Convert Cell-cylce Marker Genes Between Species

Lijiao NING

2025-05-22

Table of contents

In scRNA-seq data analysis, we often include two steps:

- Cell cycle scoring
- Then correct it as a potential batch effect or source of variation (regress out)

However, in {`Seurat`}, the `CellCycleScoring()` function uses built-in human marker genes from Tirosh et al., 2016.

When we analyze mice, zebrafish, or other model organisms, we cannot use these human genes directly. This tutorial will show you several ways to retrieve orthologs for a list of genes.

Via brute force

If you want to convert human genes to mouse genes, you can try:

```r
stringr::str_to_title(c("MCM5", "PCNA", "TYMS"))
```

```
[1] "Mcm5" "Pcna" "Tyms"
```

⚠ But it's risky:

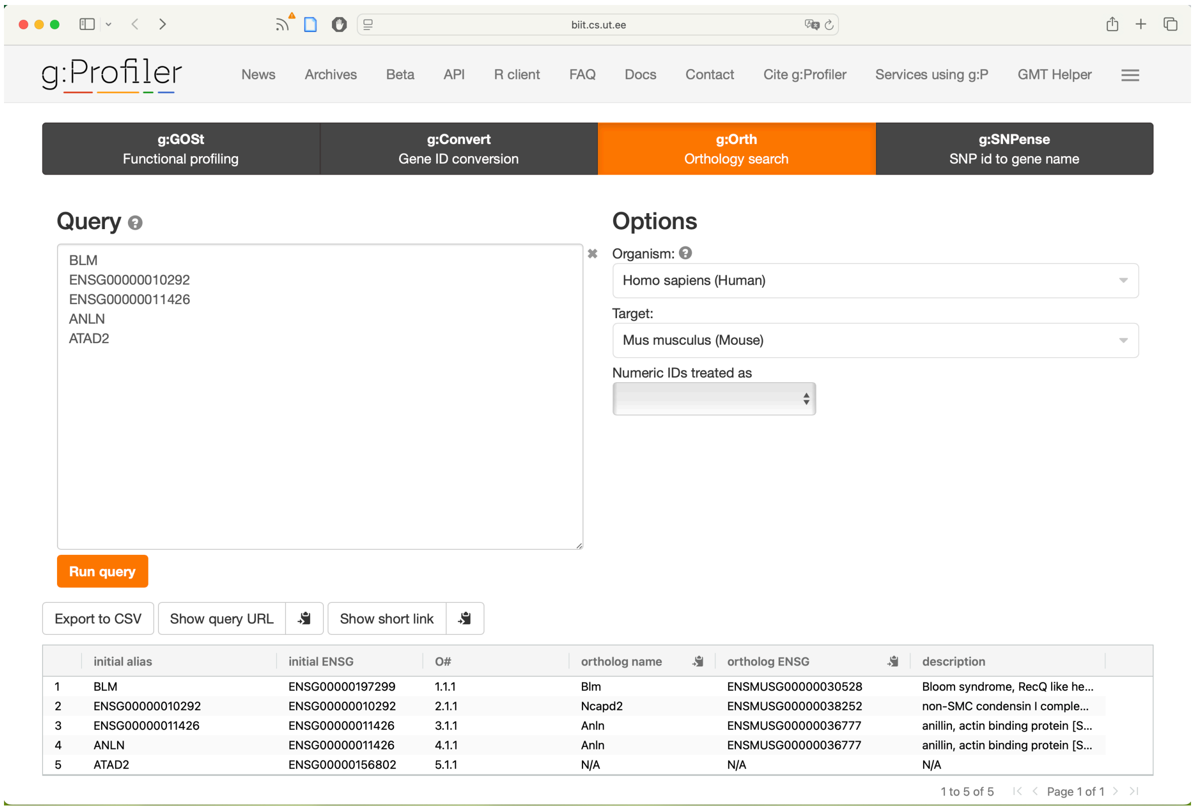
- It can create gene symbols that don't exist.
- There's no guarantee that "Pcna" is the mouse ortholog of "PCNA" (even if it often is).

Mouse and human gene symbols often follow different case conventions, but don't assume changing the case gives you the ortholog. Always use a trusted resource like **g:Profiler**, **Ensembl BioMart** to get verified ortholog mappings.

Via "g:Profiler"

Web Interface

1. Go to https://biit.cs.ut.ee/gprofiler/gorth, or click on the **g:Orth** tab on the g:Profiler homepage.
2. Paste your list of genes (one per line, can include both symbols and Ensembl IDs) into the **Query** bloc.
3. Set **Options**: select *Input organism* and *Target organism*. Then click **Run query**.
4. Click "Export to CSV" to save results. Then you can import the data for cell cycle estimation.



Query

```
BLM
ENSG00000010292
ENSG00000011426
ANLN
ATAD2
```

Options

Organism: Homo sapiens (Human)

Target: Mus musculus (Mouse)

Numeric IDs treated as

	initial alias	initial ENSG	O#	ortholog name	ortholog ENSG	description
1	BLM	ENSG00000197299	1.1.1	Blm	ENSMUSG00000030528	Bloom syndrome, RecQ like he...
2	ENSG00000010292	ENSG00000010292	2.1.1	Ncapd2	ENSMUSG00000038252	non-SMC condensin I comple...
3	ENSG00000011426	ENSG00000011426	3.1.1	Anln	ENSMUSG00000036777	anillin, actin binding protein [S...
4	ANLN	ENSG00000011426	4.1.1	Anln	ENSMUSG00000036777	anillin, actin binding protein [S...
5	ATAD2	ENSG00000156802	5.1.1	N/A	N/A	N/A

1 to 5 of 5 Page 1 of 1

No Ortholog Found?
It's completely normal that sometimes there's no ortholog match between species.

- Biological reasons:
  - Some genes are species-specific;
  - The gene might have lost its ortholog in the other species due to evolution;
  - There may be functional divergence, where the ortholog exists but has changed too much to be confidently recognized.
- Technical reasons:
  - The gene symbol or ID may be outdated, misspelled, or not annotated in the reference genome;
  - The database does not contain some genes or it's outdated.

R package {gprofiler2}

You need to install the {gprofiler2} package before.

Here is the vignette. We will use the `gorth()` function:

```r
library(gprofiler2)
suppressPackageStartupMessages(library(Seurat))

# Orthology search
mmus_s <- gorth(
  cc.genes.updated.2019$s.genes,
  source_organism = "hsapiens",
  target_organism = "mmusculus"
)$ortholog_name
mmus_s
```

```
 [1] "Mcm5"     "Pcna"     "Tyms"     "Fen1"     "Mcm7"     "Mcm4"
 [7] "Rrm1"     "Ung"      "Gins2"    "Mcm6"     "Cdca7"    "Dtl"
[13] "Prim1"    "Uhrf1"    "Cenpu"    "Hells"    "Rfc2"     "Polr1b"
[19] "Nasp"     "Rad51ap1" "Gmnn"     "Wdr76"    "Slbp"     "Ccne2"
[25] "Ubr7"     "Msh2"     "Rad51"    "Rrm2"     "Cdc45"    "Cdc6"
[31] "Exo1"     "Tipin"    "Dscc1"    "Blm"      "Casp8ap2" "Usp1"
[37] "Clspn"    "Pola1"    "Chaf1b"   "Mrpl36"   "E2f8"
```

```r
length(cc.genes.updated.2019$s.genes)
```

```
[1] 43
```

```r
length(mmus_s)
```

```
[1] 41
```

```r
mmus_g2m <- gorth(
  cc.genes.updated.2019$g2m.genes,
  source_organism = "hsapiens",
  target_organism = "mmusculus"
)$ortholog_name
mmus_g2m
```

```
 [1] "Hmgb2"   "Cdk1"    "Nusap1"  "Ube2c"   "Birc5"   "Tpx2"    "Top2a"
 [8] "Ndc80"   "Cks2"    "Nuf2"    "Cks1b"   "Mki67"   "Tmpo"    "Cenpf"
[15] "Tacc3"   "Pimreg"  "Smc4"    "Ccnb2"   "Ckap2l"  "Ckap2"   "Aurkb"
```

```
[22] "Bub1"    "Kif11"   "Anp32e"  "Tubb4b"  "Gtse1"   "Kif20b"  "Hjurp"
[29] "Cdca3"   "Jpt1"    "Cdc20"   "Ttk"     "Cdc25c"  "Kif2c"   "Rangap1"
[36] "Ncapd2"  "Dlgap5"  "Cdca2"   "Cdca8"   "Ect2"    "Kif23"   "Hmmr"
[43] "Aurka"   "Psrc1"   "Anln"    "Lbr"     "Ckap5"   "Cenpe"   "Ctcf"
[50] "Nek2"    "G2e3"    "Gas2l3"  "Cbx5"    "Cenpa"
```

```r
length(cc.genes.updated.2019$g2m.genes)
```

```
[1] 54
```

```r
length(mmus_g2m)
```

```
[1] 54
```

Organism names are constructed by concatenating the first letter of the name and the family name. Example: human - 'hsapiens', mouse - 'mmusculus'.

```r
# run cell cylce scoring
seurat_obj <- CellCycleScoring(
  seurat_obj,
  s.features = mmus_s,
  g2m.features = mmus_g2m,
  set.ident = TRUE
)
```

- If your gene list is long (hundreds+), consider using programmatic tools like R or Python for automation.
- g:Profiler uses data from Ensembl and Ensembl Genomes, it follows update of Ensembl databases.
- Ensembl updates gene annotations frequently, check the database version if you're using older datasets.

Via "Ensembl"

Web Interface

1. Open the BioMart interface https://www.ensembl.org/biomart/martview or click on the **BioMart** tab on the Ensembl homepage.

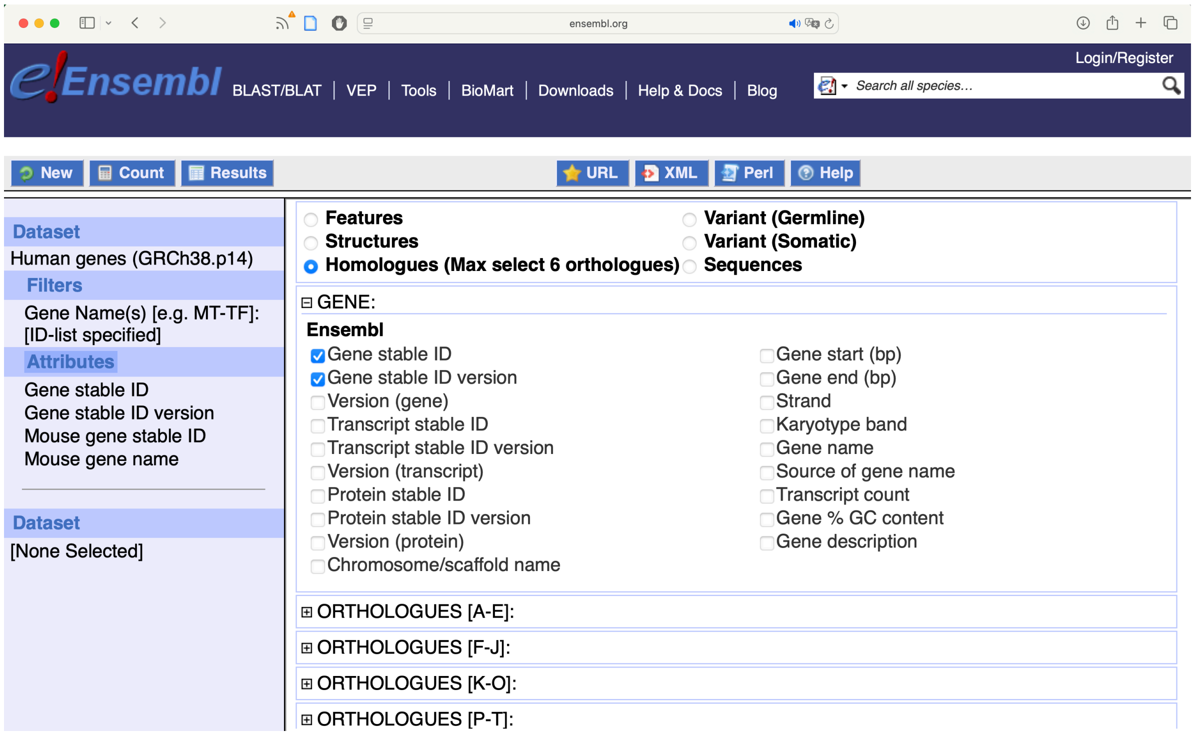
2. Choose **Database** (select "Ensembl Genes xxx") and **Dataset** (select the species of input genes).

3. Add **Filters** (Input your gene list)

- In the left menu, click on **Filters**
- Expand the **GENE** section
- Check *Input external references ID list*, find and tick the appropriate gene ID type, *e.g.*: if you use Ensembl ID -> `Gene stable ID(s)`, if you use gene symbol -> `Gene name(s)`.
- Paste your list of gene (one per line, can only include **the same ID type**).

4. Choose **Attributes** (What you want in the output)

- Click on **Attributes** in the left menu
- Select **Homologues**
- Expand the **GENE** section and tick the information you want of the input species, *e.g.*: `Gene stable ID`, `Gene name`, *etc.*
- Select the target species in the **ORTHOLOGUES** sections, and tick the information you want about the target species, *e.g.*: `Mouse gene stable ID`, `Mouse gene name`, *etc.*



Dataset
Human genes (GRCh38.p14)
Filters
Gene Name(s) [e.g. MT-TF]: [ID-list specified]
Attributes
Gene stable ID
Gene stable ID version
Mouse gene stable ID
Mouse gene name

Dataset
[None Selected]

○ Features ○ Variant (Germline)
○ Structures ○ Variant (Somatic)
[✓ Homologues (Max select 6 orthologues)] ○ Sequences

⊟ GENE:

Ensembl

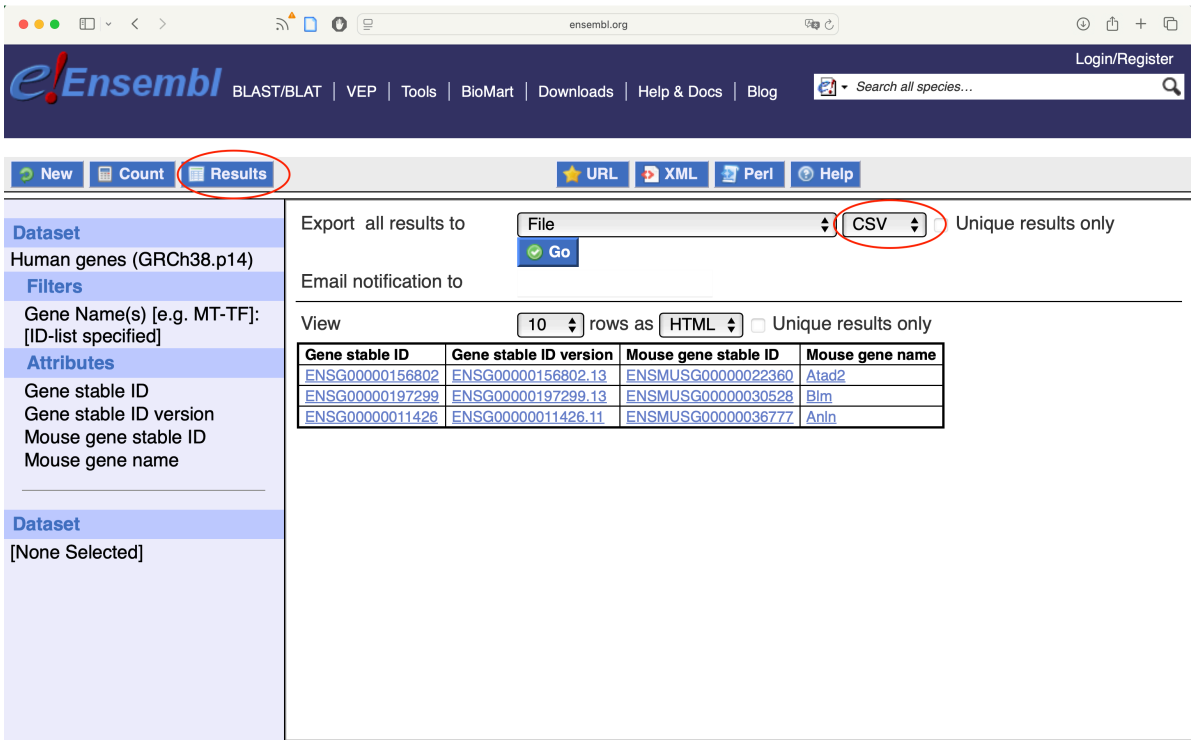
[✓ Gene stable ID]	☐ Gene start (bp)
[✓ Gene stable ID version]	☐ Gene end (bp)
☐ Version (gene)	☐ Strand
☐ Transcript stable ID	☐ Karyotype band
☐ Transcript stable ID version	☐ Gene name
☐ Version (transcript)	☐ Source of gene name
☐ Protein stable ID	☐ Transcript count
☐ Protein stable ID version	☐ Gene % GC content
☐ Version (protein)	☐ Gene description
☐ Chromosome/scaffold name	

⊞ ORTHOLOGUES [A-E]:
⊞ ORTHOLOGUES [F-J]:
⊞ ORTHOLOGUES [K-O]:
⊞ ORTHOLOGUES [P-T]:

5. Get the Results

- Click the **Results** button at the top

- Preview your results
- Select the output format and click **Go** to download the results.



Dataset
Human genes (GRCh38.p14)
Filters
Gene Name(s) [e.g. MT-TF]: [ID-list specified]
Attributes
Gene stable ID
Gene stable ID version
Mouse gene stable ID
Mouse gene name

Dataset
[None Selected]

Export all results to File CSV Unique results only

Email notification to

View 10 rows as HTML Unique results only

Gene stable ID	Gene stable ID version	Mouse gene stable ID	Mouse gene name
ENSG00000156802	ENSG00000156802.13	ENSMUSG00000022360	Atad2
ENSG00000197299	ENSG00000197299.13	ENSMUSG00000030528	Blm
ENSG00000011426	ENSG00000011426.11	ENSMUSG00000036777	Anln

R package {biomaRt}

You need to install the biomaRt before.

Principal steps: species1_symbol -> species1_ensembl_id -> getHomologs() -> species2_ensembl_id -> specie2_symbol -> CellCycleScoring()

1. Load required libraries

```
suppressPackageStartupMessages(library(dplyr))
# suppressPackageStartupMessages(library(Seurat)) # already loaded before
suppressPackageStartupMessages(library(biomaRt)) # need a version which contains the `getHom
```

2. Retrieve human cell cycle markers

```
str(cc.genes.updated.2019) # built-in human cc markers
```

```
List of 2
 $ s.genes  : chr [1:43] "MCM5" "PCNA" "TYMS" "FEN1" ...
 $ g2m.genes: chr [1:54] "HMGB2" "CDK1" "NUSAP1" "UBE2C" ...
```

```r
# built a tibble for later use
cc_genes <- tibble(
  phase = unlist(mapply(rep, c("s", "g2m"), lapply(cc.genes.updated.2019, length))),
  gene_name = unname(unlist(cc.genes.updated.2019))
)
cc_genes
```

```
# A tibble: 97 x 2
   phase gene_name
   <chr> <chr>
 1 s     MCM5
 2 s     PCNA
 3 s     TYMS
 4 s     FEN1
 5 s     MCM7
 6 s     MCM4
 7 s     RRM1
 8 s     UNG
 9 s     GINS2
10 s     MCM6
# i 87 more rows
```

Then we will use {`biomaRt`} to turn these genes into homologous genes of the target species, here we will use mouse as an example.

3. Set up marts

```r
human <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")
mouse <- useMart("ensembl", dataset = "mmusculus_gene_ensembl")
```

4. Get homologous genes

```r
s_mouse <- getLDS(
  attributes = c("hgnc_symbol"),
  filters = "hgnc_symbol",
  values = cc.genes.updated.2019$s.genes,
  mart = human,
  attributesL = c("hgnc_symbol"),
  martL = mouse,
  uniqueRows = TRUE
)
```

```
Error in `httr2::req_perform()` at biomaRt/R/utilityFunctions.R:215:5:
! HTTP 500 Internal Server Error.
```

```r
g2m_mouse <- getLDS(
  attributes = c("hgnc_symbol"),
  filters = "hgnc_symbol",
  values = cc.genes.updated.2019$g2m.genes,
  mart = human,
  attributesL = c("hgnc_symbol"),
  martL = mouse,
  uniqueRows = TRUE
)
```

```
Error in `httr2::req_perform()` at biomaRt/R/utilityFunctions.R:215:5:
! HTTP 500 Internal Server Error.
```

The `getLDS()` funciontallity started failing with the release of BioMart 106, a new function `getHomologs()` can be help. (See discussion here)

4.1 `getHomologs()` requires Ensembl gene ID as input, so we need to convert the human gene symbol to Ensembl ID.

```r
# retrieve human gene Ensembl ID
ensembl_human <- getBM(
  attributes = c(
    "ensembl_gene_id",
    "hgnc_symbol"
  ),
  filters = "hgnc_symbol",
  values = unname(unlist(cc.genes.updated.2019)),
  mart = human
)

head(ensembl_human)
```

```
  ensembl_gene_id hgnc_symbol
1 ENSG00000011426        ANLN
2 ENSG00000143401      ANP32E
3 ENSG00000156802       ATAD2
4 ENSG00000087586       AURKA
5 ENSG00000178999       AURKB
6 ENSG00000089685       BIRC5
```

```
# add the Ensembl ID back to the cell cycle tibble
cc_genes <- cc_genes |> left_join(
  ensembl_human,
  by = c("gene_name" = "hgnc_symbol")
)
cc_genes
```

```
# A tibble: 99 x 3
   phase gene_name ensembl_gene_id
   <chr> <chr>     <chr>
 1 s     MCM5      ENSG00000100297
 2 s     PCNA      ENSG00000132646
 3 s     TYMS      ENSG00000176890
 4 s     FEN1      ENSG00000168496
 5 s     MCM7      ENSG00000166508
 6 s     MCM4      ENSG00000104738
 7 s     RRM1      ENSG00000167325
 8 s     UNG       ENSG00000076248
 9 s     GINS2     ENSG00000131153
10 s     MCM6      ENSG00000076003
# i 89 more rows
```

```
# be careful, sometimes one symbol can match 0 or multiple Ensembl ID
multi_match <- count(cc_genes, gene_name) |> # multiple matches
  filter(n > 1) |>
  pull(gene_name)

filter(cc_genes, gene_name %in% multi_match | is.na(ensembl_gene_id))
```

```
# A tibble: 4 x 3
  phase gene_name ensembl_gene_id
  <chr> <chr>     <chr>
1 s     UBR7      ENSG00000012963
```

```
2 s     UBR7      ENSG00000278787
3 s     CASP8AP2  ENSG00000288475
4 s     CASP8AP2  ENSG00000118412
```

```
# if there is 0 match, we can use synonym to retrieve Ensembl ID
searchAttributes(human, "synonym") # get the attribute name
ensembl_human_synonym <- getBM(
  attributes = c(
    "ensembl_gene_id",
    "hgnc_symbol",
    "external_synonym"
  ),
  filters = "external_synonym",
  values = filter(cc_genes, is.na(ensembl_gene_id)) |> pull(gene_name),
  mart = human
)
ensembl_human_synonym

# add Ensembl ID to `cc_genes` table
for (i in ensembl_human_synonym$external_synonym) {
  cc_genes$ensembl_gene_id[cc_genes$gene_name == i] <- ensembl_human_synonym$ensembl_gene_id
} # not the best way, you can do better ;)
```

4.2 Then we can start retrieving homologous genes:

```
mouse_markers <- getHomologs(
  ensembl_gene_ids = ensembl_human$ensembl_gene_id,
  species_from = "human",
  species_to = "mouse"
)

# you may still have some gene without match,
# you can use the Ensembl web site to search manually.
filter(mouse_markers, is.na(mmusculus_homolog_ensembl_gene) | mmusculus_homolog_ensembl_gene
```

```
  ensembl_gene_id mmusculus_homolog_ensembl_gene
1 ENSG00000077514
2 ENSG00000175063
3 ENSG00000278787
4 ENSG00000288475
```

4.3 `getHomologs()` returns mouse Ensembl ID, now we need to convert them into gene symbol.

```r
ensembl_mouse <- getBM(
  attributes = c(
    "ensembl_gene_id",
    "external_gene_name"
  ),
  filters = "ensembl_gene_id",
  values = mouse_markers$mmusculus_homolog_ensembl_gene,
  mart = mouse
)
head(ensembl_mouse)
```

```
     ensembl_gene_id external_gene_name
1 ENSMUSG00000000028             Cdc45
2 ENSMUSG00000001228             Uhrf1
3 ENSMUSG00000004642              Slbp
4 ENSMUSG00000004880               Lbr
5 ENSMUSG00000005410              Mcm5
6 ENSMUSG00000005698              Ctcf
```

```r
# rename column to avoid confusion
names(ensembl_mouse) <- c("mouse_ensembl", "mouse_symbol")

# add mouse gene symbol to the marker table
mouse_markers <- left_join(
  mouse_markers,
  ensembl_mouse,
  by = c("mmusculus_homolog_ensembl_gene" = "mouse_ensembl")
)
head(mouse_markers)
```

```
  ensembl_gene_id mmusculus_homolog_ensembl_gene mouse_symbol
1 ENSG00000010292             ENSMUSG00000038252       Ncapd2
2 ENSG00000011426             ENSMUSG00000036777         Anln
3 ENSG00000012963             ENSMUSG00000041712         Ubr7
4 ENSG00000013810             ENSMUSG00000037313        Tacc3
5 ENSG00000049541             ENSMUSG00000023104         Rfc2
6 ENSG00000051180             ENSMUSG00000027323        Rad51
```

4.4 At the end, we will merge with phase information from human genes:

```r
final_markers <- full_join(cc_genes, mouse_markers, by = "ensembl_gene_id")

head(final_markers)
```

```
# A tibble: 6 x 5
  phase gene_name ensembl_gene_id mmusculus_homolog_ensembl_gene mouse_symbol
  <chr> <chr>     <chr>           <chr>                          <chr>
1 s     MCM5      ENSG00000100297 ENSMUSG00000005410             Mcm5
2 s     PCNA      ENSG00000132646 ENSMUSG00000027342             Pcna
3 s     TYMS      ENSG00000176890 ENSMUSG00000025747             Tyms
4 s     FEN1      ENSG00000168496 ENSMUSG00000024742             Fen1
5 s     MCM7      ENSG00000166508 ENSMUSG00000029730             Mcm7
6 s     MCM4      ENSG00000104738 ENSMUSG00000022673             Mcm4
```

Finally you get mouse versions of G1/S and G2/M marker genes and you can use them for cell cycle scoring.

5. Apply to Seurat object (`seurat_obj`)

```r
# extract gene lists
s_genes_mouse <- filter(
  final_markers, phase == "s" & !is.na(mouse_symbol)
) |>
  pull(mouse_symbol) |>
  unique()

g2m_genes_mouse <- filter(
  final_markers, phase == "g2m" & !is.na(mouse_symbol)
) |>
  pull(mouse_symbol) |>
  unique()
```

```r
# run cell cylce scoring
seurat_obj <- CellCycleScoring(
  seurat_obj,
  s.features = s_genes_mouse,
  g2m.features = g2m_genes_mouse,
  set.ident = TRUE
)
```