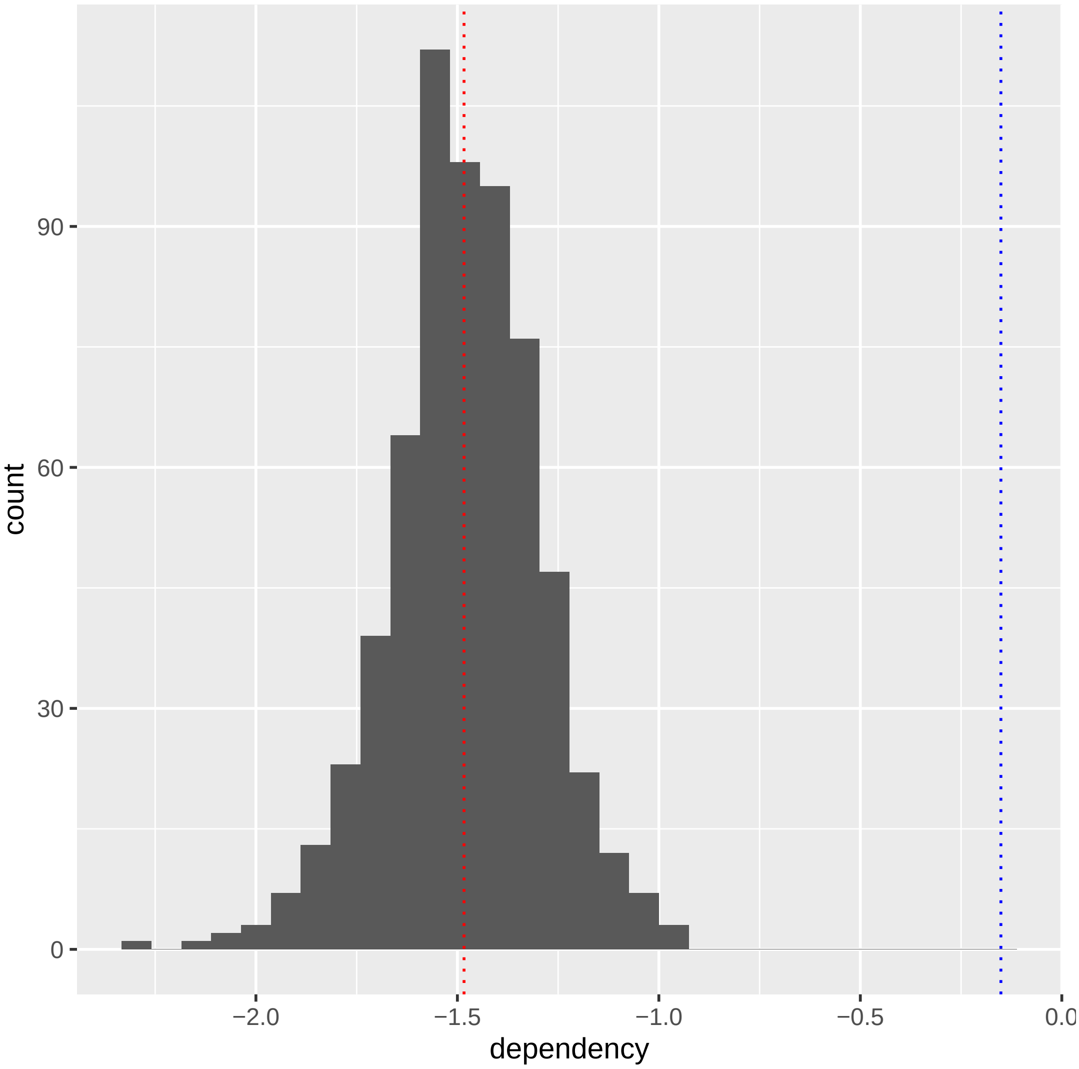

**Figure 1. Histogram of CRISPR dependency scores for gene RPL14.**

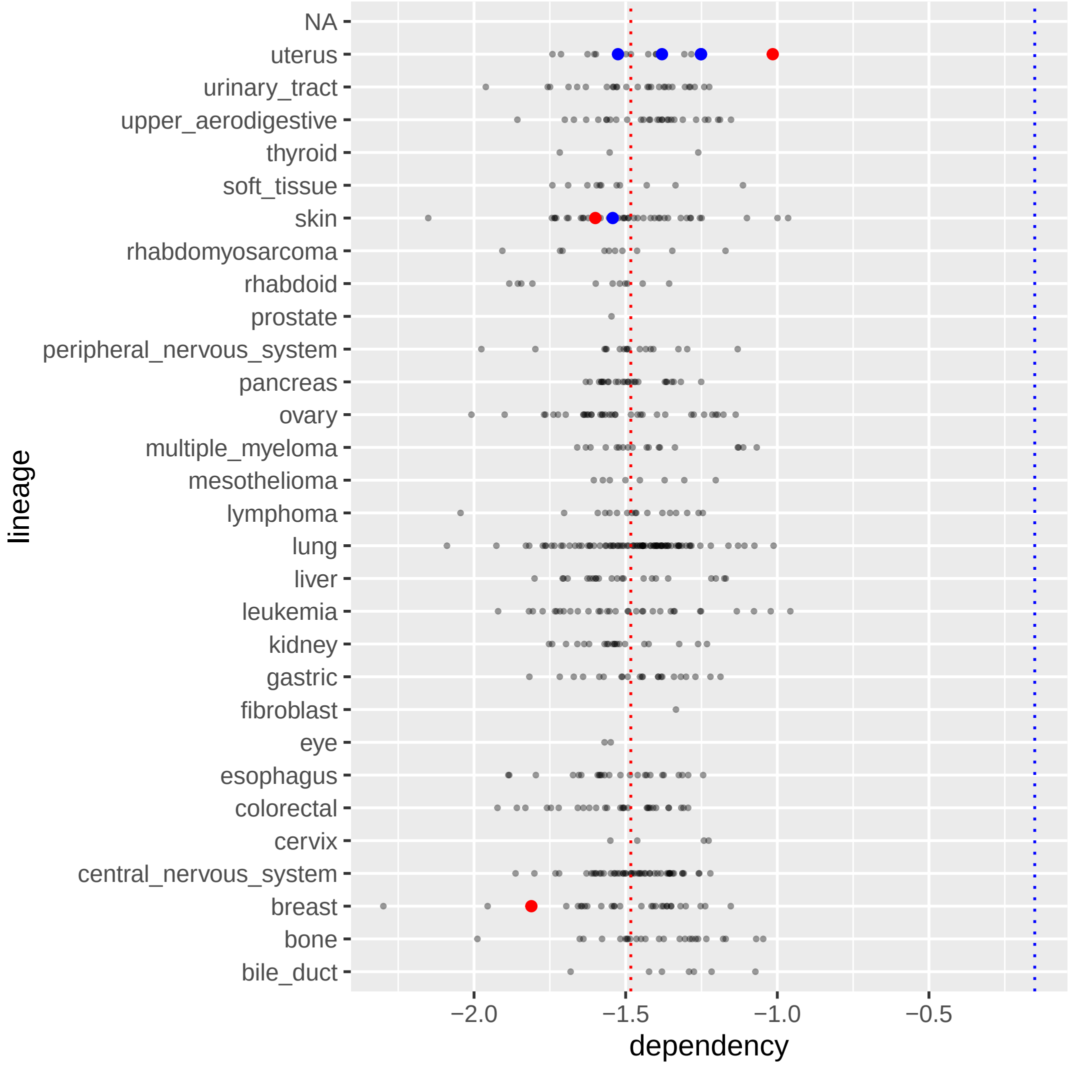

**Figure 2. Plot of CRISPR dependency scores for gene RPL14 by lineage.**

```
                                    is_cosmic_hotspot,
                                    var_annotation)),
                 by = c("depmap_id", "entrez_id"))

meta_crispr %>%
    ggplot(aes(x = dependency, y = lineage)) +
    geom_point(alpha = 0.4, size = 0.5) +
    geom_point(data = subset(meta_crispr,
                             var_annotation == "damaging"),
               color = "red") +
    geom_point(data = subset(meta_crispr,
                             var_annotation == "other non-conserving"),
               color = "blue") +
    geom_vline(xintercept = mean(meta_crispr$dependency, na.rm = TRUE),
               linetype = "dotted", color = "red") +
    geom_vline(xintercept = mean(crispr$dependency, na.rm = TRUE),
               linetype = "dotted", color = "blue")
```

Many cancer phenotypes are the result of changes in gene expression [12, 13, 14]. The extensive coverage of the `depmap` data affords visualization of genetic expression patterns across many major types of cancer. Figure 3 below shows a boxplot illustrating expression values for gene RPL14 by lineage:

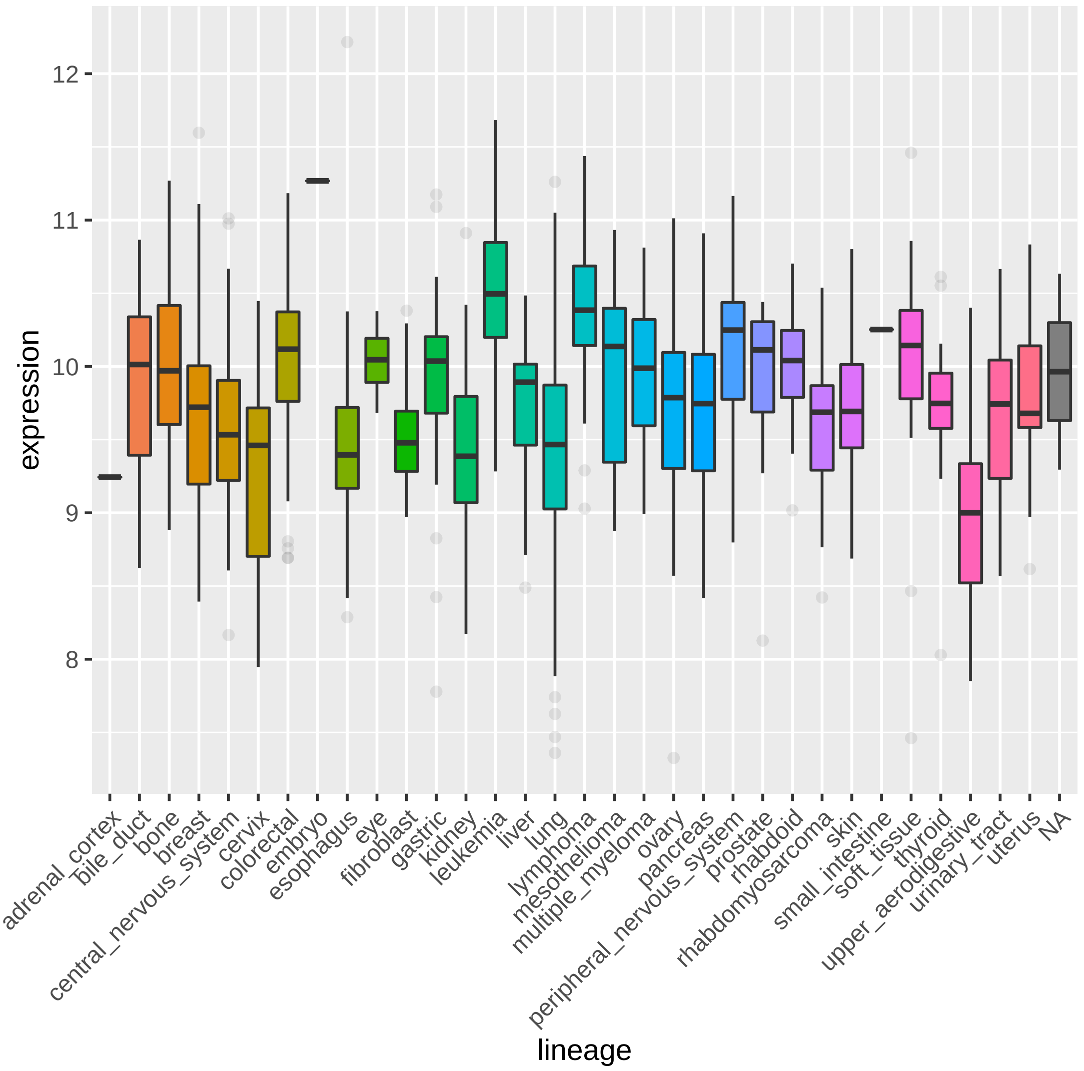

**Figure 3. Boxplot of TPM expression values for gene RPL14 by lineage.**

```
metadata %>%
  dplyr::select(depmap_id, lineage) %>%
  dplyr::full_join(TPM, by = "depmap_id") %>%
  dplyr::filter(gene_name == "RPL14") %>%
  ggplot(aes(x = lineage, y = expression, fill = lineage)) +
  geom_boxplot(outlier.alpha = 0.1) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  theme(legend.position = "none")
```

Differentially expressed genes and elevated genetic dependency in cancer cell lines have been observed [1, 7]. Therefore, such genes may present themselves as interesting research targets. Figure 4 shows a plot of expression versus CRISPR gene dependency for Rhabdomyosarcoma. The red vertical line represents the average expression for this form of cancer, while the horizontal line represents the average dependency for this cancer.

```
sarcoma <- metadata %>%
  dplyr::select(depmap_id, cell_line, primary_disease, subtype_disease) %>%
  dplyr::filter(primary_disease == "Sarcoma",
                subtype_disease == "Rhabdomyosarcoma")

crispr_sub <- crispr %>%
  dplyr::select(depmap_id, gene, gene_name, dependency)
```

```
tpm_sub <- TPM %>%
  dplyr::select(depmap_id, gene, gene_name, expression)

sarcoma_dep <- sarcoma %>%
  dplyr::left_join(crispr_sub, by = "depmap_id") %>%
  dplyr::select(-cell_line, -primary_disease,
                -subtype_disease, -gene_name)

sarcoma_exp <- sarcoma %>%
  dplyr::left_join(tpm_sub, by = "depmap_id")

sarcoma_dat_exp <- dplyr::full_join(sarcoma_dep, sarcoma_exp,
                                    by = c("depmap_id", "gene")) %>%
                   dplyr::filter(!is.na(expression))
```

```
ggplot(data = sarcoma_dat_exp, aes(x = dependency, y = expression)) +
  geom_point(alpha = 0.4, size = 0.5) +
  geom_vline(xintercept = mean(sarcoma_dat_exp$dependency, na.rm = TRUE),
             linetype = "dotted", color = "red") +
  geom_hline(yintercept = mean(sarcoma_dat_exp$expression, na.rm = TRUE),
             linetype = "dotted", color = "red") +
  ggtitle("Scatterplot of CRISPR dependency vs expression values for gene") +
  theme(axis.text.x = element_text(angle = 45))
```

Genes with the lowest depenency scores and highest TPM gene expression are found in the upper left section of the plot above. Such genes may present an example as interesting research targets. Below shows a table of such genes.

```
sarcoma_dat_exp %>%
  dplyr::select(cell_line, gene_name, dependency, expression) %>%
  dplyr::arrange(dependency, expression)
```

```
## # A tibble: 95,720 x 4
##    cell_line            gene_name dependency expression
##    <chr>                <chr>          <dbl>      <dbl>
##  1 SCMCRM2_SOFT_TISSUE RAN            -2.44       9.89
##  2 SCMCRM2_SOFT_TISSUE SNRPD1         -2.30       7.99
##  3 SCMCRM2_SOFT_TISSUE POLR2L         -2.26       6.09
##  4 SCMCRM2_SOFT_TISSUE PSMA3          -2.23       7.58
##  5 SCMCRM2_SOFT_TISSUE POLR2I         -2.20       6.51
##  6 SCMCRM2_SOFT_TISSUE ATP6V1B2       -2.19       5.44
##  7 SCMCRM2_SOFT_TISSUE CHAF1B         -2.17       2.22
##  8 SCMCRM2_SOFT_TISSUE HSPE1          -2.17       9.23
##  9 SCMCRM2_SOFT_TISSUE RRM2           -2.16       6.59
## 10 SCMCRM2_SOFT_TISSUE RPL12          -2.15      12.1
## # ... with 95,710 more rows
```

Changes in genomic copy number also play a role in some cancer phenotypes [3, 15, 16]. The `depmap` data allows the display of log genomic copy number for across many cancer lineages. Figure 5 shows such a plot for gene RPL14 for each major type of cancer lineage:

```
metadata %>%
  dplyr::select(depmap_id, lineage) %>%
  dplyr::full_join(copyNumber, by = "depmap_id") %>%
  dplyr::filter(gene_name == "RPL14") %>%
  ggplot(aes(x = lineage, y = log_copy_number, fill = lineage)) +
  geom_boxplot(outlier.alpha = 0.1) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  theme(legend.position = "none")
```

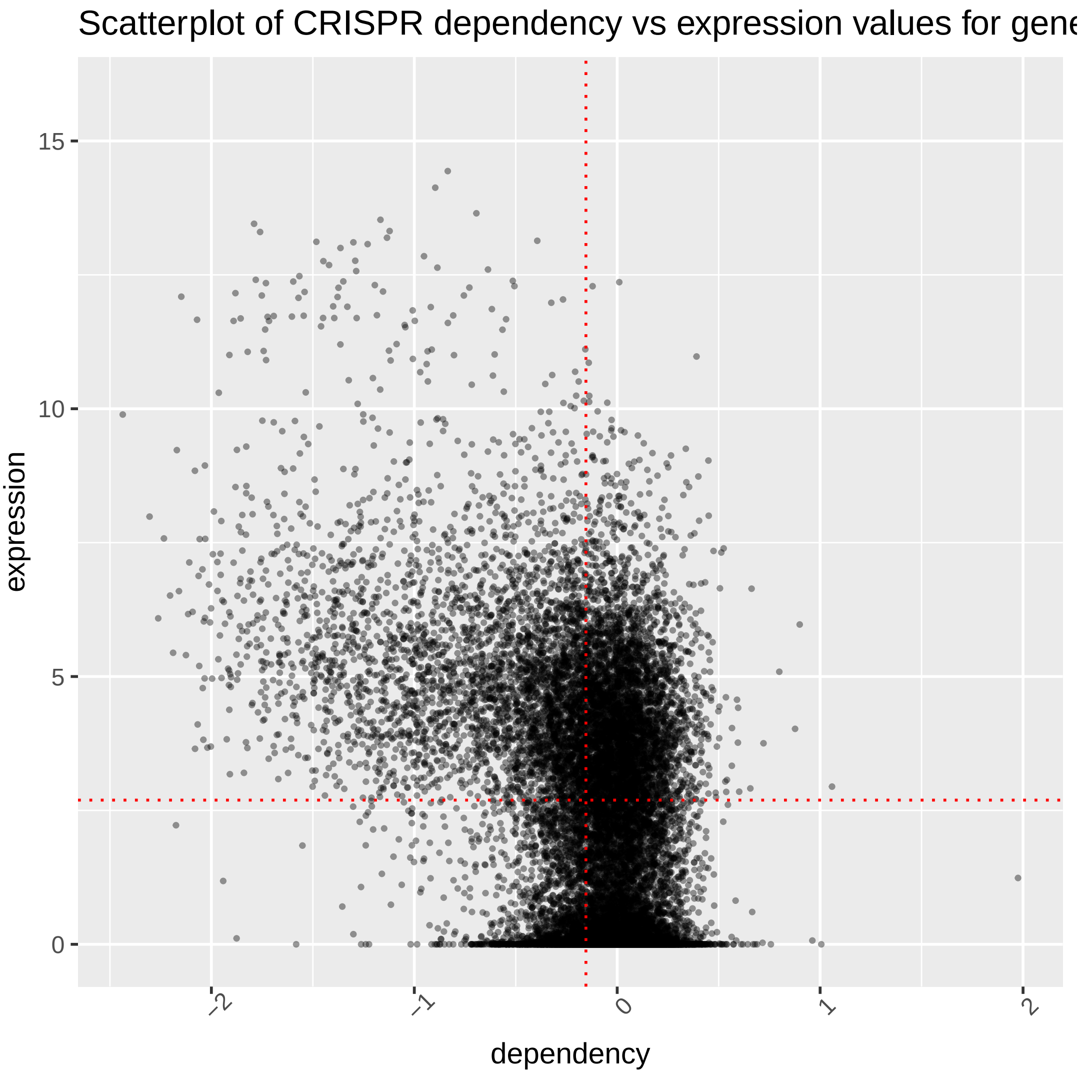

**Figure 4.** **Expression vs crispr gene dependency for Rhabdomyosarcoma.**

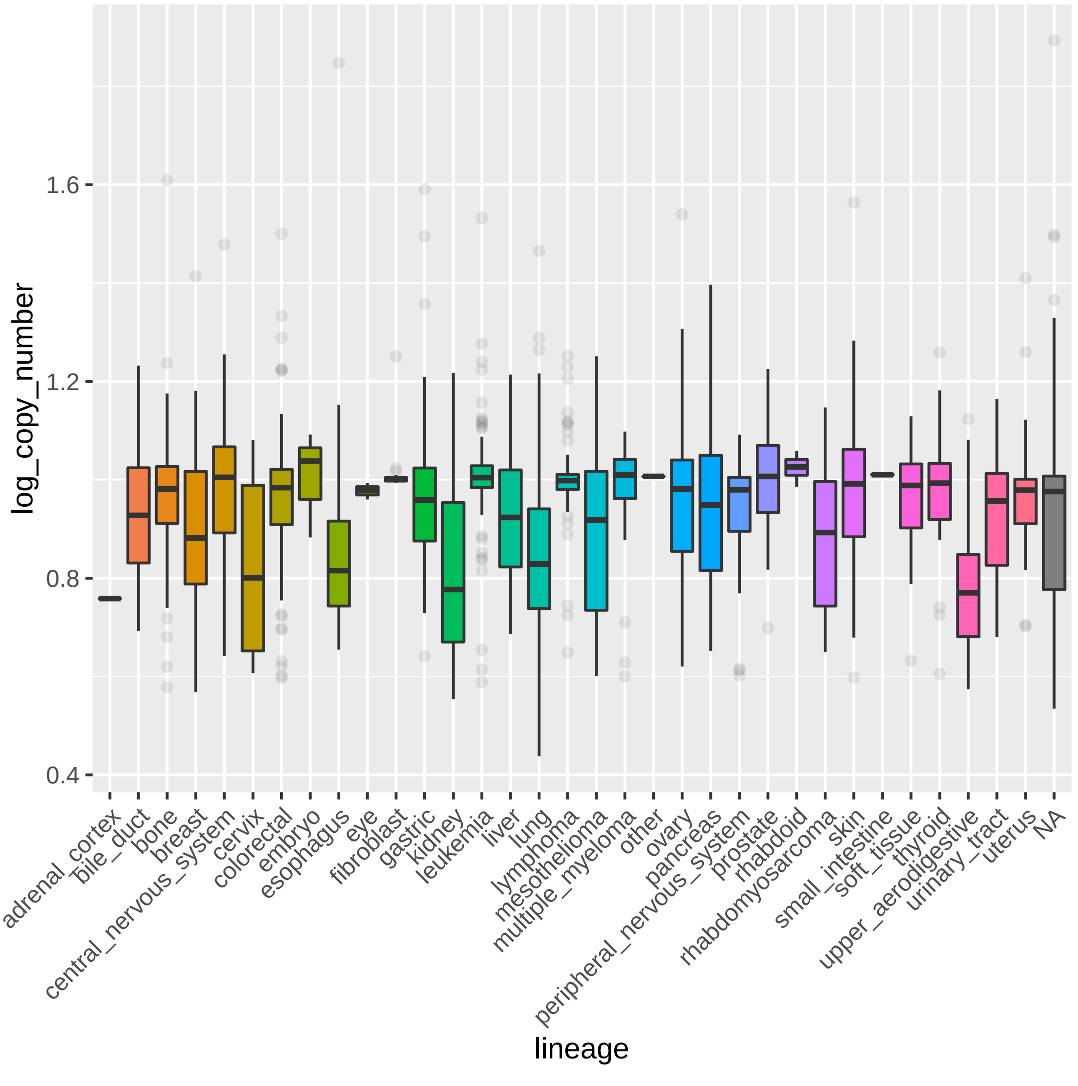

**Figure 5. Boxplot of log copy number for gene RPL14 by lineage**

## Discussion and outlook

We hope that this package will be used by cancer researchers to dig deeper into the Depmap data and to support their research. Additionally, we highly encourage future `depmap` users to combine `depmap` data with other datasets of interest, such as `TCGA` and `CCLE`.

The depmap R package will continue to be maintained in line with the biannual Bioconductor release, in addition to quarterly releases of Depmap data.

We welcome feedback and questions from the community. We also highly appreciate contributions to the code in the form of pull requests.

## Software availability

All packages used in this workflow are available from the Comprehensive R Archive Network (`https://cran.r-project.org`) or Bioconductor (`http://bioconductor.org`). The specific version numbers of R and the packages used are shown below.

To install the `depmap` package:

```
BiocManager::install('depmap')
```

```
## R version 4.0.0 (2020-04-24)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 18.04.4 LTS
## 
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/libf77blas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3
## 
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C              
##  [3] LC_TIME=fr_FR.UTF-8        LC_COLLATE=en_US.UTF-8    
##  [5] LC_MONETARY=fr_FR.UTF-8    LC_MESSAGES=en_US.UTF-8   
##  [7] LC_PAPER=fr_FR.UTF-8       LC_NAME=C                 
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C            
## [11] LC_MEASUREMENT=fr_FR.UTF-8 LC_IDENTIFICATION=C       
## 
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods  
## [8] base     
## 
## other attached packages:
## [1] stringr_1.4.0        ggplot2_3.3.0        ExperimentHub_1.14.0
## [4] AnnotationHub_2.20.0 BiocFileCache_1.12.0 dbplyr_1.4.3        
## [7] BiocGenerics_0.34.0  depmap_1.1.3         dplyr_0.8.5         
## 
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.4.6                   assertthat_0.2.1             
##  [3] digest_0.6.25                  utf8_1.1.4                   
##  [5] mime_0.9                       R6_2.4.1                     
##  [7] stats4_4.0.0                   RSQLite_2.2.0                
##  [9] evaluate_0.14                  httr_1.4.1                   
## [11] pillar_1.4.3                   rlang_0.4.5                  
## [13] curl_4.3                       rstudioapi_0.11              
## [15] blob_1.2.1                     S4Vectors_0.26.0             
## [17] rmarkdown_2.1                  labeling_0.3                 
## [19] bit_1.1-15.2                   munsell_0.5.0                
## [21] shiny_1.4.0.2                  compiler_4.0.0               
## [23] httpuv_1.5.2                   xfun_0.13                    
## [25] pkgconfig_2.0.3                htmltools_0.4.0              
## [27] tidyselect_1.0.0               tibble_3.0.1                 
## [29] interactiveDisplayBase_1.26.0 bookdown_0.18                
## [31] IRanges_2.22.1                 fansi_0.4.1                  
## [33] crayon_1.3.4                   withr_2.2.0                  
## [35] later_1.0.0                    rappdirs_0.3.1               
## [37] grid_4.0.0                   xtable_1.8-4
## [39] gtable_0.3.0                 lifecycle_0.2.0
## [41] DBI_1.1.0                    git2r_0.26.1
## [43] magrittr_1.5                 scales_1.1.0
## [45] BiocWorkflowTools_1.14.0     cli_2.0.2
## [47] stringi_1.4.6                farver_2.0.3
## [49] fs_1.4.1                     promises_1.1.0
## [51] ellipsis_0.3.0               vctrs_0.2.4
## [53] tools_4.0.0                  bit64_0.9-7
## [55] Biobase_2.48.0               glue_1.4.0
## [57] purrr_0.3.4                  BiocVersion_3.11.1
## [59] fastmap_1.0.1                yaml_2.2.1
## [61] AnnotationDbi_1.50.0         colorspace_1.4-1
## [63] BiocManager_1.30.10          memoise_1.1.0
## [65] knitr_1.28                   usethis_1.6.1
```

## Acknowledgements

## Competing interests

No competing interests were disclosed.

## Grant information

## References

[1] Aviad Tsherniak, Francisca Vazquez, Phil G Montgomery, Barbara A Weir, Gregory Kryukov, Glenn S Cowley, Stanley Gill, William F Harrington, Sasha Pantel, John M Krill-Burger, et al. Defining a cancer dependency map. *Cell*, 170(3): 564–576, 2017.

[2] Depmap Broad. Depmap achilles 20q1 public. *Broad Institute, Cambridge, MA*, 2020.

[3] Robin M Meyers, Jordan G Bryan, James M McFarland, Barbara A Weir, Ann E Sizemore, Han Xu, Neekesh V Dharia, Phillip G Montgomery, Glenn S Cowley, Sasha Pantel, et al. Computational correction of copy number effect improves specificity of crispr–cas9 essentiality screens in cancer cells. *Nature genetics*, 49(12):1779–1784, 2017.

[4] Joshua M Dempster, Jordan Rossen, Mariya Kazachkova, Joshua Pan, Guillaume Kugener, David E Root, and Aviad Tsherniak. Extracting biological insights from the project achilles genome-scale crispr screens in cancer cell lines. *BioRxiv*, page 720243, 2019.

[5] Joshua M Dempster, Clare Pacini, Sasha Pantel, Fiona M Behan, Thomas Green, John Krill-Burger, Charlotte M Beaver, Scott T Younger, Victor Zhivich, Hanna Najgebauer, et al. Agreement between two large pan-cancer crispr-cas9 gene dependency data sets. *Nature Communications*, 10(1):1–14, 2019.

[6] Glenn S Cowley, Barbara A Weir, Francisca Vazquez, Pablo Tamayo, Justine A Scott, Scott Rusin, Alexandra East-Seletsky, Levi D Ali, William FJ Gerath, Sarah E Pantel, et al. Parallel genome-scale loss of function screens in 216 cancer cell lines for the identification of context-specific genetic dependencies. *Scientific data*, 1:140035, 2014.

[7] James M McFarland, Zandra V Ho, Guillaume Kugener, Joshua M Dempster, Phillip G Montgomery, Jordan G Bryan, John M Krill-Burger, Thomas M Green, Francisca Vazquez, Jesse S Boehm, et al. Improved estimation of cancer dependencies from large-scale rnai screens using model-based normalization and data integration. *Nature communications*, 9(1):1–13, 2018.

[8] Steven M Corsello, Rohith T Nagari, Ryan D Spangler, Jordan Rossen, Mustafa Kocak, Jordan G Bryan, Ranad Humeidi, David Peck, Xiaoyun Wu, Andrew A Tang, et al. Non-oncology drugs are a source of previously unappreciated anti-cancer activity. *bioRxiv*, page 730119, 2019.

[9] Hadley Wickham and Maintainer Hadley Wickham. Package 'dplyr'. *Retreived from https://cran. rproject. org/web/packages/dplyr/dplyr. pdf*, 2020.

[10] Hadley Wickham. ggplot2. *Wiley Interdisciplinary Reviews: Computational Statistics*, 3(2):180–185, 2011.

[11] Martin Morgan and Lori Shepherd. *ExperimentHub: Client to access ExperimentHub resources*, 2020. R package version 1.14.0.

[12] Xianghua Li, Jasna Lalić, Pablo Baeza-Centurion, Riddhiman Dhar, and Ben Lehner. Changes in gene expression predictably shift and switch genetic interactions. *Nature communications*, 10(1):1–15, 2019.

[13] Enrique Hernández-Lemus, Helena Reyes-Gopar, Jesús Espinal-Enríquez, and Soledad Ochoa. The many faces of gene regulation in cancer: A computational oncogenomics outlook. *Genes*, 10(11):865, 2019.

[14] Sara J Felts, Xiaojia Tang, Benjamin Willett, Virginia P Van Keulen, Michael J Hansen, Krishna R Kalari, and Larry R Pease. Stochastic changes in gene expression promote chaotic dysregulation of homeostasis in clonal breast tumors. *Communications biology*, 2(1):1–7, 2019.

[15] Andrew J Aguirre, Robin M Meyers, Barbara A Weir, Francisca Vazquez, Cheng-Zhong Zhang, Uri Ben-David, April Cook, Gavin Ha, William F Harrington, Mihir B Doshi, et al. Genomic copy number dictates a gene-independent cell response to crispr/cas9 targeting. *Cancer discovery*, 6(8):914–929, 2016.

[16] Xin Shao, Ning Lv, Jie Liao, Jinbo Long, Rui Xue, Ni Ai, Donghang Xu, and Xiaohui Fan. Copy number variation is highly correlated with differential gene expression: a pan-cancer study. *BMC medical genetics*, 20(1):175, 2019.