The UCSC Xena platform provides an unprecedented resource for public omics data from big projects like The Cancer Genome Atlas (TCGA), however, it is hard  
for users to incorporate multiple datasets or data types, integrate the selected data with  
popular analysis tools or homebrewed code, and reproduce analysis procedures. To address this issue, we developed an R package UCSCXenaTools for enabling data retrieval, analysis integration and reproducible research for omics data from the UCSC Xena platform1.

In this technote we will outline how to use the UCSCXenaTools package to pull gene expression and clinical data from UCSC Xena for survival analysis.

**Installation**

UCSCXenaTools is available from CRAN:

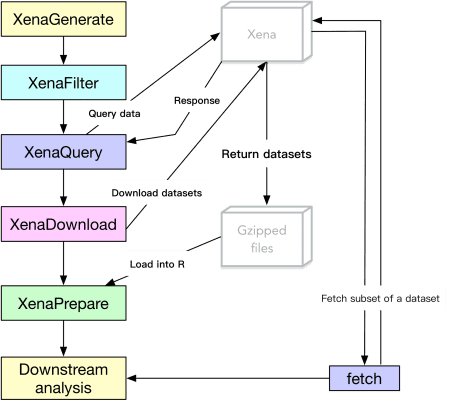
install.packages("UCSCXenaTools")

**How it works**

Before actually pulling data, understanding how UCSCXenaTools works (see Figure 1) will help users locate the most important function to use.

Generally,

* for operating datasets, we use functions whose names start with Xena
* for operating subset of a dataset, we use functions whose names start with fetch\_

*Figure 1. The UCSCXenaTools pipeline*

We will provide an example illustrating how to use UCSCXenaTools to study the effect of expression of the *KRAS* gene on prognosis of Lung Adenocarcinoma (LUAD) patients. *KRAS* is a known driver gene in LUAD. We retrieve expression data for the *KRAS* gene and survival status data for [LUAD patients from the TCGA](https://xenabrowser.net/datapages/?cohort=TCGA%20Lung%20Adenocarcinoma%20(LUAD)&removeHub=https%3A%2F%2Fxena.treehouse.gi.ucsc.edu%3A443) and use these as input to a survival analysis, frequently used in cancer research.

**Download data**

First we get information on all datasets in the TCGA LUAD cohort and store as luad\_cohort object.

suppressMessages(library(UCSCXenaTools))

suppressMessages(library(dplyr))

luad\_cohort = XenaData %>%

filter(XenaHostNames == "tcgaHub") %>% # select TCGA Hub

XenaScan("TCGA Lung Adenocarcinoma") # select LUAD cohort

luad\_cohort

#> # A tibble: 27 x 17

#> XenaHosts XenaHostNames XenaCohorts XenaDatasets SampleCount DataSubtype

#>

#> 1 https://… tcgaHub TCGA Lung … RABIT/separ… 467 Transcript…

#> 2 https://… tcgaHub TCGA Lung … RABIT/separ… 120 Transcript…

#> 3 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 151 DNA methyl…

#> 4 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 492 DNA methyl…

#> 5 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 516 copy numbe…

#> 6 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 543 somatic mu…

#> 7 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 237 protein ex…

#> 8 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 576 gene expre…

#> 9 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 60 miRNA matu…

#> 10 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 576 gene expre…

#> # … with 17 more rows, and 11 more variables: Label , Type ,

#> # AnatomicalOrigin , SampleType , Tags , ProbeMap ,

#> # LongTitle , Citation , Version , Unit ,

#> # Platform

**Download clinical dataset**

Now we download the clinical dataset of the TCGA LUAD cohort and load it into R.

cli\_query = luad\_cohort %>%

filter(DataSubtype == "phenotype") %>% # select clinical dataset

XenaGenerate() %>% # generate a XenaHub object

XenaQuery() %>%

XenaDownload()

#> This will check url status, please be patient.

#> All downloaded files will under directory /var/folders/mx/rfkl27z90c96wbmn3\_kjk8c80000gn/T//Rtmp2ihvVq.

#> The 'trans\_slash' option is FALSE, keep same directory structure as Xena.

#> Creating directories for datasets...

#> Downloading TCGA.LUAD.sampleMap/LUAD\_clinicalMatrix.gz

cli = XenaPrepare(cli\_query)

# See a few rows

head(cli)

#> # A tibble: 6 x 157

#> sampleID ABSOLUTE\_Ploidy ABSOLUTE\_Purity AKT1 ALK\_translocati… BRAF

#>

#> 1 TCGA-05… NA NA

#> 2 TCGA-05… 3.77 0.46 none p.A7…

#> 3 TCGA-05… NA NA

#> 4 TCGA-05… NA NA none p.L6…

#> 5 TCGA-05… 2.04 0.48 none none

#> 6 TCGA-05… 3.29 0.48 none p.G4…

#> # … with 151 more variables: CBL , CTNNB1 ,

#> # Canonical\_mut\_in\_KRAS\_EGFR\_ALK ,

#> # Cnncl\_mt\_n\_KRAS\_EGFR\_ALK\_RET\_ROS1\_BRAF\_ERBB2\_HRAS\_NRAS\_AKT1\_MAP2 ,

#> # EGFR , ERBB2 , ERBB4 ,

#> # Estimated\_allele\_fraction\_of\_a\_clonal\_varnt\_prsnt\_t\_1\_cpy\_pr\_cll ,

#> # Expression\_Subtype , HRAS , KRAS , MAP2K1 ,

#> # MET , NRAS , PIK3CA , PTPN11 , Pathology ,

#> # Pathology\_Updated , RET\_translocation ,

#> # ROS1\_translocation , STK11 ,

#> # WGS\_as\_of\_20120731\_0\_no\_1\_yes , `\_EVENT` ,

#> # `\_INTEGRATION` , OS.time , OS , OS.unit ,

#> # `\_PANCAN\_CNA\_PANCAN\_K8` , `\_PANCAN\_Cluster\_Cluster\_PANCAN` ,

#> # `\_PANCAN\_DNAMethyl\_LUAD` , `\_PANCAN\_DNAMethyl\_PANCAN` ,

#> # `\_PANCAN\_RPPA\_PANCAN\_K8` , `\_PANCAN\_UNC\_RNAseq\_PANCAN\_K16` ,

#> # `\_PANCAN\_miRNA\_PANCAN` , `\_PANCAN\_mirna\_LUAD` ,

#> # `\_PANCAN\_mutation\_PANCAN` , `\_PATIENT` , RFS.time ,

#> # RFS , RFS.unit , `\_TIME\_TO\_EVENT` ,

#> # `\_TIME\_TO\_EVENT\_UNIT` , `\_cohort` ,

#> # `\_primary\_disease` , `\_primary\_site` ,

#> # additional\_pharmaceutical\_therapy ,

#> # additional\_radiation\_therapy ,

#> # additional\_surgery\_locoregional\_procedure ,

#> # additional\_surgery\_metastatic\_procedure ,

#> # age\_at\_initial\_pathologic\_diagnosis ,

#> # anatomic\_neoplasm\_subdivision ,

#> # anatomic\_neoplasm\_subdivision\_other , bcr\_followup\_barcode ,

#> # bcr\_patient\_barcode , bcr\_sample\_barcode ,

#> # days\_to\_additional\_surgery\_locoregional\_procedure ,

#> # days\_to\_additional\_surgery\_metastatic\_procedure ,

#> # days\_to\_birth , days\_to\_collection , days\_to\_death ,

#> # days\_to\_initial\_pathologic\_diagnosis ,

#> # days\_to\_last\_followup ,

#> # days\_to\_new\_tumor\_event\_after\_initial\_treatment ,

#> # disease\_code , dlco\_predictive\_percent ,

#> # eastern\_cancer\_oncology\_group , egfr\_mutation\_performed ,

#> # egfr\_mutation\_result , eml4\_alk\_translocation\_method ,

#> # eml4\_alk\_translocation\_performed ,

#> # followup\_case\_report\_form\_submission\_reason ,

#> # followup\_treatment\_success , form\_completion\_date ,

#> # gender , histological\_type ,

#> # history\_of\_neoadjuvant\_treatment , icd\_10 ,

#> # icd\_o\_3\_histology , icd\_o\_3\_site ,

#> # informed\_consent\_verified , initial\_weight ,

#> # intermediate\_dimension , is\_ffpe ,

#> # karnofsky\_performance\_score , kras\_gene\_analysis\_performed ,

#> # kras\_mutation\_found , kras\_mutation\_result ,

#> # location\_in\_lung\_parenchyma , longest\_dimension ,

#> # lost\_follow\_up , new\_neoplasm\_event\_type ,

#> # new\_tumor\_event\_after\_initial\_treatment ,

#> # number\_pack\_years\_smoked , oct\_embedded , other\_dx ,

#> # pathologic\_M , pathologic\_N , pathologic\_T ,

#> # pathologic\_stage , pathology\_report\_file\_name , …

**Download *KRAS* gene expression**

To download gene expression data, first we need to select the right dataset.

ge = luad\_cohort %>%

filter(DataSubtype == "gene expression RNAseq", Label == "IlluminaHiSeq")

ge

#> # A tibble: 1 x 17

#> XenaHosts XenaHostNames XenaCohorts XenaDatasets SampleCount DataSubtype

#>

#> 1 https://… tcgaHub TCGA Lung … TCGA.LUAD.s… 576 gene expre…

#> # … with 11 more variables: Label , Type ,

#> # AnatomicalOrigin , SampleType , Tags , ProbeMap ,

#> # LongTitle , Citation , Version , Unit ,

#> # Platform

Now we fetch *KRAS* gene expression values.

# You can pass gene symbols to 'identifiers' option

# to obtain their values in a dataset.

# A matrix will be returned by 'fetch\_dense\_values' function

# with rows corresponding to genes,

# so here we extract the first row.

KRAS = fetch\_dense\_values(host = ge$XenaHosts,

dataset = ge$XenaDatasets,

identifiers = "KRAS",

use\_probeMap = TRUE) %>%

.[1, ]

#> -> Checking identifiers...

#> -> use\_probeMap is TRUE, skipping checking identifiers...

#> -> Done.

#> -> Checking samples...

#> -> Done.

#> -> Checking if the dataset has probeMap...

#> -> Done. ProbeMap is found.

head(KRAS)

#> TCGA-69-7978-01 TCGA-62-8399-01 TCGA-78-7539-01 TCGA-50-5931-11

#> 10.25 10.29 10.82 10.29

#> TCGA-73-4658-01 TCGA-44-6775-01

#> 10.36 10.03

**Merge expression data and survival status**

Next, we join the two data.frame by sampleID and keep necessary columns. Here we focus on ‘Primary Tumor’ for simplicity.

merged\_data = tibble(sampleID = names(KRAS),

KRAS\_expression = as.numeric(KRAS)) %>%

left\_join(cli, by = "sampleID") %>%

filter(sample\_type == "Primary Tumor") %>% # Keep only 'Primary Tumor'

select(sampleID, KRAS\_expression, OS.time, OS) %>%

rename(time = OS.time,

status = OS)

head(merged\_data)

#> # A tibble: 6 x 4

#> sampleID KRAS\_expression time status

#>

#> 1 TCGA-69-7978-01 10.2 134 0

#> 2 TCGA-62-8399-01 10.3 2696 0

#> 3 TCGA-78-7539-01 10.8 791 0

#> 4 TCGA-73-4658-01 10.4 1600 1

#> 5 TCGA-44-6775-01 10.0 705 0

#> 6 TCGA-44-2655-01 9.75 1324 0

**Survival analysis**

To study the effect of *KRAS* gene expression on prognosis of LUAD patients, we show two approaches:

1. use Cox model to determine the effect when *KRAS* gene expression increases
2. use Kaplan-Meier curve and log-rank test to observe the difference in different of*KRAS* gene expression status, i.e. high or low

We will use package **survival** and **survminer** to create models and plot survival curves, respectively.

library(survival)

library(survminer)

#> Loading required package: ggplot2

#> Loading required package: ggpubr

#> Loading required package: magrittr

**Cox model**

fit = coxph(Surv(time, status) ~ KRAS\_expression, data = merged\_data)

fit

#> Call:

#> coxph(formula = Surv(time, status) ~ KRAS\_expression, data = merged\_data)

#>

#> coef exp(coef) se(coef) z p

#> KRAS\_expression 0.2927 1.3400 0.1020 2.871 0.0041

#>

#> Likelihood ratio test=7.67 on 1 df, p=0.005604

#> n= 502, number of events= 183

#> (12 observations deleted due to missingness)

We can find that patients with higher *KRAS* gene expression have higher risk (34% increase per *KRAS* gene expression unit increase), and the effect of *KRAS* gene expression is statistically significant (*p*<0.05).

**Risk between expression groups**

We can also divide patients into two groups using KRAS median as a cutoff.

merged\_data = merged\_data %>%

mutate(group = case\_when(

KRAS\_expression > quantile(KRAS\_expression, 0.5) ~ 'KRAS\_High',

KRAS\_expression < quantile(KRAS\_expression, 0.5) ~ 'KRAS\_Low',

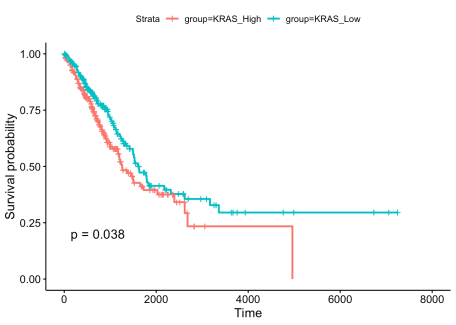
TRUE ~ NA\_character\_

))

fit = survfit(Surv(time, status) ~ group, data = merged\_data)

Then we can plot the survival curves for each group.

ggsurvplot(fit, pval = TRUE)

*Figure 2. Kaplan-Meier curve. Survival probability vs Time (days)*

The Kaplan-Meier plot shows what percent of patients are alive at a time point. We can clearly see that patients in ‘KRAS\_Low’ group have better survival than patients in ‘KRAS\_High’ group because the survival probability of ‘KRAS\_High’ group is always lower than ‘KRAS\_Low’ group over time (the unit is ‘day’ here). The difference between the two groups is statistically significant (*p*<0.05 by log-rank test).