# Votre rapport

## Contents

```
################################################################################
# Analyse pré-post (6 participants)
# ------------------------------------------------------------------------------
# 1. Charge les jeux de données preinter.csv / postinter.csv.
# 2. Sépare les variables :
#       • AE*  = sentiment d'auto-efficacité
#       • COP* = régulation émotionnelle
# 3. Pour chaque variable :
#       • Shapiro-Wilk normalité (pré & post) --> si valeurs non identiques
#       • t-test Welch non appareillé          → t-value, p-value
#       • d de Cohen                           → magnitude (très faible → très forte)
# 5. Concatène les résultats dans deux tableaux,
################################################################################

#
# instalation automatique des librairies
#
pkgs <- c(
  "tidyverse",  # contient dplyr, readr, purrr, stringr, etc.
  "effsize",    # pour la fonction cohen.d()
  "broom",      # (optionnel) tidy() pour mettre en forme les résultats de tests
  "knitr"       # pour kable(), la mise en forme des tableaux
)
to_install <- pkgs[!(pkgs %in% rownames(installed.packages()))]
if (length(to_install)) install.packages(to_install, dependencies = TRUE)

#
# Chargement des librairies
#
invisible(lapply(pkgs, library, character.only = TRUE))
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.2     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
#
# Lecture des fichiers
#
pre  <- read_delim(
  "preinter.csv",
  delim           = ";", # le séparateur est un ; exce francophone
  show_col_types  = FALSE, # on n'affiche pas le type de données par colones
  locale          = locale(encoding = "UTF-8"), # caractères classique
  col_types       = cols(
    `n°`     = col_skip(),     # on ignore la colonne ID
    .default = col_double()    # toutes les autres colonnes sont définis comme numériques
  )
)

post <- read_delim(
  "postinter.csv",
  delim           = ";",
  show_col_types  = FALSE,
  locale          = locale(encoding = "UTF-8"),
  col_types       = cols(
    `n°`     = col_skip(),
    .default = col_double()
  )
)



#
# (c) Détection des familles de variables
#
vars_ae  <- names(pre)  %>% str_subset("^AE")
vars_cop <- names(pre)  %>% str_subset("^COP")

# verification que les 2 fichiers (pre/post) contiennent les mêmes variables
vars_ae  <- intersect(vars_ae,  names(post))
vars_cop <- intersect(vars_cop, names(post))

cat("Variables de sentiment d'auto-efficacité (AE) identifiées :\n")
```

```
## Variables de sentiment d'auto-efficacité (AE) identifiées :
```

```r
cat(paste0("- ", vars_ae), sep = "\n")
```

```
## - AE 1
## - AE 2
## - AE 3
## - AE 4
## - AE 5
## - AE 6
## - AE 7
```

```r
cat("\nVariables de régulation émotionnelle (COP) identifiées :\n")
```

```
##
## Variables de régulation émotionnelle (COP) identifiées :
```

```
cat(paste0("- ", vars_cop), sep = "\n")
```

```
## - COP1
## - COP2
## - COP3
## - COP4
## - COP5
## - COP6
## - COP7
## - COP8
## - COP9
## - COP10
## - COP11
## - COP12
## - COP13
## - COP14
## - COP15
## - COP16
## - COP17
## - COP18
```

```
# création des tableaux vides
table_ae <- tibble(
  Variable          = character(),
  `Pré intervention moyenne (ET)`  = character(),
  `Post intervention moyenne (ET)` = character(),
  t                 = numeric(),
  p                 = numeric(),
  `d de Cohen`      = numeric(),
  `Taille d'effet`  = character(),
  `Shapiro pré (p)` = numeric(),
  `Shapiro post (p)`= numeric()
)

table_cop <- tibble(
  Variable          = character(),
  `Pré intervention moyenne (ET)`  = character(),
  `Post intervention moyenne (ET)` = character(),
  t                 = numeric(),
  p                 = numeric(),
  `d de Cohen`      = numeric(),
  `Taille d'effet`  = character(),
  `Shapiro pré (p)` = numeric(),
  `Shapiro post (p)`= numeric()
)

# --- Analyse AE (sentiment d'auto-efficacité) ---
for (var in vars_ae) {
  # Extraction des données pré- et post-intervention
  pre_vals  <- (pre[[var]])
```

```r
  post_vals <- (post[[var]])

  #    Test de normalité Shapiro-Wilk (pré)
  shapiro_pre_test  <- shapiro.test(pre_vals)
  p_shapiro_pre     <- shapiro_pre_test$p.value
  # Affichage du W et de la p-value
  cat(sprintf(
    "Shapiro-Wilk pré [%s] : W = %.3f, p = %.3f\n",
    var,
    shapiro_pre_test$statistic,
    signif(p_shapiro_pre, 3)
  ))

  #    Test de normalité Shapiro-Wilk (post)
  shapiro_post_test <- shapiro.test(post_vals)
  p_shapiro_post    <- shapiro_post_test$p.value
  # Affichage du W et de la p-value
  cat(sprintf(
    "Shapiro-Wilk post [%s] : W = %.3f, p = %.3f\n\n",
    var,
    shapiro_post_test$statistic,
    signif(p_shapiro_post, 3)
  ))

  #    Test t  appareillé (Welch)
  t_res <- t.test(post_vals, pre_vals, paired = TRUE)

  #    Calcul du d de Cohen
#    d de Cohen pour données appariées
d_res <- effsize::cohen.d(
  post_vals,        # vecteur post-intervention
  pre_vals,         # vecteur pré-intervention
  paired = TRUE,    #  appariées
)
  #    Interprétation de la taille d'effet je sais pas si c'est ca que tu veux associer d à un type d'ef
  magnitude <- cut(
    abs(d_res$estimate),
    breaks = c(0, 0.2, 0.5, 0.8, 1.3, Inf),
    labels = c("très faible", "faible", "moyenne", "forte", "très forte"),
    right  = FALSE
  )

  #    remplissage du tableau AE
  table_ae <- table_ae %>%
    add_row(
      Variable           = var,
      `Pré intervention moyenne (ET)` = sprintf("%.2f (%.2f)", mean(pre_vals),  sd(pre_vals)),
      `Post intervention moyenne (ET)`= sprintf("%.2f (%.2f)", mean(post_vals), sd(post_vals)),
      t                  = round(t_res$statistic, 3),
      p                  = signif(t_res$p.value,  3),
      `d de Cohen`       = round(d_res$estimate,   3),
      `Taille d'effet`   = as.character(magnitude),
      `Shapiro pré (p)`  = signif(p_shapiro_pre,   3),
```

```
      `Shapiro post (p)` = signif(p_shapiro_post,  3)
    )
}
```

```
## Shapiro-Wilk pré [AE 1] : W = 0.823, p = 0.093
## Shapiro-Wilk post [AE 1] : W = 0.822, p = 0.091
##
## Shapiro-Wilk pré [AE 2] : W = 0.832, p = 0.111
## Shapiro-Wilk post [AE 2] : W = 0.775, p = 0.035
##
## Shapiro-Wilk pré [AE 3] : W = 0.866, p = 0.212
## Shapiro-Wilk post [AE 3] : W = 0.640, p = 0.001
##
## Shapiro-Wilk pré [AE 4] : W = 0.960, p = 0.820
## Shapiro-Wilk post [AE 4] : W = 0.822, p = 0.091
##
## Shapiro-Wilk pré [AE 5] : W = 0.640, p = 0.001
## Shapiro-Wilk post [AE 5] : W = 0.822, p = 0.091
##
## Shapiro-Wilk pré [AE 6] : W = 0.918, p = 0.493
## Shapiro-Wilk post [AE 6] : W = 0.781, p = 0.039
##
## Shapiro-Wilk pré [AE 7] : W = 0.891, p = 0.324
## Shapiro-Wilk post [AE 7] : W = 0.950, p = 0.737
```

```
# --- COP (régulation émotionnelle) ---
for (var in vars_cop) {
  # Extraction des données pré- et post-intervention
  pre_vals  <- (pre[[var]])
  post_vals <- (post[[var]])

  #    Test de normalité Shapiro-Wilk (pré)
  if (length(unique(pre_vals)) > 1) {
    sh_pre <- shapiro.test(pre_vals)
    p_shapiro_pre <- sh_pre$p.value
    cat(sprintf(
      "Shapiro-Wilk pré [%s] : W = %.3f, p = %.3f\n",
      var,
      sh_pre$statistic,
      signif(p_shapiro_pre, 3)
    ))
  } else {
    # Cas où toutes les réponses sont identiques ou trop peu nombreuses
    p_shapiro_pre <- NA_real_
    cat(sprintf(
      "Shapiro-Wilk pré [%s] : NA (valeurs identiques)\n",
      var
    ))
  }

  #    Test de normalité Shapiro-Wilk (post)
  if (length(unique(post_vals)) > 1) {
    sh_post <- shapiro.test(post_vals)
```

```
    p_shapiro_post <- sh_post$p.value
    cat(sprintf(
      "Shapiro-Wilk post [%s] : W = %.3f, p = %.3f\n\n",
      var,
      sh_post$statistic,
      signif(p_shapiro_post, 3)
    ))
  } else {
    p_shapiro_post <- NA_real_
    cat(sprintf(
      "Shapiro-Wilk post [%s] : NA (valeurs identiques)\n\n",
      var
    ))
  }
  #   Test t appareillé
  t_res <- t.test(post_vals, pre_vals, paired = TRUE)

  #   Calcul du d de Cohen
  d_res <- effsize::cohen.d(
  post_vals,        # vecteur post-intervention
  pre_vals,         # vecteur pré-intervention
  paired = TRUE,    #  appariées
)
  #   Interprétation de la taille d'effet
  magnitude <- cut(
    abs(d_res$estimate),
    breaks = c(0, 0.2, 0.5, 0.8, 1.3, Inf),
    labels = c("très faible", "faible", "moyenne", "forte", "très forte"),
    right  = FALSE
  )

  #   Remplissage du tableau COP
  table_cop <- table_cop %>%
    add_row(
      Variable           = var,
      `Pré intervention moyenne (ET)` = sprintf("%.2f (%.2f)", mean(pre_vals),  sd(pre_vals)),
      `Post intervention moyenne (ET)`= sprintf("%.2f (%.2f)", mean(post_vals), sd(post_vals)),
      t                  = round(t_res$statistic, 3),
      p                  = signif(t_res$p.value,  3),
      `d de Cohen`       = round(d_res$estimate,   3),
      `Taille d'effet`   = as.character(magnitude),
      `Shapiro pré (p)`  = signif(p_shapiro_pre,   3),
      `Shapiro post (p)` = signif(p_shapiro_post,  3)
    )
}
```

```
## Shapiro-Wilk pré [COP1] : W = 0.751, p = 0.020
## Shapiro-Wilk post [COP1] : W = 0.640, p = 0.001
##
## Shapiro-Wilk pré [COP2] : W = 0.921, p = 0.514
## Shapiro-Wilk post [COP2] : W = 0.866, p = 0.212
##
## Shapiro-Wilk pré [COP3] : W = 0.805, p = 0.065
```

```
## Shapiro-Wilk post [COP3] : NA (valeurs identiques)
##
## Shapiro-Wilk pré [COP4] : W = 0.640, p = 0.001
## Shapiro-Wilk post [COP4] : W = 0.640, p = 0.001
##
## Shapiro-Wilk pré [COP5] : W = 0.827, p = 0.101
## Shapiro-Wilk post [COP5] : W = 0.775, p = 0.035
##
## Shapiro-Wilk pré [COP6] : W = 0.907, p = 0.415
## Shapiro-Wilk post [COP6] : W = 0.721, p = 0.010
##
## Shapiro-Wilk pré [COP7] : W = 0.831, p = 0.110
## Shapiro-Wilk post [COP7] : W = 0.773, p = 0.033
##
## Shapiro-Wilk pré [COP8] : W = 0.869, p = 0.223
## Shapiro-Wilk post [COP8] : W = 0.908, p = 0.425
##
## Shapiro-Wilk pré [COP9] : W = 0.701, p = 0.006
## Shapiro-Wilk post [COP9] : W = 0.640, p = 0.001
##
## Shapiro-Wilk pré [COP10] : W = 0.640, p = 0.001
## Shapiro-Wilk post [COP10] : W = 0.683, p = 0.004
##
## Shapiro-Wilk pré [COP11] : W = 0.683, p = 0.004
## Shapiro-Wilk post [COP11] : W = 0.640, p = 0.001
##
## Shapiro-Wilk pré [COP12] : W = 0.908, p = 0.421
## Shapiro-Wilk post [COP12] : W = 0.866, p = 0.212
##
## Shapiro-Wilk pré [COP13] : W = 0.822, p = 0.091
## Shapiro-Wilk post [COP13] : W = 0.640, p = 0.001
##
## Shapiro-Wilk pré [COP14] : W = 0.683, p = 0.004
## Shapiro-Wilk post [COP14] : W = 0.701, p = 0.006
##
## Shapiro-Wilk pré [COP15] : W = 0.640, p = 0.001
## Shapiro-Wilk post [COP15] : W = 0.640, p = 0.001
##
## Shapiro-Wilk pré [COP16] : W = 0.822, p = 0.091
## Shapiro-Wilk post [COP16] : W = 0.683, p = 0.004
##
## Shapiro-Wilk pré [COP17] : W = 0.683, p = 0.004
## Shapiro-Wilk post [COP17] : W = 0.496, p = 0.000
##
## Shapiro-Wilk pré [COP18] : W = 0.908, p = 0.421
## Shapiro-Wilk post [COP18] : W = 0.702, p = 0.006
```

```
#
# (f) Affichage final
#
cat("## Tableau AE - auto-efficacité\n")
```

```
## ## Tableau AE - auto-efficacité
```

```
print(knitr::kable(table_ae, align="lcccc", caption="Items AE"))
```

```
##
##
## Table: Items AE
##
## |Variable | Pré intervention moyenne (ET) | Post intervention moyenne (ET) |   t   |   p    |d de Col
## |:--------|:-----------------------------:|:------------------------------:|:-----:|:------:|:------
## |AE 1     |          5.33 (1.37)          |          6.33 (0.82)           | 2.739 | 0.0409 |0.747
## |AE 2     |          5.50 (1.76)          |          6.17 (0.98)           | 1.085 | 0.3280 |0.434
## |AE 3     |          6.17 (0.75)          |          6.33 (0.52)           | 0.542 | 0.6110 |0.254
## |AE 4     |          5.50 (1.05)          |          6.33 (0.82)           | 2.712 | 0.0422 |0.857
## |AE 5     |          5.67 (1.03)          |          6.33 (0.82)           | 2.000 | 0.1020 |0.700
## |AE 6     |          4.67 (2.07)          |          5.33 (1.86)           | 0.791 | 0.4650 |0.338
## |AE 7     |          4.17 (2.40)          |          4.33 (1.86)           | 0.210 | 0.8420 |0.076
```

```
cat("\n## Tableau COP - régulation émotionnelle\n")
```

```
##
## ## Tableau COP - régulation émotionnelle
```

```
print(knitr::kable(table_cop, align="lcccc", caption="Items COP"))
```

```
##
##
## Table: Items COP
##
## |Variable | Pré intervention moyenne (ET) | Post intervention moyenne (ET) |   t    |   p    |d de Co
## |:--------|:-----------------------------:|:------------------------------:|:------:|:------:|:-----
## |COP1     |          3.83 (1.47)          |          4.67 (0.52)           | 1.185  | 0.2890 |0.795
## |COP2     |          3.17 (1.33)          |          3.67 (1.51)           | 0.808  | 0.4560 |0.351
## |COP3     |          3.83 (1.33)          |          5.00 (0.00)           | 2.150  | 0.0842 |1.241
## |COP4     |          1.67 (0.52)          |          1.67 (1.03)           | 0.000  | 1.0000 |0.000
## |COP5     |          2.00 (0.63)          |          1.83 (0.98)           | -0.307 | 0.7710 |-0.204
## |COP6     |          3.33 (1.21)          |          4.00 (1.55)           | 1.348  | 0.2350 |0.467
## |COP7     |          2.00 (1.26)          |          1.83 (1.17)           | -0.542 | 0.6110 |-0.136
## |COP8     |          2.83 (1.72)          |          2.83 (1.60)           | 0.000  | 1.0000 |0.000
## |COP9     |          4.50 (0.84)          |          4.67 (0.52)           | 0.542  | 0.6110 |0.229
## |COP10    |          4.67 (0.52)          |          4.50 (0.55)           | -1.000 | 0.3630 |-0.312
## |COP11    |          1.50 (0.55)          |          1.33 (0.52)           | -0.542 | 0.6110 |-0.313
## |COP12    |          3.17 (1.17)          |          3.33 (1.51)           | 0.307  | 0.7710 |0.122
## |COP13    |          3.67 (0.82)          |          4.67 (0.52)           | 3.873  | 0.0117 |1.356
## |COP14    |          1.50 (0.55)          |          1.50 (0.84)           | 0.000  | 1.0000 |0.000
## |COP15    |          4.67 (0.52)          |          4.67 (0.52)           |  NaN   |  NaN   |NaN
## |COP16    |          4.33 (0.82)          |          4.50 (0.55)           | 1.000  | 0.3630 |0.188
## |COP17    |          1.50 (0.55)          |          1.33 (0.82)           | -0.349 | 0.7410 |-0.243
## |COP18    |          2.17 (1.17)          |          1.83 (1.33)           | -0.791 | 0.4650 |-0.264
```