task8

2025-09-30

```
## Install & load `tidybiology` (GitHub) and tidyverse
# We try to install tidybiology from GitHub if it's not already installed.
# If installation fails (e.g., no internet), we fall back to a small demo dataset
# so the document can still knit.

use_demo <- FALSE  # this will be set to TRUE if we cannot load tidybiology

# Helper to safely require a package ------------------------------------------------
safe_require <- function(pkg) {
  suppressPackageStartupMessages(requireNamespace(pkg, quietly = TRUE))
}

# 1) Ensure a GitHub installer is available (remotes preferred; devtools works too)
if (!safe_require("remotes") && !safe_require("devtools")) {
  install.packages("remotes")
}

# 2) Install tidybiology if missing
if (!safe_require("tidybiology")) {
  msg <- tryCatch({
    if (safe_require("remotes")) {
      remotes::install_github("hirscheylab/tidybiology", upgrade = "never", dependencies = TRU
E)
    } else {
      devtools::install_github("hirscheylab/tidybiology", upgrade = "never", dependencies = TR
UE)
    }
    NULL
  }, error = function(e) e$message)
  if (!safe_require("tidybiology")) {
    message("Could not install/load 'tidybiology' from GitHub.\nReason: ", msg,
            "\nUsing a small demo dataset so this document can knit. ",
            "Please install 'tidybiology' later and re-knit for the real data.")
    use_demo <- TRUE
  }
}

# Load plotting & wrangling packages
suppressPackageStartupMessages({
  library(ggplot2)
  library(dplyr)
  library(tidyr)
  library(stringr)
})
```

```r
# --- GUARANTEE: define chr & prt FIRST, then detect columns ---------------------

# 0) Load real data if tidybiology is available; otherwise use a demo fallback
if (!use_demo && requireNamespace("tidybiology", quietly = TRUE)) {
  data("chromosome", package = "tidybiology")
  data("proteins",   package = "tidybiology")
  chr <- get("chromosome")
  prt <- get("proteins")
  message("Loaded REAL data: chromosome (", nrow(chr), "×", ncol(chr),
          "), proteins (", nrow(prt), "×", ncol(prt), ").")
} else {
  set.seed(42)
  chr <- data.frame(
    chromosome = paste0("chr", 1:22),
    length_bp  = round(runif(22, 40, 250))*1e6,
    variations = rpois(22, 5e4),
    protein_coding_genes = rpois(22, 1600),
    miRNAs     = rpois(22, 80),
    check.names = FALSE
  )
  len <- rpois(3000, 350) + 50
  prt <- data.frame(
    length = len,
    mass   = (len * 110) + rnorm(3000, 0, 6000),
    check.names = FALSE
  )
  message("USING DEMO data. Install 'tidybiology' and re-knit to analyze the real datasets.")
}
```

```
## Loaded REAL data: chromosome (24×14), proteins (20430×8).
```

```r
# 1) Helpers ------------------------------------------------------------------
library(stringr); library(dplyr)

pick_col <- function(df, patterns) {
  nm <- names(df); lo <- tolower(nm)
  first_hit <- function(p) {
    i <- which(str_detect(lo, p))
    if (length(i)) i[1] else NA_integer_
  }
  idx <- NA_integer_
  for (p in patterns) {
    i <- first_hit(p)
    if (is.finite(i)) { idx <- i; break }
  }
  if (is.na(idx)) NA_character_ else nm[idx]
}

# 2) Detect chromosome columns ------------------------------------------------
size_col_chr <- pick_col(chr, c("length", "size", "bp", "base"))
var_col      <- pick_col(chr, c("variation", "variant", "\\bsnp\\b", "\\bsnv\\b", "\\bvar
\\b"))
prot_col     <- pick_col(chr, c("protein.*coding", "protein_coding", "\\bprotein\\b"))
mirna_col    <- pick_col(chr, c("\\bmi[-_ ]?rna\\b", "mirna", "micro\\s*rna", "microrna", "mir
nas?"))

if (is.na(size_col_chr))
  stop("Could not find the chromosome size/length column in 'chromosome' data.\n",
       "Columns available: ", paste(names(chr), collapse = ", "))

# --- (Optional) MANUAL OVERRIDE: if you know the exact miRNA column name, set it:
# mirna_col <- "miRNAs"   # <- e.g. set to your real column name and remove '#'

# 3) Safely coerce detected numeric columns -----------------------------------
cols_to_num <- c(size_col_chr, var_col, prot_col, mirna_col)
cols_to_num <- intersect(cols_to_num[!is.na(cols_to_num) & nzchar(cols_to_num)], names(chr))
if (length(cols_to_num)) {
  chr[cols_to_num] <- lapply(chr[cols_to_num], function(z) suppressWarnings(as.numeric(z)))
}

# 4) Detect proteins columns --------------------------------------------------
len_col_prt  <- pick_col(prt, c("^length$", "length", "\\blen\\b"))
mass_col_prt <- pick_col(prt, c("^mass$", "mass", "weight", "mw", "kda", "dalton"))
if (is.na(len_col_prt) || is.na(mass_col_prt))
  stop("Could not find 'length' and/or 'mass' in 'proteins' data.\n",
       "Columns available: ", paste(names(prt), collapse = ", "))

# 5) Flags for downstream plotting --------------------------------------------
has_prot  <- !is.na(prot_col)  && prot_col  %in% names(chr)
has_mirna <- !is.na(mirna_col) && mirna_col %in% names(chr)

# 6) Diagnostics (printed to page) --------------------------------------------
message("Chromosome columns: ", paste(names(chr), collapse = ", "))
```

```
## Chromosome columns: id, length_mm, basepairs, variations, protein_codinggenes, pseudo_gene
s, totallongnc_rna, totalsmallnc_rna, mi_rna, r_rna, sn_rna, sno_rna, miscnc_rna, centromerepo
sition_mbp
```

```
message("Detected -> size: ", size_col_chr,
        " | variations: ", ifelse(is.na(var_col), "NOT FOUND", var_col),
        " | protein_coding: ", ifelse(is.na(prot_col), "NOT FOUND", prot_col),
        " | miRNAs: ", ifelse(is.na(mirna_col), "NOT FOUND", mirna_col))
```

```
## Detected -> size: length_mm | variations: variations | protein_coding: protein_codinggenes
| miRNAs: mi_rna
```

```
message("Proteins -> length: ", len_col_prt, " | mass: ", mass_col_prt)
```

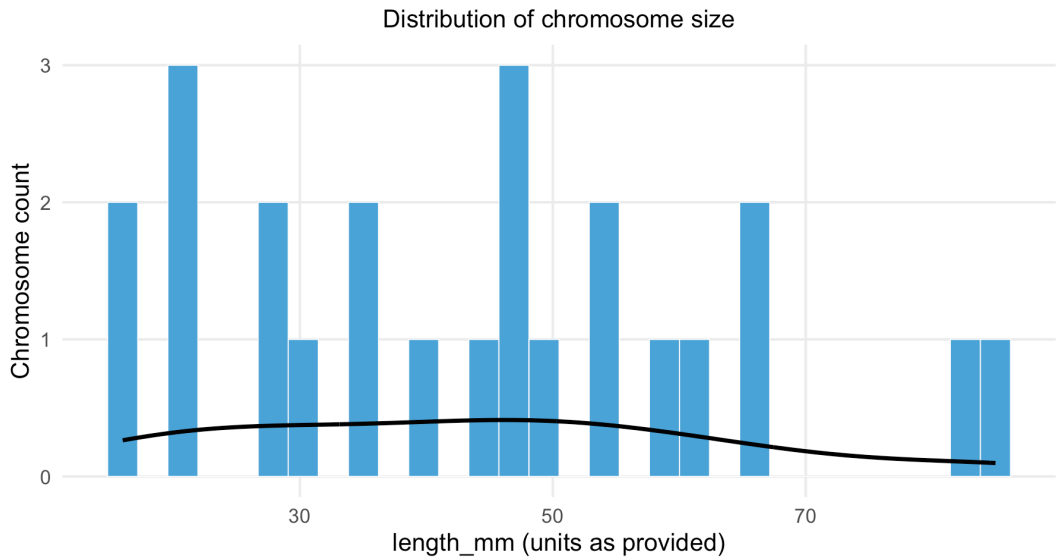
```
## Proteins -> length: length | mass: mass
```

```
## a summary statistics
library(tidyr); library(dplyr)

# Map "friendly names" -> detected real columns (may be NA if not found) --------
targets <- c(variations = var_col,
             protein_coding_genes = prot_col,
             miRNAs = mirna_col)

# Keep only targets that actually exist in the data ------------------------------
present <- targets[!is.na(targets) & nzchar(targets) & targets %in% names(chr)]

if (!length(present)) {
  stop("None of the requested columns (variations / protein_coding_genes / miRNAs) were found
in 'chromosome'.")
}

# Compute mean/median/max for the present targets --------------------------------
chr_summary <- chr %>%
  summarise(
    across(any_of(unname(present)),
           list(mean   = ~mean(.,   na.rm = TRUE),
                median = ~median(., na.rm = TRUE),
                max    = ~max(.,    na.rm = TRUE)))
  ) %>%
  pivot_longer(everything(),
               names_to = c("column", ".value"),
               names_sep = "_(?=(mean|median|max)$)") %>%
  mutate(variable = names(present)[match(column, unname(present))]) %>%  # back-map to friendl
y names
  select(variable, mean, median, max)

knitr::kable(chr_summary, caption = "Chromosome: summary stats (mean / median / max)")
```

Chromosome: summary stats (mean / median / max)

variable	mean	median	max
variations	6.484572e+06	6172346	12945965
protein_coding_genes	8.499583e+02	836	2058
miRNAs	7.316667e+01	75	134

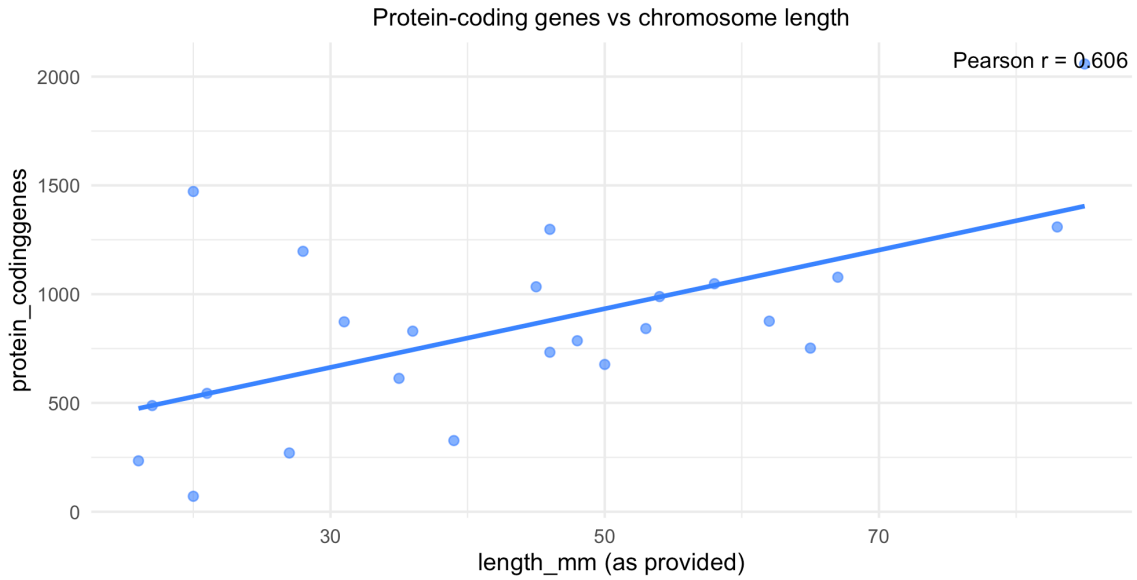
```
## b How does chromosome size distribute?
ggplot(chr, aes(x = .data[[size_col_chr]])) +
  geom_histogram(bins = 30, color = "white", linewidth = 0.2, fill = "#4EA5D9") +
  geom_density(aes(y = after_stat(count)), linewidth = 0.8, alpha = 0.3) +
  labs(title = "Distribution of chromosome size",
       x = paste0(size_col_chr, " (units as provided)"),
       y = "Chromosome count") +
  theme_minimal(base_size = 10) +
  theme(plot.title = element_text(size = 10, hjust = 0.5),
        panel.grid.minor = element_blank())
```



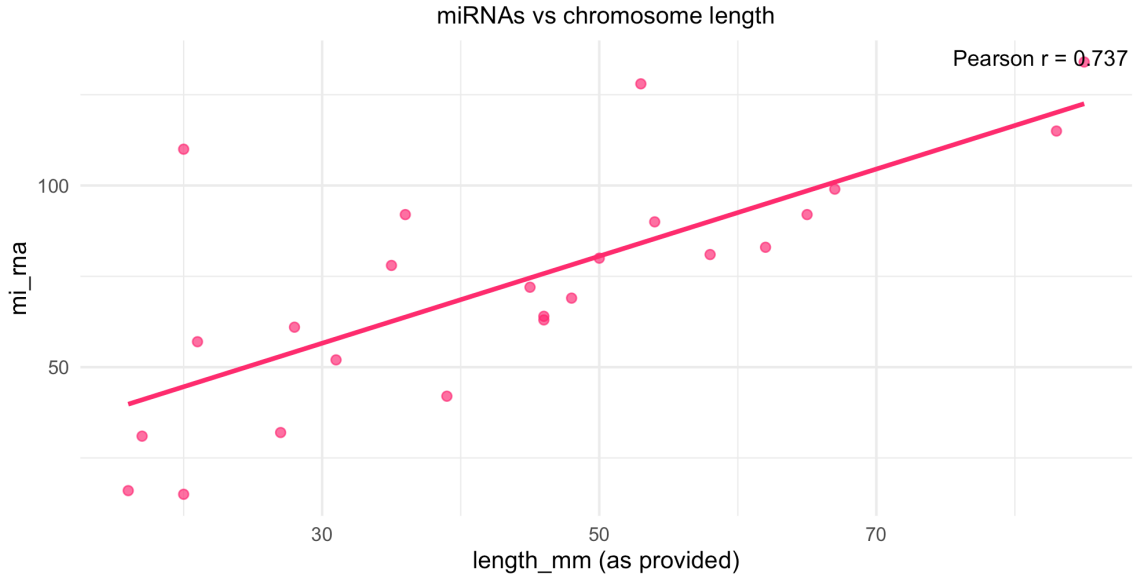
(c) Correlation with chromosome length (protein-

coding genes / miRNAs)

```
## c Correlation with chromosome length (protein-coding genes / miRNAs)

scatter_lm <- function(df, xcol, ycol, pt_col, line_col, title_txt){
  r <- suppressWarnings(cor(df[[xcol]], df[[ycol]], use = "pairwise.complete.obs", method = "p
earson"))
  ggplot(df, aes(x = .data[[xcol]], y = .data[[ycol]])) +
    geom_point(alpha = 0.65, size = 1.6, color = pt_col) +
    geom_smooth(method = "lm", se = FALSE, linewidth = 0.9, color = line_col) +
    labs(title = title_txt,
         x = paste0(xcol, " (as provided)"),
         y = ycol) +
    annotate("text", x = Inf, y = Inf,
             label = sprintf("Pearson r = %.3f", r),
             hjust = 1.02, vjust = 1.6, size = 3.2) +
    theme_minimal(base_size = 10) +
    theme(plot.title = element_text(size = 10, hjust = 0.5))
}

# 1) Protein-coding genes vs chromosome length
if (has_prot) {
  print(scatter_lm(chr, size_col_chr, prot_col,
                   pt_col = "#3A86FF", line_col = "#3A86FF",
                   title_txt = "Protein-coding genes vs chromosome length"))
} else {
  print(ggplot() + theme_void() +
          annotate("text", x = 0, y = 0,
                   label = "No 'protein-coding' column detected;\nskipping this plot.",
                   size = 4))
}
```



```
# 2) miRNAs vs chromosome length
if (has_mirna) {
  print(scatter_lm(chr, size_col_chr, mirna_col,
                   pt_col = "#FF006E", line_col = "#FF006E",
                   title_txt = "miRNAs vs chromosome length"))
} else {
  print(ggplot() + theme_void() +
          annotate("text", x = 0, y = 0,
                   label = "No 'miRNA' column detected;\nskipping this plot.",
                   size = 4))
}
```




```
## d Proteins: summary stats & visualization (length vs mass)

prot_summary <- prt %>%
  summarise(
    across(any_of(c(len_col_prt, mass_col_prt)),
           list(mean = ~mean(., na.rm = TRUE),
                median = ~median(., na.rm = TRUE),
                max = ~max(., na.rm = TRUE)))
  ) %>%
  pivot_longer(everything(),
               names_to = c("column", ".value"),
               names_sep = "_(?=(mean|median|max)$)") %>%
  mutate(variable = c(length = len_col_prt, mass = mass_col_prt)[column]) %>%
  select(variable, mean, median, max)

knitr::kable(prot_summary, caption = "Proteins: summary stats for length & mass")
```

Proteins: summary stats for length & mass

variable	mean	median	max
length	557.1603	414.0	34350
mass	62061.3791	46140.5	3816030

```r
# Ensure 'scales' is available (install once if missing)
if (!requireNamespace("scales", quietly = TRUE)) install.packages("scales")

ggplot(prt, aes(x = .data[[len_col_prt]], y = .data[[mass_col_prt]])) +
  geom_point(alpha = 0.35, size = 1.4, color = "#4ECDC4") +
  geom_smooth(method = "loess", se = FALSE, linewidth = 1, color = "#FF6B6B", span = 0.8) +
  scale_x_continuous(trans = "log10", labels = scales::label_comma()) +  # <-- use fully-quali
fied labeler
  scale_y_continuous(trans = "log10", labels = scales::label_comma()) +  # <-- same here
  labs(title = "Protein mass vs length (log–log view)",
       x = paste0(len_col_prt, " (log10)"),
       y = paste0(mass_col_prt, " (log10)")) +
  theme_minimal(base_size = 11) +
  theme(plot.title = element_text(size = 11, hjust = 0.5, face = "bold"),
        panel.grid.minor = element_blank(),
        panel.grid.major.x = element_line(linewidth = 0.2, linetype = 3),
        panel.grid.major.y = element_line(linewidth = 0.2, linetype = 3))
```