Recherche des sources de variabilite qui influencent la qualité des Runs RNASeq

Mickael Coquerelle

2025-06-24

```
library(tidyverse)
```

1. Préparation du df et des données d'entrée

Ce qui est intéressant en biologie et particulièrement en génétique, c'est de constater à quel point dans la littérature on prends des précautions sur la véracité des relations de cause à effets entre une variable A et une variable B. D'ailleurs dans la majorité des articles ou sites que j'ai pu parcourir on parle rarement de cause à effet mais de potentiel corrélation (ce qui est ne. Du reste, on peut vite se perdre dans les méandres méthodologiques statistiques tant il y'a d'approche et de conditions à respecter pour démontrer une significativité dans la corrélation de deux variables, tant il y'a de garde fou à considérés (hypothèse de normalité, homoscédasticité , correction des p-value ...). Au vu de la période de stage et du temps imparti, j'ai essayer de faire un focus sur certaines variables explicatives qui m'ont paru pertinente à récolter avec l'aide de l'équipe du laboratoire, il à fallu faire certains choix, dans le tableau ci-dessous je propose un résumé des données que je considères intéressante à explorer.

```
fichier <- "Stats_Log_merge_with_deltas.csv"

df_raw <- read_csv(fichier,
                   locale = locale(encoding = "ISO-8859-1"),
                   guess_max = 100,
                   show_col_types = TRUE)
```

3. Choix des variables explicatives et réponses pour l'exploration

Ci-dessous, je stocke dans deux vecteurs les variables explicatives et les variables réponses que j'ai choisies pour mon analyse de corrélation. Il a fallu faire des choix : cette sélection n'est pas exhaustive. On peut bien sûr discuter de la pertinence de telle ou telle métrique, mais, dans le cadre de ces analyses univariées, je cherche à balayer rapidement les éléments qui me semblent les plus susceptibles d'influencer la robustesse expérimentale.

J'ai dû faire ces choix en tenant compte du temps imparti et du cadre du stage, car il y avait beaucoup de points à aborder, mais aussi en fonction de la disponibilité et de l'accessibilité des données. Comme je l'évoque plus haut dans mon rapport, lorsqu'on cherche à évaluer la corrélation entre deux variables, il faut poser des limites, et ici, le critère temporel a joué un rôle évident :

```
vars_exp <- c("Ville_Prescripteur", "Date_Prelevement", "Date_Recep", "Date_extraction",
              "Concentration_ARN", "Purete_proteique", "Date_Lib", "Date_Lancement",
              "Delta_Run_Prel", "Delta_Run_Recep", "Delta_Run_Ext", "Delta_Run_Lib", "STAR_Type")

vars_resp <- c("STAR_Total_reads", "STAR_Unique_reads", "STAR_Unique_pct", "STAR_Multi_reads",
```

```
                "STAR_Multi_pct", "STAR_No_map_reads", "STAR_No_map_pct_sum",
                "CRAC_Total_reads", "CRAC_Unique_reads", "CRAC_Unique_pct", "CRAC_Multi_reads",
                "CRAC_Multi_pct", "CRAC_No_map_reads", "CRAC_No_map_pct")
```

J'ai également choisi de filtrer les variables qualitatives par manque de temps. Même si je les stocke dans les vecteurs initiaux, je n'ai pas eu le temps de les explorer pour l'instant, mais elles sont incluses afin de pouvoir étudier leur impact ultérieurement.

```
vars_exp <- intersect(vars_exp, colnames(df_raw))
vars_resp <- intersect(vars_resp, colnames(df_raw))

vars_exp_num <- vars_exp[sapply(df_raw[vars_exp], is.numeric)]
vars_resp_num <- vars_resp[sapply(df_raw[vars_resp], is.numeric)]
```

4. Fonction personnalisée pour les corrélations

```
corr_metrics <- function(x, y) {
  ok <- !is.na(x) & !is.na(y)
  x_ok <- x[ok]
  y_ok <- y[ok]

  if(length(x_ok) < 10) return(c(cor = NA, R2 = NA, pval = NA))

  cor_val <- cor(x_ok, y_ok, method = "pearson")
  lmfit <- lm(y_ok ~ x_ok)
  sum_lm <- summary(lmfit)
  R2 <- sum_lm$r.squared
  pval <- coef(summary(lmfit))[2,4]

  c(cor = cor_val, R2 = R2, pval = pval)
}
```

5. Calcul des matrices de corrélation

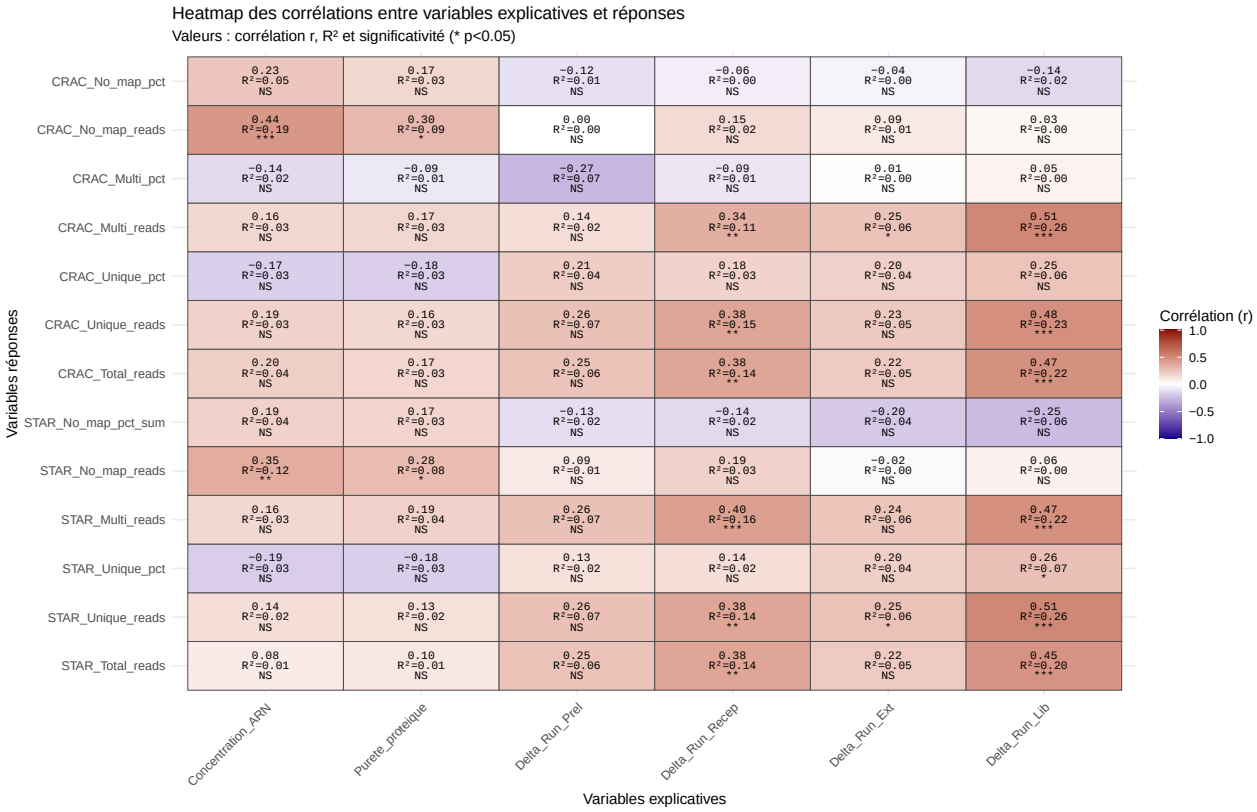
```
cor_mat  <- matrix(NA_real_, nrow = length(vars_resp_num), ncol = length(vars_exp_num),
                   dimnames = list(vars_resp_num, vars_exp_num))
R2_mat   <- cor_mat
pval_mat <- cor_mat

for (resp in vars_resp_num) {
  for (expv in vars_exp_num) {
    m <- corr_metrics(df_raw[[expv]], df_raw[[resp]])
    cor_mat[resp, expv]  <- m["cor"]
    R2_mat[resp, expv]   <- m["R2"]
    pval_mat[resp, expv] <- m["pval"]
  }
}
```

6. Formatage des résultats pour ggplot2

```
df_corr <- as.data.frame(as.table(cor_mat))
colnames(df_corr) <- c("Response", "Explicative", "Correlation")

df_corr$R2 <- as.vector(R2_mat)
df_corr$pval <- as.vector(pval_mat)

df_corr <- df_corr %>%
  mutate(
    R2_label = ifelse(!is.na(R2), sprintf("R²=%.2f", R2), ""),
    signif = case_when(
      is.na(pval) ~ "NS",
      pval < 0.001 ~ "***",
      pval < 0.01  ~ "**",
      pval < 0.05  ~ "*",
      TRUE         ~ "NS"
    ),
    label = ifelse(!is.na(Correlation),
                   paste0(sprintf("%.2f", Correlation), "\n", R2_label, "\n", signif),
                   "")
  )
```

7. Génération de la heatmap des corrélations

```
heatmap_plot <- ggplot(df_corr, aes(x = Explicative, y = Response, fill = Correlation)) +
  geom_tile(color = "grey30") +
  geom_text(aes(label = label), size = 3, lineheight = 0.8, family = "mono") +
  scale_fill_gradient2(low = "darkblue", mid = "white", high = "darkred", midpoint = 0,
                       na.value = "grey90", limits = c(-1, 1), name = "Corrélation (r)") +
  theme_minimal(base_size = 12) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1)) +
  labs(
    title = "Heatmap des corrélations entre variables explicatives et réponses",
    subtitle = "Valeurs : corrélation r, R² et significativité (* p<0.05)",
    x = "Variables explicatives",
    y = "Variables réponses"
  )

print(heatmap_plot)
```



8. Sauvegarde de la figure et des résultats

```
ggsave("Heatmap_Correlation.png",
       heatmap_plot, width = 14, height = 9, dpi = 300)

write.csv(df_corr %>% select(Response, Explicative, Correlation, R2, pval, signif),
          file = "Correlation_Results_numeric_only.csv",
          row.names = FALSE, fileEncoding = "UTF-8")
```