ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

DIPARTIMENTO DI ARCHITETTURA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA EDILE-ARCHITETTURA

TESI DI LAUREA

in
Science of Climate Change

ANALISI DEI TREND CLIMATICI DI CAPOLUOGHI ITALIANI PER LO SVILUPPO DI STRATEGIE DI ADATTAMENTO AL RISCALDAMENTO CLIMATICO IN AMBITO URBANO

CANDIDATO:
Marta Benetti

RELATORE:
Chiar.mo Prof. Alberto Montanari

ALLEGATO

Anno Accademico 2025/26
Sessione I

Indice

Allegato 1: Testi dei programmi per l'analisi climatica 1

Milano – Stazione di Milano Brera 2

Torino – Stazione di Torino Giardini Reali 19

Roma – Stazione di Roma Macao 37

Napoli – Stazione di Napoli Capodichino 55

Palermo Osservatorio Astronomico 76

Allegato 2: Grafici ottenuti dall'analisi climatica dei cinque capoluoghi 97

Allegato 1: Testi dei programmi per l'analisi climatica

In questo allegato sono riportati gli script dei cinque programmi sviluppati nell'ambiente RStudio per l'analisi climatica delle città di Roma, Milano, Napoli, Torino e Palermo.

Gli script sono presentati nell'ordine in cui sono stati sviluppati, così da mostrare il processo evolutivo che ha portato dal programma principale, realizzato per la città di Milano, allo sviluppo degli altri quattro. Ciascuno script risponde, infatti, alle specifiche esigenze di omogeneizzazione riscontrate nei diversi dataset.

Milano – Stazione di Milano Brera

```
################################################################
# 0. SETUP
################################################################

# Pulizia ambiente
rm(list = ls())

# Librerie necessarie
library(readxl)     # lettura Excel
library(dplyr)      # manipolazione dati
library(lubridate)  # gestione date
library(ggplot2)    # grafici
library(forecast)   # ACF
library(zoo)        # aggregazioni
library(ggtext)     # testi extra su assi dei grafici

# Definire working directory                                  -> INSERIRE CARTELLA DI LAVORO
setwd("……..")

getwd()
list.files()

# Lettura file (funziona sia per Excel che CSV)                    -> INSERIRE DIRECTORY FILE
EXCEL
file_name <- "…… .csv"

if(grepl("\\.xlsx$", file_name)){
  dataframe <- read_excel(file_name)
} else {
  dataframe <- read.csv(file_name)
}

# Uniformare nomi colonne / pulizia nome colonne
names(dataframe) <- tolower(names(dataframe))


# Rinomina colonne realmente presenti nel file
dataframe <- dataframe %>%
  rename(
    year = matches("year|anno"),
    month = matches("month|mese"),
    day = matches("day|giorno"),
    prec = matches("prec|rain|precipit"),
    tempmax = matches("tmax|tempmax|max"),
    tempmin = matches("tmin|tempmin|min")
  )


# Costruzione variabile data
dataframe$vdata <- as.Date(with(dataframe, paste(year, month, day, sep = "-")))

# Ordinamento per data
dataframe <- dataframe %>% arrange(vdata)

# Controllo qualità dati
summary(dataframe)
colSums(is.na(dataframe))
```

```
#--------------------------
# GESTIONE VALORI MANCANTI:
#--------------------------

# Sostituzione valori mancanti codificati (-99.9) con NA - per tempmin
dataframe$tempmin[dataframe$tempmin == -99.9] <- NA
summary(dataframe$tempmin)

# Sostituzione valori mancanti codificati (-99.9) con NA - per tempmax
dataframe$tempmax[dataframe$tempmax == -99.9] <- NA
summary(dataframe$tempmax)

# Sostituzione valori mancanti codificati (-99.9) con NA - per precipitazioni
dataframe$prec[dataframe$prec == -99.9] <- NA
summary(dataframe$prec)

# Ripristino lunghezza dei vettori dopo l'introduzione dei valori NA
mod <- lm(tempmin ~ vdata,
          data = dataframe,
          na.action = na.exclude)

dataframe$trend_lin <- predict(mod, newdata = dataframe)

# Correzzione dei dati di temperature massime con valori fuoriscala (>1000)
dataframe$tempmax[is.na(dataframe$tempmax)==F & dataframe$tempmax>1000] <-
  dataframe$tempmax[is.na(dataframe$tempmax)==F & dataframe$tempmax>1000]/1000

plot(dataframe$tempmax)

#-----------------------------------------------
# CALCOLO TEMPERATURA MEDIA
#-----------------------------------------------
dataframe$tempmedia <- (dataframe$tempmax + dataframe$tempmin)/2


################################################################
# 1. PARAMETRI ESTETICI GRAFICI
################################################################

# DISTANZA NOMI ASSI DAGLI ASSI
margine_nome_asse_x <- 20
margine_nome_asse_y <- 20

# DIMENSIONI GRAFICI
larghezza_grafico <- 24
altezza_grafico <- 10
dpi_grafico <- 300

# COLORI PER FAMIGLIA DI GRAFICI
# Serie
colore_quadrati_serie <- "#D9D9D9"
colore_griglia_serie <- "white"
colore_titolo_serie <- "#848484"

# Trend lineare
colore_quadrati_trend_lin <- "#d8daee"
colore_griglia_trend_lin <- "white"
colore_titolo_trend_lin <- "#a82d2d"
```

```
# Trend polinomiale - Colore di default se non specificato il grado
colore_quadrati_poly <- "#d7e0ee"
colore_griglia_poly <- "white"
colore_titolo_poly <- "#045cea"

# Trend polinomiale - grado 2
colore_quadrati_poly_2 <- "#d7e5ee"
colore_quadriglia_poly_2 <- "white"
colore_titolo_poly_2 <- "#045cea"

# Trend polinomiale - grado 3
colore_quadrati_poly_3 <- "#d7e9ee"
colore_griglia_poly_3 <- "white"
colore_titolo_poly_3 <- "#045cea"

# Trend polinomiale - grado 4
colore_quadrati_poly_4 <- "#d7ecee"
colore_griglia_poly_4 <- "white"
colore_titolo_poly_4 <- "#045cea"

# Trend polinomiale - grado 5
colore_quadrati_poly_5 <- "#d7eeee"
colore_griglia_poly_5 <- "white"
colore_titolo_poly_5 <- "#045cea"

# ACF
# Colore barre ACF
colore_acf <- "gray"

colore_quadrati_acf <- "#f0d6d6"
colore_griglia_acf <- "white"
colore_titolo_acf <- "#8a2f2f"

# RESIDUI                                                    -> EXTRA
# Colore barre residui
colore_residui <- "gray"

colore_quadrati_residui <- "#f0e2d4"
colore_griglia_residui <- "white"
colore_titolo_residui <- "#8a6235"

# TEMA BASE
tema_base <- theme_grey() +
  theme(
    # Sfondo esterno grafico
    plot.background = element_rect(fill = "white", color = NA),
    # Sfondo pannello grafico
    panel.background = element_rect(fill = "white", color = NA),
    # Assi
    axis.line = element_line(color = "black"),
    # Nome asse X
    axis.title.x = element_text(margin = margin(t = margine_nome_asse_x)),
    # Nome asse Y
    axis.title.y = element_text(margin = margin(r = margine_nome_asse_y)),
    # Valori sugli assi
    axis.text = ggtext::element_markdown(
      size = 11,
      color = "black"
    ),
```

```
    axis.text.x = ggtext::element_markdown(),
    # Titolo grafico
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5)
  )

# TEMA PER FAMIGLIE
# Serie
tema_serie <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_serie),
        panel.grid.major = element_line(color = colore_griglia_serie),
        panel.grid.minor = element_line(color = colore_griglia_serie),
        plot.title = element_text(color = colore_titolo_serie))

# Trend lineare
tema_trend_lin <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_trend_lin),
        panel.grid.major = element_line(color = colore_griglia_trend_lin),
        panel.grid.minor = element_line(color = colore_griglia_trend_lin),
        plot.title = element_text(color = colore_titolo_trend_lin))

# Trend polinomiale base
tema_trend_poly <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_poly))

# Trend polinomiale per gradi
tema_trend_poly_2 <- tema_trend_poly
tema_trend_poly_3 <- tema_trend_poly
tema_trend_poly_4 <- tema_trend_poly
tema_trend_poly_5 <- tema_trend_poly

# Autocorrelazione
tema_acf <- tema_base +
  theme(

    panel.background = element_rect(
      fill = colore_quadrati_acf,
      color = NA
    ),

    panel.grid.major = element_line(
      color = colore_griglia_acf,
      linewidth = 0.5
    ),

    panel.grid.minor = element_line(
      color = colore_griglia_acf,
      linewidth = 0.25
    ),

    plot.title = element_text(
      color = colore_titolo_acf
    )
  )


# Residui
tema_residui <- tema_base +
  theme(

    panel.background = element_rect(
```

```
    fill = colore_quadrati_residui,
    color = NA
   ),

   panel.grid.major = element_line(
    color = colore_griglia_residui,
    linewidth = 0.5
   ),

   panel.grid.minor = element_line(
    color = colore_griglia_residui,
    linewidth = 0.25
   ),

   plot.title = element_text(
    color = colore_titolo_residui
   )
  )


#ASSE X PERSONALIZZATO
scala_asse_x_50anni <- function(vettore_date){

  vettore_date <- as.Date(vettore_date)

  anno_min <- year(min(vettore_date, na.rm = TRUE))
  anno_max <- year(max(vettore_date, na.rm = TRUE))

  breaks_finali <- sort(unique(c(
   seq(floor(anno_min/50)*50, ceiling(anno_max/50)*50, by = 50),
   anno_min,
   anno_max
  )))

  scale_x_date(
   breaks = as.Date(paste0(breaks_finali, "-01-01")),
   labels = function(x){

    anni <- format(x, "%Y")

    ifelse(
     anni %in% c(as.character(anno_min), as.character(anno_max)),
     paste0("<span style='color:#4D4D4D'>", anni, "</span>"),
     anni
    )
   },
   guide = guide_axis(n.dodge = 1)     #(se cambio 1 con 2 mi sfalsa i valori dei numeri sull'asse)
  )
}

#SETUP PER PLOT AUTOMATICO DEI GRAFICI IN PDF E PNG:
# Cartella output: Creo la cartella all'interno della quale salverò tutti i miei grafici
dir.create("grafici", showWarnings = FALSE)

# Contatore automatico dei grafici: Assegno un prefisso numerico al nome con cui vengono salvati i grafici
per ordine e per evitare sovrascrizioni
counter <- 1

#------------------------------------------------------------
# TABELLA RISULTATI TREND LINEARI
```

```
# Dataframe vuoto per salvare:
# - titolo grafico
# - pendenza
# - errore standard

tabella_trend <- data.frame(
  grafico = character(),
  pendenza = numeric(),
  errore_standard = numeric(),
  stringsAsFactors = FALSE
)
#------------------------------------------------------------

nome_progressivo <- function(nome){
  prefisso <- sprintf("%02d_", counter)
  assign("counter", counter + 1, envir = .GlobalEnv)
  return(paste0(prefisso, nome))
}

# Funzione salvataggio dei grafici
plot_e_salva <- function(grafico, nome_file){

  nome_file <- nome_progressivo(nome_file)

  print(grafico)

  ggsave(paste0("grafici/", nome_file, ".png"),
         plot = grafico,
         width = larghezza_grafico,
         height = altezza_grafico,
         dpi = dpi_grafico)

  ggsave(paste0("grafici/", nome_file, ".pdf"),
         plot = grafico,
         width = larghezza_grafico,
         height = altezza_grafico)
}


################################################################
# 2. ANALISI GIORNALIERA
################################################################

# Funzione per nomi asse Y in italiano
nome_variabile_plot <- function(var){

  if(var == "tempmax"){
    return("TEMPERATURA MASSIMA (°C)")
  }

  if(var == "tempmin"){
    return("TEMPERATURA MINIMA (°C)")
  }

  if(var == "tempmedia"){
    return("TEMPERATURA MEDIA (°C)")
  }

  if(var == "prec"){
    return("PRECIPITAZIONE (mm)")
```

```
  }

  return(toupper(gsub("_", "-", var)))
}

analisi_giornaliera <- function(dataframe, variabile, nome){

  nome_y <- nome_variabile_plot(variabile)


  # Serie-plot esplorativo
  g1 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie giornaliera -", nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_serie

  plot_e_salva(g1, paste0(variabile, "_serie"))


  # Trend lineare
  modello_lin <- lm(dataframe[[variabile]] ~ as.numeric(dataframe$vdata),
                    na.action = na.exclude)

  dataframe$trend_lin <- predict(modello_lin)

  g2 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_lin),
              color = "#a82d2d",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend lineare -", nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_trend_lin

  plot_e_salva(g2,
               paste0(variabile, "_trend_lin"))

  # Calcolo pendenza ed errore standard del trend lineare
  pendenza_giornaliera <- coef(modello_lin)[2]
  errore_standard_giornaliero <- summary(modello_lin)$coefficients[2,2]
  pendenza_annua <- pendenza_giornaliera * 365.25
  errore_standard_annuo <- errore_standard_giornaliero * 365.25
```

```
print (pendenza_giornaliera)
print (errore_standard_giornaliero)
print(pendenza_annua)
print(errore_standard_annuo)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE IN TABELLA
tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste("Trend lineare -", nome),
    pendenza = pendenza_annua,
    errore_standard = errore_standard_annuo
  )
)
#------------------------------------------------------------

# Trend polinomiale (da grado 2 a grado 5)
for(grado in 2:5){

  #(per colorare diversamente i quadrati di lo sfondo dei grafici dei diversi gradi)
  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  modello_poly <- lm(dataframe[[variabile]] ~
                       poly(as.numeric(dataframe$vdata), grado),
                     na.action = na.exclude)

  dataframe$trend_poly <- predict(modello_poly)

  g_poly <- ggplot(dataframe,
                   aes(x = vdata,
                       y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_poly),
              color = "#045cea",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend polinomiale grado",
                    grado,
                    "-",
                    nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_trend_poly_scorrimento

  plot_e_salva(g_poly,
```

```
            paste0(variabile,
                   "_trend_poly_grado_",
                   grado))
  }
}

# Applicazioni
analisi_giornaliera(dataframe, "tempmax", "Temperatura Massima")
analisi_giornaliera(dataframe, "tempmin", "Temperatura Minima")
analisi_giornaliera(dataframe, "tempmedia", "Temperatura Media")
analisi_giornaliera(dataframe, "prec", "Precipitazioni")


################################################################
# 3. AGGREGAZIONI
################################################################

# Variabile anno-mese
dataframe$year_month <- as.yearmon(dataframe$vdata)

# AGGREGAZIONE MENSILE
mensile <- dataframe %>%
  group_by(year_month) %>%
  summarise(
    tempmax = mean(tempmax, na.rm = TRUE),
    tempmin = mean(tempmin, na.rm = TRUE),
    tempmedia = mean(tempmedia, na.rm = TRUE),
    prec = sum(prec, na.rm = TRUE)
  )

# Variabile tempo numerica (per regressioni)
mensile$time <- as.numeric(mensile$year_month)

# Conversione corretta per grafici (yearmon -> Date)
mensile$date <- as.Date(mensile$year_month, frac = 0)


#-----------------------------------------------
# GRAFICI MENSILI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){

  nome_y <- nome_variabile_plot(variabile)

  # Serie
  g_mensile <- ggplot(mensile,
                      aes(x = date,
                          y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie mensile -", nome_y)
    ) +

    scala_asse_x_50anni(mensile$date) +
```

```
  tema_serie

plot_e_salva(g_mensile,
             paste0("mensile_serie_", variabile))


# Trend lineare
lm_mensile <- lm(as.formula(paste0(variabile, " ~ time")),
                 data = mensile,
                 na.action = na.exclude)

mensile$trend_lin <- predict(lm_mensile)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE MENSILE IN TABELLA
  pendenza_mensile <- coef(lm_mensile)[2]

errore_standard_mensile <- summary(
  lm_mensile
)$coefficients[2,2]

tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste(
      "Trend lineare mensile -",
      nome_y
    ),

    pendenza = pendenza_mensile,

    errore_standard = errore_standard_mensile
  )
)
#------------------------------------------------------------

  g_mensile_lin <- ggplot(mensile,
                          aes(x = date,
                              y = .data[[variabile]])) +

  geom_line() +

  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare mensile -", nome_y)
  ) +

  scala_asse_x_50anni(mensile$date) +

  tema_trend_lin

plot_e_salva(g_mensile_lin,
             paste0("mensile_trend_lineare_", variabile))
```

```
  # Trend polinomiale
  for(grado in 2:5){

    tema_trend_poly_scorrimento <- switch(
      as.character(grado),
      "2" = tema_trend_poly_2,
      "3" = tema_trend_poly_3,
      "4" = tema_trend_poly_4,
      "5" = tema_trend_poly_5,
      tema_trend_poly
    )

    lm_poly <- lm(as.formula(
      paste0(variabile, " ~ poly(time, ", grado, ")")
    ),
    data = mensile,
    na.action = na.exclude)

    mensile$trend_poly <- predict(lm_poly)

    g_poly <- ggplot(mensile,
                     aes(x = date,
                         y = .data[[variabile]])) +

      geom_line() +

      geom_line(aes(y = trend_poly),
                color = "#045cea",
                linewidth = 1.5) +

      labs(
        x = "ANNO",
        y = nome_y,
        title = paste("Trend polinomiale mensile grado", grado, "-", nome_y)
      ) +

      scala_asse_x_50anni(mensile$date) +

      tema_trend_poly_scorrimento

    plot_e_salva(g_poly,
                 paste0("mensile_trend_poly_grado_", grado, "_", variabile))
  }
}


# AGGREGAZIONE ANNUALE
annuale <- dataframe %>%
  group_by(year) %>%
  summarise(
    tempmax = mean(tempmax, na.rm = TRUE),
    tempmin = mean(tempmin, na.rm = TRUE),
    tempmedia = mean(tempmedia, na.rm = TRUE),
    prec = sum(prec, na.rm = TRUE)
  )

# Tempo centrato
annuale$year_c <- annuale$year - mean(annuale$year)
```

```
#-----------------------------------------------
# GRAFICI ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){

 nome_y <- nome_variabile_plot(variabile)

 # Serie
 g_annuale <- ggplot(annuale,
                     aes(x = year,
                         y = .data[[variabile]])) +

  geom_line() +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Serie annuale -", nome_y)
  ) +

  scale_x_continuous(
    breaks = sort(unique(c(
      seq(
        floor(min(annuale$year) / 50) * 50,
        ceiling(max(annuale$year) / 50) * 50,
        by = 50
      ),
      min(annuale$year),
      max(annuale$year)
    )))
  ) +

  tema_serie

 plot_e_salva(g_annuale,
              paste0("annuale_serie_", variabile))


 # Trend lineare
 lm_annuale <- lm(as.formula(paste0(variabile, " ~ year_c")),
                  data = annuale)

 annuale$trend_lin <- predict(lm_annuale)

 #-------------------------------------------------------
 # SALVATAGGIO RISULTATI TREND LINEARE ANNUALE IN TABELLA
  pendenza_annuale <- coef(lm_annuale)[2]

 errore_standard_annuale <- summary(
   lm_annuale
 )$coefficients[2,2]

 tabella_trend <<- rbind(
   tabella_trend,

   data.frame(
     grafico = paste(
       "Trend lineare annuale -",
```

```
      nome_y
    ),

    pendenza = pendenza_annuale,

    errore_standard = errore_standard_annuale
  )
)
#-----------------------------------------------------------

g_annuale_lin <- ggplot(annuale,
                        aes(x = year,
                            y = .data[[variabile]])) +

  geom_line() +

  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare annuale -", nome_y)
  ) +

  scale_x_continuous(
    breaks = sort(unique(c(
      seq(
        floor(min(annuale$year) / 50) * 50,
        ceiling(max(annuale$year) / 50) * 50,
        by = 50
      ),
      min(annuale$year),
      max(annuale$year)
    )))
  ) +

  tema_trend_lin

plot_e_salva(g_annuale_lin,
             paste0("annuale_trend_lineare_", variabile))


# Trend polinomiale
for(grado in 2:5){

  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  lm_poly_ann <- lm(as.formula(
    paste0(variabile, " ~ poly(year_c, ", grado, ")")
  ),
  data = annuale,
```

```
    na.action = na.exclude)

    annuale$trend_poly <- predict(lm_poly_ann)

    g_poly_ann <- ggplot(annuale,
                         aes(x = year,
                             y = .data[[variabile]])) +

      geom_line() +

      geom_line(aes(y = trend_poly),
                color = "#045cea",
                linewidth = 1.5) +

      labs(
        x = "ANNO",
        y = nome_y,
        title = paste("Trend polinomiale annuale grado", grado, "-", nome_y)
      ) +

      scale_x_continuous(
        breaks = sort(unique(c(
          seq(
            floor(min(annuale$year) / 50) * 50,
            ceiling(max(annuale$year) / 50) * 50,
            by = 50
          ),
          min(annuale$year),
          max(annuale$year)
        )))
      ) +

      tema_trend_poly_scorrimento

    plot_e_salva(g_poly_ann,
                 paste0("annuale_trend_poly_grado_", grado, "_", variabile))
  }
}

################################################################
# 4. AUTOCORRELAZIONE DELLE SERIE
################################################################

#-----------------------------------------------
# FUNZIONE AUTOCORRELAZIONE
#-----------------------------------------------

acf_salva <- function(serie,
                      nome,
                      tema_grafico,
                      colore_barre_positive = "gray",
                      colore_barre_negative = "gray30",
                      nome_lag = "LAG"){

  # Calcolo autocorrelazione
  acf_obj <- acf(
    serie,
    plot = FALSE,
    na.action = na.pass
  )
```

```
# Creazione dataframe ACF
df_acf <- data.frame(
  lag = as.numeric(acf_obj$lag),
  acf = as.numeric(acf_obj$acf)
)

# Colore barre positivo/negativo
df_acf$colore_barre <- ifelse(
  df_acf$acf >= 0,
  colore_barre_positive,
  colore_barre_negative
)

# Nome progressivo SOLO per file
nome_file <- nome_progressivo(nome)

# Grafico ACF con ggplot
g_acf <- ggplot(
  df_acf,
  aes(x = lag,
      y = acf)
) +

  geom_col(
    aes(fill = colore_barre),
    show.legend = FALSE
  ) +

  scale_fill_identity() +

  geom_hline(
    yintercept = 0,
    color = "black"
  ) +

  labs(
    title = nome,
    x = nome_lag,
    y = "AUTOCORRELAZIONE"
  ) +

  tema_grafico


# Salvataggio PNG
ggsave(
  paste0("grafici/", nome_file, ".png"),
  plot = g_acf,
  width = larghezza_grafico,
  height = altezza_grafico,
  dpi = dpi_grafico
)

# Salvataggio PDF
ggsave(
  paste0("grafici/", nome_file, ".pdf"),
  plot = g_acf,
  width = larghezza_grafico,
  height = altezza_grafico
```

```
  )

  # Visualizzazione grafico
  print(g_acf)
}


#-----------------------------------------------
# AUTOCORRELAZIONE SERIE MENSILI
#-----------------------------------------------

for(variabile in c("tempmax",
                   "tempmin",
                   "tempmedia",
                   "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF mensile
  acf_salva(
    serie = mensile[[variabile]],

    nome = paste(
      "Autocorrelazione mensile -",
      nome_variabile
    ),

    tema_grafico = tema_acf,

    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG MENSILE"
  )
}


#-----------------------------------------------
# AUTOCORRELAZIONE SERIE ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax",
                   "tempmin",
                   "tempmedia",
                   "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF annuale
  acf_salva(
    serie = annuale[[variabile]],

    nome = paste(
      "Autocorrelazione annuale -",
      nome_variabile
    ),

    tema_grafico = tema_acf,
```

```
    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG ANNUALE"
  )
}


################################################################
# 6. ESPORTAZIONE TABELLA TREND LINEARI
################################################################

# Libreria per esportazione tabella PDF
library(gridExtra)
library(grid)

# Creazione oggetto tabella
tabella_grafica <- tableGrob(
  tabella_trend,
  rows = NULL
)

# Esportazione PDF
pdf(
  "grafici/tabella_trend_lineari.pdf",
  width = 14,
  height = 8
)

grid.draw(tabella_grafica)

dev.off()
```

Torino – Stazione di Torino Giardini Reali

```
################################################################
# 0. SETUP
################################################################

# Pulizia ambiente
rm(list = ls())

# Librerie necessarie
library(readxl)
library(dplyr)
library(lubridate)
library(ggplot2)
library(forecast)
library(zoo)
library(ggtext)

# Definire working directory
setwd("…….. ")

getwd()
list.files()

#---------------------------------------------------------
# LETTURA FILE
#---------------------------------------------------------

file_name <- "…….. .csv"

if(grepl("\\.xlsx$", file_name)){
  dataframe <- read_excel(file_name)
} else {

  dataframe <- read.csv(
    file_name,
    stringsAsFactors = FALSE,
    check.names = FALSE
  )
}

#---------------------------------------------------------
# PULIZIA NOMI COLONNE
#---------------------------------------------------------

names(dataframe) <- trimws(names(dataframe))

# Verifica colonne presenti
print(names(dataframe))

#---------------------------------------------------------
# RINOMINA COLONNE DEL NUOVO FORMATO
#---------------------------------------------------------

dataframe <- dataframe %>%
  rename(
    vdata = `DATA`,
    tempmax = `Temperatura massima omogenea (°C)`,
    tempmin = `Temperatura minima omogenea (°C)`
```

```
  )

# Imposto che tempmax e temp min siano lette come numeri e non testo
dataframe$tempmax <- as.numeric(dataframe$tempmax)
dataframe$tempmin <- as.numeric(dataframe$tempmin)

#---------------------------------------------------------
# GESTIONE DATA
#---------------------------------------------------------

dataframe$vdata <- as.Date(dataframe$vdata)

# Creazione variabili richieste dal resto dello script
dataframe$year  <- year(dataframe$vdata)
dataframe$month <- month(dataframe$vdata)
dataframe$day   <- day(dataframe$vdata)

#---------------------------------------------------------
# ORDINAMENTO
#---------------------------------------------------------

dataframe <- dataframe %>%
  arrange(vdata)

#---------------------------------------------------------
# PRECIPITAZIONI
#---------------------------------------------------------
# Se il dataset non contiene precipitazioni:

dataframe$prec <- 1

#---------------------------------------------------------
# CONTROLLO QUALITA'
#---------------------------------------------------------

summary(dataframe)
colSums(is.na(dataframe))

#---------------------------------------------------------
# GESTIONE VALORI MANCANTI
#---------------------------------------------------------

# Eventuali codifiche mancanti
dataframe$tempmin[dataframe$tempmin == -99.9] <- NA
dataframe$tempmax[dataframe$tempmax == -99.9] <- NA

summary(dataframe$tempmin)
summary(dataframe$tempmax)

#---------------------------------------------------------
# TREND PRELIMINARE (utilizzato in seguito)
#---------------------------------------------------------

mod <- lm(
  tempmin ~ vdata,
  data = dataframe,
  na.action = na.exclude
)

dataframe$trend_lin <- predict(
```

```
  mod,
  newdata = dataframe
)


#------------------------------------------------------------
# TEMPERATURA MEDIA
#------------------------------------------------------------

dataframe$tempmedia <-
  (dataframe$tempmax + dataframe$tempmin) / 2


################################################################
# 1. PARAMETRI ESTETICI GRAFICI
################################################################

# DISTANZA NOMI ASSI DAGLI ASSI
margine_nome_asse_x <- 20
margine_nome_asse_y <- 20

# DIMENSIONI GRAFICI
larghezza_grafico <- 24
altezza_grafico <- 10
dpi_grafico <- 300

# COLORI PER FAMIGLIA DI GRAFICI
# Serie
colore_quadrati_serie <- "#D9D9D9"
colore_griglia_serie <- "white"
colore_titolo_serie <- "#848484"

# Trend lineare
colore_quadrati_trend_lin <- "#d8daee"
colore_griglia_trend_lin <- "white"
colore_titolo_trend_lin <- "#a82d2d"

# Trend polinomiale - Colore di default se non specificato il grado
colore_quadrati_poly <- "#d7e0ee"
colore_griglia_poly <- "white"
colore_titolo_poly <- "#045cea"

# Trend polinomiale - grado 2
colore_quadrati_poly_2 <- "#d7e5ee"
colore_quadriglia_poly_2 <- "white"
colore_titolo_poly_2 <- "#045cea"

# Trend polinomiale - grado 3
colore_quadrati_poly_3 <- "#d7e9ee"
colore_griglia_poly_3 <- "white"
colore_titolo_poly_3 <- "#045cea"

# Trend polinomiale - grado 4
colore_quadrati_poly_4 <- "#d7ecee"
colore_griglia_poly_4 <- "white"
colore_titolo_poly_4 <- "#045cea"

# Trend polinomiale - grado 5
colore_quadrati_poly_5 <- "#d7eeee"
colore_griglia_poly_5 <- "white"
```

```
colore_titolo_poly_5 <- "#045cea"

# ACF
# Colore barre ACF
colore_acf <- "gray"

colore_quadrati_acf <- "#f0d6d6"
colore_griglia_acf <- "white"
colore_titolo_acf <- "#8a2f2f"

# RESIDUI                                                    -> EXTRA
# Colore barre residui
colore_residui <- "gray"

colore_quadrati_residui <- "#f0e2d4"
colore_griglia_residui <- "white"
colore_titolo_residui <- "#8a6235"

# TEMA BASE
tema_base <- theme_grey() +
  theme(
    # Sfondo esterno grafico
    plot.background = element_rect(fill = "white", color = NA),
    # Sfondo pannello grafico
    panel.background = element_rect(fill = "white", color = NA),
    # Assi
    axis.line = element_line(color = "black"),
    # Nome asse X
    axis.title.x = element_text(margin = margin(t = margine_nome_asse_x)),
    # Nome asse Y
    axis.title.y = element_text(margin = margin(r = margine_nome_asse_y)),
    # Valori sugli assi
    axis.text = ggtext::element_markdown(
      size = 11,
      color = "black"
    ),
    axis.text.x = ggtext::element_markdown(),
    # Titolo grafico
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5)
  )

# TEMA PER FAMIGLIE
# Serie
tema_serie <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_serie),
        panel.grid.major = element_line(color = colore_griglia_serie),
        panel.grid.minor = element_line(color = colore_griglia_serie),
        plot.title = element_text(color = colore_titolo_serie))

# Trend lineare
tema_trend_lin <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_trend_lin),
        panel.grid.major = element_line(color = colore_griglia_trend_lin),
        panel.grid.minor = element_line(color = colore_griglia_trend_lin),
        plot.title = element_text(color = colore_titolo_trend_lin))

# Trend polinomiale base
tema_trend_poly <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_poly))
```

```
# Trend polinomiale per gradi
tema_trend_poly_2 <- tema_trend_poly
tema_trend_poly_3 <- tema_trend_poly
tema_trend_poly_4 <- tema_trend_poly
tema_trend_poly_5 <- tema_trend_poly

# Autocorrelazione
tema_acf <- tema_base +
  theme(

    panel.background = element_rect(
      fill = colore_quadrati_acf,
      color = NA
    ),

    panel.grid.major = element_line(
      color = colore_griglia_acf,
      linewidth = 0.5
    ),

    panel.grid.minor = element_line(
      color = colore_griglia_acf,
      linewidth = 0.25
    ),

    plot.title = element_text(
      color = colore_titolo_acf
    )
  )


# Residui
tema_residui <- tema_base +
  theme(

    panel.background = element_rect(
      fill = colore_quadrati_residui,
      color = NA
    ),

    panel.grid.major = element_line(
      color = colore_griglia_residui,
      linewidth = 0.5
    ),

    panel.grid.minor = element_line(
      color = colore_griglia_residui,
      linewidth = 0.25
    ),

    plot.title = element_text(
      color = colore_titolo_residui
    )
  )


#ASSE X PERSONALIZZATO
scala_asse_x_50anni <- function(vettore_date){

  vettore_date <- as.Date(vettore_date)
```

```
  anno_min <- year(min(vettore_date, na.rm = TRUE))
  anno_max <- year(max(vettore_date, na.rm = TRUE))

  breaks_finali <- sort(unique(c(
    seq(floor(anno_min/50)*50, ceiling(anno_max/50)*50, by = 50),
    anno_min,
    anno_max
  )))

  scale_x_date(
    breaks = as.Date(paste0(breaks_finali, "-01-01")),
    labels = function(x){

      anni <- format(x, "%Y")

      ifelse(
        anni %in% c(as.character(anno_min), as.character(anno_max)),
        paste0("<span style='color:#4D4D4D'>", anni, "</span>"),
        anni
      )
    },
    guide = guide_axis(n.dodge = 1)     #(se cambio 1 con 2 mi sfalsa i valori dei numeri sull'asse)
  )
}

#SETUP PER PLOT AUTOMATICO DEI GRAFICI IN PDF E PNG:
# Cartella output: Creo la cartella all'interno della quale salverò tutti i miei grafici
dir.create("grafici", showWarnings = FALSE)

# Contatore automatico dei grafici: Assegno un prefisso numerico al nome con cui vengono salvati i grafici
per ordine e per evitare sovrascrizioni
counter <- 1

#------------------------------------------------------------
# TABELLA RISULTATI TREND LINEARI
# Dataframe vuoto per salvare:
# - titolo grafico
# - pendenza
# - errore standard

tabella_trend <- data.frame(
  grafico = character(),
  pendenza = numeric(),
  errore_standard = numeric(),
  stringsAsFactors = FALSE
)
#------------------------------------------------------------

nome_progressivo <- function(nome){
  prefisso <- sprintf("%02d_", counter)
  assign("counter", counter + 1, envir = .GlobalEnv)
  return(paste0(prefisso, nome))
}

# Funzione salvataggio dei grafici
plot_e_salva <- function(grafico, nome_file){

  nome_file <- nome_progressivo(nome_file)
```

```
  print(grafico)

  ggsave(paste0("grafici/", nome_file, ".png"),
         plot = grafico,
         width = larghezza_grafico,
         height = altezza_grafico,
         dpi = dpi_grafico)

  ggsave(paste0("grafici/", nome_file, ".pdf"),
         plot = grafico,
         width = larghezza_grafico,
         height = altezza_grafico)
}


################################################################
# 2. ANALISI GIORNALIERA
################################################################

# Funzione per nomi asse Y in italiano
nome_variabile_plot <- function(var){

  if(var == "tempmax"){
    return("TEMPERATURA MASSIMA (°C)")
  }

  if(var == "tempmin"){
    return("TEMPERATURA MINIMA (°C)")
  }

  if(var == "tempmedia"){
    return("TEMPERATURA MEDIA (°C)")
  }

  if(var == "prec"){
    return("PRECIPITAZIONE (mm)")
  }

  return(toupper(gsub("_", "-", var)))
}

analisi_giornaliera <- function(dataframe, variabile, nome){

  nome_y <- nome_variabile_plot(variabile)


  # Serie-plot esplorativo
  g1 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie giornaliera -", nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_serie
```

```
plot_e_salva(g1, paste0(variabile, "_serie"))


# Trend lineare
modello_lin <- lm(dataframe[[variabile]] ~ as.numeric(dataframe$vdata),
                  na.action = na.exclude)

dataframe$trend_lin <- predict(modello_lin)

g2 <- ggplot(dataframe,
             aes(x = vdata,
                 y = .data[[variabile]])) +

  geom_line() +

  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare -", nome)
  ) +
  scala_asse_x_50anni(dataframe$vdata) +
  tema_trend_lin

plot_e_salva(g2,
             paste0(variabile, "_trend_lin"))

# Calcolo pendenza ed errore standard del trend lineare
pendenza_giornaliera <- coef(modello_lin)[2]
errore_standard_giornaliero <- summary(modello_lin)$coefficients[2,2]
pendenza_annua <- pendenza_giornaliera * 365.25
errore_standard_annuo <- errore_standard_giornaliero * 365.25

print (pendenza_giornaliera)
print (errore_standard_giornaliero)
print(pendenza_annua)
print(errore_standard_annuo)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE IN TABELLA
tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste("Trend lineare -", nome),
    pendenza = pendenza_annua,
    errore_standard = errore_standard_annuo
  )
)
#------------------------------------------------------------

# Trend polinomiale (da grado 2 a grado 5)
for(grado in 2:5){

  #(per colorare diversamente i quadrati di lo sfondo dei grafici dei diversi gradi)
  tema_trend_poly_scorrimento <- switch(
```

```
      as.character(grado),
      "2" = tema_trend_poly_2,
      "3" = tema_trend_poly_3,
      "4" = tema_trend_poly_4,
      "5" = tema_trend_poly_5,
      tema_trend_poly
    )

    modello_poly <- lm(dataframe[[variabile]] ~
                         poly(as.numeric(dataframe$vdata), grado),
                       na.action = na.exclude)

    dataframe$trend_poly <- predict(modello_poly)

    g_poly <- ggplot(dataframe,
                     aes(x = vdata,
                         y = .data[[variabile]])) +

      geom_line() +

      geom_line(aes(y = trend_poly),
                color = "#045cea",
                linewidth = 1.5) +

      labs(
        x = "ANNO",
        y = nome_y,
        title = paste("Trend polinomiale grado",
                      grado,
                      "-",
                      nome)
      ) +
      scala_asse_x_50anni(dataframe$vdata) +
      tema_trend_poly_scorrimento

    plot_e_salva(g_poly,
                 paste0(variabile,
                        "_trend_poly_grado_",
                        grado))
  }
}

# Applicazioni
analisi_giornaliera(dataframe, "tempmax", "Temperatura Massima")
analisi_giornaliera(dataframe, "tempmin", "Temperatura Minima")
analisi_giornaliera(dataframe, "tempmedia", "Temperatura Media")
analisi_giornaliera(dataframe, "prec", "Precipitazioni")


################################################################
# 3. AGGREGAZIONI
################################################################

# Variabile anno-mese
dataframe$year_month <- as.yearmon(dataframe$vdata)

# AGGREGAZIONE MENSILE
mensile <- dataframe %>%
  group_by(year_month) %>%
  summarise(
```

```
  tempmax = mean(tempmax, na.rm = TRUE),
  tempmin = mean(tempmin, na.rm = TRUE),
  tempmedia = mean(tempmedia, na.rm = TRUE),
  prec = sum(prec, na.rm = TRUE)
 )

# Variabile tempo numerica (per regressioni)
mensile$time <- as.numeric(mensile$year_month)

# Conversione corretta per grafici (yearmon -> Date)
mensile$date <- as.Date(mensile$year_month, frac = 0)


#------------------------------------------------
# GRAFICI MENSILI
#------------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){

 nome_y <- nome_variabile_plot(variabile)

 # Serie
 g_mensile <- ggplot(mensile,
                     aes(x = date,
                         y = .data[[variabile]])) +

  geom_line() +

  labs(
   x = "ANNO",
   y = nome_y,
   title = paste("Serie mensile -", nome_y)
  ) +

  scala_asse_x_50anni(mensile$date) +

  tema_serie

 plot_e_salva(g_mensile,
              paste0("mensile_serie_", variabile))


 # Trend lineare
 lm_mensile <- lm(as.formula(paste0(variabile, " ~ time")),
                  data = mensile,
                  na.action = na.exclude)

 mensile$trend_lin <- predict(lm_mensile)

 #------------------------------------------------------------
 # SALVATAGGIO RISULTATI TREND LINEARE MENSILE IN TABELLA
 pendenza_mensile <- coef(lm_mensile)[2]

 errore_standard_mensile <- summary(
  lm_mensile
 )$coefficients[2,2]

 tabella_trend <<- rbind(
  tabella_trend,
```

```
  data.frame(
    grafico = paste(
      "Trend lineare mensile -",
      nome_y
    ),

    pendenza = pendenza_mensile,

    errore_standard = errore_standard_mensile
  )
)
#-----------------------------------------------------------

g_mensile_lin <- ggplot(mensile,
                        aes(x = date,
                            y = .data[[variabile]])) +

  geom_line() +

  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare mensile -", nome_y)
  ) +

  scala_asse_x_50anni(mensile$date) +

  tema_trend_lin

plot_e_salva(g_mensile_lin,
             paste0("mensile_trend_lineare_", variabile))


# Trend polinomiale
for(grado in 2:5){

  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  lm_poly <- lm(as.formula(
    paste0(variabile, " ~ poly(time, ", grado, ")")
  ),
  data = mensile,
  na.action = na.exclude)

  mensile$trend_poly <- predict(lm_poly)

  g_poly <- ggplot(mensile,
                   aes(x = date,
                       y = .data[[variabile]])) +
```

```
    geom_line() +

    geom_line(aes(y = trend_poly),
              color = "#045cea",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend polinomiale mensile grado", grado, "-", nome_y)
    ) +

    scala_asse_x_50anni(mensile$date) +

    tema_trend_poly_scorrimento

  plot_e_salva(g_poly,
               paste0("mensile_trend_poly_grado_", grado, "_", variabile))
 }
}


# AGGREGAZIONE ANNUALE
annuale <- dataframe %>%
 group_by(year) %>%
 summarise(
  tempmax = mean(tempmax, na.rm = TRUE),
  tempmin = mean(tempmin, na.rm = TRUE),
  tempmedia = mean(tempmedia, na.rm = TRUE),
  prec = sum(prec, na.rm = TRUE)
 )

# Tempo centrato
annuale$year_c <- annuale$year - mean(annuale$year)


#-----------------------------------------------
# GRAFICI ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){

 nome_y <- nome_variabile_plot(variabile)

 # Serie
 g_annuale <- ggplot(annuale,
                     aes(x = year,
                         y = .data[[variabile]])) +

  geom_line() +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Serie annuale -", nome_y)
  ) +

  scale_x_continuous(
    breaks = sort(unique(c(
```

```
    seq(
      floor(min(annuale$year) / 50) * 50,
      ceiling(max(annuale$year) / 50) * 50,
      by = 50
    ),
    min(annuale$year),
    max(annuale$year)
  )))
 ) +

 tema_serie

plot_e_salva(g_annuale,
             paste0("annuale_serie_", variabile))


# Trend lineare
lm_annuale <- lm(as.formula(paste0(variabile, " ~ year_c")),
                 data = annuale)

annuale$trend_lin <- predict(lm_annuale)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE ANNUALE IN TABELLA
pendenza_annuale <- coef(lm_annuale)[2]

errore_standard_annuale <- summary(
 lm_annuale
)$coefficients[2,2]

tabella_trend <<- rbind(
 tabella_trend,

 data.frame(
   grafico = paste(
     "Trend lineare annuale -",
     nome_y
   ),

   pendenza = pendenza_annuale,

   errore_standard = errore_standard_annuale
 )
)
#------------------------------------------------------------

g_annuale_lin <- ggplot(annuale,
                        aes(x = year,
                            y = .data[[variabile]])) +

 geom_line() +

 geom_line(aes(y = trend_lin),
           color = "#a82d2d",
           linewidth = 1.5) +

 labs(
   x = "ANNO",
   y = nome_y,
   title = paste("Trend lineare annuale -", nome_y)
```

```
  ) +

  scale_x_continuous(
    breaks = sort(unique(c(
      seq(
        floor(min(annuale$year) / 50) * 50,
        ceiling(max(annuale$year) / 50) * 50,
        by = 50
      ),
      min(annuale$year),
      max(annuale$year)
    )))
  ) +

  tema_trend_lin

plot_e_salva(g_annuale_lin,
             paste0("annuale_trend_lineare_", variabile))


# Trend polinomiale
for(grado in 2:5){

  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  lm_poly_ann <- lm(as.formula(
    paste0(variabile, " ~ poly(year_c, ", grado, ")")
  ),
  data = annuale,
  na.action = na.exclude)

  annuale$trend_poly <- predict(lm_poly_ann)

  g_poly_ann <- ggplot(annuale,
                       aes(x = year,
                           y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_poly),
              color = "#045cea",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend polinomiale annuale grado", grado, "-", nome_y)
    ) +

    scale_x_continuous(
      breaks = sort(unique(c(
        seq(
          floor(min(annuale$year) / 50) * 50,
```

```
          ceiling(max(annuale$year) / 50) * 50,
          by = 50
        ),
        min(annuale$year),
        max(annuale$year)
      )))
    ) +

    tema_trend_poly_scorrimento

  plot_e_salva(g_poly_ann,
               paste0("annuale_trend_poly_grado_", grado, "_", variabile))
 }
}

################################################################
# 4. AUTOCORRELAZIONE DELLE SERIE
################################################################

#-----------------------------------------------
# FUNZIONE AUTOCORRELAZIONE
#-----------------------------------------------

acf_salva <- function(serie,
                      nome,
                      tema_grafico,
                      colore_barre_positive = "gray",
                      colore_barre_negative = "gray30",
                      nome_lag = "LAG"){

 # Calcolo autocorrelazione
 acf_obj <- acf(
  serie,
  plot = FALSE,
  na.action = na.pass
 )

 # Creazione dataframe ACF
 df_acf <- data.frame(
  lag = as.numeric(acf_obj$lag),
  acf = as.numeric(acf_obj$acf)
 )

 # Colore barre positivo/negativo
 df_acf$colore_barre <- ifelse(
  df_acf$acf >= 0,
  colore_barre_positive,
  colore_barre_negative
 )

 # Nome progressivo SOLO per file
 nome_file <- nome_progressivo(nome)

 # Grafico ACF con ggplot
 g_acf <- ggplot(
  df_acf,
  aes(x = lag,
      y = acf)
 ) +
```

```
  geom_col(
    aes(fill = colore_barre),
    show.legend = FALSE
  ) +

  scale_fill_identity() +

  geom_hline(
    yintercept = 0,
    color = "black"
  ) +

  labs(
    title = nome,
    x = nome_lag,
    y = "AUTOCORRELAZIONE"
  ) +

  tema_grafico


 # Salvataggio PNG
 ggsave(
   paste0("grafici/", nome_file, ".png"),
   plot = g_acf,
   width = larghezza_grafico,
   height = altezza_grafico,
   dpi = dpi_grafico
 )

 # Salvataggio PDF
 ggsave(
   paste0("grafici/", nome_file, ".pdf"),
   plot = g_acf,
   width = larghezza_grafico,
   height = altezza_grafico
 )

 # Visualizzazione grafico
 print(g_acf)
}


#-----------------------------------------------
# AUTOCORRELAZIONE SERIE MENSILI
#-----------------------------------------------

for(variabile in c("tempmax",
                   "tempmin",
                   "tempmedia",
                   "prec")){

 # Nome leggibile variabile
 nome_variabile <- nome_variabile_plot(variabile)

 # Creazione grafico ACF mensile
 acf_salva(
   serie = mensile[[variabile]],

   nome = paste(
```

```
    "Autocorrelazione mensile -",
    nome_variabile
   ),

   tema_grafico = tema_acf,

   colore_barre_positive = "gray",
   colore_barre_negative = "gray30",

   nome_lag = "LAG MENSILE"
  )
}


#-----------------------------------------------
# AUTOCORRELAZIONE SERIE ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax",
                   "tempmin",
                   "tempmedia",
                   "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF annuale
  acf_salva(
    serie = annuale[[variabile]],

    nome = paste(
      "Autocorrelazione annuale -",
      nome_variabile
    ),

    tema_grafico = tema_acf,

    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG ANNUALE"
  )
}


#################################################################
# 6. ESPORTAZIONE TABELLA TREND LINEARI
#################################################################

# Libreria per esportazione tabella PDF
library(gridExtra)
library(grid)

# Creazione oggetto tabella
tabella_grafica <- tableGrob(
  tabella_trend,
  rows = NULL
)

# Esportazione PDF
```

```
pdf(
  "grafici/tabella_trend_lineari.pdf",
  width = 14,
  height = 8
)

grid.draw(tabella_grafica)

dev.off()
```

Roma – Stazione di Roma Macao

```
################################################################
# 0. SETUP
################################################################

# Pulizia ambiente
rm(list = ls())

# Librerie necessarie
library(readxl)
library(dplyr)
library(lubridate)
library(ggplot2)
library(forecast)
library(zoo)
library(ggtext)
library(readr)

# Definire working directory
setwd("…….. ")

getwd()
list.files()

#---------------------------------------------------------
# LETTURA FILE
#---------------------------------------------------------

# PULIZIA DEL FILE CSV
raw <- readLines("…… .txt")

# rimuove virgolette esterne
raw <- gsub('^"|"$', '', raw)

file_name <- "file_pulito.csv"
writeLines(raw, file_name)

# Lettura file separato da ;
dataframe <- read.csv2(
  file_name,
  fileEncoding = "UTF-8-BOM",
  stringsAsFactors = FALSE,
  check.names = FALSE
)

#---------------------------------------------------------
# PULIZIA NOMI COLONNE
#---------------------------------------------------------
names(dataframe) <- trimws(names(dataframe))

# Verifica colonne presenti
print(names(dataframe))

#---------------------------------------------------------
# FILTRO STAZIONE
#---------------------------------------------------------
```

```
dataframe <- dataframe %>%
  filter(nome == "Roma Macao")

#------------------------------------------------------------
# CONVERSIONE VARIABILI NUMERICHE
#------------------------------------------------------------

dataframe$tmin <- as.numeric(dataframe$tmin)
dataframe$tmax <- as.numeric(dataframe$tmax)

dataframe$giorno <- as.integer(dataframe$giorno)
dataframe$mese   <- as.integer(dataframe$mese)
dataframe$anno   <- as.integer(dataframe$anno)

#------------------------------------------------------------
# CREAZIONE VARIABILI COMPATIBILI CON IL RESTO DELLO SCRIPT
#------------------------------------------------------------

dataframe <- dataframe %>%
  mutate(
    vdata = as.Date(
      sprintf(
        "%04d-%02d-%02d",
        anno,
        mese,
        giorno
      )
    ),

    tempmin = tmin,
    tempmax = tmax
  )

#------------------------------------------------------------
# GESTIONE DATA
#------------------------------------------------------------

dataframe$year  <- year(dataframe$vdata)
dataframe$month <- month(dataframe$vdata)
dataframe$day   <- day(dataframe$vdata)

#------------------------------------------------------------
# ORDINAMENTO
#------------------------------------------------------------

dataframe <- dataframe %>%
  arrange(vdata)

#------------------------------------------------------------
# PRECIPITAZIONI
#------------------------------------------------------------

# Variabile mantenuta per compatibilità con il resto dello script
dataframe$prec <- 1

#------------------------------------------------------------
# CONTROLLO QUALITA'
#------------------------------------------------------------
```

```
summary(dataframe)
colSums(is.na(dataframe))

#------------------------------------------------------------
# GESTIONE VALORI MANCANTI
#------------------------------------------------------------

# Eventuali codifiche mancanti
dataframe$tempmin[dataframe$tempmin == -99.9] <- NA
dataframe$tempmax[dataframe$tempmax == -99.9] <- NA

summary(dataframe$tempmin)
summary(dataframe$tempmax)

#------------------------------------------------------------
# TREND PRELIMINARE (utilizzato in seguito)
#------------------------------------------------------------

mod <- lm(
  tempmin ~ vdata,
  data = dataframe,
  na.action = na.exclude
)

dataframe$trend_lin <- predict(
  mod,
  newdata = dataframe
)

#------------------------------------------------------------
# TEMPERATURA MEDIA
#------------------------------------------------------------

dataframe$tempmedia <-
  (dataframe$tempmax + dataframe$tempmin) / 2


################################################################
# 1. PARAMETRI ESTETICI GRAFICI
################################################################

# DISTANZA NOMI ASSI DAGLI ASSI
margine_nome_asse_x <- 20
margine_nome_asse_y <- 20

# DIMENSIONI GRAFICI
larghezza_grafico <- 20
altezza_grafico <- 10
dpi_grafico <- 300

# COLORI PER FAMIGLIA DI GRAFICI
# Serie
colore_quadrati_serie <- "#D9D9D9"
colore_griglia_serie <- "white"
colore_titolo_serie <- "#848484"

# Trend lineare
colore_quadrati_trend_lin <- "#d8daee"
colore_griglia_trend_lin <- "white"
colore_titolo_trend_lin <- "#a82d2d"
```

```
# Trend polinomiale - Colore di default se non specificato il grado
colore_quadrati_poly <- "#d7e0ee"
colore_griglia_poly <- "white"
colore_titolo_poly <- "#045cea"

# Trend polinomiale - grado 2
colore_quadrati_poly_2 <- "#d7e5ee"
colore_quadriglia_poly_2 <- "white"
colore_titolo_poly_2 <- "#045cea"

# Trend polinomiale - grado 3
colore_quadrati_poly_3 <- "#d7e9ee"
colore_griglia_poly_3 <- "white"
colore_titolo_poly_3 <- "#045cea"

# Trend polinomiale - grado 4
colore_quadrati_poly_4 <- "#d7ecee"
colore_griglia_poly_4 <- "white"
colore_titolo_poly_4 <- "#045cea"

# Trend polinomiale - grado 5
colore_quadrati_poly_5 <- "#d7eeee"
colore_griglia_poly_5 <- "white"
colore_titolo_poly_5 <- "#045cea"

# ACF
# Colore barre ACF
colore_acf <- "gray"

colore_quadrati_acf <- "#f0d6d6"
colore_griglia_acf <- "white"
colore_titolo_acf <- "#8a2f2f"

# RESIDUI                                                    -> EXTRA
# Colore barre residui
colore_residui <- "gray"

colore_quadrati_residui <- "#f0e2d4"
colore_griglia_residui <- "white"
colore_titolo_residui <- "#8a6235"

# TEMA BASE
tema_base <- theme_grey() +
  theme(
    # Sfondo esterno grafico
    plot.background = element_rect(fill = "white", color = NA),
    # Sfondo pannello grafico
    panel.background = element_rect(fill = "white", color = NA),
    # Assi
    axis.line = element_line(color = "black"),
    # Nome asse X
    axis.title.x = element_text(margin = margin(t = margine_nome_asse_x)),
    # Nome asse Y
    axis.title.y = element_text(margin = margin(r = margine_nome_asse_y)),
    # Valori sugli assi
    axis.text = ggtext::element_markdown(
      size = 11,
      color = "black"
    ),
```

```
    axis.text.x = ggtext::element_markdown(),
    # Titolo grafico
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5)
  )

# TEMA PER FAMIGLIE
# Serie
tema_serie <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_serie),
        panel.grid.major = element_line(color = colore_griglia_serie),
        panel.grid.minor = element_line(color = colore_griglia_serie),
        plot.title = element_text(color = colore_titolo_serie))

# Trend lineare
tema_trend_lin <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_trend_lin),
        panel.grid.major = element_line(color = colore_griglia_trend_lin),
        panel.grid.minor = element_line(color = colore_griglia_trend_lin),
        plot.title = element_text(color = colore_titolo_trend_lin))

# Trend polinomiale base
tema_trend_poly <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_poly))

# Trend polinomiale per gradi
tema_trend_poly_2 <- tema_trend_poly
tema_trend_poly_3 <- tema_trend_poly
tema_trend_poly_4 <- tema_trend_poly
tema_trend_poly_5 <- tema_trend_poly

# Autocorrelazione
tema_acf <- tema_base +
  theme(

    panel.background = element_rect(
      fill = colore_quadrati_acf,
      color = NA
    ),

    panel.grid.major = element_line(
      color = colore_griglia_acf,
      linewidth = 0.5
    ),

    panel.grid.minor = element_line(
      color = colore_griglia_acf,
      linewidth = 0.25
    ),

    plot.title = element_text(
      color = colore_titolo_acf
    )
  )


# Residui
tema_residui <- tema_base +
  theme(

    panel.background = element_rect(
```

```
    fill = colore_quadrati_residui,
    color = NA
   ),

   panel.grid.major = element_line(
    color = colore_griglia_residui,
    linewidth = 0.5
   ),

   panel.grid.minor = element_line(
    color = colore_griglia_residui,
    linewidth = 0.25
   ),

   plot.title = element_text(
    color = colore_titolo_residui
   )
  )


#ASSE X PERSONALIZZATO
scala_asse_x_50anni <- function(vettore_date){

  vettore_date <- as.Date(vettore_date)

  anno_min <- year(min(vettore_date, na.rm = TRUE))
  anno_max <- year(max(vettore_date, na.rm = TRUE))

  breaks_finali <- sort(unique(c(
   seq(floor(anno_min/20)*20, ceiling(anno_max/20)*20, by = 20),
   anno_min,
   anno_max
  )))

  scale_x_date(
   breaks = as.Date(paste0(breaks_finali, "-01-01")),
   labels = function(x){

    anni <- format(x, "%Y")

    ifelse(
     anni %in% c(as.character(anno_min), as.character(anno_max)),
     paste0("<span style='color:#4D4D4D'>", anni, "</span>"),
     anni
    )
   },
   guide = guide_axis(n.dodge = 1)     #(se cambio 1 con 2 mi sfalsa i valori dei numeri sull'asse)
  )
}

#SETUP PER PLOT AUTOMATICO DEI GRAFICI IN PDF E PNG:
# Cartella output: Creo la cartella all'interno della quale salverò tutti i miei grafici
dir.create("grafici", showWarnings = FALSE)

# Contatore automatico dei grafici: Assegno un prefisso numerico al nome con cui vengono salvati i grafici
per ordine e per evitare sovrascrizioni
counter <- 1

#------------------------------------------------------------
# TABELLA RISULTATI TREND LINEARI
```

```
# Dataframe vuoto per salvare:
# - titolo grafico
# - pendenza
# - errore standard

tabella_trend <- data.frame(
  grafico = character(),
  pendenza = numeric(),
  errore_standard = numeric(),
  stringsAsFactors = FALSE
)
#------------------------------------------------------------

nome_progressivo <- function(nome){
  prefisso <- sprintf("%02d_", counter)
  assign("counter", counter + 1, envir = .GlobalEnv)
  return(paste0(prefisso, nome))
}

# Funzione salvataggio dei grafici
plot_e_salva <- function(grafico, nome_file){

  nome_file <- nome_progressivo(nome_file)

  print(grafico)

  ggsave(paste0("grafici/", nome_file, ".png"),
         plot = grafico,
         width = larghezza_grafico,
         height = altezza_grafico,
         dpi = dpi_grafico)

  ggsave(paste0("grafici/", nome_file, ".pdf"),
         plot = grafico,
         width = larghezza_grafico,
         height = altezza_grafico)
}


################################################################
# 2. ANALISI GIORNALIERA
################################################################

# Funzione per nomi asse Y in italiano
nome_variabile_plot <- function(var){

  if(var == "tempmax"){
    return("TEMPERATURA MASSIMA (°C)")
  }

  if(var == "tempmin"){
    return("TEMPERATURA MINIMA (°C)")
  }

  if(var == "tempmedia"){
    return("TEMPERATURA MEDIA (°C)")
  }

  if(var == "prec"){
    return("PRECIPITAZIONE (mm)")
```

```
  }

  return(toupper(gsub("_", "-", var)))
}

analisi_giornaliera <- function(dataframe, variabile, nome){

  nome_y <- nome_variabile_plot(variabile)


  # Serie-plot esplorativo
  g1 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie giornaliera -", nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_serie

  plot_e_salva(g1, paste0(variabile, "_serie"))


  # Trend lineare
  modello_lin <- lm(dataframe[[variabile]] ~ as.numeric(dataframe$vdata),
                    na.action = na.exclude)

  dataframe$trend_lin <- predict(modello_lin)

  g2 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_lin),
              color = "#a82d2d",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend lineare -", nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_trend_lin

  plot_e_salva(g2,
               paste0(variabile, "_trend_lin"))

  # Calcolo pendenza ed errore standard del trend lineare
  pendenza_giornaliera <- coef(modello_lin)[2]
  errore_standard_giornaliero <- summary(modello_lin)$coefficients[2,2]
  pendenza_annua <- pendenza_giornaliera * 365.25
  errore_standard_annuo <- errore_standard_giornaliero * 365.25
```

```
print (pendenza_giornaliera)
print (errore_standard_giornaliero)
print(pendenza_annua)
print(errore_standard_annuo)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE IN TABELLA
tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste("Trend lineare -", nome),
    pendenza = pendenza_annua,
    errore_standard = errore_standard_annuo
  )
)
#------------------------------------------------------------

# Trend polinomiale (da grado 2 a grado 5)
for(grado in 2:5){

  #(per colorare diversamente i quadrati di lo sfondo dei grafici dei diversi gradi)
  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  modello_poly <- lm(dataframe[[variabile]] ~
                       poly(as.numeric(dataframe$vdata), grado),
                     na.action = na.exclude)

  dataframe$trend_poly <- predict(modello_poly)

  g_poly <- ggplot(dataframe,
                   aes(x = vdata,
                       y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_poly),
              color = "#045cea",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend polinomiale grado",
                    grado,
                    "-",
                    nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_trend_poly_scorrimento

  plot_e_salva(g_poly,
```

```
            paste0(variabile,
                   "_trend_poly_grado_",
                   grado))
  }
}

# Applicazioni
analisi_giornaliera(dataframe, "tempmax", "Temperatura Massima")
analisi_giornaliera(dataframe, "tempmin", "Temperatura Minima")
analisi_giornaliera(dataframe, "tempmedia", "Temperatura Media")
analisi_giornaliera(dataframe, "prec", "Precipitazioni")


############################################################
# 3. AGGREGAZIONI
############################################################

# Variabile anno-mese
dataframe$year_month <- as.yearmon(dataframe$vdata)

# AGGREGAZIONE MENSILE
mensile <- dataframe %>%
  group_by(year_month) %>%
  summarise(
    tempmax = mean(tempmax, na.rm = TRUE),
    tempmin = mean(tempmin, na.rm = TRUE),
    tempmedia = mean(tempmedia, na.rm = TRUE),
    prec = sum(prec, na.rm = TRUE)
  )

# Variabile tempo numerica (per regressioni)
mensile$time <- as.numeric(mensile$year_month)

# Conversione corretta per grafici (yearmon -> Date)
mensile$date <- as.Date(mensile$year_month, frac = 0)


#-----------------------------------------------
# GRAFICI MENSILI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){

  nome_y <- nome_variabile_plot(variabile)

  # Serie
  g_mensile <- ggplot(mensile,
                      aes(x = date,
                          y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie mensile -", nome_y)
    ) +

    scala_asse_x_50anni(mensile$date) +
```

```
  tema_serie

plot_e_salva(g_mensile,
             paste0("mensile_serie_", variabile))


# Trend lineare
lm_mensile <- lm(as.formula(paste0(variabile, " ~ time")),
                 data = mensile,
                 na.action = na.exclude)

mensile$trend_lin <- predict(lm_mensile)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE MENSILE IN TABELLA
pendenza_mensile <- coef(lm_mensile)[2]

errore_standard_mensile <- summary(
  lm_mensile
)$coefficients[2,2]

tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste(
      "Trend lineare mensile -",
      nome_y
    ),

    pendenza = pendenza_mensile,

    errore_standard = errore_standard_mensile
  )
)
#------------------------------------------------------------

g_mensile_lin <- ggplot(mensile,
                        aes(x = date,
                            y = .data[[variabile]])) +

  geom_line() +

  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare mensile -", nome_y)
  ) +

  scala_asse_x_50anni(mensile$date) +

  tema_trend_lin

plot_e_salva(g_mensile_lin,
             paste0("mensile_trend_lineare_", variabile))
```

```
  # Trend polinomiale
  for(grado in 2:5){

    tema_trend_poly_scorrimento <- switch(
      as.character(grado),
      "2" = tema_trend_poly_2,
      "3" = tema_trend_poly_3,
      "4" = tema_trend_poly_4,
      "5" = tema_trend_poly_5,
      tema_trend_poly
    )

    lm_poly <- lm(as.formula(
      paste0(variabile, " ~ poly(time, ", grado, ")")
    ),
    data = mensile,
    na.action = na.exclude)

    mensile$trend_poly <- predict(lm_poly)

    g_poly <- ggplot(mensile,
                     aes(x = date,
                         y = .data[[variabile]])) +

      geom_line() +

      geom_line(aes(y = trend_poly),
                color = "#045cea",
                linewidth = 1.5) +

      labs(
        x = "ANNO",
        y = nome_y,
        title = paste("Trend polinomiale mensile grado", grado, "-", nome_y)
      ) +

      scala_asse_x_50anni(mensile$date) +

      tema_trend_poly_scorrimento

    plot_e_salva(g_poly,
                 paste0("mensile_trend_poly_grado_", grado, "_", variabile))
  }
}


# AGGREGAZIONE ANNUALE
annuale <- dataframe %>%
  group_by(year) %>%
  summarise(
    tempmax = mean(tempmax, na.rm = TRUE),
    tempmin = mean(tempmin, na.rm = TRUE),
    tempmedia = mean(tempmedia, na.rm = TRUE),
    prec = sum(prec, na.rm = TRUE)
  )

# Tempo centrato
annuale$year_c <- annuale$year - mean(annuale$year)
```

```
#-----------------------------------------------
# GRAFICI ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){

  nome_y <- nome_variabile_plot(variabile)

  # Serie
  g_annuale <- ggplot(annuale,
                      aes(x = year,
                          y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie annuale -", nome_y)
    ) +

    scale_x_continuous(
      breaks = sort(unique(c(
        seq(
          floor(min(annuale$year) / 20) * 20,
          ceiling(max(annuale$year) / 20) * 20,
          by = 20
        ),
        min(annuale$year),
        max(annuale$year)
      )))
    ) +

    tema_serie

  plot_e_salva(g_annuale,
               paste0("annuale_serie_", variabile))


  # Trend lineare
  lm_annuale <- lm(as.formula(paste0(variabile, " ~ year_c")),
                   data = annuale)

  annuale$trend_lin <- predict(lm_annuale)

  #-------------------------------------------------------
  # SALVATAGGIO RISULTATI TREND LINEARE ANNUALE IN TABELLA
  pendenza_annuale <- coef(lm_annuale)[2]

  errore_standard_annuale <- summary(
    lm_annuale
  )$coefficients[2,2]

  tabella_trend <<- rbind(
    tabella_trend,

    data.frame(
      grafico = paste(
        "Trend lineare annuale -",
```

```
      nome_y
    ),

    pendenza = pendenza_annuale,

    errore_standard = errore_standard_annuale
  )
)
#-------------------------------------------------------------

g_annuale_lin <- ggplot(annuale,
                        aes(x = year,
                            y = .data[[variabile]])) +

  geom_line() +

  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare annuale -", nome_y)
  ) +

  scale_x_continuous(
    breaks = sort(unique(c(
      seq(
        floor(min(annuale$year) / 20) * 20,
        ceiling(max(annuale$year) / 20) * 20,
        by = 20
      ),
      min(annuale$year),
      max(annuale$year)
    )))
  ) +

  tema_trend_lin

plot_e_salva(g_annuale_lin,
             paste0("annuale_trend_lineare_", variabile))


# Trend polinomiale
for(grado in 2:5){

  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  lm_poly_ann <- lm(as.formula(
    paste0(variabile, " ~ poly(year_c, ", grado, ")")
  ),
  data = annuale,
```

```
    na.action = na.exclude)

    annuale$trend_poly <- predict(lm_poly_ann)

    g_poly_ann <- ggplot(annuale,
                         aes(x = year,
                             y = .data[[variabile]])) +

      geom_line() +

      geom_line(aes(y = trend_poly),
                color = "#045cea",
                linewidth = 1.5) +

      labs(
        x = "ANNO",
        y = nome_y,
        title = paste("Trend polinomiale annuale grado", grado, "-", nome_y)
      ) +

      scale_x_continuous(
        breaks = sort(unique(c(
          seq(
            floor(min(annuale$year) / 20) * 20,
            ceiling(max(annuale$year) / 20) * 20,
            by = 20
          ),
          min(annuale$year),
          max(annuale$year)
        )))
      ) +

      tema_trend_poly_scorrimento

    plot_e_salva(g_poly_ann,
                 paste0("annuale_trend_poly_grado_", grado, "_", variabile))
  }
}

################################################################
# 4. AUTOCORRELAZIONE DELLE SERIE
################################################################

#-----------------------------------------------
# FUNZIONE AUTOCORRELAZIONE
#-----------------------------------------------

acf_salva <- function(serie,
                      nome,
                      tema_grafico,
                      colore_barre_positive = "gray",
                      colore_barre_negative = "gray30",
                      nome_lag = "LAG"){

  # Calcolo autocorrelazione
  acf_obj <- acf(
    serie,
    plot = FALSE,
    na.action = na.pass
  )
```

```
# Creazione dataframe ACF
df_acf <- data.frame(
  lag = as.numeric(acf_obj$lag),
  acf = as.numeric(acf_obj$acf)
)

# Colore barre positivo/negativo
df_acf$colore_barre <- ifelse(
  df_acf$acf >= 0,
  colore_barre_positive,
  colore_barre_negative
)

# Nome progressivo SOLO per file
nome_file <- nome_progressivo(nome)

# Grafico ACF con ggplot
g_acf <- ggplot(
  df_acf,
  aes(x = lag,
      y = acf)
) +

  geom_col(
    aes(fill = colore_barre),
    show.legend = FALSE
  ) +

  scale_fill_identity() +

  geom_hline(
    yintercept = 0,
    color = "black"
  ) +

  labs(
    title = nome,
    x = nome_lag,
    y = "AUTOCORRELAZIONE"
  ) +

  tema_grafico


# Salvataggio PNG
ggsave(
  paste0("grafici/", nome_file, ".png"),
  plot = g_acf,
  width = larghezza_grafico,
  height = altezza_grafico,
  dpi = dpi_grafico
)

# Salvataggio PDF
ggsave(
  paste0("grafici/", nome_file, ".pdf"),
  plot = g_acf,
  width = larghezza_grafico,
  height = altezza_grafico
```

```
  )

  # Visualizzazione grafico
  print(g_acf)
}


#-----------------------------------------------
# AUTOCORRELAZIONE SERIE MENSILI
#-----------------------------------------------

for(variabile in c("tempmax",
                   "tempmin",
                   "tempmedia",
                   "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF mensile
  acf_salva(
    serie = mensile[[variabile]],

    nome = paste(
      "Autocorrelazione mensile -",
      nome_variabile
    ),

    tema_grafico = tema_acf,

    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG MENSILE"
  )
}


#-----------------------------------------------
# AUTOCORRELAZIONE SERIE ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax",
                   "tempmin",
                   "tempmedia",
                   "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF annuale
  acf_salva(
    serie = annuale[[variabile]],

    nome = paste(
      "Autocorrelazione annuale -",
      nome_variabile
    ),

    tema_grafico = tema_acf,
```

```
    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG ANNUALE"
  )
}

################################################################
# 6. ESPORTAZIONE TABELLA TREND LINEARI
################################################################

# Libreria per esportazione tabella PDF
library(gridExtra)
library(grid)

# Creazione oggetto tabella
tabella_grafica <- tableGrob(
  tabella_trend,
  rows = NULL
)

# Esportazione PDF
pdf(
  "grafici/tabella_trend_lineari.pdf",
  width = 14,
  height = 8
)

grid.draw(tabella_grafica)

dev.off()
```

Napoli – Stazione di Napoli Capodichino

```
################################################################
# 0. SETUP
################################################################

# Pulizia ambiente
rm(list = ls())

# Librerie necessarie
library(readxl)
library(dplyr)
library(lubridate)
library(ggplot2)
library(forecast)
library(zoo)
library(ggtext)
library(readr)

# Definire working directory
setwd("……")

getwd()
list.files()
#-------------------------------------------------------------
# ANNO MINIMO DA ANALIZZARE       ->da togliere se voglio le serie con buchi
#-------------------------------------------------------------

anno_inizio_analisi <- 1972

# Se NULL verrà utilizzata tutta la serie
# anno_inizio_analisi <- NULL


#-------------------------------------------------------------
# LETTURA FILE
#-------------------------------------------------------------

# PULIZIA DEL FILE CSV
raw <- readLines("…… .txt")

# rimuove virgolette esterne
raw <- gsub('^"|"$', '', raw)

file_name <- "file_pulito.csv"
writeLines(raw, file_name)

# Lettura file separato da ;
dataframe <- read.csv2(
  file_name,
  fileEncoding = "UTF-8-BOM",
  stringsAsFactors = FALSE,
  check.names = FALSE
)

#-------------------------------------------------------------
# PULIZIA NOMI COLONNE
#-------------------------------------------------------------
names(dataframe) <- trimws(names(dataframe))
```

```
# Verifica colonne presenti
print(names(dataframe))

#---------------------------------------------------------
# FILTRO STAZIONE
#---------------------------------------------------------

dataframe <- dataframe %>%
  filter(nome == "NAPOLI/CAPODICHINO")

#---------------------------------------------------------
# CONVERSIONE VARIABILI NUMERICHE
#---------------------------------------------------------

dataframe$tmin <- as.numeric(dataframe$tmin)
dataframe$tmax <- as.numeric(dataframe$tmax)

dataframe$giorno <- as.integer(dataframe$giorno)
dataframe$mese   <- as.integer(dataframe$mese)
dataframe$anno   <- as.integer(dataframe$anno)

#---------------------------------------------------------
# CREAZIONE VARIABILI COMPATIBILI CON IL RESTO DELLO SCRIPT
#---------------------------------------------------------

dataframe <- dataframe %>%
  mutate(
    vdata = as.Date(
      sprintf(
        "%04d-%02d-%02d",
        anno,
        mese,
        giorno
      )
    ),

    tempmin = tmin,
    tempmax = tmax
  )
#---------------------------------------------------------
# FILTRO PER ANNO INIZIALE ANALISI      ->da togliere se voglio le serie con buchi
#---------------------------------------------------------

if(!is.null(anno_inizio_analisi)){

  dataframe <- dataframe %>%
    filter(anno >= anno_inizio_analisi)

}

#---------------------------------------------------------
# GESTIONE DATA
#---------------------------------------------------------

dataframe$year  <- year(dataframe$vdata)
dataframe$month <- month(dataframe$vdata)
dataframe$day   <- day(dataframe$vdata)

#---------------------------------------------------------
```

```
# ORDINAMENTO
#-------------------------------------------------------------

dataframe <- dataframe %>%
  arrange(vdata)

#-------------------------------------------------------------
# PRECIPITAZIONI
#-------------------------------------------------------------

# Variabile mantenuta per compatibilità con il resto dello script
dataframe$prec <- 1

#-------------------------------------------------------------
# CONTROLLO QUALITA'
#-------------------------------------------------------------

summary(dataframe)
colSums(is.na(dataframe))

#-------------------------------------------------------------
# GESTIONE VALORI MANCANTI
#-------------------------------------------------------------

# Eventuali codifiche mancanti
dataframe$tempmin[dataframe$tempmin == -99.9] <- NA
dataframe$tempmax[dataframe$tempmax == -99.9] <- NA

summary(dataframe$tempmin)
summary(dataframe$tempmax)

#-------------------------------------------------------------
# TREND PRELIMINARE (utilizzato in seguito)
#-------------------------------------------------------------

mod <- lm(
  tempmin ~ vdata,
  data = dataframe,
  na.action = na.exclude
)

dataframe$trend_lin <- predict(
  mod,
  newdata = dataframe
)

#-------------------------------------------------------------
# TEMPERATURA MEDIA
#-------------------------------------------------------------

dataframe$tempmedia <-
  (dataframe$tempmax + dataframe$tempmin) / 2


################################################################
# 1. PARAMETRI ESTETICI GRAFICI
################################################################

# DISTANZA NOMI ASSI DAGLI ASSI
margine_nome_asse_x <- 20
```

```
margine_nome_asse_y <- 20

# DIMENSIONI GRAFICI
larghezza_grafico <- 20
altezza_grafico <- 10
dpi_grafico <- 300

# COLORI PER FAMIGLIA DI GRAFICI
# Serie
colore_quadrati_serie <- "#D9D9D9"
colore_griglia_serie <- "white"
colore_titolo_serie <- "#848484"

# Trend lineare
colore_quadrati_trend_lin <- "#d8daee"
colore_griglia_trend_lin <- "white"
colore_titolo_trend_lin <- "#a82d2d"

# Trend polinomiale - Colore di default se non specificato il grado
colore_quadrati_poly <- "#d7e0ee"
colore_griglia_poly <- "white"
colore_titolo_poly <- "#045cea"

# Trend polinomiale - grado 2
colore_quadrati_poly_2 <- "#d7e5ee"
colore_quadriglia_poly_2 <- "white"
colore_titolo_poly_2 <- "#045cea"

# Trend polinomiale - grado 3
colore_quadrati_poly_3 <- "#d7e9ee"
colore_griglia_poly_3 <- "white"
colore_titolo_poly_3 <- "#045cea"

# Trend polinomiale - grado 4
colore_quadrati_poly_4 <- "#d7ecee"
colore_griglia_poly_4 <- "white"
colore_titolo_poly_4 <- "#045cea"

# Trend polinomiale - grado 5
colore_quadrati_poly_5 <- "#d7eeee"
colore_griglia_poly_5 <- "white"
colore_titolo_poly_5 <- "#045cea"

# ACF
# Colore barre ACF
colore_acf <- "gray"

colore_quadrati_acf <- "#f0d6d6"
colore_griglia_acf <- "white"
colore_titolo_acf <- "#8a2f2f"

# RESIDUI                                          -> EXTRA
# Colore barre residui
colore_residui <- "gray"

colore_quadrati_residui <- "#f0e2d4"
colore_griglia_residui <- "white"
colore_titolo_residui <- "#8a6235"

# TEMA BASE
```

```
tema_base <- theme_grey() +
  theme(
    # Sfondo esterno grafico
    plot.background = element_rect(fill = "white", color = NA),
    # Sfondo pannello grafico
    panel.background = element_rect(fill = "white", color = NA),
    # Assi
    axis.line = element_line(color = "black"),
    # Nome asse X
    axis.title.x = element_text(margin = margin(t = margine_nome_asse_x)),
    # Nome asse Y
    axis.title.y = element_text(margin = margin(r = margine_nome_asse_y)),
    # Valori sugli assi
    axis.text = ggtext::element_markdown(
      size = 11,
      color = "black"
    ),
    axis.text.x = ggtext::element_markdown(),
    # Titolo grafico
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5)
  )

# TEMA PER FAMIGLIE
# Serie
tema_serie <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_serie),
        panel.grid.major = element_line(color = colore_griglia_serie),
        panel.grid.minor = element_line(color = colore_griglia_serie),
        plot.title = element_text(color = colore_titolo_serie))

# Trend lineare
tema_trend_lin <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_trend_lin),
        panel.grid.major = element_line(color = colore_griglia_trend_lin),
        panel.grid.minor = element_line(color = colore_griglia_trend_lin),
        plot.title = element_text(color = colore_titolo_trend_lin))

# Trend polinomiale base
tema_trend_poly <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_poly))

# Trend polinomiale per gradi
tema_trend_poly_2 <- tema_trend_poly
tema_trend_poly_3 <- tema_trend_poly
tema_trend_poly_4 <- tema_trend_poly
tema_trend_poly_5 <- tema_trend_poly

# Autocorrelazione
tema_acf <- tema_base +
  theme(

    panel.background = element_rect(
      fill = colore_quadrati_acf,
      color = NA
    ),

    panel.grid.major = element_line(
      color = colore_griglia_acf,
      linewidth = 0.5
    ),
```

```
    panel.grid.minor = element_line(
      color = colore_griglia_acf,
      linewidth = 0.25
    ),

    plot.title = element_text(
      color = colore_titolo_acf
    )
  )


# Residui
tema_residui <- tema_base +
  theme(

    panel.background = element_rect(
      fill = colore_quadrati_residui,
      color = NA
    ),

    panel.grid.major = element_line(
      color = colore_griglia_residui,
      linewidth = 0.5
    ),

    panel.grid.minor = element_line(
      color = colore_griglia_residui,
      linewidth = 0.25
    ),

    plot.title = element_text(
      color = colore_titolo_residui
    )
  )


#ASSE X PERSONALIZZATO
scala_asse_x_50anni <- function(vettore_date){

  vettore_date <- as.Date(vettore_date)

  anno_min <- year(min(vettore_date, na.rm = TRUE))
  anno_max <- year(max(vettore_date, na.rm = TRUE))

  breaks_finali <- sort(unique(c(
    seq(floor(anno_min/20)*20, ceiling(anno_max/20)*20, by = 20),
    anno_min,
    anno_max
  )))

  scale_x_date(
    breaks = as.Date(paste0(breaks_finali, "-01-01")),
    labels = function(x){

      anni <- format(x, "%Y")

      ifelse(
        anni %in% c(as.character(anno_min), as.character(anno_max)),
        paste0("<span style='color:#4D4D4D'>", anni, "</span>"),
```

```
      anni
    )
  },
  guide = guide_axis(n.dodge = 1)     #(se cambio 1 con 2 mi sfalsa i valori dei numeri sull'asse)
 )
}

#SETUP PER PLOT AUTOMATICO DEI GRAFICI IN PDF E PNG:
# Cartella output: Creo la cartella all'interno della quale salverò tutti i miei grafici
dir.create("grafici", showWarnings = FALSE)

# Contatore automatico dei grafici: Assegno un prefisso numerico al nome con cui vengono salvati i grafici
per ordine e per evitare sovrascrizioni
counter <- 1

#------------------------------------------------------------
# TABELLA RISULTATI TREND LINEARI
# Dataframe vuoto per salvare:
# - titolo grafico
# - pendenza
# - errore standard

tabella_trend <- data.frame(
 grafico = character(),
 pendenza = numeric(),
 errore_standard = numeric(),
 stringsAsFactors = FALSE
)
#------------------------------------------------------------

nome_progressivo <- function(nome){
 prefisso <- sprintf("%02d_", counter)
 assign("counter", counter + 1, envir = .GlobalEnv)
 return(paste0(prefisso, nome))
}

# Funzione salvataggio dei grafici
plot_e_salva <- function(grafico, nome_file){

 nome_file <- nome_progressivo(nome_file)

 print(grafico)

 ggsave(paste0("grafici/", nome_file, ".png"),
        plot = grafico,
        width = larghezza_grafico,
        height = altezza_grafico,
        dpi = dpi_grafico)

 ggsave(paste0("grafici/", nome_file, ".pdf"),
        plot = grafico,
        width = larghezza_grafico,
        height = altezza_grafico)
}


################################################################
# 2. ANALISI GIORNALIERA
################################################################
```

```
# Funzione per nomi asse Y in italiano
nome_variabile_plot <- function(var){

  if(var == "tempmax"){
    return("TEMPERATURA MASSIMA (°C)")
  }

  if(var == "tempmin"){
    return("TEMPERATURA MINIMA (°C)")
  }

  if(var == "tempmedia"){
    return("TEMPERATURA MEDIA (°C)")
  }

  if(var == "prec"){
    return("PRECIPITAZIONE (mm)")
  }

  return(toupper(gsub("_", "-", var)))
}

analisi_giornaliera <- function(dataframe, variabile, nome){

  nome_y <- nome_variabile_plot(variabile)


  # Serie-plot esplorativo
  g1 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie giornaliera -", nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_serie

  plot_e_salva(g1, paste0(variabile, "_serie"))


  # Trend lineare
  modello_lin <- lm(dataframe[[variabile]] ~ as.numeric(dataframe$vdata),
                    na.action = na.exclude)

  dataframe$trend_lin <- predict(modello_lin)

  g2 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_lin),
              color = "#a82d2d",
              linewidth = 1.5) +
```

```
  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare -", nome)
  ) +
  scala_asse_x_50anni(dataframe$vdata) +
  tema_trend_lin

plot_e_salva(g2,
             paste0(variabile, "_trend_lin"))

# Calcolo pendenza ed errore standard del trend lineare
pendenza_giornaliera <- coef(modello_lin)[2]
errore_standard_giornaliero <- summary(modello_lin)$coefficients[2,2]
pendenza_annua <- pendenza_giornaliera * 365.25
errore_standard_annuo <- errore_standard_giornaliero * 365.25

print (pendenza_giornaliera)
print (errore_standard_giornaliero)
print(pendenza_annua)
print(errore_standard_annuo)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE IN TABELLA
tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste("Trend lineare -", nome),
    pendenza = pendenza_annua,
    errore_standard = errore_standard_annuo
  )
)
#------------------------------------------------------------

# Trend polinomiale (da grado 2 a grado 5)
for(grado in 2:5){

  #(per colorare diversamente i quadrati di lo sfondo dei grafici dei diversi gradi)
  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  modello_poly <- lm(dataframe[[variabile]] ~
                       poly(as.numeric(dataframe$vdata), grado),
                     na.action = na.exclude)

  dataframe$trend_poly <- predict(modello_poly)

  g_poly <- ggplot(dataframe,
                   aes(x = vdata,
                       y = .data[[variabile]])) +

    geom_line() +
```

```
    geom_line(aes(y = trend_poly),
              color = "#045cea",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend polinomiale grado",
                    grado,
                    "-",
                    nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_trend_poly_scorrimento

  plot_e_salva(g_poly,
               paste0(variabile,
                      "_trend_poly_grado_",
                      grado))
 }
}

# Applicazioni
analisi_giornaliera(dataframe, "tempmax", "Temperatura Massima")
analisi_giornaliera(dataframe, "tempmin", "Temperatura Minima")
analisi_giornaliera(dataframe, "tempmedia", "Temperatura Media")
analisi_giornaliera(dataframe, "prec", "Precipitazioni")


################################################################
# 3. AGGREGAZIONI
################################################################

# Variabile anno-mese
dataframe$year_month <- as.yearmon(dataframe$vdata)

# AGGREGAZIONE MENSILE
mensile <- dataframe %>%
 group_by(year_month) %>%
 summarise(
   tempmax = mean(tempmax, na.rm = TRUE),
   tempmin = mean(tempmin, na.rm = TRUE),
   tempmedia = mean(tempmedia, na.rm = TRUE),
   prec = sum(prec, na.rm = TRUE)
 )

# Variabile tempo numerica (per regressioni)
mensile$time <- as.numeric(mensile$year_month)

# Conversione corretta per grafici (yearmon -> Date)
mensile$date <- as.Date(mensile$year_month, frac = 0)


#-----------------------------------------------
# GRAFICI MENSILI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){
```

```
nome_y <- nome_variabile_plot(variabile)

# Serie
g_mensile <- ggplot(mensile,
                    aes(x = date,
                        y = .data[[variabile]])) +

  geom_line() +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Serie mensile -", nome_y)
  ) +

  scala_asse_x_50anni(mensile$date) +

  tema_serie

plot_e_salva(g_mensile,
             paste0("mensile_serie_", variabile))


# Trend lineare
lm_mensile <- lm(as.formula(paste0(variabile, " ~ time")),
                 data = mensile,
                 na.action = na.exclude)

mensile$trend_lin <- predict(lm_mensile)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE MENSILE IN TABELLA
pendenza_mensile <- coef(lm_mensile)[2]

errore_standard_mensile <- summary(
  lm_mensile
)$coefficients[2,2]

tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste(
      "Trend lineare mensile -",
      nome_y
    ),

    pendenza = pendenza_mensile,

    errore_standard = errore_standard_mensile
  )
)
#------------------------------------------------------------

g_mensile_lin <- ggplot(mensile,
                        aes(x = date,
                            y = .data[[variabile]])) +

  geom_line() +
```

```
  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare mensile -", nome_y)
  ) +

  scala_asse_x_50anni(mensile$date) +

  tema_trend_lin

plot_e_salva(g_mensile_lin,
             paste0("mensile_trend_lineare_", variabile))


# Trend polinomiale
for(grado in 2:5){

  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  lm_poly <- lm(as.formula(
    paste0(variabile, " ~ poly(time, ", grado, ")")
  ),
  data = mensile,
  na.action = na.exclude)

  mensile$trend_poly <- predict(lm_poly)

  g_poly <- ggplot(mensile,
                   aes(x = date,
                       y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_poly),
              color = "#045cea",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend polinomiale mensile grado", grado, "-", nome_y)
    ) +

    scala_asse_x_50anni(mensile$date) +

    tema_trend_poly_scorrimento

  plot_e_salva(g_poly,
               paste0("mensile_trend_poly_grado_", grado, "_", variabile))
```

```
  }
}


# AGGREGAZIONE ANNUALE
annuale <- dataframe %>%
  group_by(year) %>%
  summarise(
    tempmax = mean(tempmax, na.rm = TRUE),
    tempmin = mean(tempmin, na.rm = TRUE),
    tempmedia = mean(tempmedia, na.rm = TRUE),
    prec = sum(prec, na.rm = TRUE)
  )

# Tempo centrato
annuale$year_c <- annuale$year - mean(annuale$year)


#-----------------------------------------------
# GRAFICI ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){

  nome_y <- nome_variabile_plot(variabile)

  # Serie
  g_annuale <- ggplot(annuale,
                      aes(x = year,
                          y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie annuale -", nome_y)
    ) +

    scale_x_continuous(
      breaks = sort(unique(c(
        seq(
          floor(min(annuale$year) / 20) * 20,
          ceiling(max(annuale$year) / 20) * 20,
          by = 20
        ),
        min(annuale$year),
        max(annuale$year)
      )))
    ) +

    tema_serie

  plot_e_salva(g_annuale,
               paste0("annuale_serie_", variabile))


  # Trend lineare
  lm_annuale <- lm(as.formula(paste0(variabile, " ~ year_c")),
                   data = annuale)
```

```
annuale$trend_lin <- predict(lm_annuale)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE ANNUALE IN TABELLA
pendenza_annuale <- coef(lm_annuale)[2]

errore_standard_annuale <- summary(
  lm_annuale
)$coefficients[2,2]

tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste(
      "Trend lineare annuale -",
      nome_y
    ),

    pendenza = pendenza_annuale,

    errore_standard = errore_standard_annuale
  )
)
#------------------------------------------------------------

g_annuale_lin <- ggplot(annuale,
                        aes(x = year,
                            y = .data[[variabile]])) +

  geom_line() +

  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare annuale -", nome_y)
  ) +

  scale_x_continuous(
    breaks = sort(unique(c(
      seq(
        floor(min(annuale$year) / 20) * 20,
        ceiling(max(annuale$year) / 20) * 20,
        by = 20
      ),
      min(annuale$year),
      max(annuale$year)
    )))
  ) +

  tema_trend_lin

plot_e_salva(g_annuale_lin,
             paste0("annuale_trend_lineare_", variabile))
```

```
  # Trend polinomiale
  for(grado in 2:5){

    tema_trend_poly_scorrimento <- switch(
      as.character(grado),
      "2" = tema_trend_poly_2,
      "3" = tema_trend_poly_3,
      "4" = tema_trend_poly_4,
      "5" = tema_trend_poly_5,
      tema_trend_poly
    )

    lm_poly_ann <- lm(as.formula(
      paste0(variabile, " ~ poly(year_c, ", grado, ")")
    ),
    data = annuale,
    na.action = na.exclude)

    annuale$trend_poly <- predict(lm_poly_ann)

    g_poly_ann <- ggplot(annuale,
                         aes(x = year,
                             y = .data[[variabile]])) +

      geom_line() +

      geom_line(aes(y = trend_poly),
                color = "#045cea",
                linewidth = 1.5) +

      labs(
        x = "ANNO",
        y = nome_y,
        title = paste("Trend polinomiale annuale grado", grado, "-", nome_y)
      ) +

      scale_x_continuous(
        breaks = sort(unique(c(
          seq(
            floor(min(annuale$year) / 20) * 20,
            ceiling(max(annuale$year) / 20) * 20,
            by = 20
          ),
          min(annuale$year),
          max(annuale$year)
        )))
      ) +

      tema_trend_poly_scorrimento

    plot_e_salva(g_poly_ann,
                 paste0("annuale_trend_poly_grado_", grado, "_", variabile))
  }
}

#################################################################
# 4. AUTOCORRELAZIONE DELLE SERIE
#################################################################
```

```
#-----------------------------------------------
# FUNZIONE AUTOCORRELAZIONE
#-----------------------------------------------

acf_salva <- function(serie,
                      nome,
                      tema_grafico,
                      colore_barre_positive = "gray",
                      colore_barre_negative = "gray30",
                      nome_lag = "LAG"){

 # Calcolo autocorrelazione
 acf_obj <- acf(
   serie,
   plot = FALSE,
   na.action = na.pass
 )

 # Creazione dataframe ACF
 df_acf <- data.frame(
   lag = as.numeric(acf_obj$lag),
   acf = as.numeric(acf_obj$acf)
 )

 # Colore barre positivo/negativo
 df_acf$colore_barre <- ifelse(
   df_acf$acf >= 0,
   colore_barre_positive,
   colore_barre_negative
 )

 # Nome progressivo SOLO per file
 nome_file <- nome_progressivo(nome)

 # Grafico ACF con ggplot
 g_acf <- ggplot(
   df_acf,
   aes(x = lag,
       y = acf)
 ) +

   geom_col(
     aes(fill = colore_barre),
     show.legend = FALSE
   ) +

   scale_fill_identity() +

   geom_hline(
     yintercept = 0,
     color = "black"
   ) +

   labs(
     title = nome,
     x = nome_lag,
     y = "AUTOCORRELAZIONE"
   ) +

   tema_grafico
```

```
  # Salvataggio PNG
  ggsave(
    paste0("grafici/", nome_file, ".png"),
    plot = g_acf,
    width = larghezza_grafico,
    height = altezza_grafico,
    dpi = dpi_grafico
  )

  # Salvataggio PDF
  ggsave(
    paste0("grafici/", nome_file, ".pdf"),
    plot = g_acf,
    width = larghezza_grafico,
    height = altezza_grafico
  )

  # Visualizzazione grafico
  print(g_acf)
}


#-----------------------------------------------
# AUTOCORRELAZIONE SERIE MENSILI
#-----------------------------------------------

for(variabile in c("tempmax",
                   "tempmin",
                   "tempmedia",
                   "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF mensile
  acf_salva(
    serie = mensile[[variabile]],

    nome = paste(
      "Autocorrelazione mensile -",
      nome_variabile
    ),

    tema_grafico = tema_acf,

    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG MENSILE"
  )
}


#-----------------------------------------------
# AUTOCORRELAZIONE SERIE ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax",
```

```
                "tempmin",
                "tempmedia",
                "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF annuale
  acf_salva(
    serie = annuale[[variabile]],

    nome = paste(
      "Autocorrelazione annuale -",
      nome_variabile
    ),

    tema_grafico = tema_acf,

    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG ANNUALE"
  )
}

################################################################
# 6. ESPORTAZIONE TABELLA TREND LINEARI
################################################################

# Libreria per esportazione tabella PDF
library(gridExtra)
library(grid)

# Creazione oggetto tabella
tabella_grafica <- tableGrob(
  tabella_trend,
  rows = NULL
)

# Esportazione PDF
pdf(
  "grafici/tabella_trend_lineari.pdf",
  width = 14,
  height = 8
)

grid.draw(tabella_grafica)

dev.off()


################################################################
# EXTRA:CONTROLLO ANNI/GIORNI MANCANTI NELLE SERIE
################################################################
# ======================================================
# CONTROLLO COMPLETEZZA DELLA SERIE TEMPORALE
# ======================================================

library(dplyr)
library(gridExtra)
```

```
# ------------------------------------------------------
# 1. ANNI COMPLETAMENTE ASSENTI
# ------------------------------------------------------

anni_attesi <- seq(
  min(dataframe$year, na.rm = TRUE),
  max(dataframe$year, na.rm = TRUE)
)

anni_presenti <- sort(unique(dataframe$year))

anni_assenti <- setdiff(
  anni_attesi,
  anni_presenti
)

anni_assenti_df <- data.frame(
  anno_assente = anni_assenti
)

cat("\n=====================================\n")
cat("ANNI COMPLETAMENTE ASSENTI\n")
cat("=====================================\n")

if(length(anni_assenti) == 0){
  cat("Nessun anno completamente assente.\n")
} else {
  print(anni_assenti_df)
}

# ------------------------------------------------------
# 2. GIORNI MANCANTI
# ------------------------------------------------------

date_attese <- seq(
  min(dataframe$vdata, na.rm = TRUE),
  max(dataframe$vdata, na.rm = TRUE),
  by = "day"
)

date_mancanti <- setdiff(
  date_attese,
  dataframe$vdata
)

giorni_mancanti <- data.frame(
  data = date_mancanti,
  anno = as.integer(format(date_mancanti, "%Y")),
  mese = as.integer(format(date_mancanti, "%m")),
  giorno = as.integer(format(date_mancanti, "%d"))
)

cat("\n=====================================\n")
cat("GIORNI MANCANTI\n")
cat("=====================================\n")

cat(
  "Numero totale giorni mancanti:",
  nrow(giorni_mancanti),
```

```
  "\n"
)

# ------------------------------------------------------
# 3. ELENCO ANNI CHE CONTENGONO GIORNI MANCANTI
# ------------------------------------------------------

anni_con_giorni_mancanti <- sort(
  unique(giorni_mancanti$anno)
)

anni_con_giorni_mancanti_df <- data.frame(
  anno = anni_con_giorni_mancanti
)

cat("\n=====================================\n")
cat("ANNI CON ALMENO UN GIORNO MANCANTE\n")
cat("=====================================\n")

print(anni_con_giorni_mancanti_df)

# ------------------------------------------------------
# 4. DETTAGLIO: QUANTI GIORNI MANCANO IN OGNI ANNO
# ------------------------------------------------------

riepilogo_anni <- giorni_mancanti %>%
  group_by(anno) %>%
  summarise(
    giorni_mancanti = n(),
    .groups = "drop"
  ) %>%
  arrange(desc(giorni_mancanti))

cat("\n=====================================\n")
cat("GIORNI MANCANTI PER ANNO\n")
cat("=====================================\n")

print(riepilogo_anni)

# ------------------------------------------------------
# 5. ESPORTAZIONE PDF
# ------------------------------------------------------

pdf(
  "grafici_anni_completamente_assenti.pdf",
  width = 8,
  height = 10
)

grid.table(anni_assenti_df)

dev.off()


pdf(
  "grafici_anni_con_giorni_mancanti.pdf",
  width = 8,
  height = 10
)
```

```
grid.table(anni_con_giorni_mancanti_df)

dev.off()


pdf(
  "grafici_riepilogo_giorni_mancanti_per_anno.pdf",
  width = 10,
  height = 12
)

grid.table(riepilogo_anni)

dev.off()


pdf(
  "grafici_elenco_completo_giorni_mancanti.pdf",
  width = 12,
  height = 12
)

grid.table(giorni_mancanti)

dev.off()

cat("\n=====================================\n")
cat("PDF GENERATI\n")
cat("=====================================\n")
cat("1. grafici_anni_completamente_assenti.pdf\n")
cat("2. grafici_anni_con_giorni_mancanti.pdf\n")
cat("3. grafici_riepilogo_giorni_mancanti_per_anno.pdf\n")
cat("4. grafici_elenco_completo_giorni_mancanti.pdf\n")
cat("=====================================\n")


# Controllo che Rstudio abbia letto tutti i dati e che quindi i dati mancanti non provengano da un errore di
lettura
df_completo <- read.csv2(
  "file_pulito.csv",
  fileEncoding = "UTF-8-BOM",
  stringsAsFactors = FALSE,
  check.names = FALSE
)

sum(df_completo$nome == "NAPOLI/CAPODICHINO", na.rm = TRUE)
```

Palermo Osservatorio Astronomico

```
################################################################
# 0. SETUP
################################################################

# Pulizia ambiente
rm(list = ls())

# Librerie necessarie
library(readxl)
library(dplyr)
library(lubridate)
library(ggplot2)
library(forecast)
library(zoo)
library(ggtext)
library(readr)

# Definire working directory
setwd("…….")

getwd()
list.files()

#-------------------------------------------------------------
# ANNO MINIMO DA ANALIZZARE       ->da togliere se voglio le serie con buchi
#-------------------------------------------------------------

anno_inizio_analisi <- 1941

# Se NULL verrà utilizzata tutta la serie
# anno_inizio_analisi <- NULL


#-------------------------------------------------------------
# LETTURA FILE
#-------------------------------------------------------------

# PULIZIA DEL FILE CSV
raw <- readLines("…… .txt")

# rimuove virgolette esterne
raw <- gsub('^"|"$', '', raw)

file_name <- "file_pulito.csv"
writeLines(raw, file_name)

# Lettura file separato da ;
dataframe <- read.csv2(
  file_name,
  fileEncoding = "UTF-8-BOM",
  stringsAsFactors = FALSE,
  check.names = FALSE
)

#-------------------------------------------------------------
# PULIZIA NOMI COLONNE
#-------------------------------------------------------------
names(dataframe) <- trimws(names(dataframe))
```

```
# Verifica colonne presenti
print(names(dataframe))

#------------------------------------------------------------
# FILTRO STAZIONE
#------------------------------------------------------------

dataframe <- dataframe %>%
  filter(nome == "Palermo Osservatorio Astronomico")

#------------------------------------------------------------
# CONVERSIONE VARIABILI NUMERICHE
#------------------------------------------------------------

dataframe$tmin <- as.numeric(dataframe$tmin)
dataframe$tmax <- as.numeric(dataframe$tmax)

dataframe$giorno <- as.integer(dataframe$giorno)
dataframe$mese   <- as.integer(dataframe$mese)
dataframe$anno   <- as.integer(dataframe$anno)

#------------------------------------------------------------
# CREAZIONE VARIABILI COMPATIBILI CON IL RESTO DELLO SCRIPT
#------------------------------------------------------------

dataframe <- dataframe %>%
  mutate(
    vdata = as.Date(
      sprintf(
        "%04d-%02d-%02d",
        anno,
        mese,
        giorno
      )
    ),

    tempmin = tmin,
    tempmax = tmax
  )
#------------------------------------------------------------
# FILTRO PER ANNO INIZIALE ANALISI      ->da togliere se voglio le serie con buchi
#------------------------------------------------------------

if(!is.null(anno_inizio_analisi)){

  dataframe <- dataframe %>%
    filter(anno >= anno_inizio_analisi)

}

#------------------------------------------------------------
# GESTIONE DATA
#------------------------------------------------------------

dataframe$year  <- year(dataframe$vdata)
dataframe$month <- month(dataframe$vdata)
dataframe$day   <- day(dataframe$vdata)

#------------------------------------------------------------
```

```
# ORDINAMENTO
#-------------------------------------------------------------

dataframe <- dataframe %>%
  arrange(vdata)

#-------------------------------------------------------------
# PRECIPITAZIONI
#-------------------------------------------------------------

# Variabile mantenuta per compatibilità con il resto dello script
dataframe$prec <- 1

#-------------------------------------------------------------
# CONTROLLO QUALITA'
#-------------------------------------------------------------

summary(dataframe)
colSums(is.na(dataframe))

#-------------------------------------------------------------
# GESTIONE VALORI MANCANTI
#-------------------------------------------------------------

# Eventuali codifiche mancanti
dataframe$tempmin[dataframe$tempmin == -99.9] <- NA
dataframe$tempmax[dataframe$tempmax == -99.9] <- NA

summary(dataframe$tempmin)
summary(dataframe$tempmax)

#-------------------------------------------------------------
# TREND PRELIMINARE (utilizzato in seguito)
#-------------------------------------------------------------

mod <- lm(
  tempmin ~ vdata,
  data = dataframe,
  na.action = na.exclude
)

dataframe$trend_lin <- predict(
  mod,
  newdata = dataframe
)

#-------------------------------------------------------------
# TEMPERATURA MEDIA
#-------------------------------------------------------------

dataframe$tempmedia <-
  (dataframe$tempmax + dataframe$tempmin) / 2


################################################################
# 1. PARAMETRI ESTETICI GRAFICI
################################################################

# DISTANZA NOMI ASSI DAGLI ASSI
margine_nome_asse_x <- 20
```

```
margine_nome_asse_y <- 20

# DIMENSIONI GRAFICI
larghezza_grafico <- 20
altezza_grafico <- 10
dpi_grafico <- 300

# COLORI PER FAMIGLIA DI GRAFICI
# Serie
colore_quadrati_serie <- "#D9D9D9"
colore_griglia_serie <- "white"
colore_titolo_serie <- "#848484"

# Trend lineare
colore_quadrati_trend_lin <- "#d8daee"
colore_griglia_trend_lin <- "white"
colore_titolo_trend_lin <- "#a82d2d"

# Trend polinomiale - Colore di default se non specificato il grado
colore_quadrati_poly <- "#d7e0ee"
colore_griglia_poly <- "white"
colore_titolo_poly <- "#045cea"

# Trend polinomiale - grado 2
colore_quadrati_poly_2 <- "#d7e5ee"
colore_quadriglia_poly_2 <- "white"
colore_titolo_poly_2 <- "#045cea"

# Trend polinomiale - grado 3
colore_quadrati_poly_3 <- "#d7e9ee"
colore_griglia_poly_3 <- "white"
colore_titolo_poly_3 <- "#045cea"

# Trend polinomiale - grado 4
colore_quadrati_poly_4 <- "#d7ecee"
colore_griglia_poly_4 <- "white"
colore_titolo_poly_4 <- "#045cea"

# Trend polinomiale - grado 5
colore_quadrati_poly_5 <- "#d7eeee"
colore_griglia_poly_5 <- "white"
colore_titolo_poly_5 <- "#045cea"

# ACF
# Colore barre ACF
colore_acf <- "gray"

colore_quadrati_acf <- "#f0d6d6"
colore_griglia_acf <- "white"
colore_titolo_acf <- "#8a2f2f"

# RESIDUI                                                    -> EXTRA
# Colore barre residui
colore_residui <- "gray"

colore_quadrati_residui <- "#f0e2d4"
colore_griglia_residui <- "white"
colore_titolo_residui <- "#8a6235"

# TEMA BASE
```

```
tema_base <- theme_grey() +
  theme(
    # Sfondo esterno grafico
    plot.background = element_rect(fill = "white", color = NA),
    # Sfondo pannello grafico
    panel.background = element_rect(fill = "white", color = NA),
    # Assi
    axis.line = element_line(color = "black"),
    # Nome asse X
    axis.title.x = element_text(margin = margin(t = margine_nome_asse_x)),
    # Nome asse Y
    axis.title.y = element_text(margin = margin(r = margine_nome_asse_y)),
    # Valori sugli assi
    axis.text = ggtext::element_markdown(
      size = 11,
      color = "black"
    ),
    axis.text.x = ggtext::element_markdown(),
    # Titolo grafico
    plot.title = element_text(size = 16, face = "bold", hjust = 0.5)
  )

# TEMA PER FAMIGLIE
# Serie
tema_serie <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_serie),
        panel.grid.major = element_line(color = colore_griglia_serie),
        panel.grid.minor = element_line(color = colore_griglia_serie),
        plot.title = element_text(color = colore_titolo_serie))

# Trend lineare
tema_trend_lin <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_trend_lin),
        panel.grid.major = element_line(color = colore_griglia_trend_lin),
        panel.grid.minor = element_line(color = colore_griglia_trend_lin),
        plot.title = element_text(color = colore_titolo_trend_lin))

# Trend polinomiale base
tema_trend_poly <- tema_base +
  theme(panel.background = element_rect(fill = colore_quadrati_poly))

# Trend polinomiale per gradi
tema_trend_poly_2 <- tema_trend_poly
tema_trend_poly_3 <- tema_trend_poly
tema_trend_poly_4 <- tema_trend_poly
tema_trend_poly_5 <- tema_trend_poly

# Autocorrelazione
tema_acf <- tema_base +
  theme(

    panel.background = element_rect(
      fill = colore_quadrati_acf,
      color = NA
    ),

    panel.grid.major = element_line(
      color = colore_griglia_acf,
      linewidth = 0.5
    ),
```

```
    panel.grid.minor = element_line(
      color = colore_griglia_acf,
      linewidth = 0.25
    ),

    plot.title = element_text(
      color = colore_titolo_acf
    )
  )


# Residui
tema_residui <- tema_base +
  theme(

    panel.background = element_rect(
      fill = colore_quadrati_residui,
      color = NA
    ),

    panel.grid.major = element_line(
      color = colore_griglia_residui,
      linewidth = 0.5
    ),

    panel.grid.minor = element_line(
      color = colore_griglia_residui,
      linewidth = 0.25
    ),

    plot.title = element_text(
      color = colore_titolo_residui
    )
  )


#ASSE X PERSONALIZZATO
scala_asse_x_50anni <- function(vettore_date){

  vettore_date <- as.Date(vettore_date)

  anno_min <- year(min(vettore_date, na.rm = TRUE))
  anno_max <- year(max(vettore_date, na.rm = TRUE))

  breaks_finali <- sort(unique(c(
    seq(floor(anno_min/20)*20, ceiling(anno_max/20)*20, by = 20),
    anno_min,
    anno_max
  )))

  scale_x_date(
    breaks = as.Date(paste0(breaks_finali, "-01-01")),
    labels = function(x){

      anni <- format(x, "%Y")

      ifelse(
        anni %in% c(as.character(anno_min), as.character(anno_max)),
        paste0("<span style='color:#4D4D4D'>", anni, "</span>"),
```

```
      anni
    )
  },
  guide = guide_axis(n.dodge = 1)     #(se cambio 1 con 2 mi sfalsa i valori dei numeri sull'asse)
 )
}

#SETUP PER PLOT AUTOMATICO DEI GRAFICI IN PDF E PNG:
# Cartella output: Creo la cartella all'interno della quale salverò tutti i miei grafici
dir.create("grafici", showWarnings = FALSE)

# Contatore automatico dei grafici: Assegno un prefisso numerico al nome con cui vengono salvati i grafici
per ordine e per evitare sovrascrizioni
counter <- 1

#------------------------------------------------------------
# TABELLA RISULTATI TREND LINEARI
# Dataframe vuoto per salvare:
# - titolo grafico
# - pendenza
# - errore standard

tabella_trend <- data.frame(
  grafico = character(),
  pendenza = numeric(),
  errore_standard = numeric(),
  stringsAsFactors = FALSE
)
#------------------------------------------------------------

nome_progressivo <- function(nome){
  prefisso <- sprintf("%02d_", counter)
  assign("counter", counter + 1, envir = .GlobalEnv)
  return(paste0(prefisso, nome))
}

# Funzione salvataggio dei grafici
plot_e_salva <- function(grafico, nome_file){

  nome_file <- nome_progressivo(nome_file)

  print(grafico)

  ggsave(paste0("grafici/", nome_file, ".png"),
         plot = grafico,
         width = larghezza_grafico,
         height = altezza_grafico,
         dpi = dpi_grafico)

  ggsave(paste0("grafici/", nome_file, ".pdf"),
         plot = grafico,
         width = larghezza_grafico,
         height = altezza_grafico)
}


################################################################
# 2. ANALISI GIORNALIERA
################################################################
```

```
# Funzione per nomi asse Y in italiano
nome_variabile_plot <- function(var){

  if(var == "tempmax"){
    return("TEMPERATURA MASSIMA (°C)")
  }

  if(var == "tempmin"){
    return("TEMPERATURA MINIMA (°C)")
  }

  if(var == "tempmedia"){
    return("TEMPERATURA MEDIA (°C)")
  }

  if(var == "prec"){
    return("PRECIPITAZIONE (mm)")
  }

  return(toupper(gsub("_", "-", var)))
}

analisi_giornaliera <- function(dataframe, variabile, nome){

  nome_y <- nome_variabile_plot(variabile)


  # Serie-plot esplorativo
  g1 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie giornaliera -", nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_serie

  plot_e_salva(g1, paste0(variabile, "_serie"))


  # Trend lineare
  modello_lin <- lm(dataframe[[variabile]] ~ as.numeric(dataframe$vdata),
                    na.action = na.exclude)

  dataframe$trend_lin <- predict(modello_lin)

  g2 <- ggplot(dataframe,
               aes(x = vdata,
                   y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_lin),
              color = "#a82d2d",
              linewidth = 1.5) +
```

```
  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare -", nome)
  ) +
  scala_asse_x_50anni(dataframe$vdata) +
  tema_trend_lin

plot_e_salva(g2,
             paste0(variabile, "_trend_lin"))

# Calcolo pendenza ed errore standard del trend lineare
pendenza_giornaliera <- coef(modello_lin)[2]
errore_standard_giornaliero <- summary(modello_lin)$coefficients[2,2]
pendenza_annua <- pendenza_giornaliera * 365.25
errore_standard_annuo <- errore_standard_giornaliero * 365.25

print (pendenza_giornaliera)
print (errore_standard_giornaliero)
print(pendenza_annua)
print(errore_standard_annuo)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE IN TABELLA
tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste("Trend lineare -", nome),
    pendenza = pendenza_annua,
    errore_standard = errore_standard_annuo
  )
)
#------------------------------------------------------------

# Trend polinomiale (da grado 2 a grado 5)
for(grado in 2:5){

  #(per colorare diversamente i quadrati di lo sfondo dei grafici dei diversi gradi)
  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  modello_poly <- lm(dataframe[[variabile]] ~
                       poly(as.numeric(dataframe$vdata), grado),
                     na.action = na.exclude)

  dataframe$trend_poly <- predict(modello_poly)

  g_poly <- ggplot(dataframe,
                   aes(x = vdata,
                       y = .data[[variabile]])) +

    geom_line() +
```

```
    geom_line(aes(y = trend_poly),
              color = "#045cea",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend polinomiale grado",
                    grado,
                    "-",
                    nome)
    ) +
    scala_asse_x_50anni(dataframe$vdata) +
    tema_trend_poly_scorrimento

  plot_e_salva(g_poly,
               paste0(variabile,
                      "_trend_poly_grado_",
                      grado))
 }
}

# Applicazioni
analisi_giornaliera(dataframe, "tempmax", "Temperatura Massima")
analisi_giornaliera(dataframe, "tempmin", "Temperatura Minima")
analisi_giornaliera(dataframe, "tempmedia", "Temperatura Media")
analisi_giornaliera(dataframe, "prec", "Precipitazioni")


#####################################################################
# 3. AGGREGAZIONI
#####################################################################

# Variabile anno-mese
dataframe$year_month <- as.yearmon(dataframe$vdata)

# AGGREGAZIONE MENSILE
mensile <- dataframe %>%
 group_by(year_month) %>%
 summarise(
   tempmax = mean(tempmax, na.rm = TRUE),
   tempmin = mean(tempmin, na.rm = TRUE),
   tempmedia = mean(tempmedia, na.rm = TRUE),
   prec = sum(prec, na.rm = TRUE)
 )

# Variabile tempo numerica (per regressioni)
mensile$time <- as.numeric(mensile$year_month)

# Conversione corretta per grafici (yearmon -> Date)
mensile$date <- as.Date(mensile$year_month, frac = 0)


#-----------------------------------------------
# GRAFICI MENSILI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){
```

```
nome_y <- nome_variabile_plot(variabile)

# Serie
g_mensile <- ggplot(mensile,
                    aes(x = date,
                        y = .data[[variabile]])) +

  geom_line() +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Serie mensile -", nome_y)
  ) +

  scala_asse_x_50anni(mensile$date) +

  tema_serie

plot_e_salva(g_mensile,
             paste0("mensile_serie_", variabile))


# Trend lineare
lm_mensile <- lm(as.formula(paste0(variabile, " ~ time")),
                 data = mensile,
                 na.action = na.exclude)

mensile$trend_lin <- predict(lm_mensile)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE MENSILE IN TABELLA
pendenza_mensile <- coef(lm_mensile)[2]

errore_standard_mensile <- summary(
  lm_mensile
)$coefficients[2,2]

tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste(
      "Trend lineare mensile -",
      nome_y
    ),

    pendenza = pendenza_mensile,

    errore_standard = errore_standard_mensile
  )
)
#------------------------------------------------------------

g_mensile_lin <- ggplot(mensile,
                        aes(x = date,
                            y = .data[[variabile]])) +

  geom_line() +
```

```
  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare mensile -", nome_y)
  ) +

  scala_asse_x_50anni(mensile$date) +

  tema_trend_lin

plot_e_salva(g_mensile_lin,
             paste0("mensile_trend_lineare_", variabile))


# Trend polinomiale
for(grado in 2:5){

  tema_trend_poly_scorrimento <- switch(
    as.character(grado),
    "2" = tema_trend_poly_2,
    "3" = tema_trend_poly_3,
    "4" = tema_trend_poly_4,
    "5" = tema_trend_poly_5,
    tema_trend_poly
  )

  lm_poly <- lm(as.formula(
    paste0(variabile, " ~ poly(time, ", grado, ")")
  ),
  data = mensile,
  na.action = na.exclude)

  mensile$trend_poly <- predict(lm_poly)

  g_poly <- ggplot(mensile,
                   aes(x = date,
                       y = .data[[variabile]])) +

    geom_line() +

    geom_line(aes(y = trend_poly),
              color = "#045cea",
              linewidth = 1.5) +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Trend polinomiale mensile grado", grado, "-", nome_y)
    ) +

    scala_asse_x_50anni(mensile$date) +

    tema_trend_poly_scorrimento

  plot_e_salva(g_poly,
               paste0("mensile_trend_poly_grado_", grado, "_", variabile))
```

```
  }
}


# AGGREGAZIONE ANNUALE
annuale <- dataframe %>%
  group_by(year) %>%
  summarise(
    tempmax = mean(tempmax, na.rm = TRUE),
    tempmin = mean(tempmin, na.rm = TRUE),
    tempmedia = mean(tempmedia, na.rm = TRUE),
    prec = sum(prec, na.rm = TRUE)
  )

# Tempo centrato
annuale$year_c <- annuale$year - mean(annuale$year)


#-----------------------------------------------
# GRAFICI ANNUALI
#-----------------------------------------------

for(variabile in c("tempmax", "tempmin", "tempmedia", "prec")){

  nome_y <- nome_variabile_plot(variabile)

  # Serie
  g_annuale <- ggplot(annuale,
                      aes(x = year,
                          y = .data[[variabile]])) +

    geom_line() +

    labs(
      x = "ANNO",
      y = nome_y,
      title = paste("Serie annuale -", nome_y)
    ) +

    scale_x_continuous(
      breaks = sort(unique(c(
        seq(
          floor(min(annuale$year) / 20) * 20,
          ceiling(max(annuale$year) / 20) * 20,
          by = 20
        ),
        min(annuale$year),
        max(annuale$year)
      )))
    ) +

    tema_serie

  plot_e_salva(g_annuale,
               paste0("annuale_serie_", variabile))


  # Trend lineare
  lm_annuale <- lm(as.formula(paste0(variabile, " ~ year_c")),
                   data = annuale)
```

```
annuale$trend_lin <- predict(lm_annuale)

#------------------------------------------------------------
# SALVATAGGIO RISULTATI TREND LINEARE ANNUALE IN TABELLA
pendenza_annuale <- coef(lm_annuale)[2]

errore_standard_annuale <- summary(
  lm_annuale
)$coefficients[2,2]

tabella_trend <<- rbind(
  tabella_trend,

  data.frame(
    grafico = paste(
      "Trend lineare annuale -",
      nome_y
    ),

    pendenza = pendenza_annuale,

    errore_standard = errore_standard_annuale
  )
)
#------------------------------------------------------------

g_annuale_lin <- ggplot(annuale,
                        aes(x = year,
                            y = .data[[variabile]])) +

  geom_line() +

  geom_line(aes(y = trend_lin),
            color = "#a82d2d",
            linewidth = 1.5) +

  labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend lineare annuale -", nome_y)
  ) +

  scale_x_continuous(
    breaks = sort(unique(c(
      seq(
        floor(min(annuale$year) / 20) * 20,
        ceiling(max(annuale$year) / 20) * 20,
        by = 20
      ),
      min(annuale$year),
      max(annuale$year)
    )))
  ) +

  tema_trend_lin

plot_e_salva(g_annuale_lin,
             paste0("annuale_trend_lineare_", variabile))
```

```
 # Trend polinomiale
 for(grado in 2:5){

  tema_trend_poly_scorrimento <- switch(
   as.character(grado),
   "2" = tema_trend_poly_2,
   "3" = tema_trend_poly_3,
   "4" = tema_trend_poly_4,
   "5" = tema_trend_poly_5,
   tema_trend_poly
  )

  lm_poly_ann <- lm(as.formula(
   paste0(variabile, " ~ poly(year_c, ", grado, ")")
  ),
  data = annuale,
  na.action = na.exclude)

  annuale$trend_poly <- predict(lm_poly_ann)

  g_poly_ann <- ggplot(annuale,
                       aes(x = year,
                           y = .data[[variabile]])) +

   geom_line() +

   geom_line(aes(y = trend_poly),
             color = "#045cea",
             linewidth = 1.5) +

   labs(
    x = "ANNO",
    y = nome_y,
    title = paste("Trend polinomiale annuale grado", grado, "-", nome_y)
   ) +

   scale_x_continuous(
    breaks = sort(unique(c(
     seq(
      floor(min(annuale$year) / 20) * 20,
      ceiling(max(annuale$year) / 20) * 20,
      by = 20
     ),
     min(annuale$year),
     max(annuale$year)
    )))
   ) +

   tema_trend_poly_scorrimento

  plot_e_salva(g_poly_ann,
               paste0("annuale_trend_poly_grado_", grado, "_", variabile))
 }
}

################################################################
# 4. AUTOCORRELAZIONE DELLE SERIE
################################################################
```

```
#-----------------------------------------------
# FUNZIONE AUTOCORRELAZIONE
#-----------------------------------------------

acf_salva <- function(serie,
                      nome,
                      tema_grafico,
                      colore_barre_positive = "gray",
                      colore_barre_negative = "gray30",
                      nome_lag = "LAG"){

 # Calcolo autocorrelazione
 acf_obj <- acf(
  serie,
  plot = FALSE,
  na.action = na.pass
 )

 # Creazione dataframe ACF
 df_acf <- data.frame(
  lag = as.numeric(acf_obj$lag),
  acf = as.numeric(acf_obj$acf)
 )

 # Colore barre positivo/negativo
 df_acf$colore_barre <- ifelse(
  df_acf$acf >= 0,
  colore_barre_positive,
  colore_barre_negative
 )

 # Nome progressivo SOLO per file
 nome_file <- nome_progressivo(nome)

 # Grafico ACF con ggplot
 g_acf <- ggplot(
  df_acf,
  aes(x = lag,
      y = acf)
 ) +

  geom_col(
   aes(fill = colore_barre),
   show.legend = FALSE
  ) +

  scale_fill_identity() +

  geom_hline(
   yintercept = 0,
   color = "black"
  ) +

  labs(
   title = nome,
   x = nome_lag,
   y = "AUTOCORRELAZIONE"
  ) +

  tema_grafico
```

```
  # Salvataggio PNG
  ggsave(
    paste0("grafici/", nome_file, ".png"),
    plot = g_acf,
    width = larghezza_grafico,
    height = altezza_grafico,
    dpi = dpi_grafico
  )

  # Salvataggio PDF
  ggsave(
    paste0("grafici/", nome_file, ".pdf"),
    plot = g_acf,
    width = larghezza_grafico,
    height = altezza_grafico
  )

  # Visualizzazione grafico
  print(g_acf)
}


#------------------------------------------------
# AUTOCORRELAZIONE SERIE MENSILI
#------------------------------------------------

for(variabile in c("tempmax",
                   "tempmin",
                   "tempmedia",
                   "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF mensile
  acf_salva(
    serie = mensile[[variabile]],

    nome = paste(
      "Autocorrelazione mensile -",
      nome_variabile
    ),

    tema_grafico = tema_acf,

    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG MENSILE"
  )
}


#------------------------------------------------
# AUTOCORRELAZIONE SERIE ANNUALI
#------------------------------------------------

for(variabile in c("tempmax",
```

```
                "tempmin",
                "tempmedia",
                "prec")){

  # Nome leggibile variabile
  nome_variabile <- nome_variabile_plot(variabile)

  # Creazione grafico ACF annuale
  acf_salva(
    serie = annuale[[variabile]],

    nome = paste(
      "Autocorrelazione annuale -",
      nome_variabile
    ),

    tema_grafico = tema_acf,

    colore_barre_positive = "gray",
    colore_barre_negative = "gray30",

    nome_lag = "LAG ANNUALE"
  )
}

################################################################
# 6. ESPORTAZIONE TABELLA TREND LINEARI
################################################################

# Libreria per esportazione tabella PDF
library(gridExtra)
library(grid)

# Creazione oggetto tabella
tabella_grafica <- tableGrob(
  tabella_trend,
  rows = NULL
)

# Esportazione PDF
pdf(
  "grafici/tabella_trend_lineari.pdf",
  width = 14,
  height = 8
)

grid.draw(tabella_grafica)

dev.off()


################################################################
# EXTRA:CONTROLLO ANNI/GIORNI MANCANTI NELLE SERIE
################################################################
# ======================================================
# CONTROLLO COMPLETEZZA DELLA SERIE TEMPORALE
# ======================================================

library(dplyr)
library(gridExtra)
```

```
# ------------------------------------------------------
# 1. ANNI COMPLETAMENTE ASSENTI
# ------------------------------------------------------

anni_attesi <- seq(
  min(dataframe$year, na.rm = TRUE),
  max(dataframe$year, na.rm = TRUE)
)

anni_presenti <- sort(unique(dataframe$year))

anni_assenti <- setdiff(
  anni_attesi,
  anni_presenti
)

anni_assenti_df <- data.frame(
  anno_assente = anni_assenti
)

cat("\n======================================\n")
cat("ANNI COMPLETAMENTE ASSENTI\n")
cat("======================================\n")

if(length(anni_assenti) == 0){
  cat("Nessun anno completamente assente.\n")
} else {
  print(anni_assenti_df)
}

# ------------------------------------------------------
# 2. GIORNI MANCANTI
# ------------------------------------------------------

date_attese <- seq(
  min(dataframe$vdata, na.rm = TRUE),
  max(dataframe$vdata, na.rm = TRUE),
  by = "day"
)

date_mancanti <- setdiff(
  date_attese,
  dataframe$vdata
)

giorni_mancanti <- data.frame(
  data = date_mancanti,
  anno = as.integer(format(date_mancanti, "%Y")),
  mese = as.integer(format(date_mancanti, "%m")),
  giorno = as.integer(format(date_mancanti, "%d"))
)

cat("\n======================================\n")
cat("GIORNI MANCANTI\n")
cat("======================================\n")

cat(
  "Numero totale giorni mancanti:",
  nrow(giorni_mancanti),
```

```
  "\n"
)

# ------------------------------------------------------
# 3. ELENCO ANNI CHE CONTENGONO GIORNI MANCANTI
# ------------------------------------------------------

anni_con_giorni_mancanti <- sort(
  unique(giorni_mancanti$anno)
)

anni_con_giorni_mancanti_df <- data.frame(
  anno = anni_con_giorni_mancanti
)

cat("\n=====================================\n")
cat("ANNI CON ALMENO UN GIORNO MANCANTE\n")
cat("=====================================\n")

print(anni_con_giorni_mancanti_df)

# ------------------------------------------------------
# 4. DETTAGLIO: QUANTI GIORNI MANCANO IN OGNI ANNO
# ------------------------------------------------------

riepilogo_anni <- giorni_mancanti %>%
  group_by(anno) %>%
  summarise(
    giorni_mancanti = n(),
    .groups = "drop"
  ) %>%
  arrange(desc(giorni_mancanti))

cat("\n=====================================\n")
cat("GIORNI MANCANTI PER ANNO\n")
cat("=====================================\n")

print(riepilogo_anni)

# ------------------------------------------------------
# 5. ESPORTAZIONE PDF
# ------------------------------------------------------

pdf(
  "grafici_anni_completamente_assenti.pdf",
  width = 8,
  height = 10
)

grid.table(anni_assenti_df)

dev.off()


pdf(
  "grafici_anni_con_giorni_mancanti.pdf",
  width = 8,
  height = 10
)
```

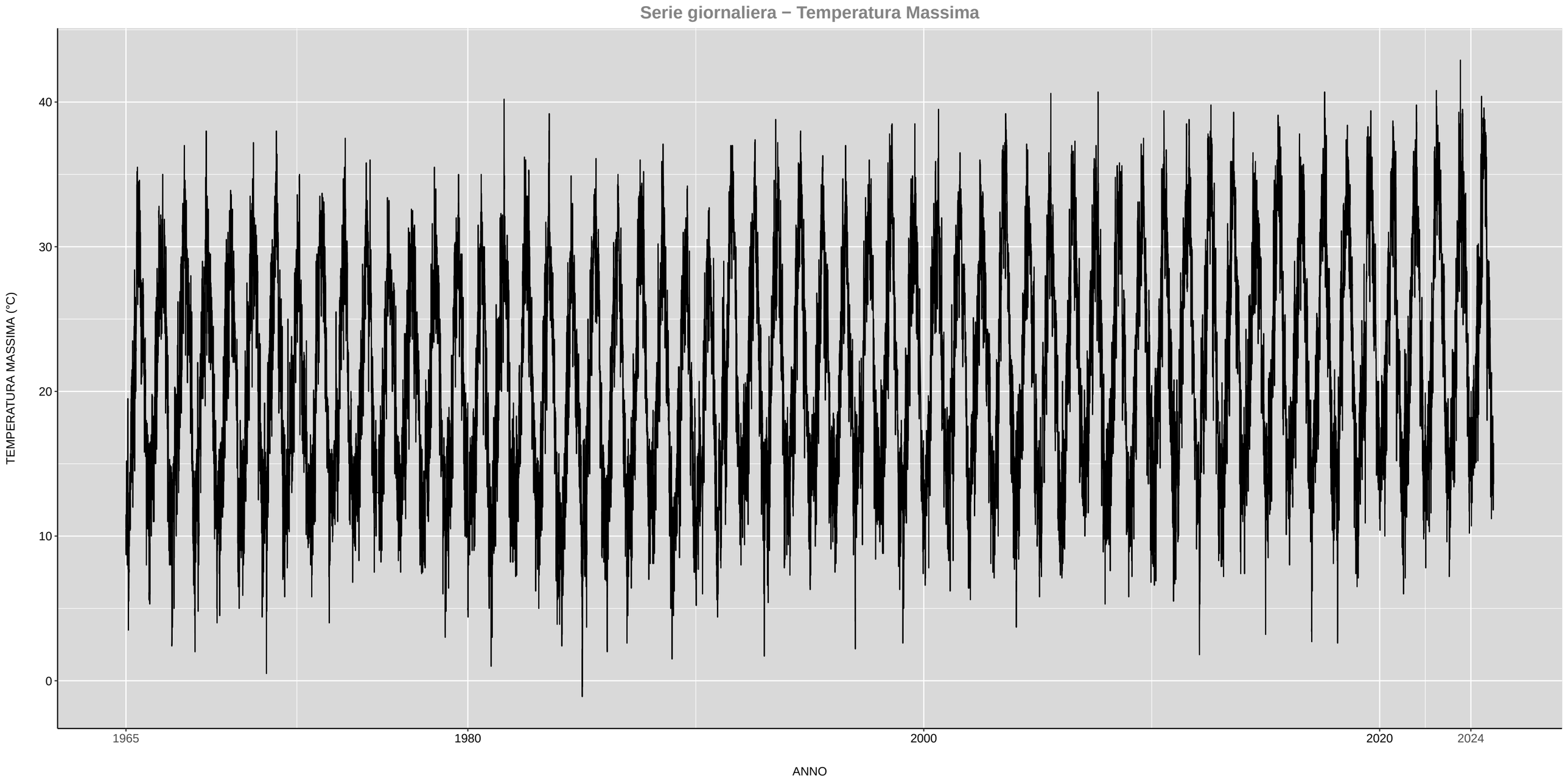
```
grid.table(anni_con_giorni_mancanti_df)

dev.off()


pdf(
  "grafici_riepilogo_giorni_mancanti_per_anno.pdf",
  width = 10,
  height = 12
)

grid.table(riepilogo_anni)

dev.off()


pdf(
  "grafici_elenco_completo_giorni_mancanti.pdf",
  width = 12,
  height = 12
)

grid.table(giorni_mancanti)

dev.off()

cat("\n=====================================\n")
cat("PDF GENERATI\n")
cat("=====================================\n")
cat("1. grafici_anni_completamente_assenti.pdf\n")
cat("2. grafici_anni_con_giorni_mancanti.pdf\n")
cat("3. grafici_riepilogo_giorni_mancanti_per_anno.pdf\n")
cat("4. grafici_elenco_completo_giorni_mancanti.pdf\n")
cat("=====================================\n")


# Controllo che Rstudio abbia letto tutti i dati e che quindi i dati mancanti non provengano da un errore di
lettura
df_completo <- read.csv2(
  "file_pulito.csv",
  fileEncoding = "UTF-8-BOM",
  stringsAsFactors = FALSE,
  check.names = FALSE
)

sum(df_completo$nome == "Palermo Osservatorio Astronomico", na.rm = TRUE)
```

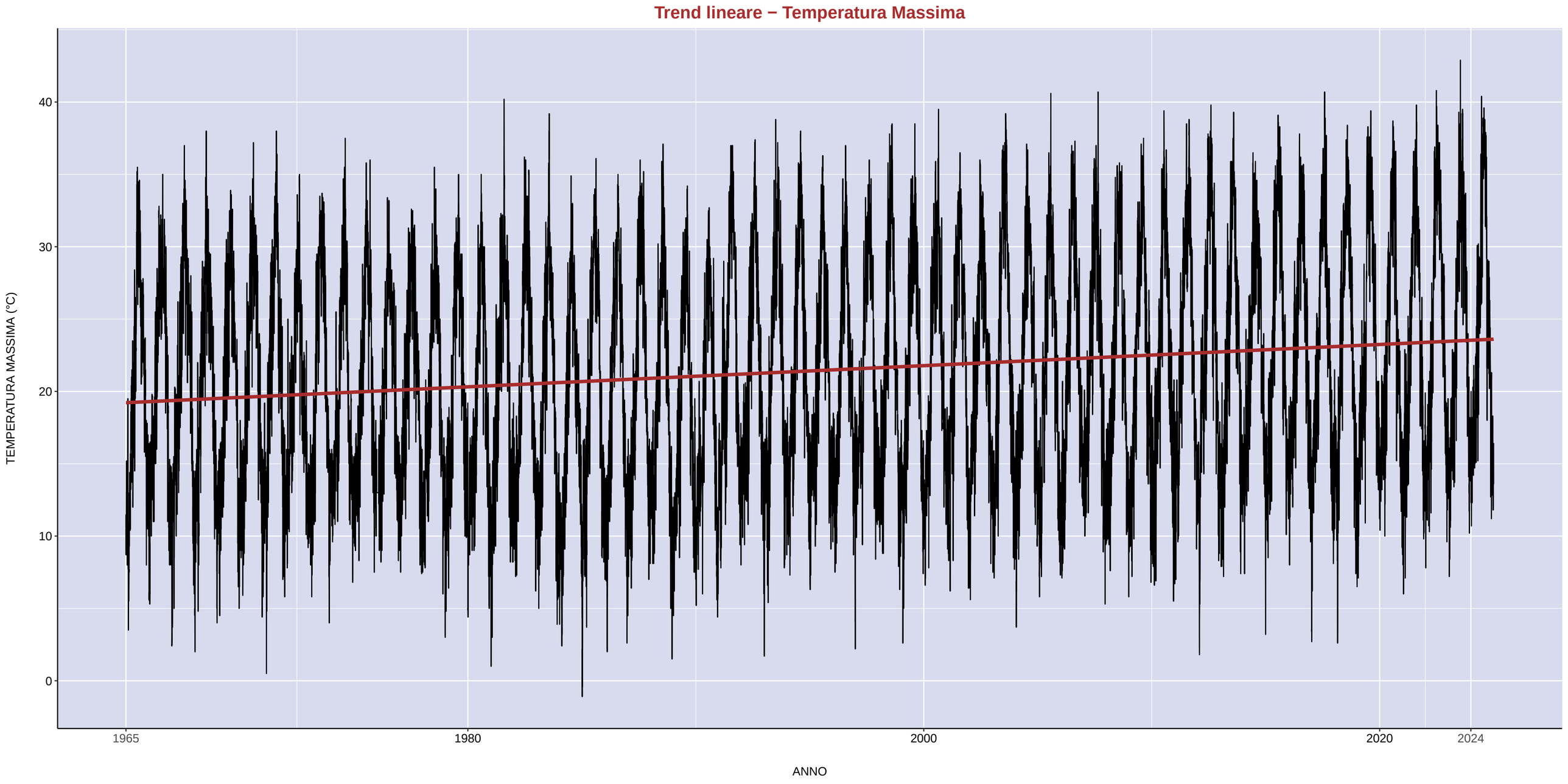
Allegato 2: Grafici ottenuti dall'analisi climatica dei cinque capoluoghi

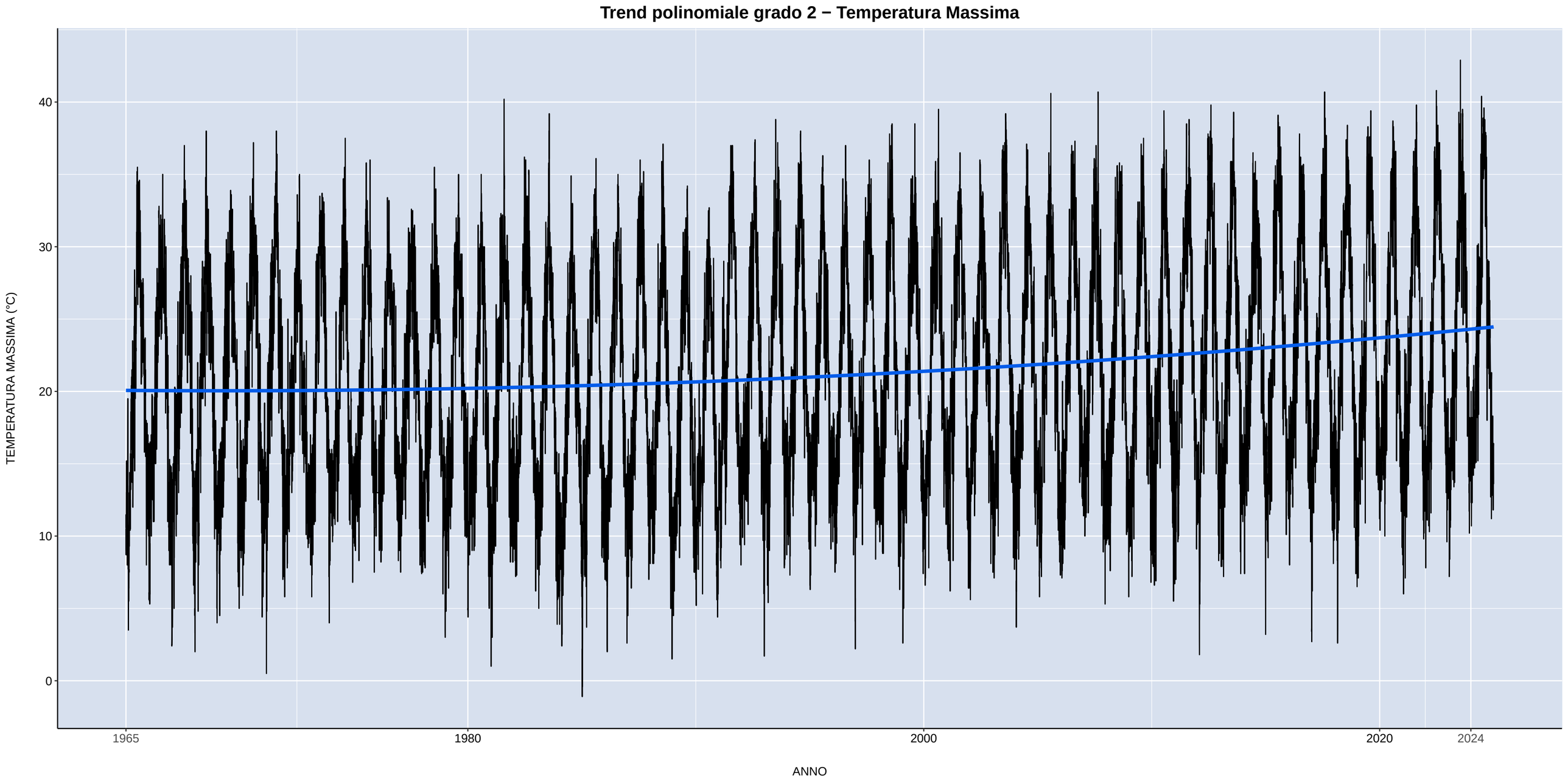
In questo secondo allegato vengono riportati i grafici relativi all'analisi climatica, ottenuti per i cinque capoluoghi italiani. Questo allegato ha l'obiettivo di fornire al lettore i grafici nel loro formato ottimale e nella loro risoluzione originale, restituendo la qualità grafica che non è stato possibile riportare a pieno nell'elaborato di tesi.

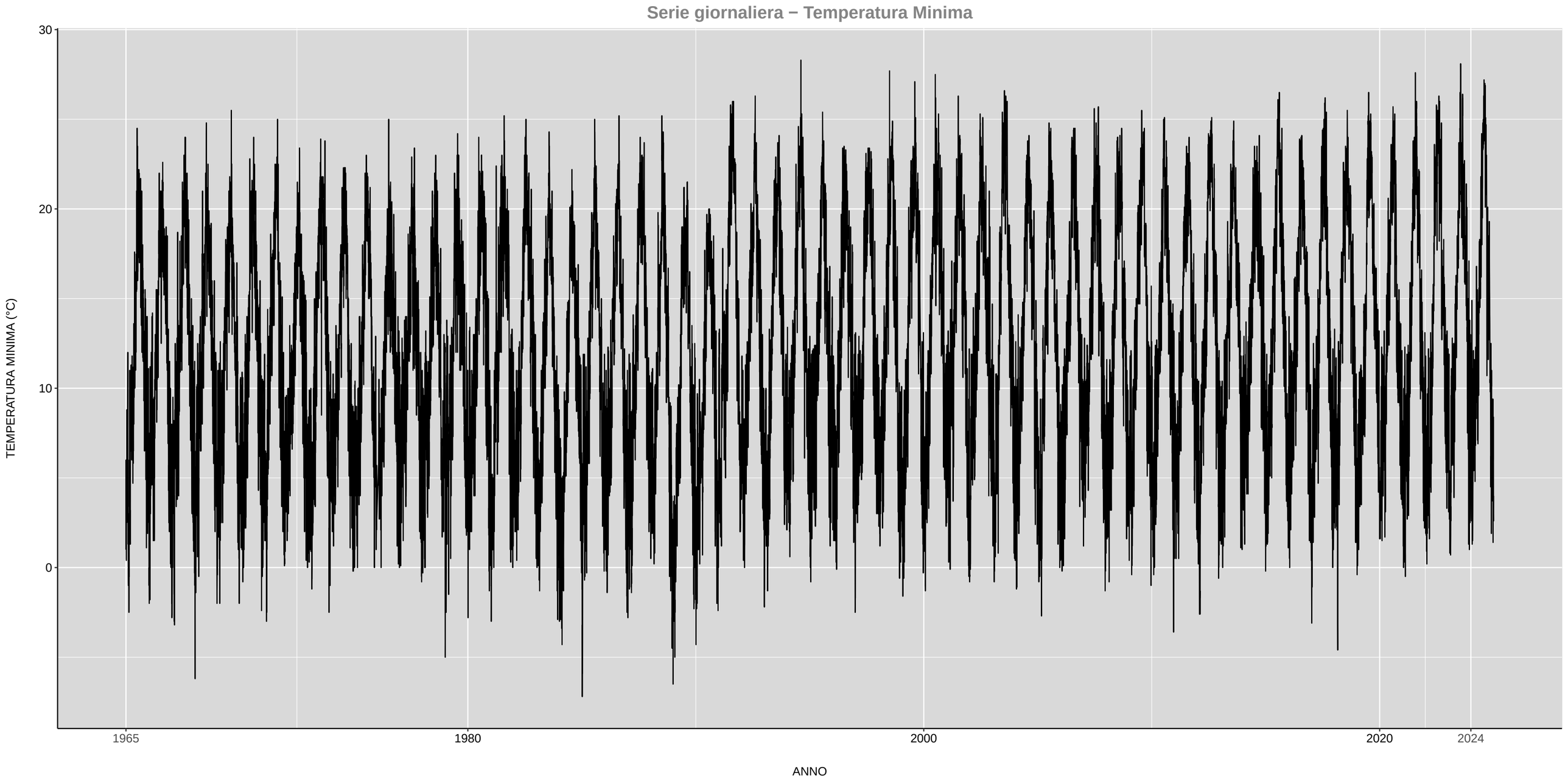
Verranno riportati, in ordine, i grafici delle città di Roma, Milano, Napoli e Palermo.

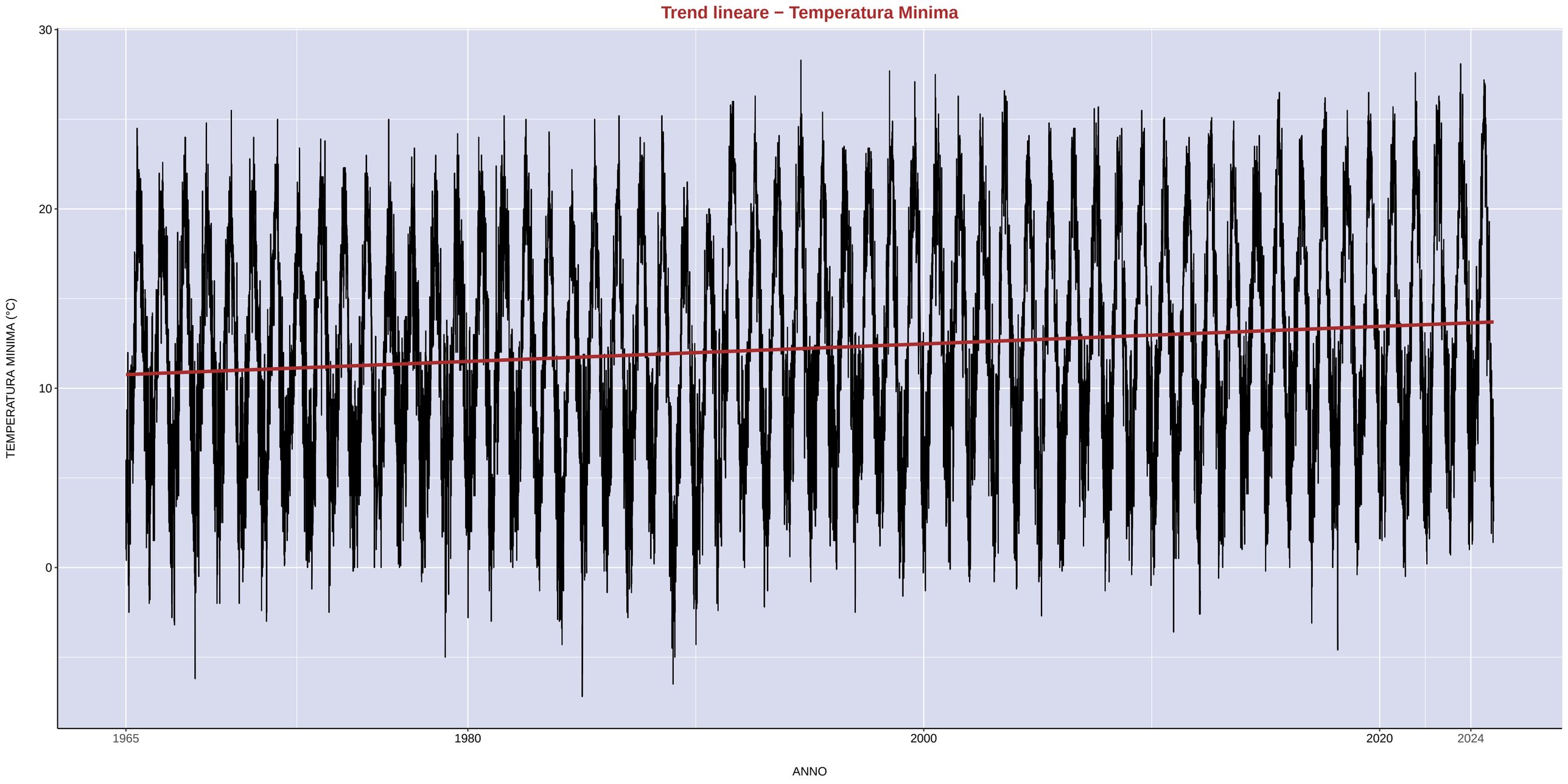
Roma – Stazione di Roma Macao

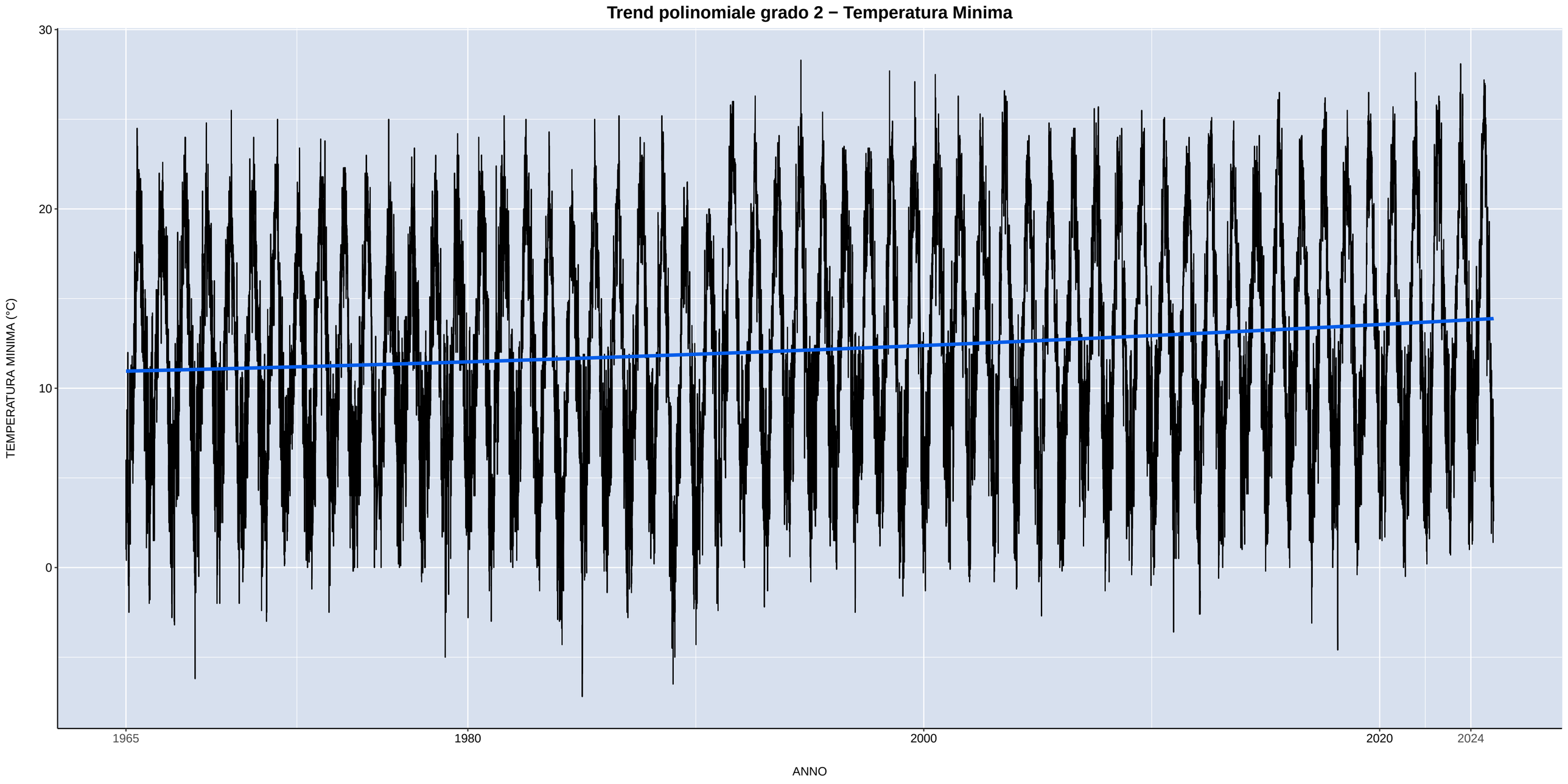

Serie giornaliera – Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1965
1980
2000
2020
2024
ANNO

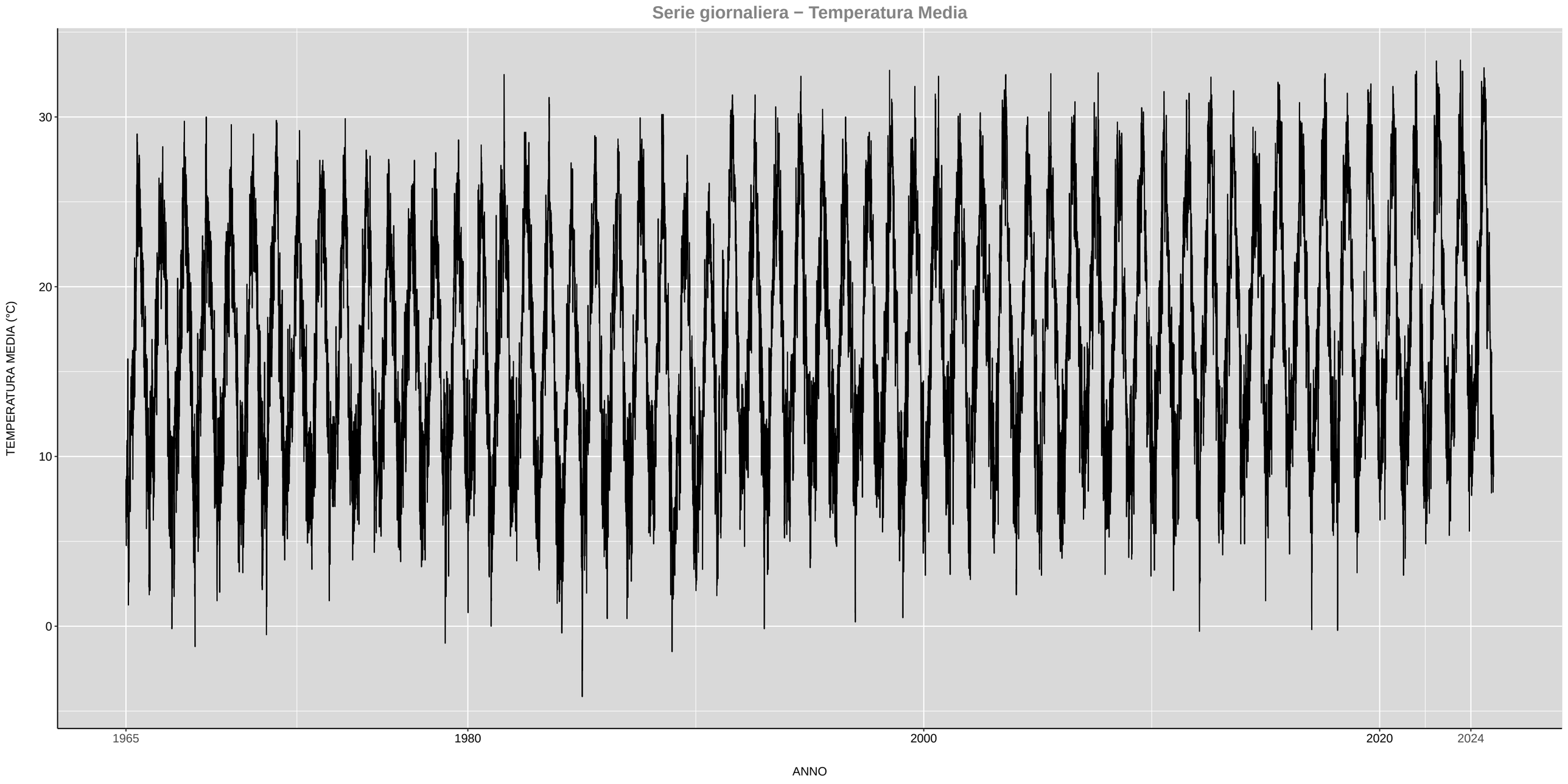

Trend lineare − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1965
1980
2000
2020
2024
ANNO

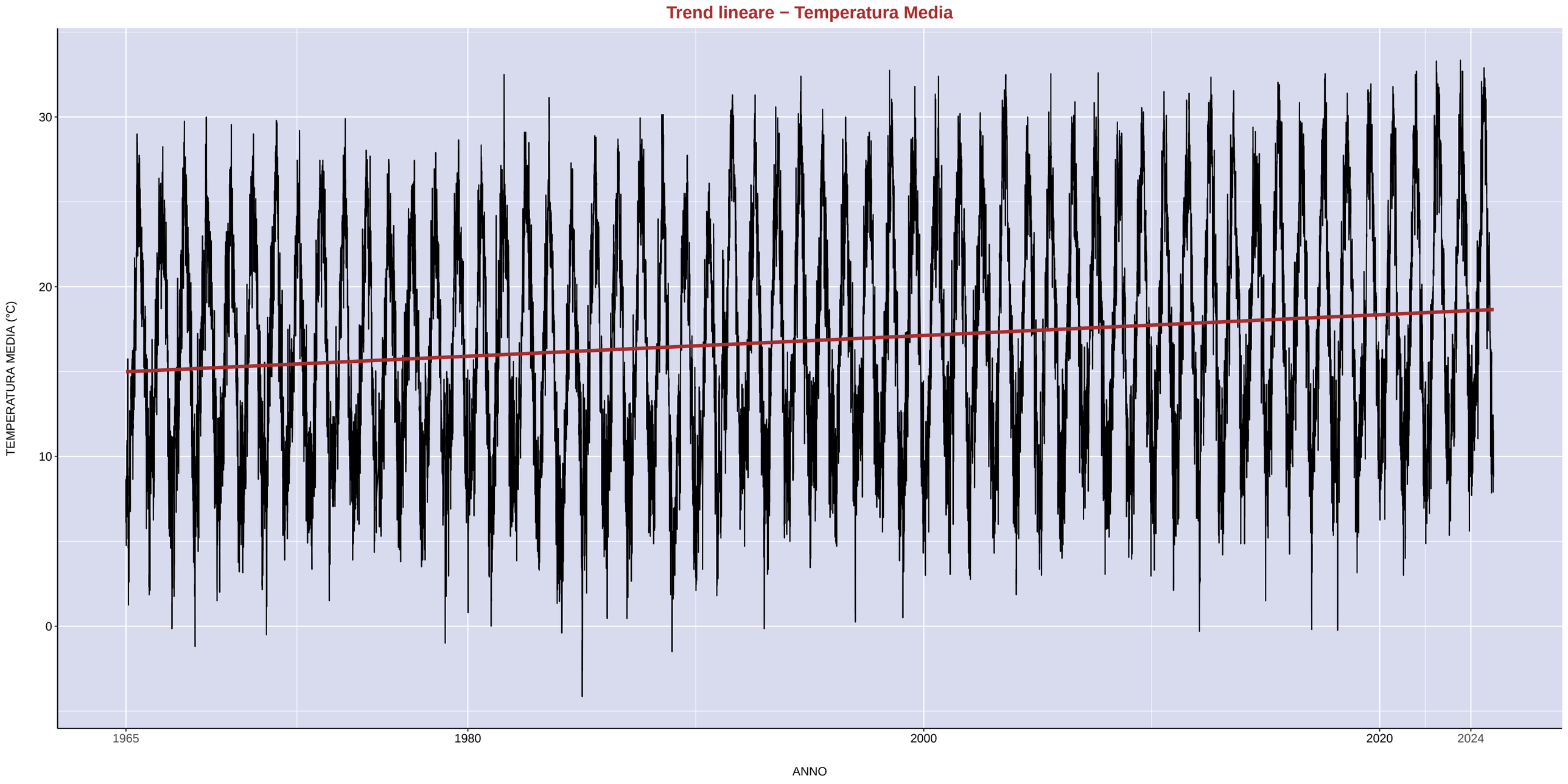

Trend polinomiale grado 2 − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1965
1980
2000
2020
2024
ANNO

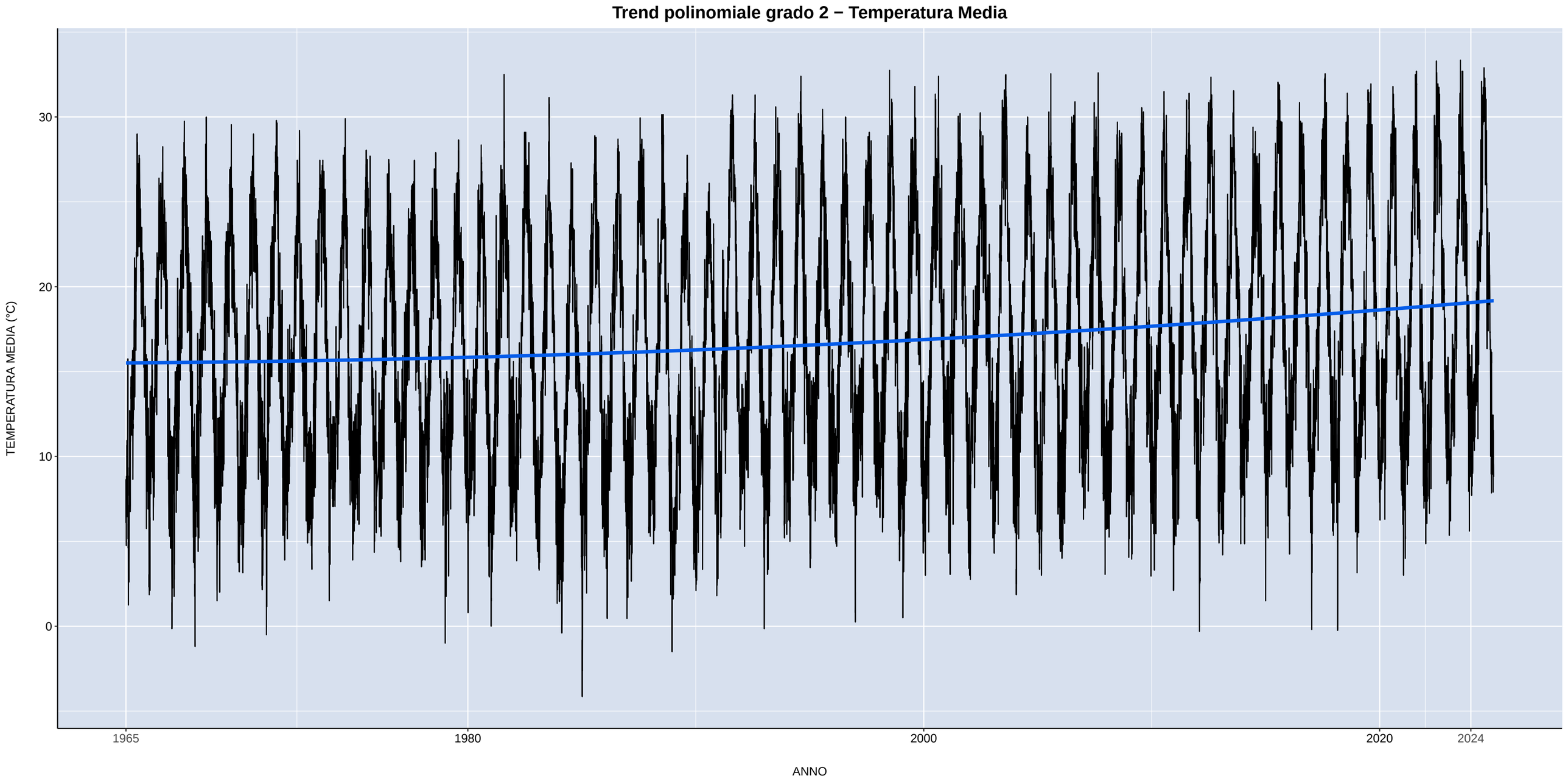

Serie giornaliera − Temperatura Minima
30
20
10
0
TEMPERATURA MINIMA (°C)
1965
1980
2000
2020
2024
ANNO

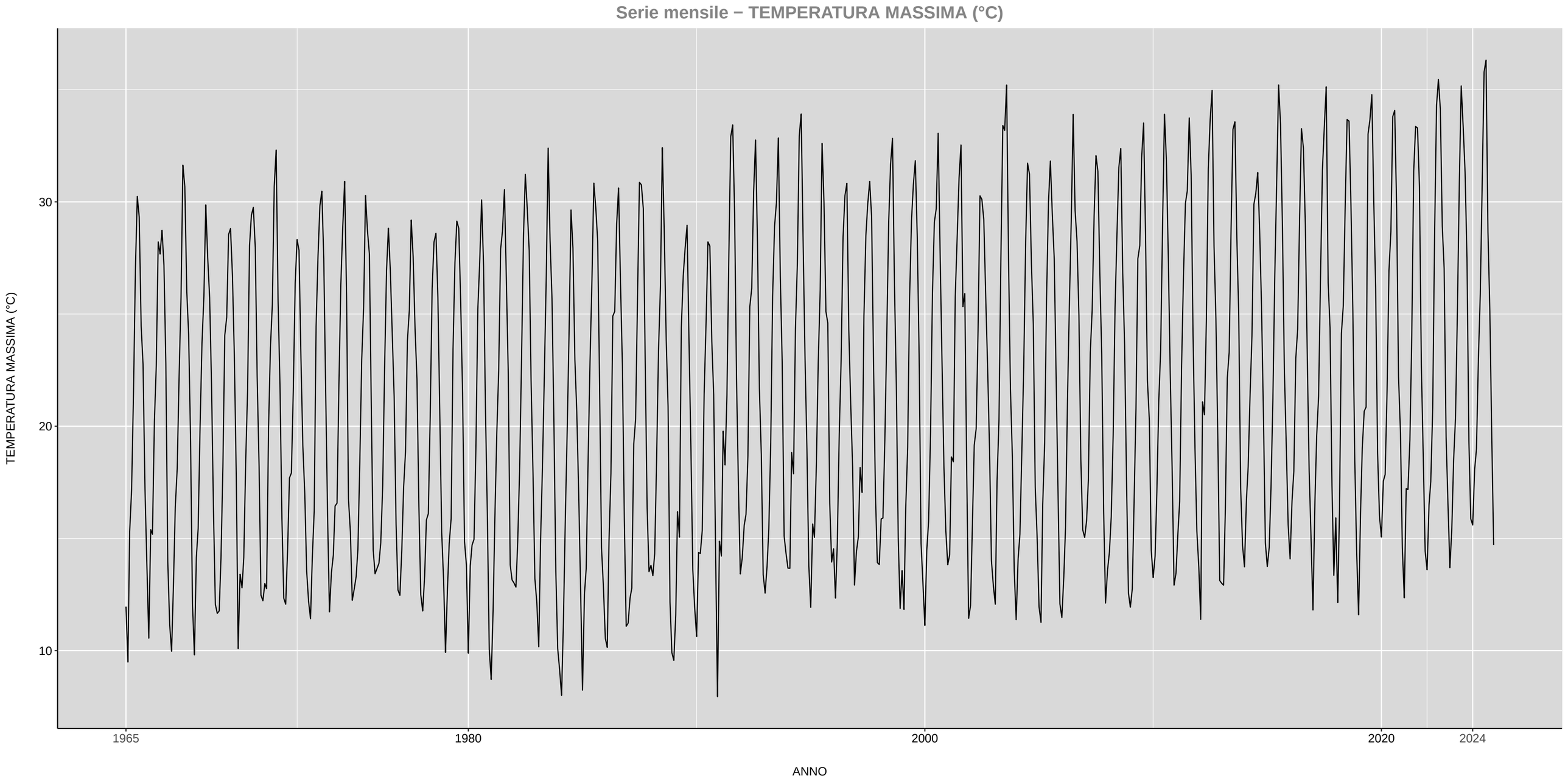

Trend lineare − Temperatura Minima
30
20
10
0
TEMPERATURA MINIMA (°C)
1965
1980
2000
2020
2024
ANNO

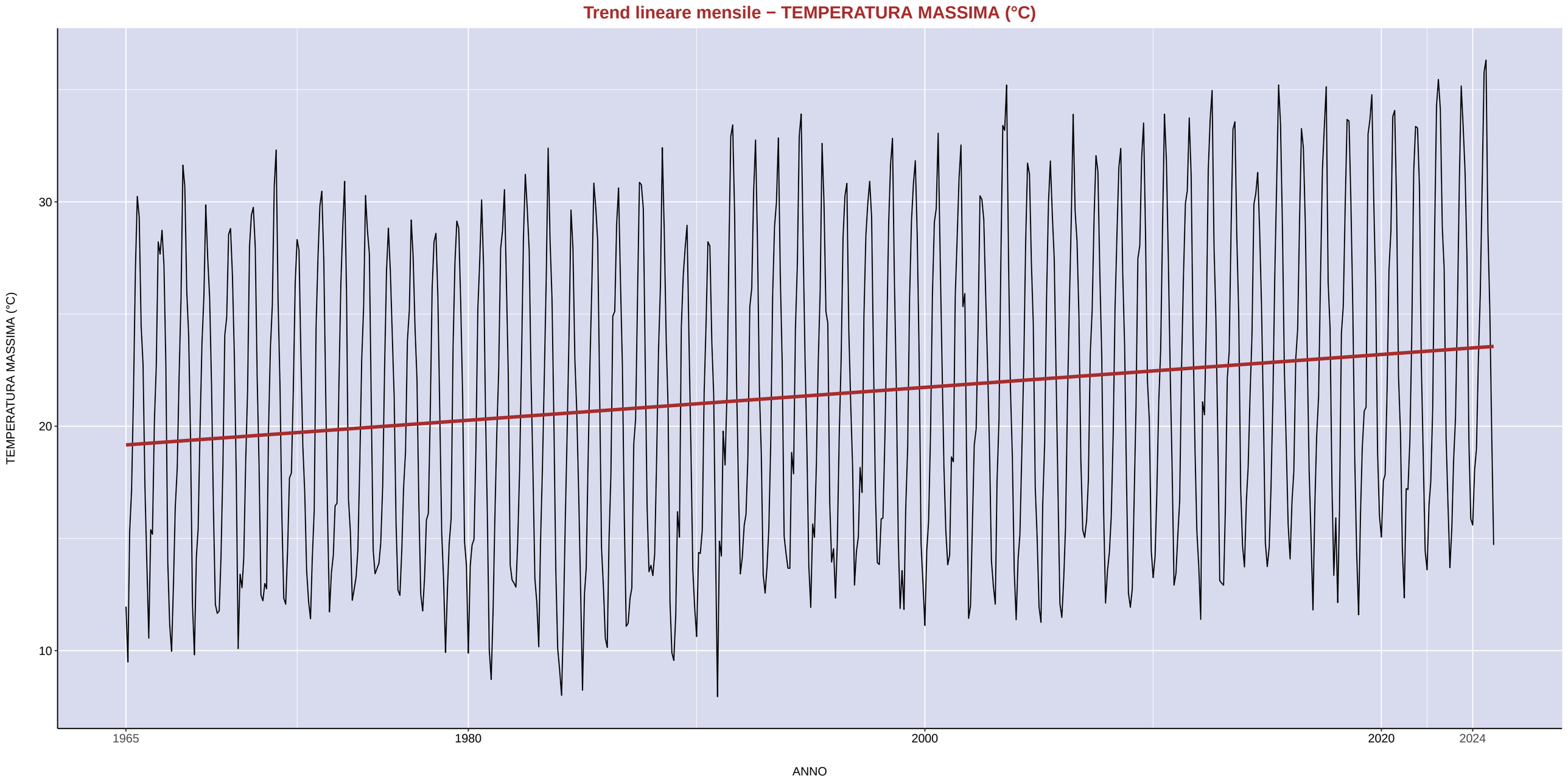

Trend polinomiale grado 2 − Temperatura Minima
30
20
10
0
TEMPERATURA MINIMA (°C)
1965
1980
2000
2020
2024
ANNO

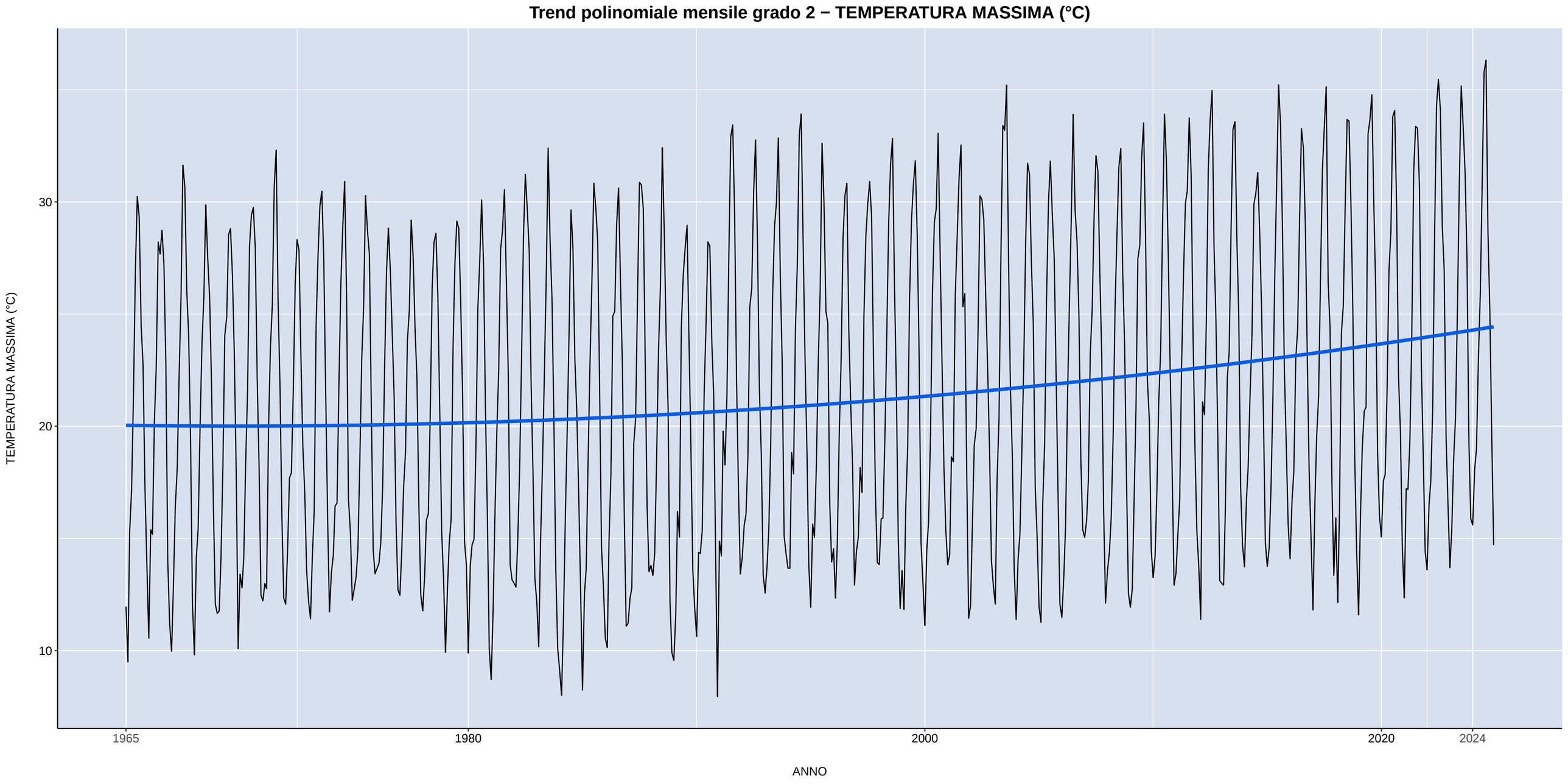

Serie giornaliera − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
1965
1980
2000
2020
2024
ANNO

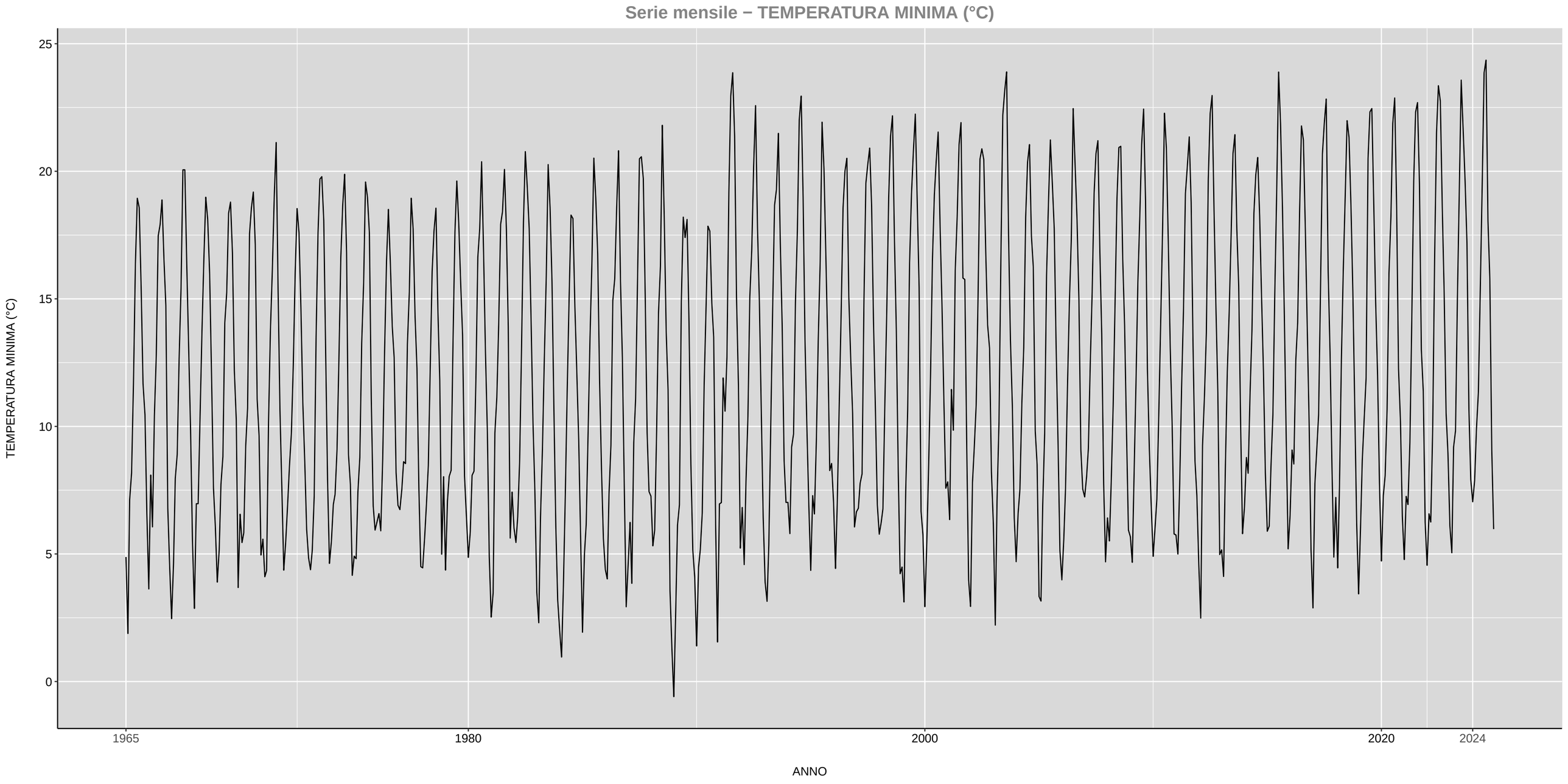

Trend lineare − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
1965
1980
2000
2020
2024
ANNO


Trend polinomiale grado 2 − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
1965
1980
2000
2020
2024
ANNO


Serie mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
1965
1980
2000
2020
2024
ANNO

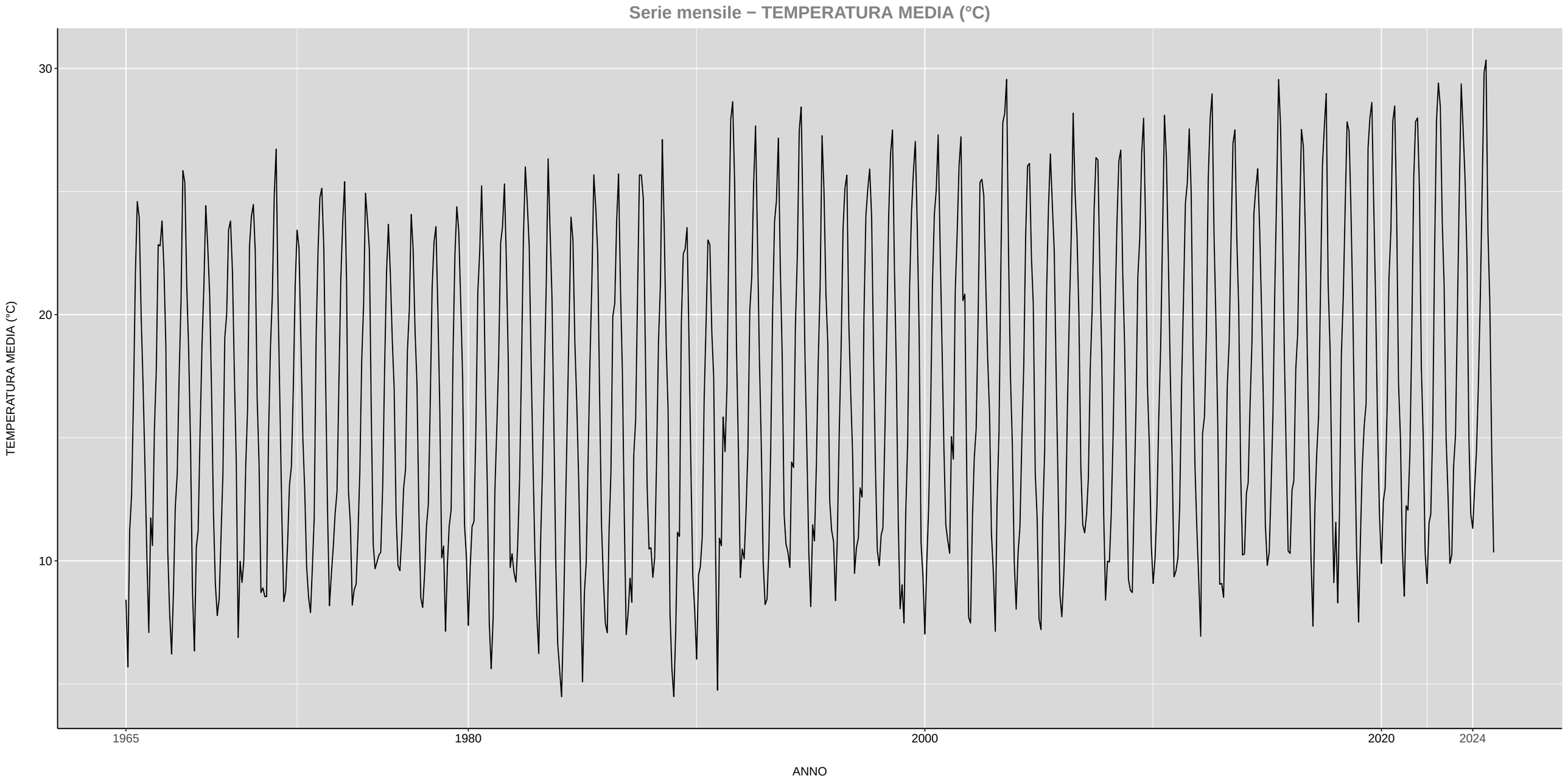

Trend lineare mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
1965
1980
2000
2020
2024
ANNO


Trend polinomiale mensile grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
1965
1980
2000
2020
2024
ANNO

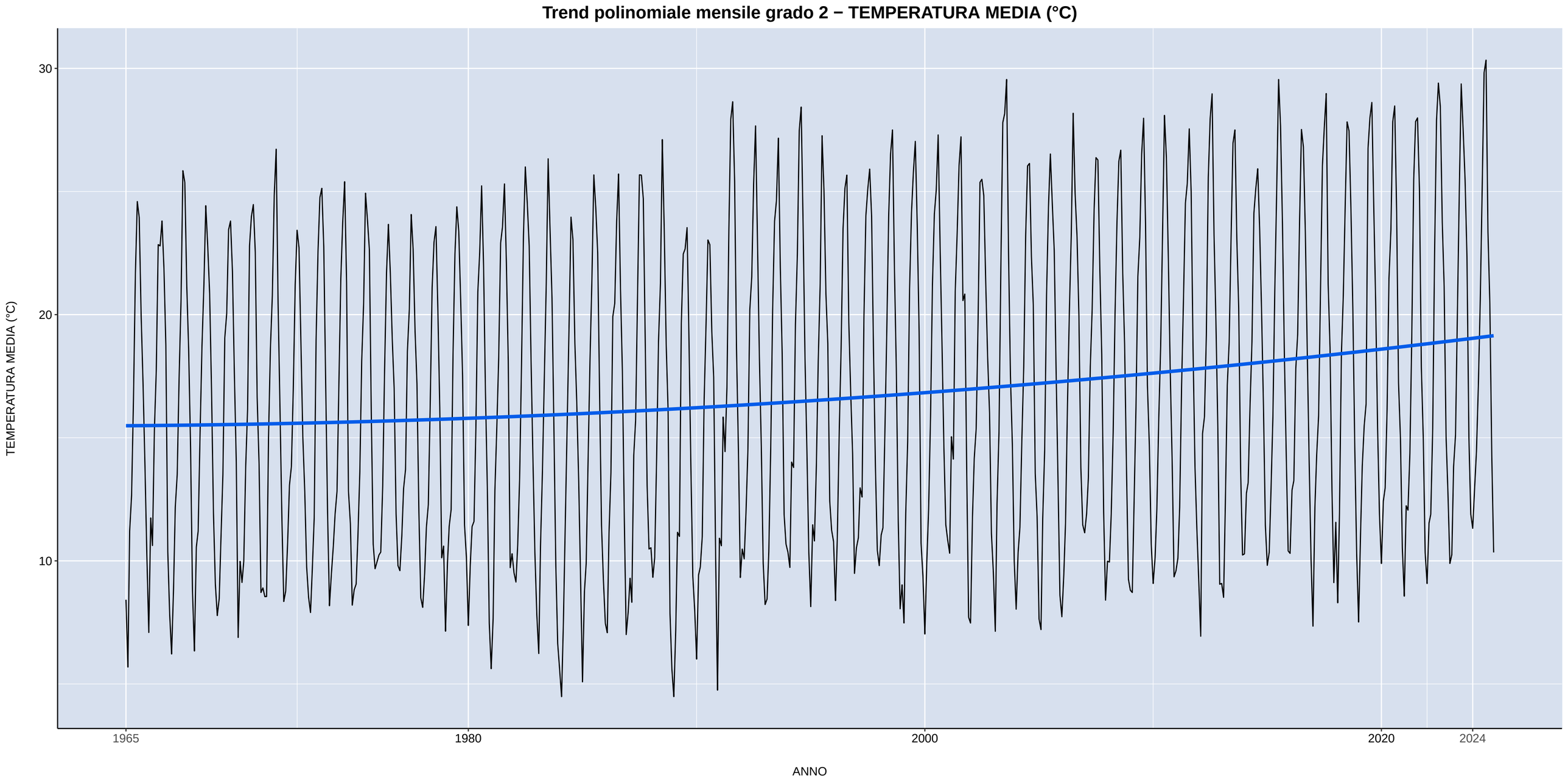

Serie mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
25
20
15
10
5
0
1965
1980
2000
2020
2024
ANNO

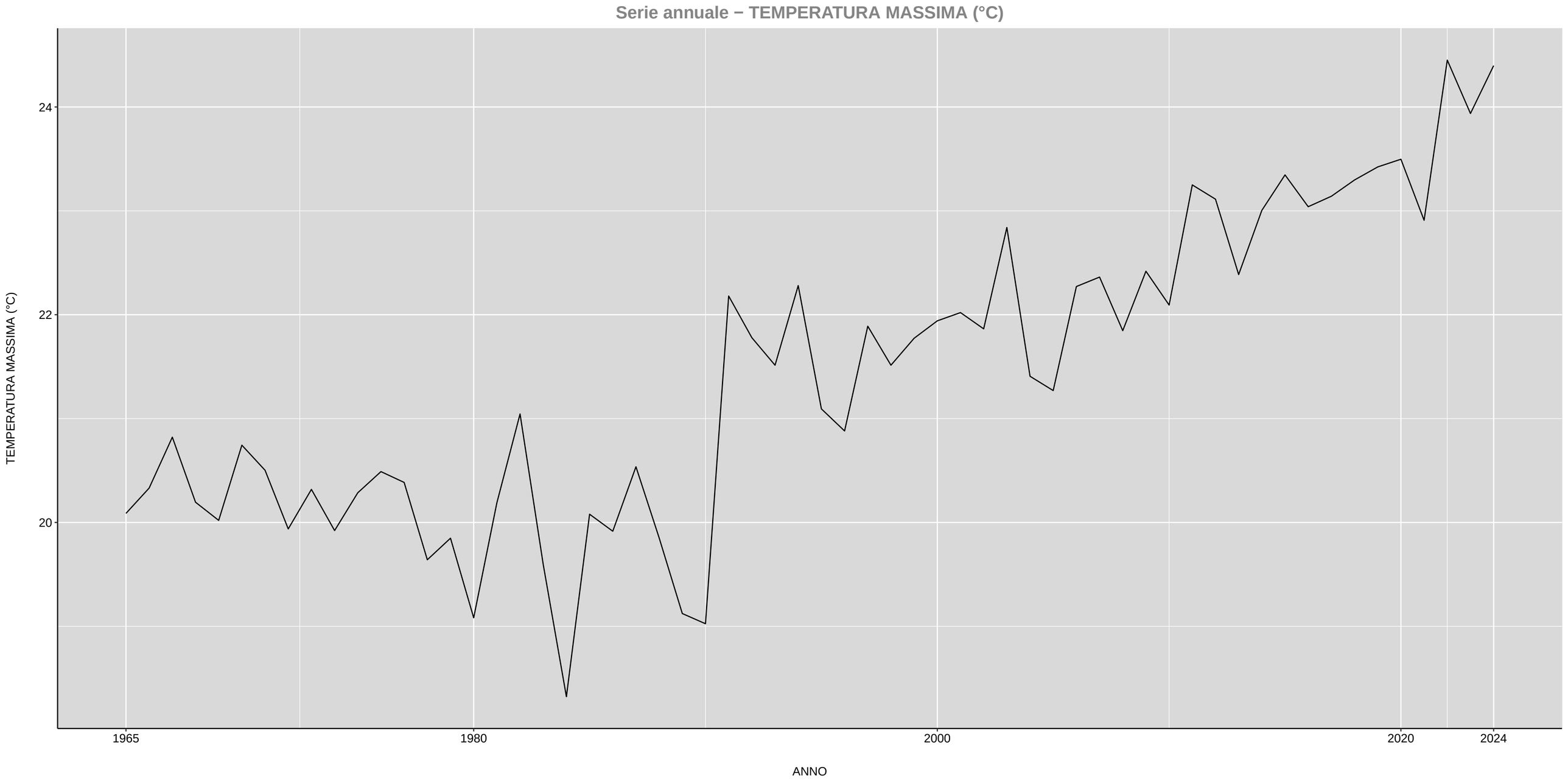
Trend lineare mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
25
20
15
10
5
0
1965
1980
2000
2020
2024
ANNO

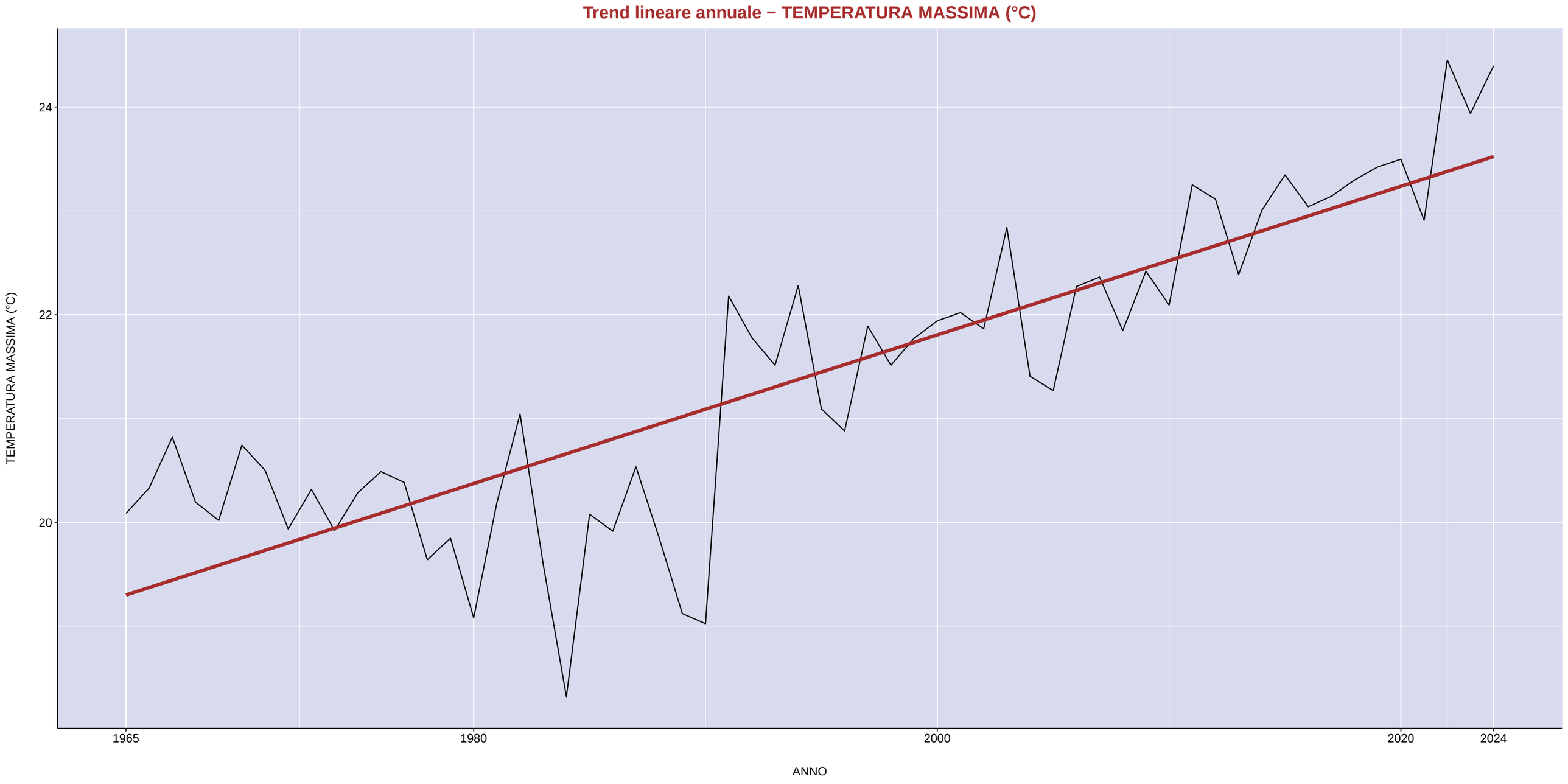
Trend polinomiale mensile grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
25
20
15
10
5
0
1965
1980
2000
2020
2024
ANNO

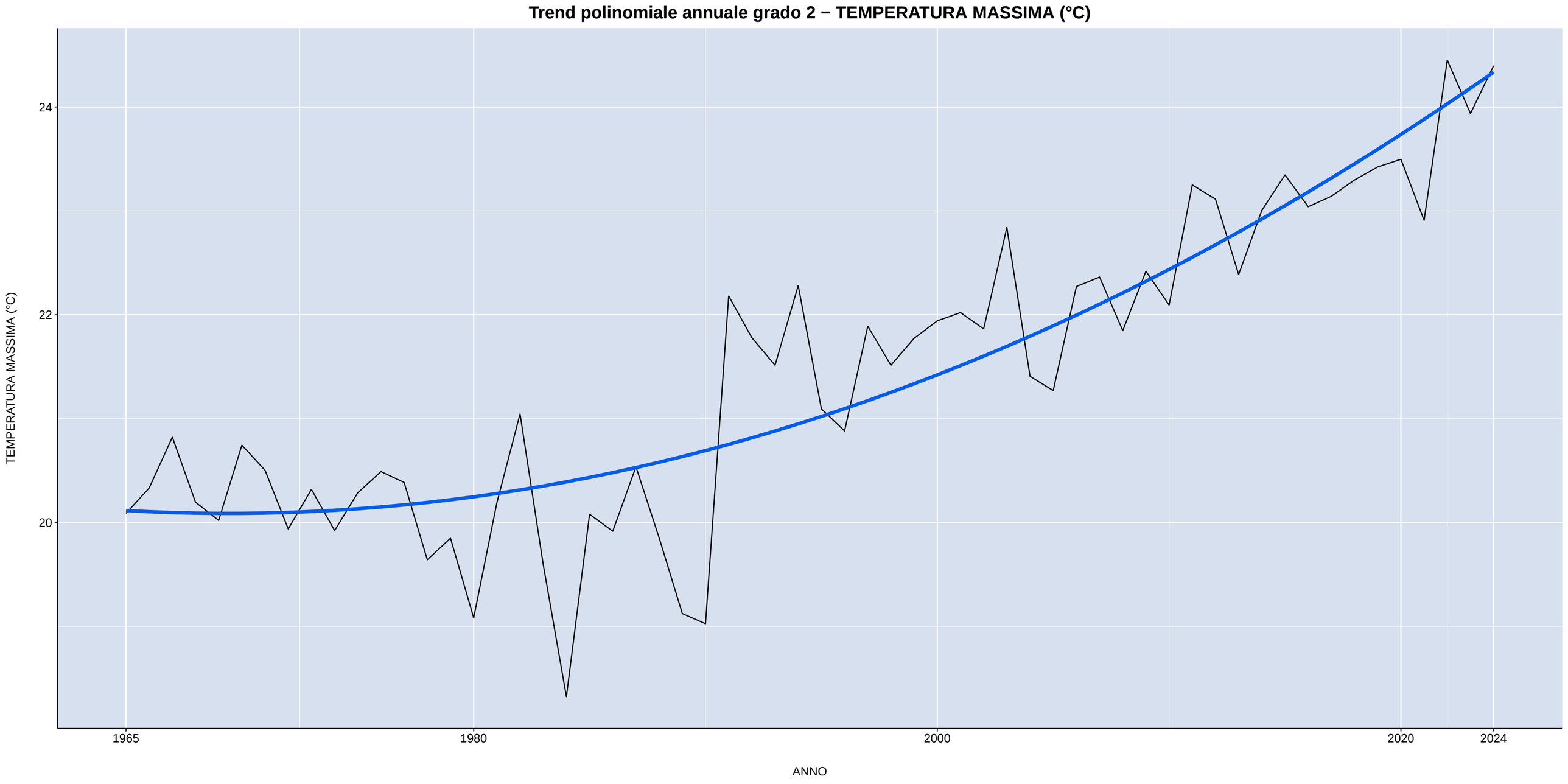

Serie mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
1965
1980
2000
2020
2024
ANNO

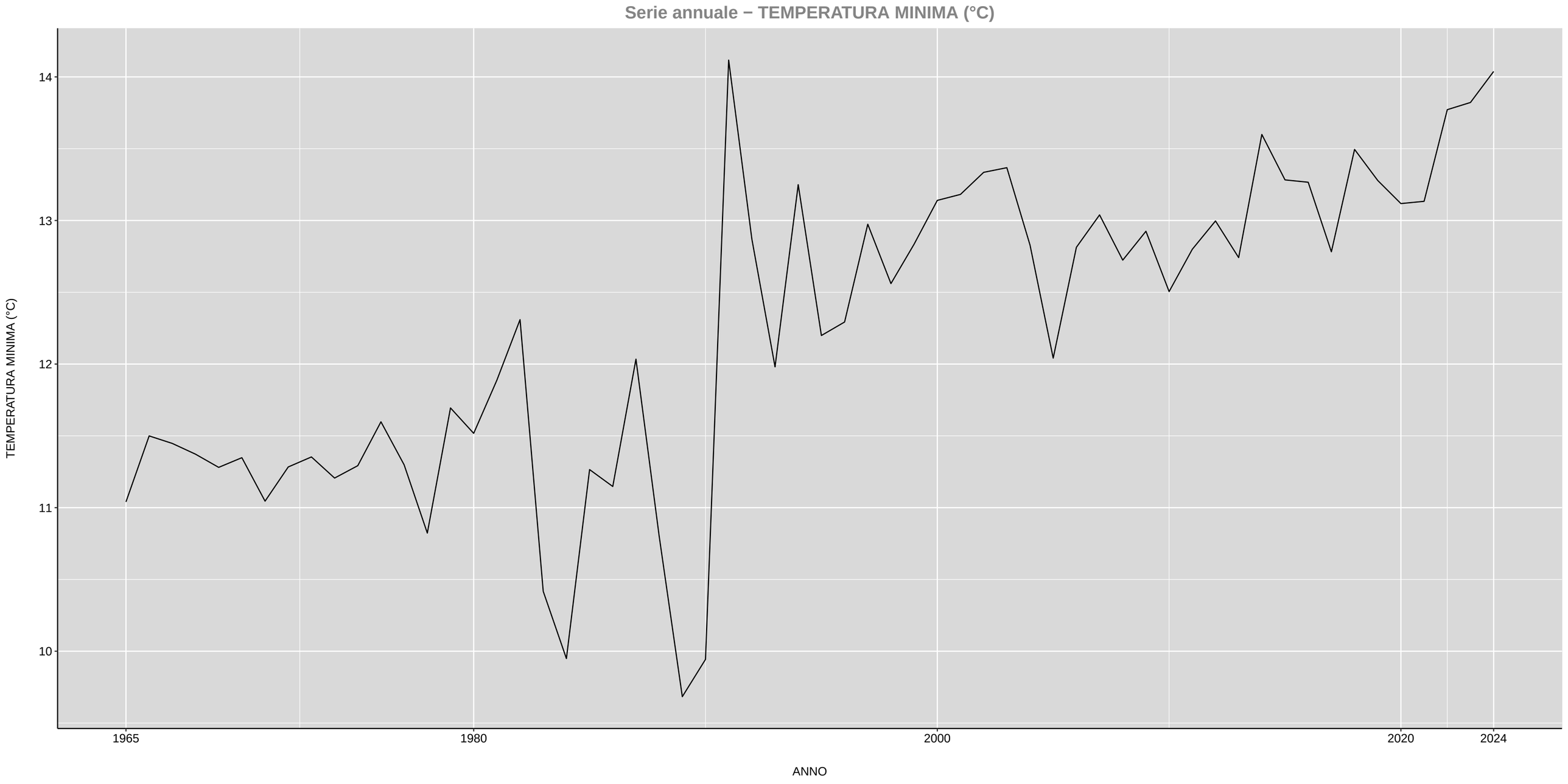
Trend lineare mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
1965
1980
2000
2020
2024
ANNO

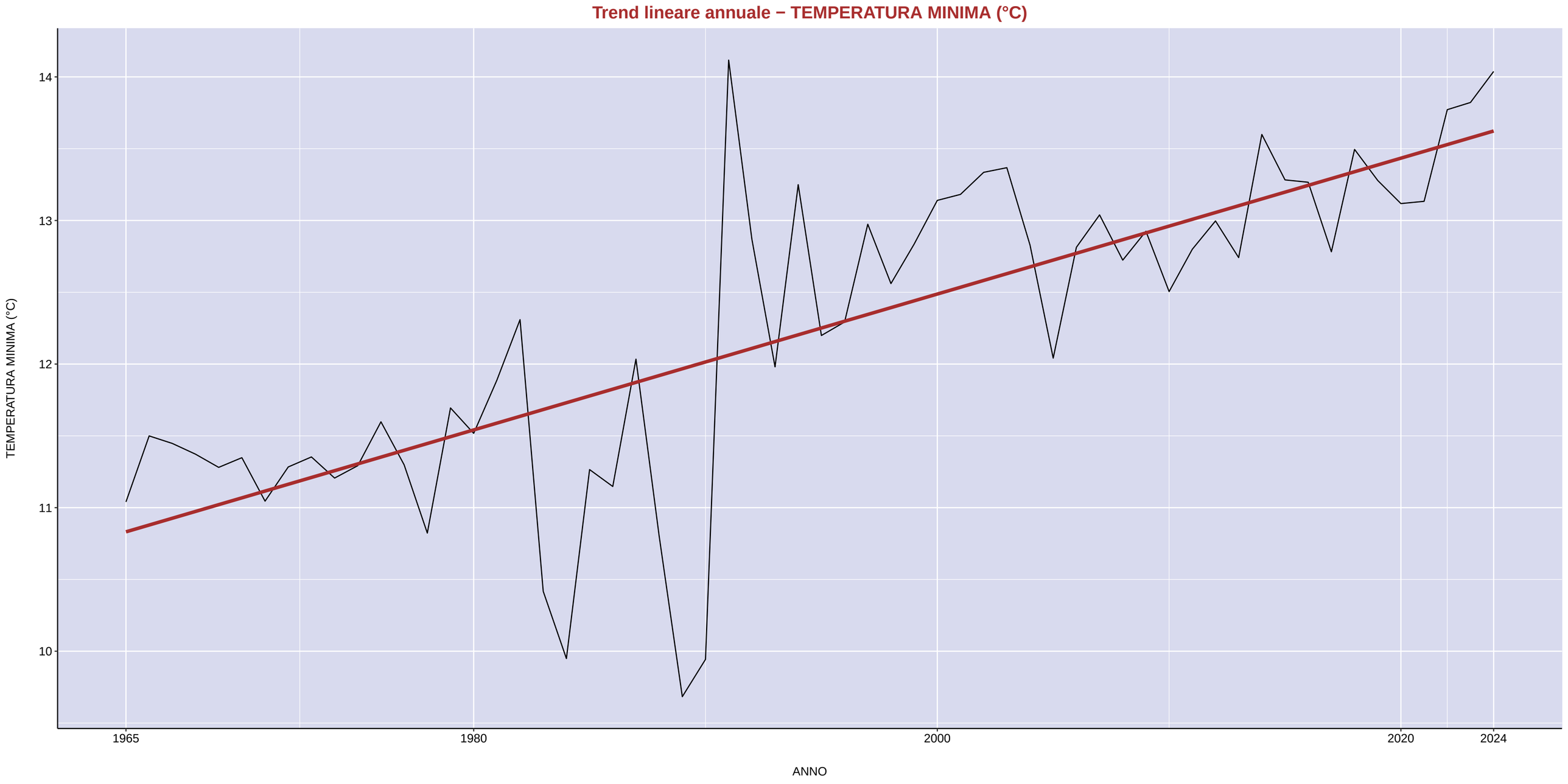

Trend polinomiale mensile grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
1965
1980
2000
2020
2024
ANNO

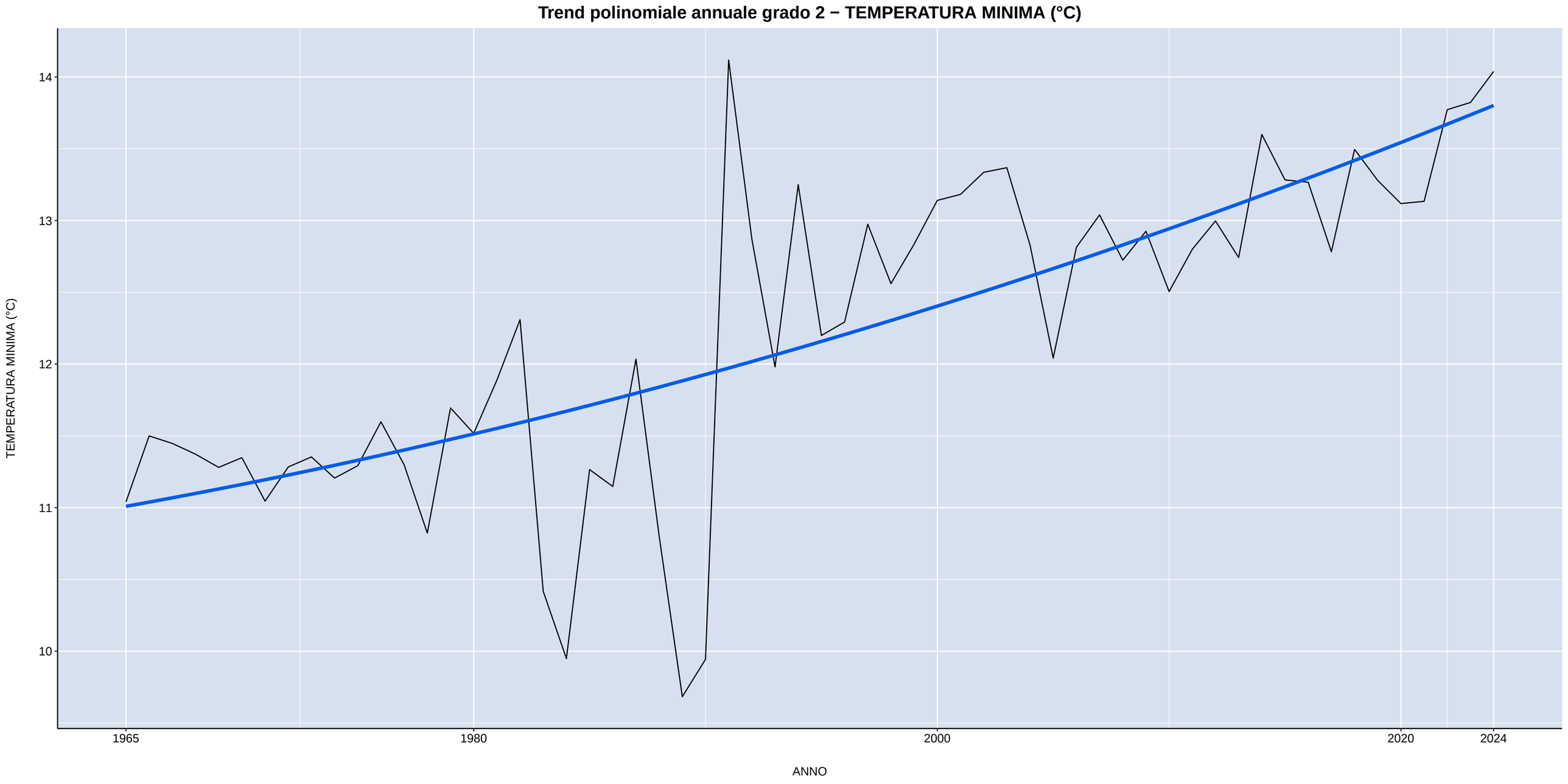

Serie annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
24
22
20
1965
1980
2000
2020
2024
ANNO

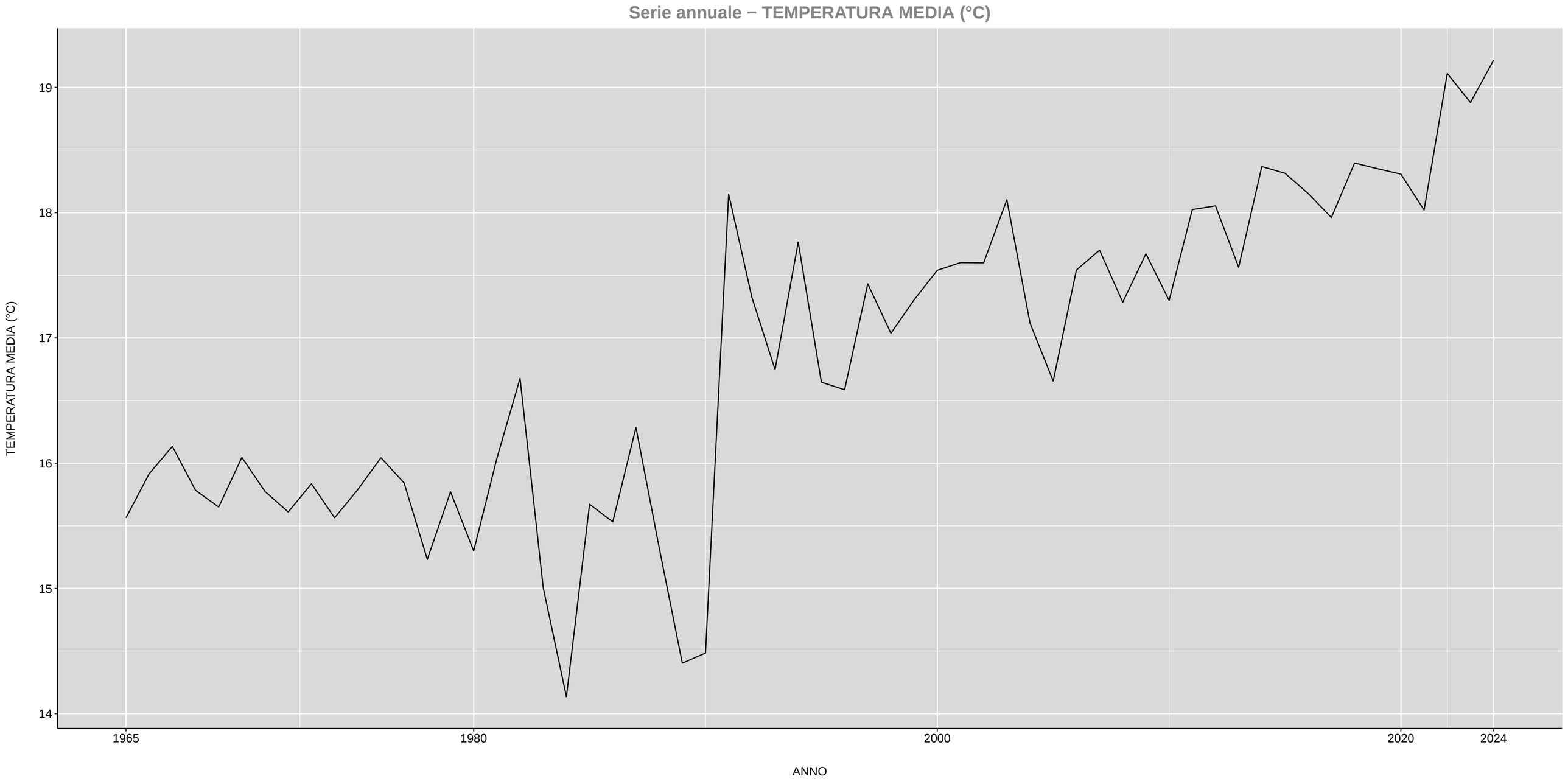

Trend lineare annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
24
22
20
1965
1980
2000
2020
2024
ANNO

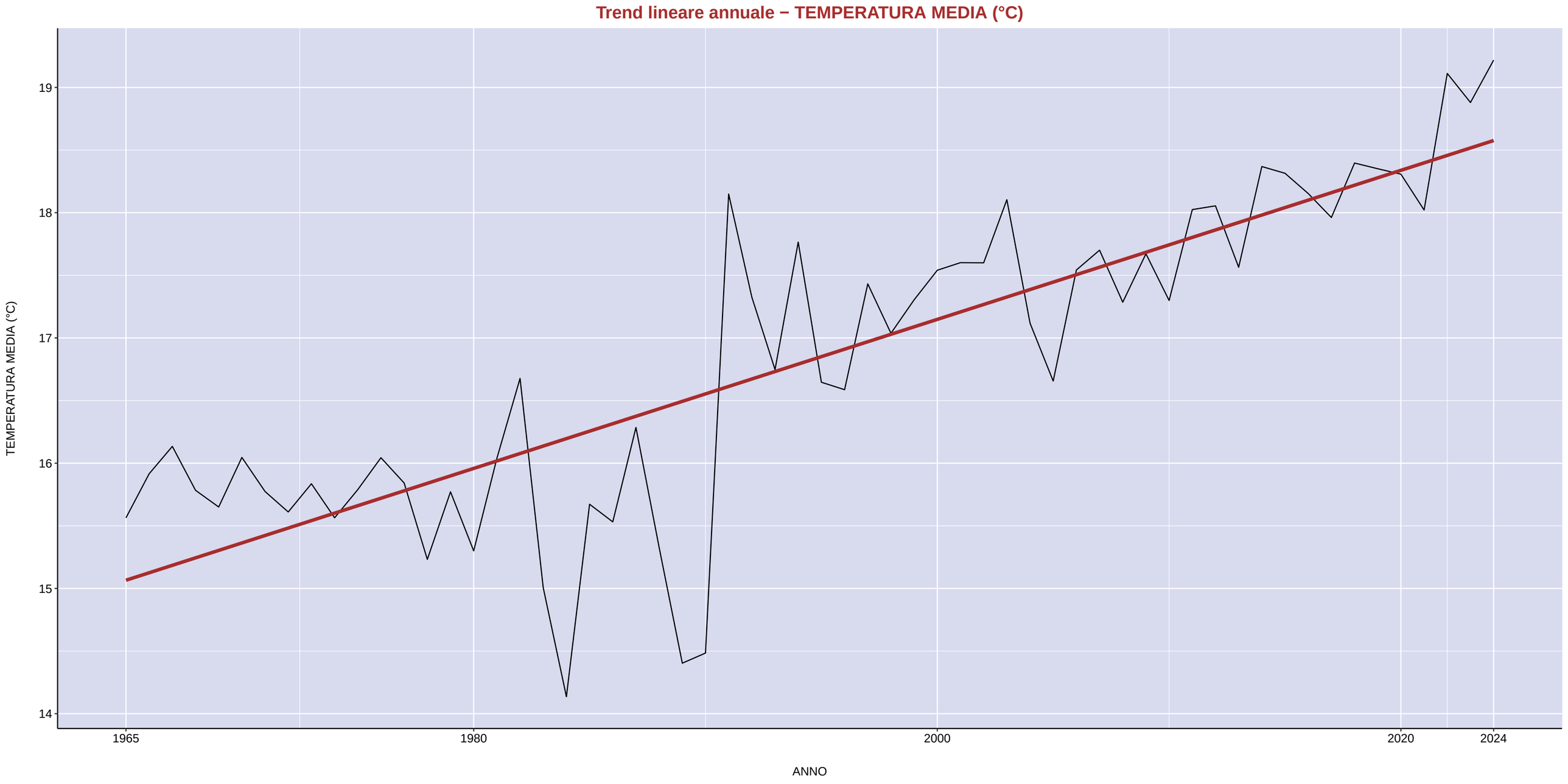

Trend polinomiale annuale grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
24
22
20
1965
1980
2000
2020
2024
ANNO

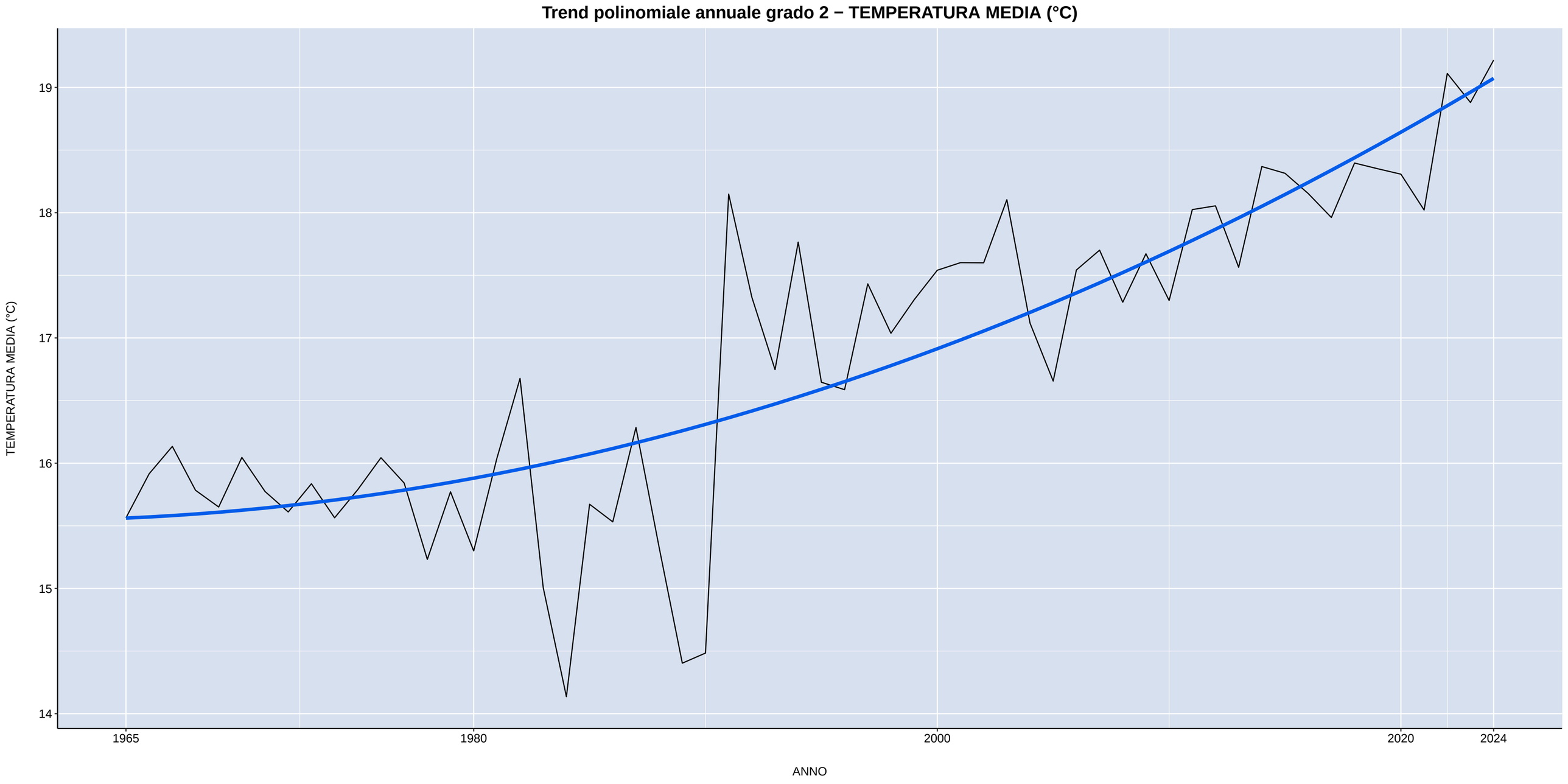

Serie annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
14
13
12
11
10
1965
1980
2000
2020
2024
ANNO

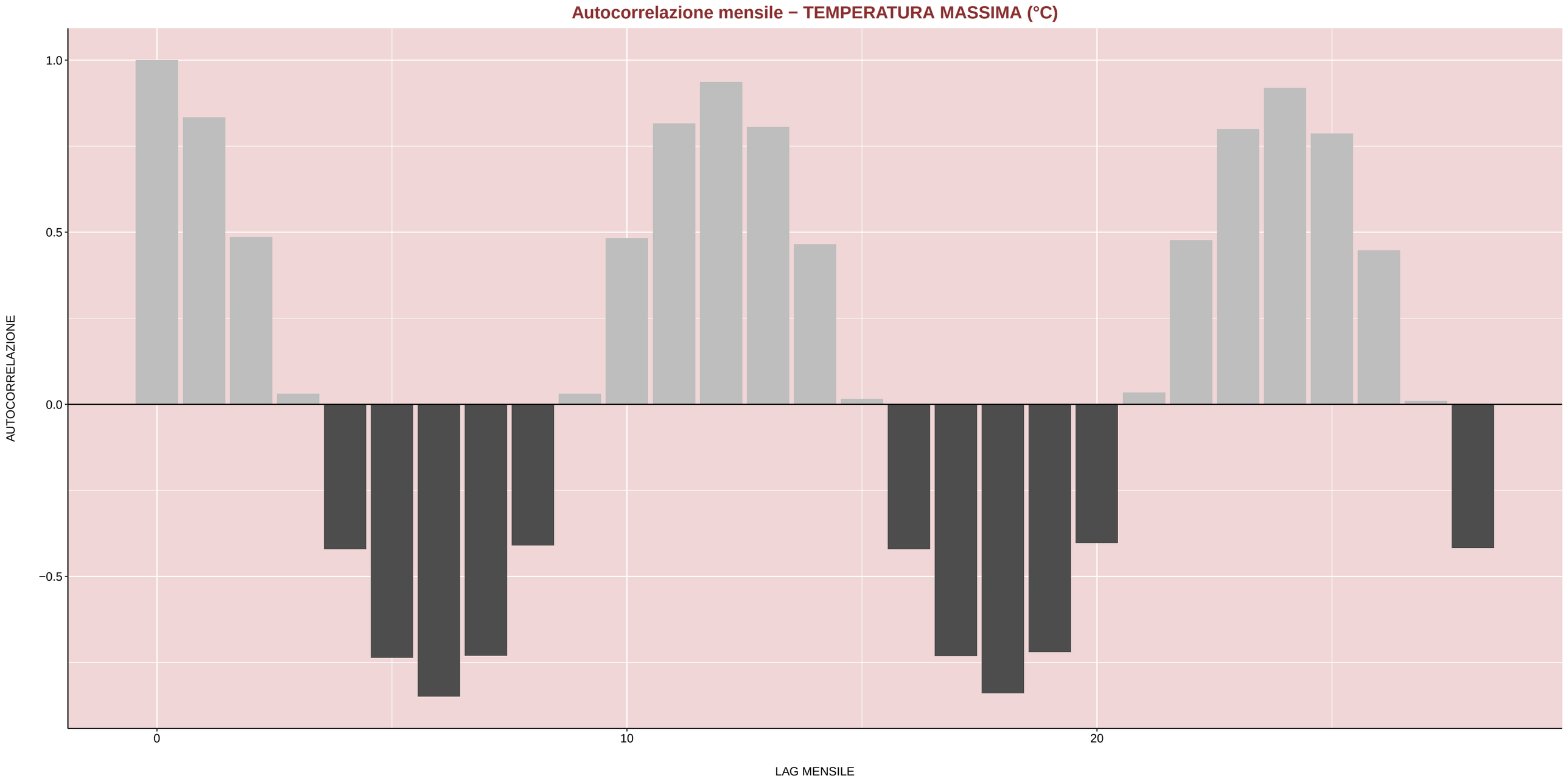

Trend lineare annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
14
13
12
11
10
1965
1980
2000
2020
2024
ANNO

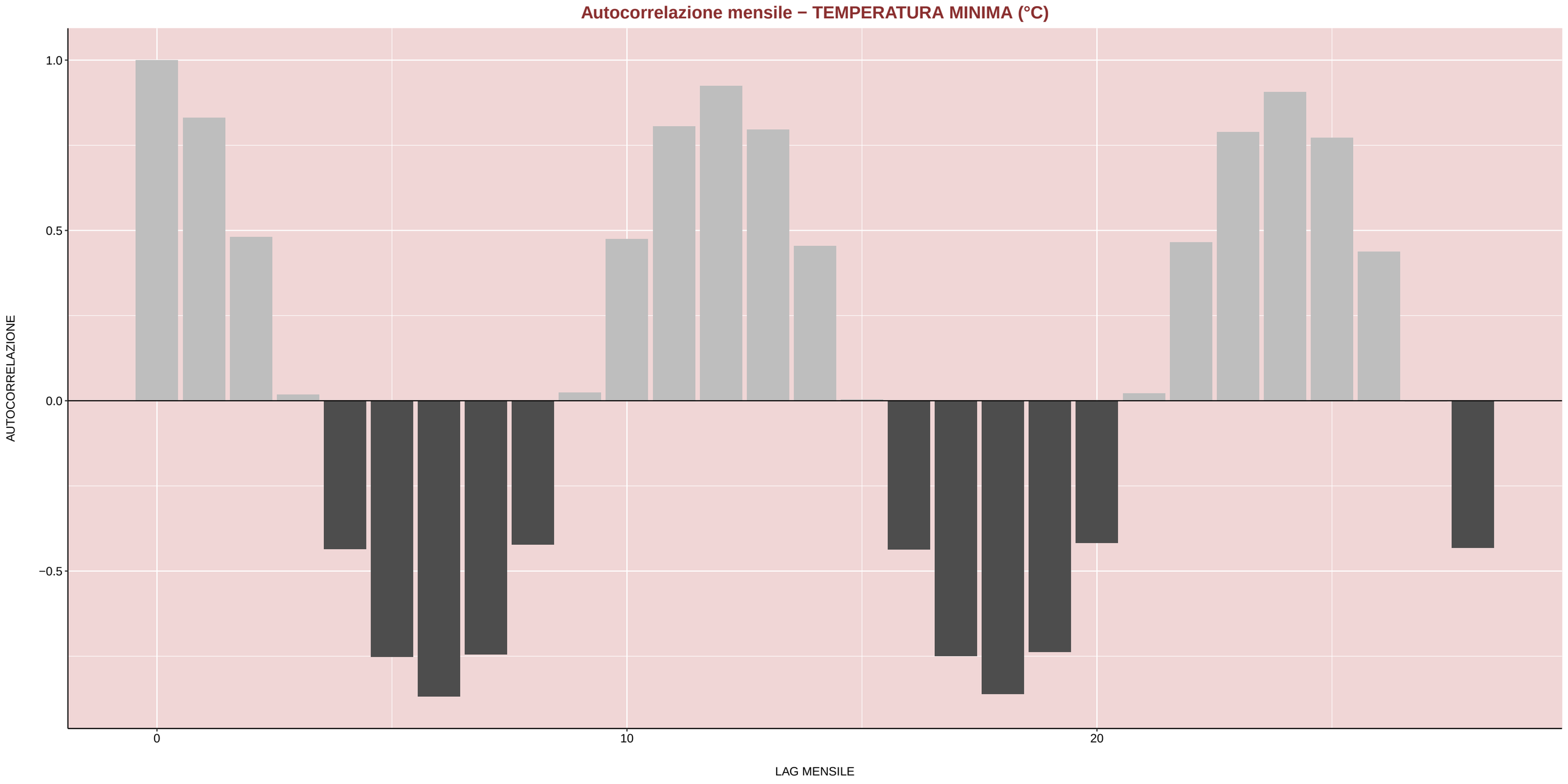

Trend polinomiale annuale grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
14
13
12
11
10
1965
1980
2000
2020
2024
ANNO

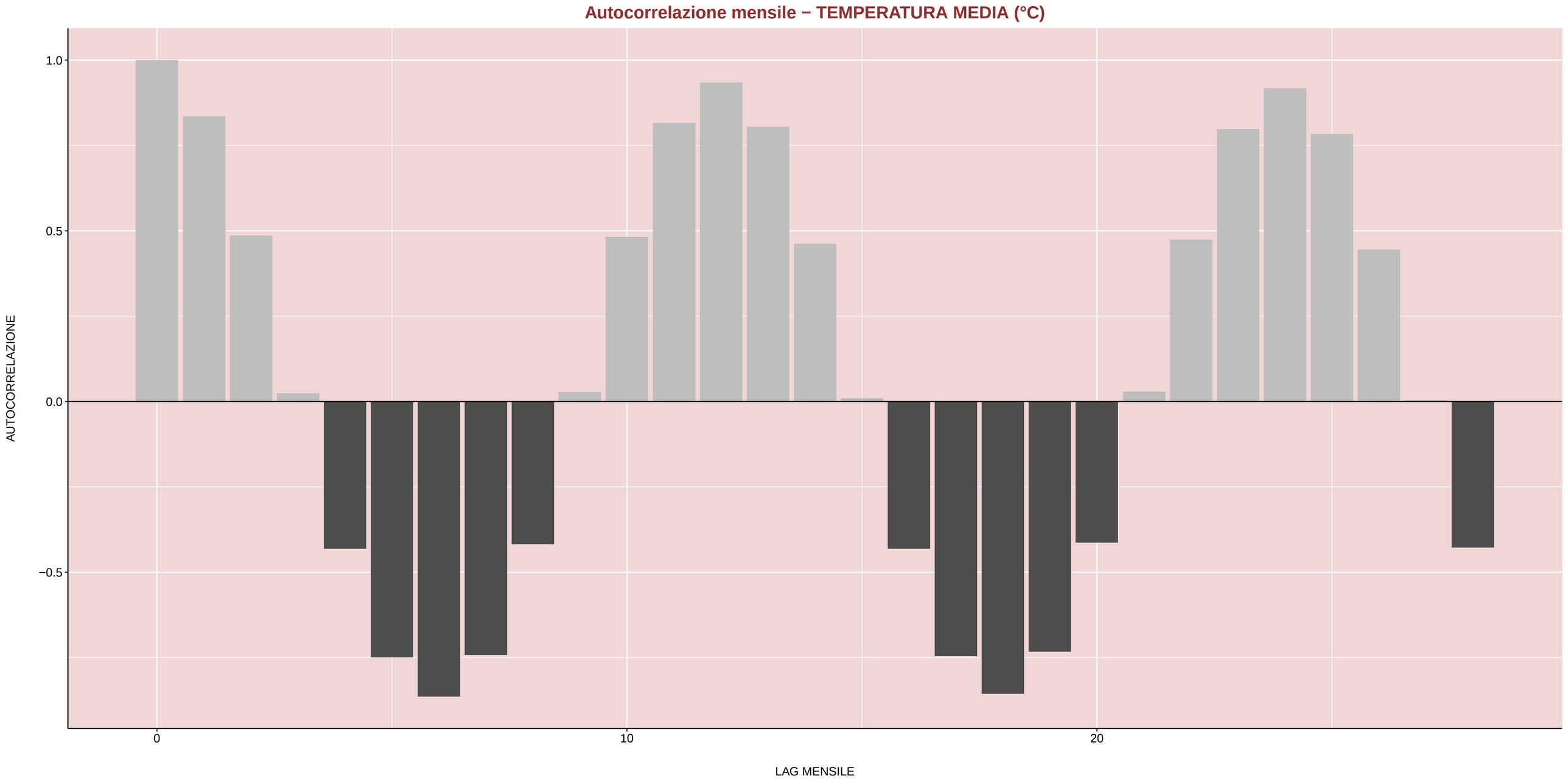

Serie annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
19
18
17
16
15
14
1965
1980
2000
2020
2024
ANNO

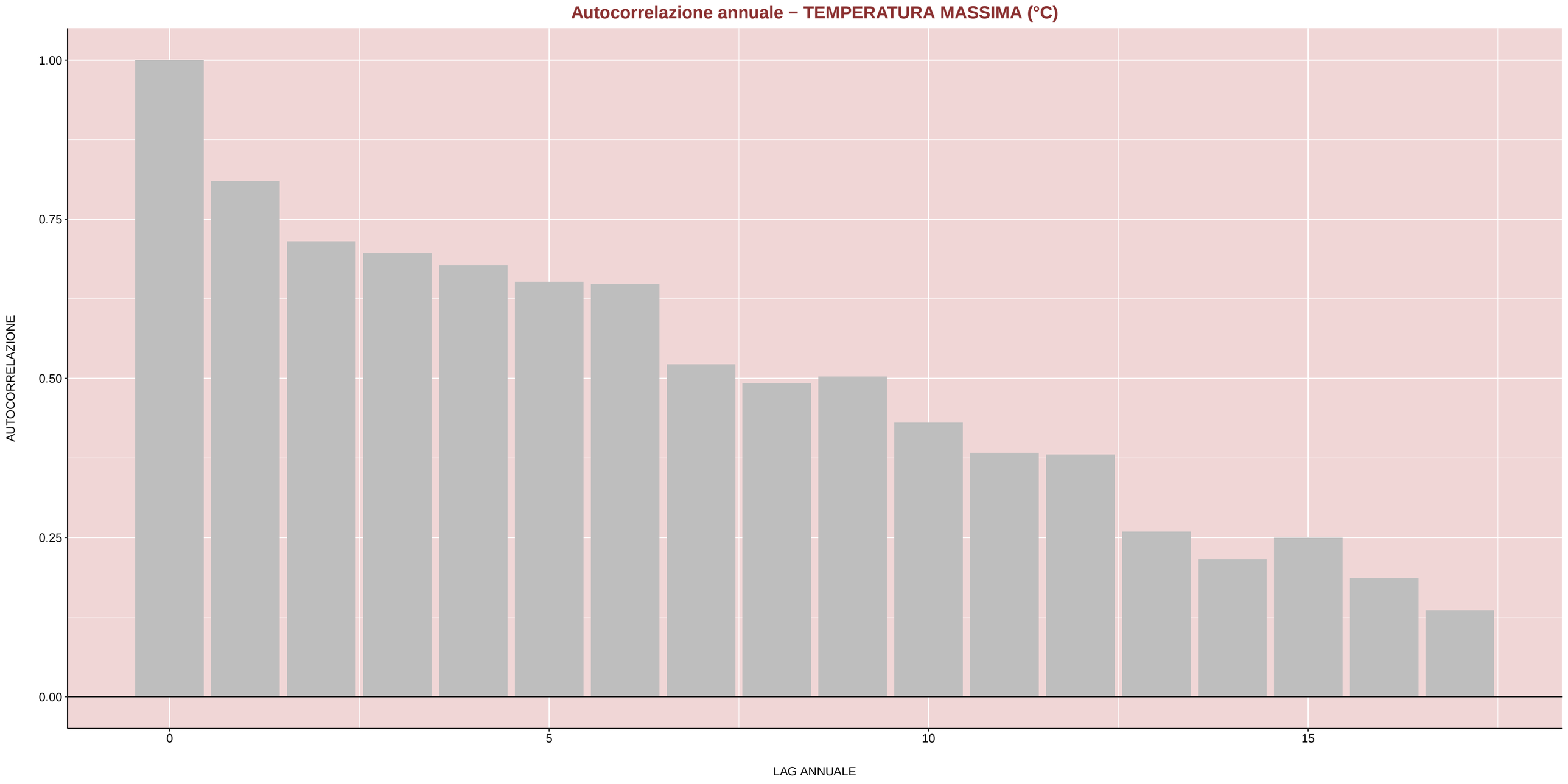

Trend lineare annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
19
18
17
16
15
14
1965
1980
2000
2020
2024
ANNO

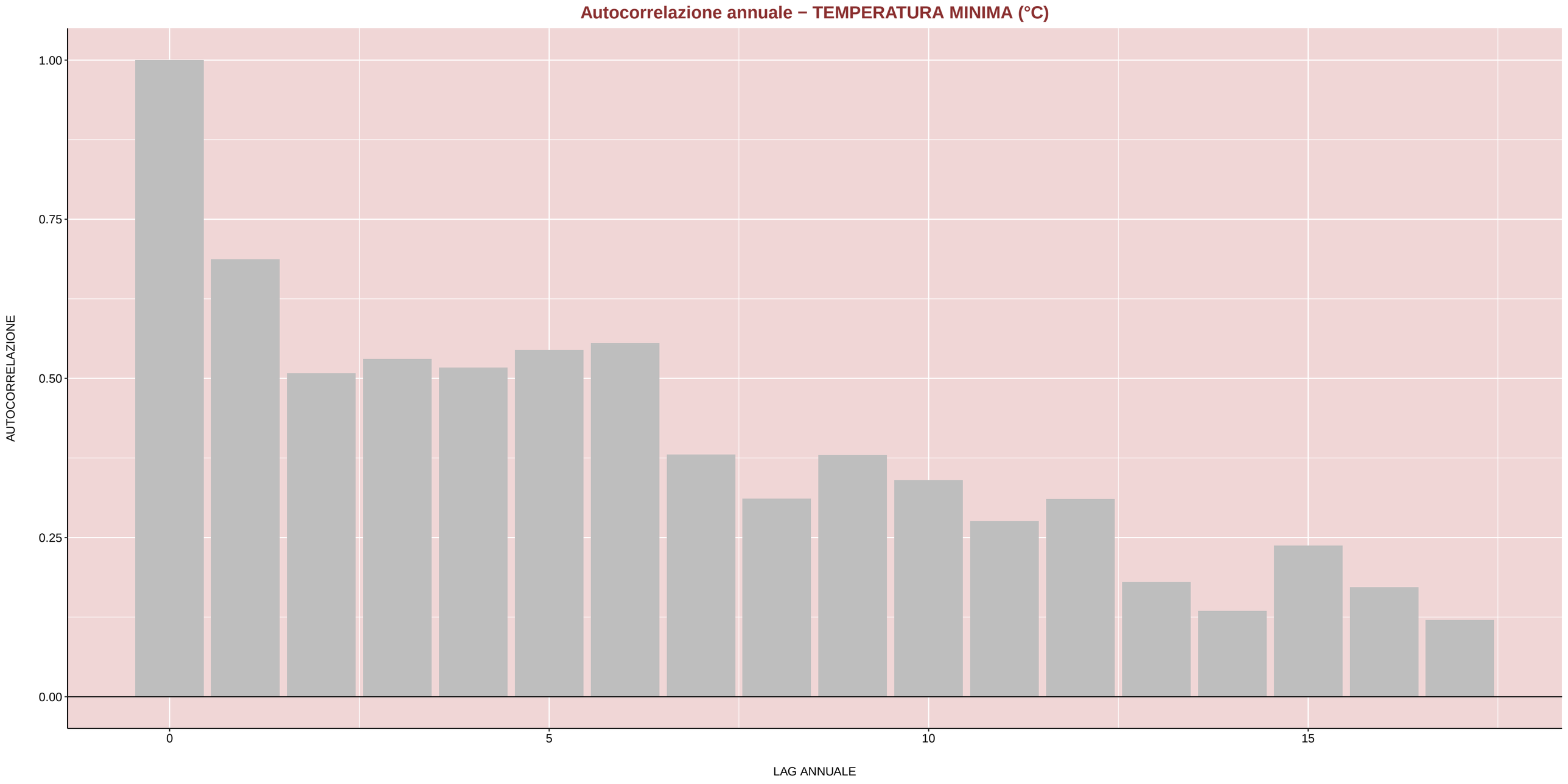

Trend polinomiale annuale grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
19
18
17
16
15
14
1965
1980
2000
2020
2024
ANNO

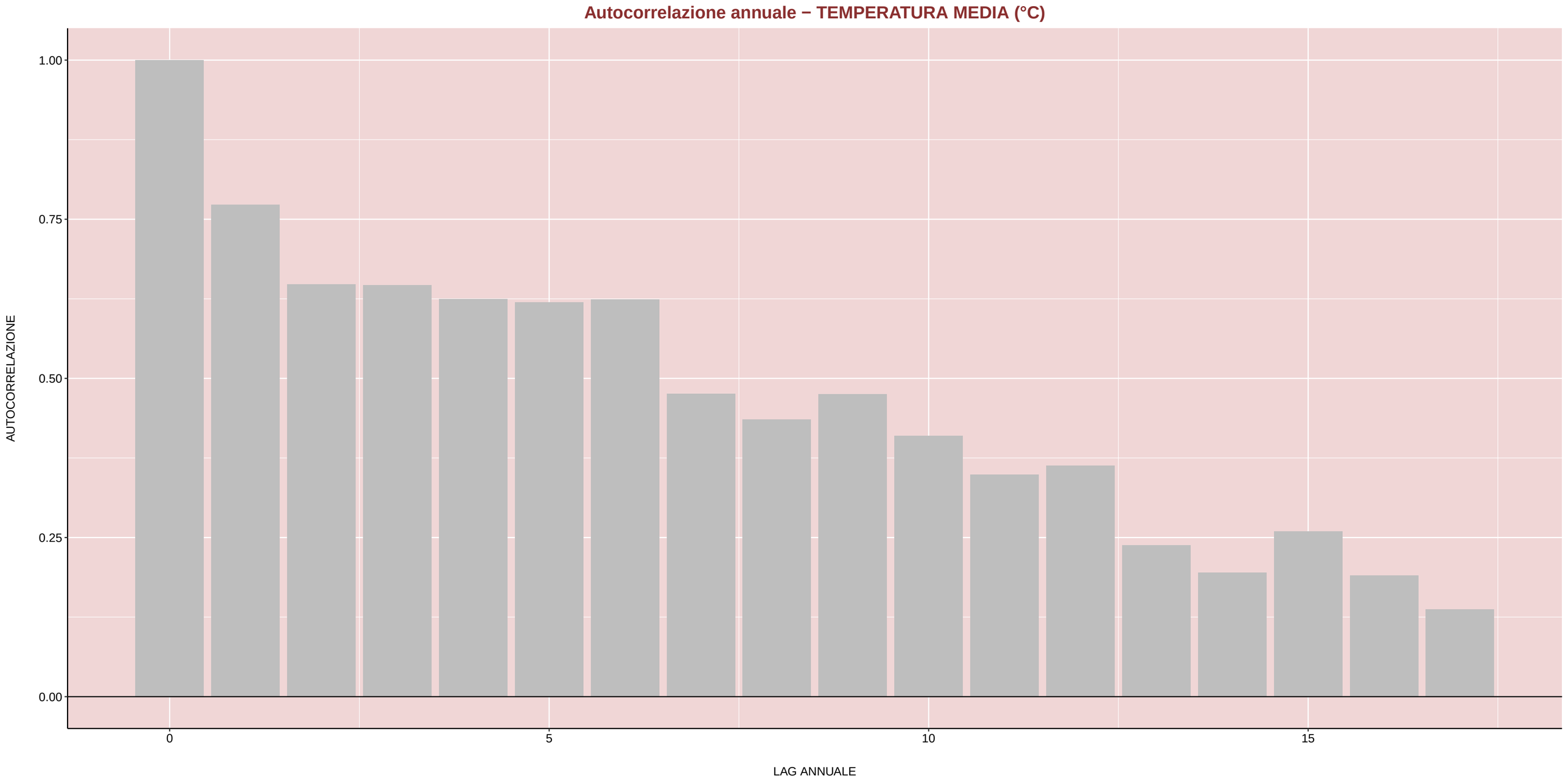

Autocorrelazione mensile − TEMPERATURA MASSIMA (°C)
1.0
0.5
0.0
−0.5
AUTOCORRELAZIONE
0
10
20
LAG MENSILE


Autocorrelazione mensile − TEMPERATURA MINIMA (°C)
1.0
0.5
0.0
−0.5
AUTOCORRELAZIONE
0
10
20
LAG MENSILE

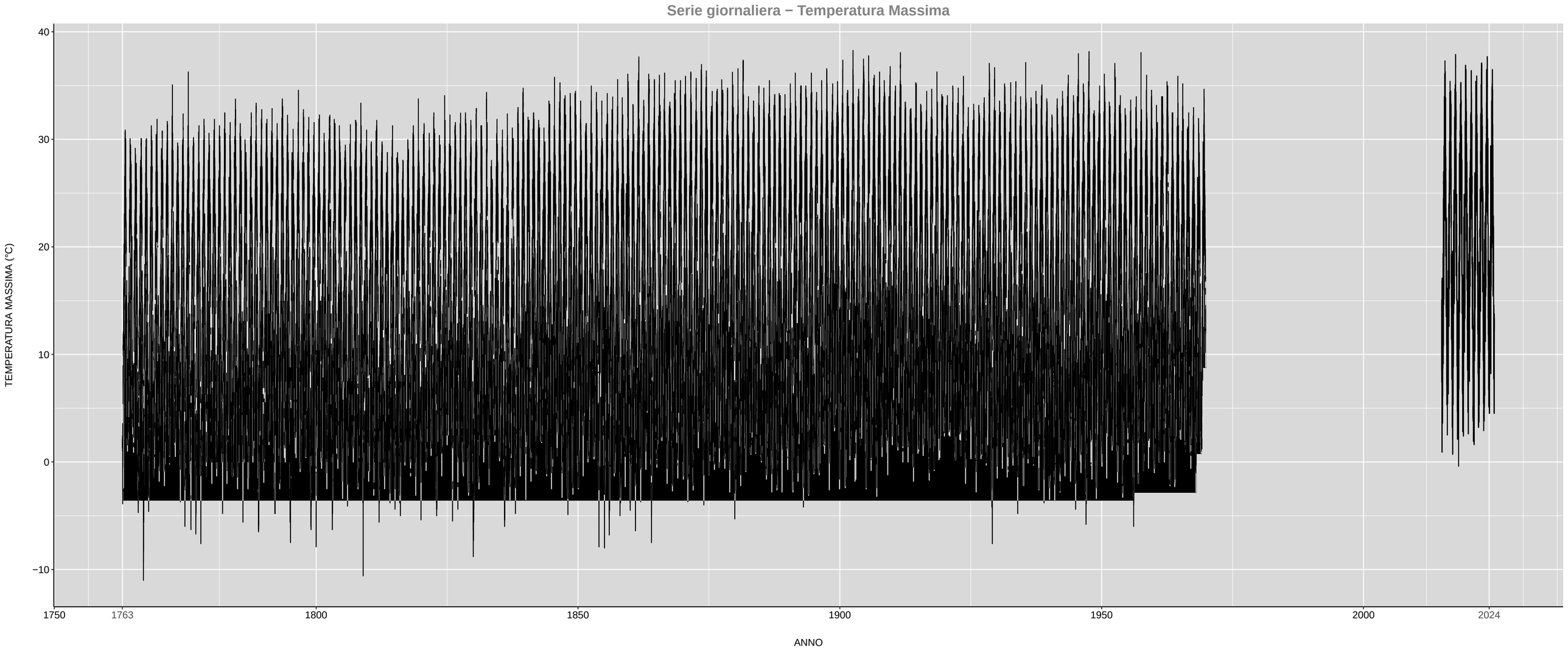

Autocorrelazione mensile − TEMPERATURA MEDIA (°C)
1.0
0.5
0.0
−0.5
AUTOCORRELAZIONE
0
10
20
LAG MENSILE

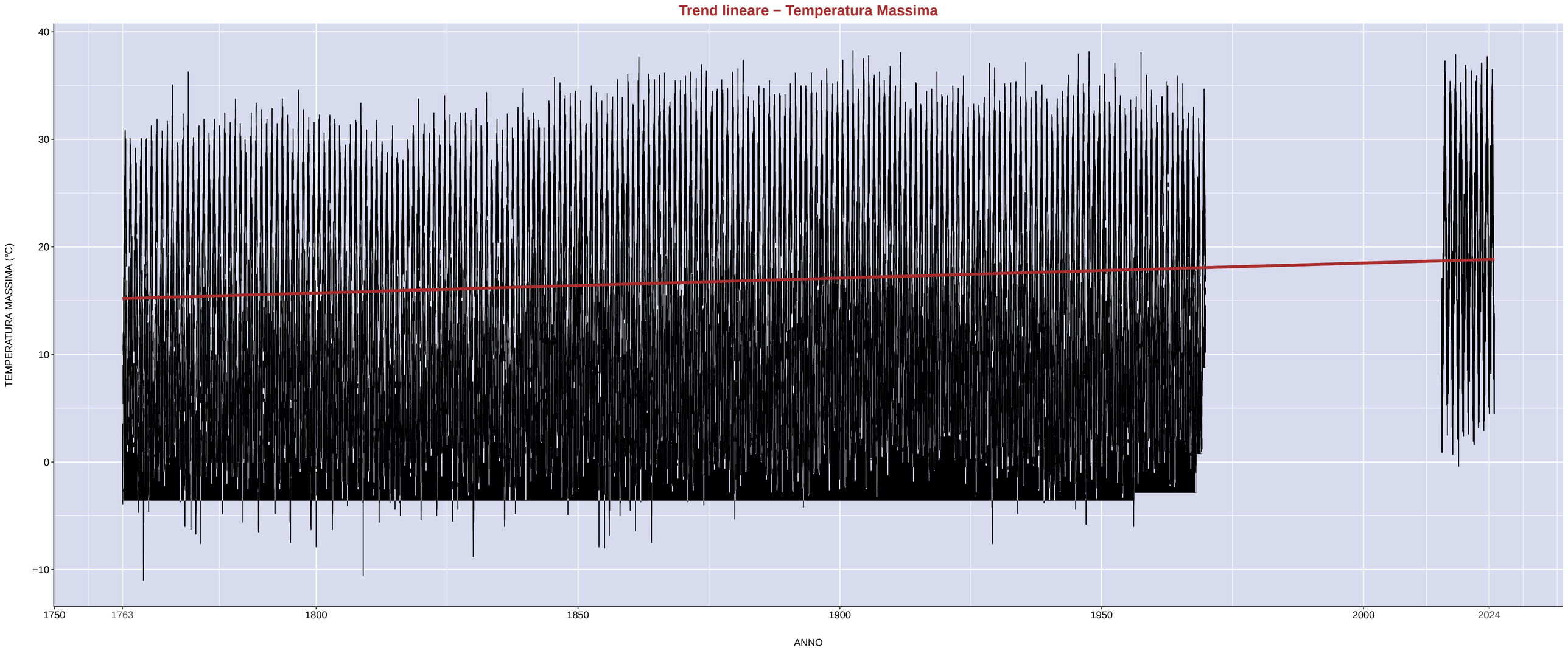

Autocorrelazione annuale − TEMPERATURA MASSIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
LAG ANNUALE

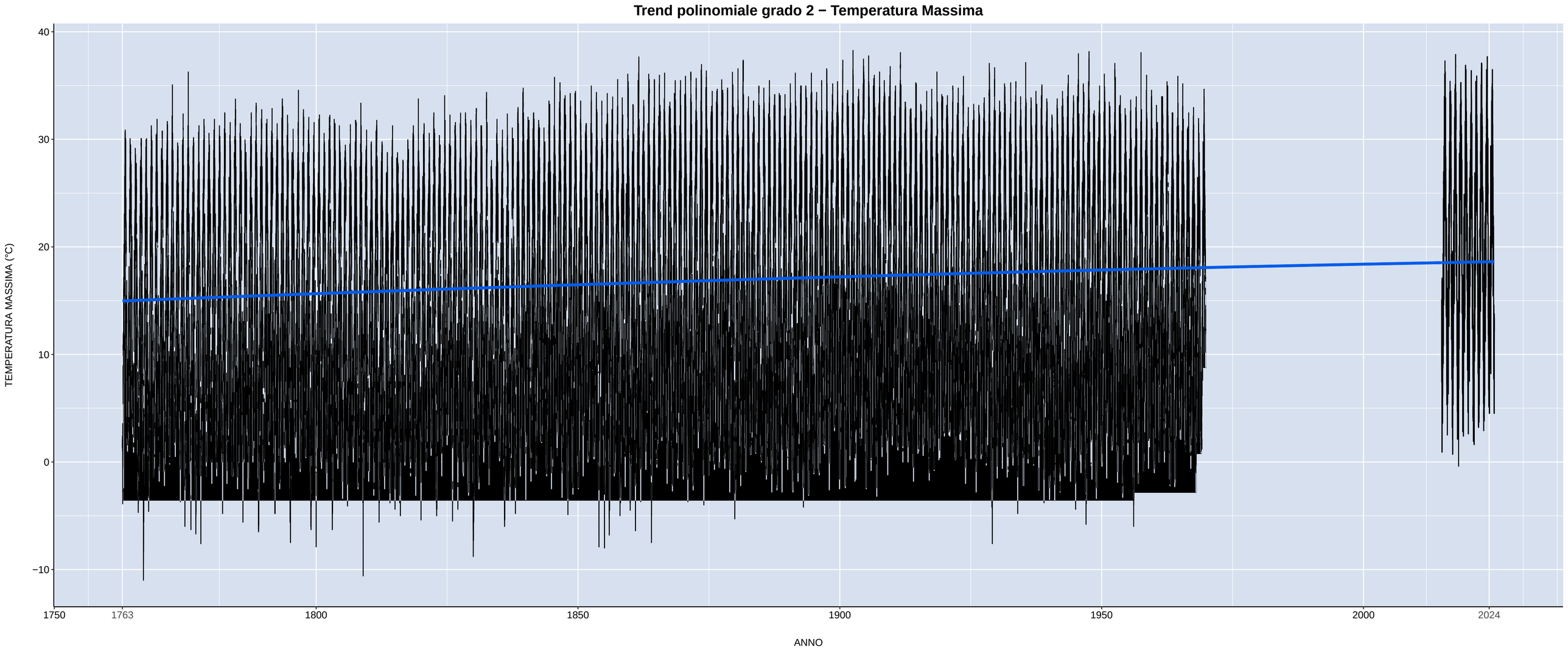

Autocorrelazione annuale − TEMPERATURA MINIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
LAG ANNUALE

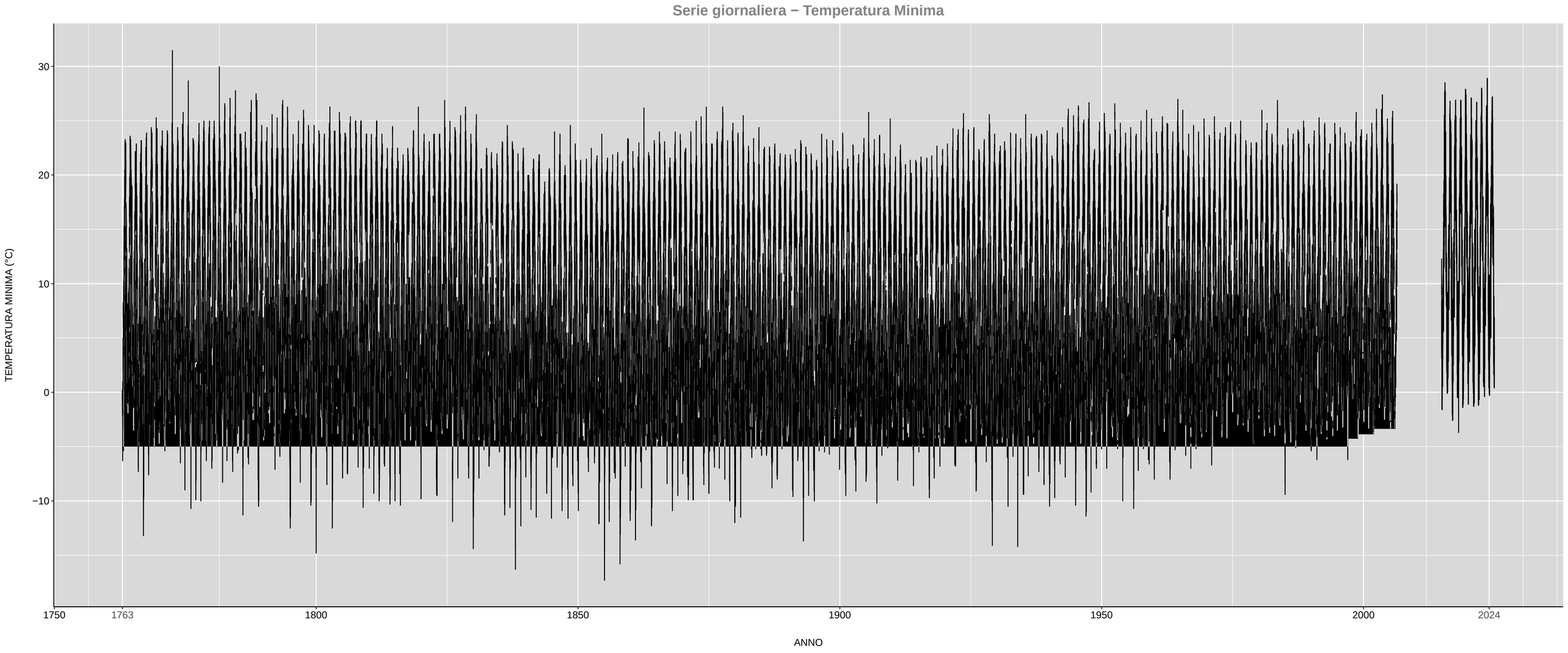

Autocorrelazione annuale − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
LAG ANNUALE

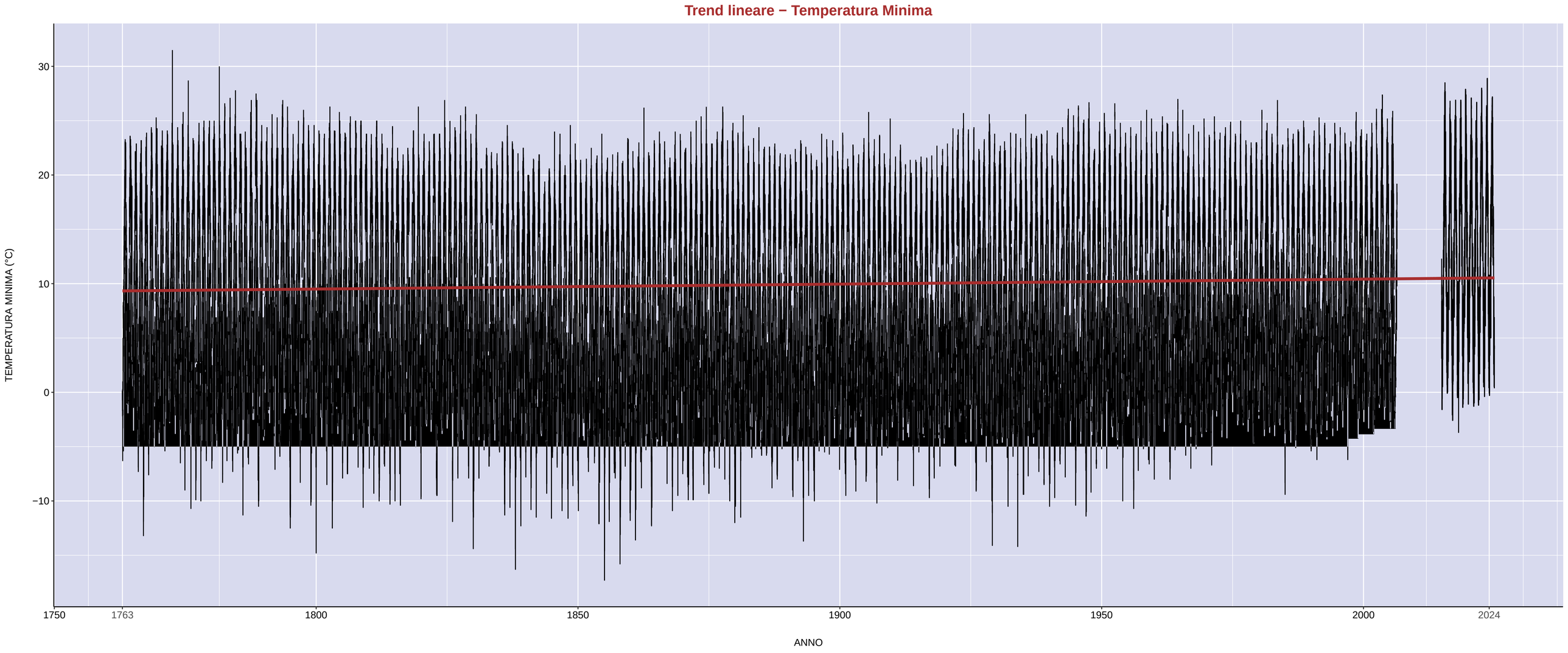
Milano – Stazione di Milano Brera

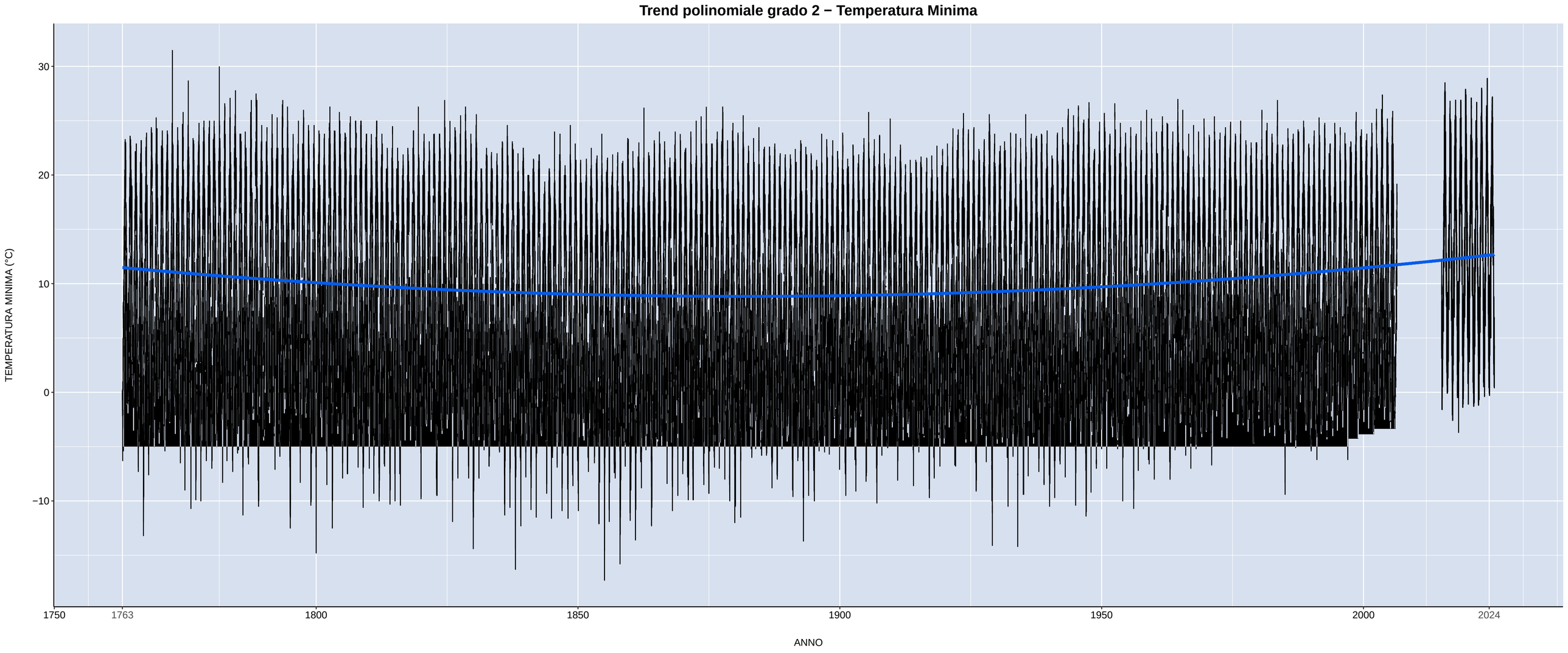

Serie giornaliera − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

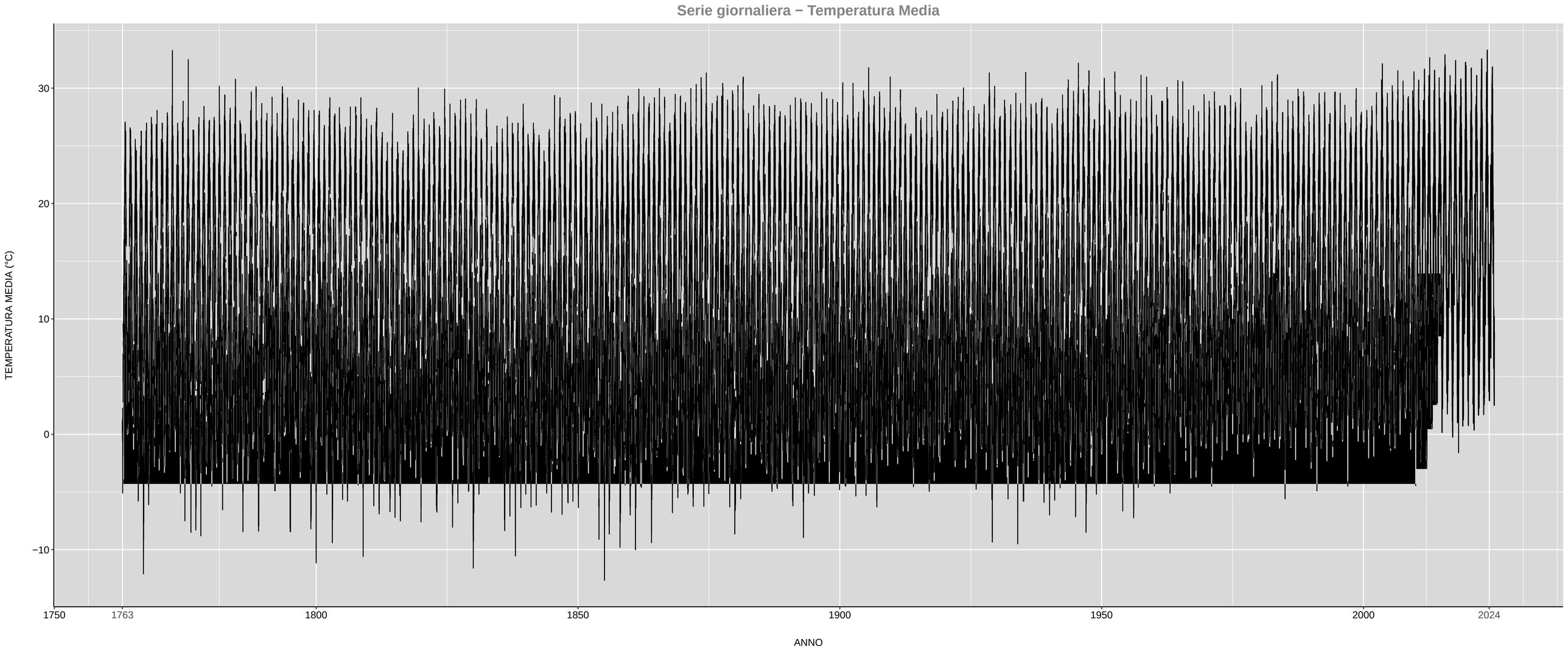

Trend lineare − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

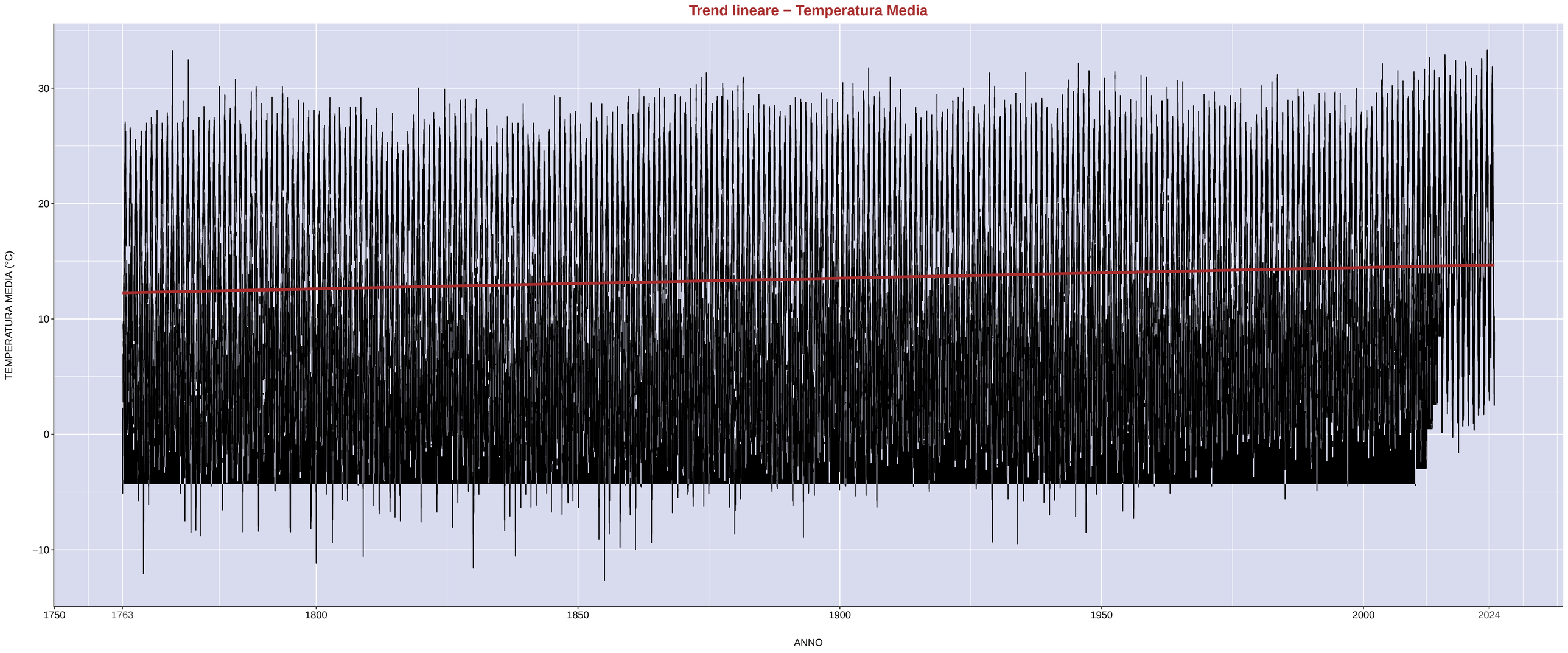

Trend polinomiale grado 2 − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

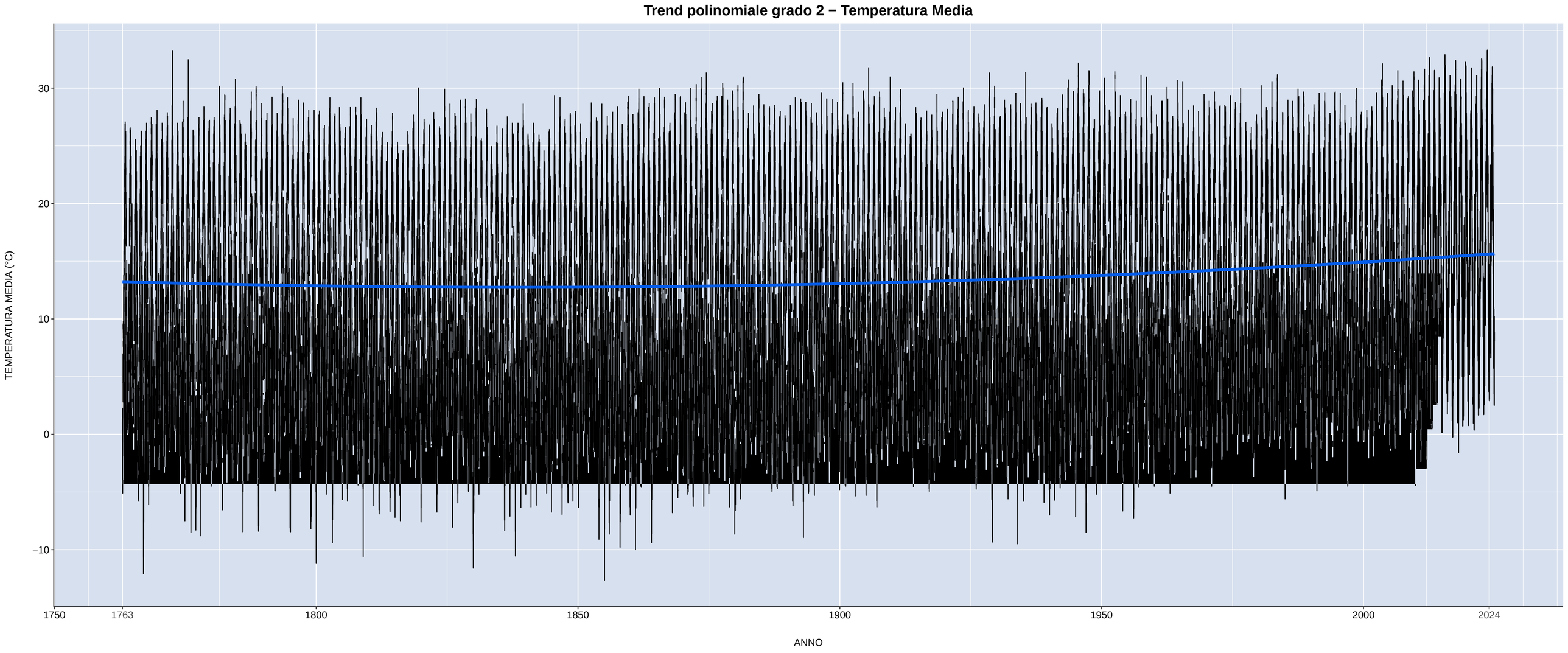

Serie giornaliera − Temperatura Minima
TEMPERATURA MINIMA (°C)
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

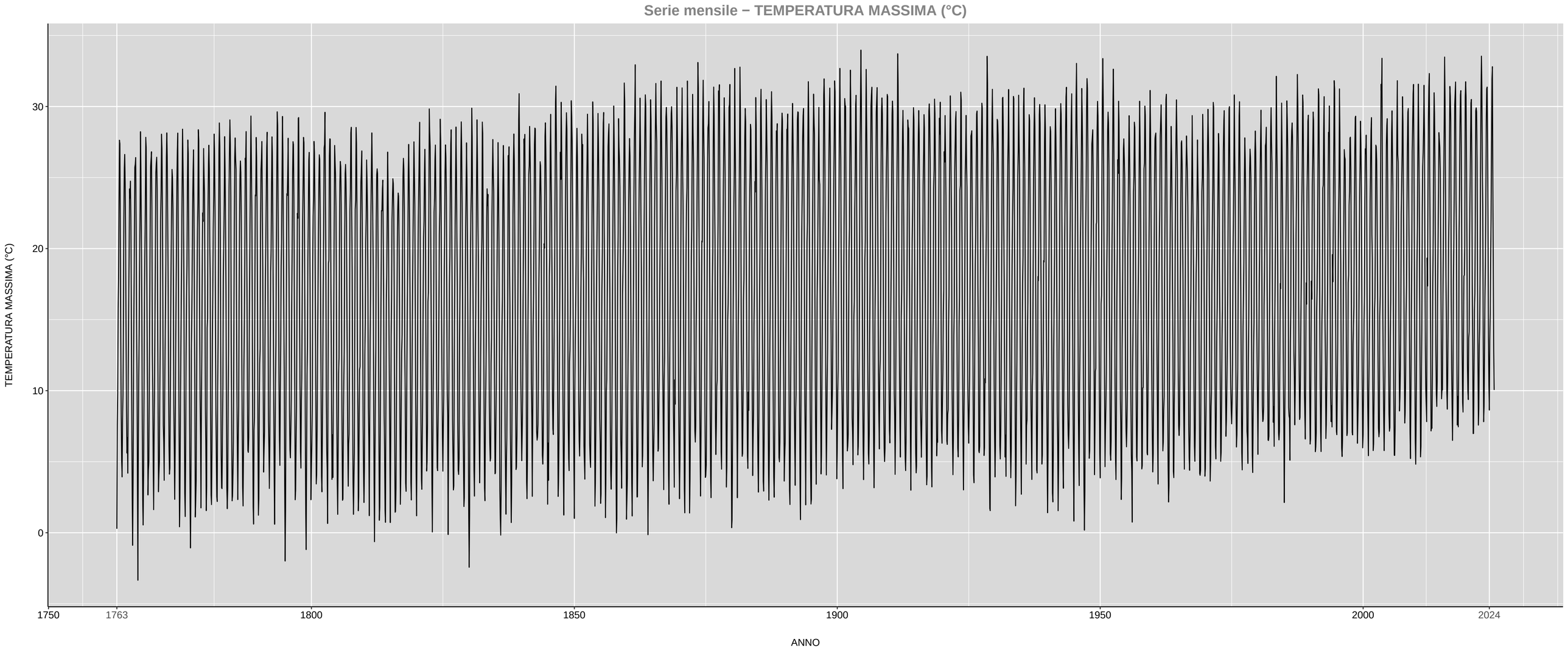

Trend lineare − Temperatura Minima
TEMPERATURA MINIMA (°C)
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

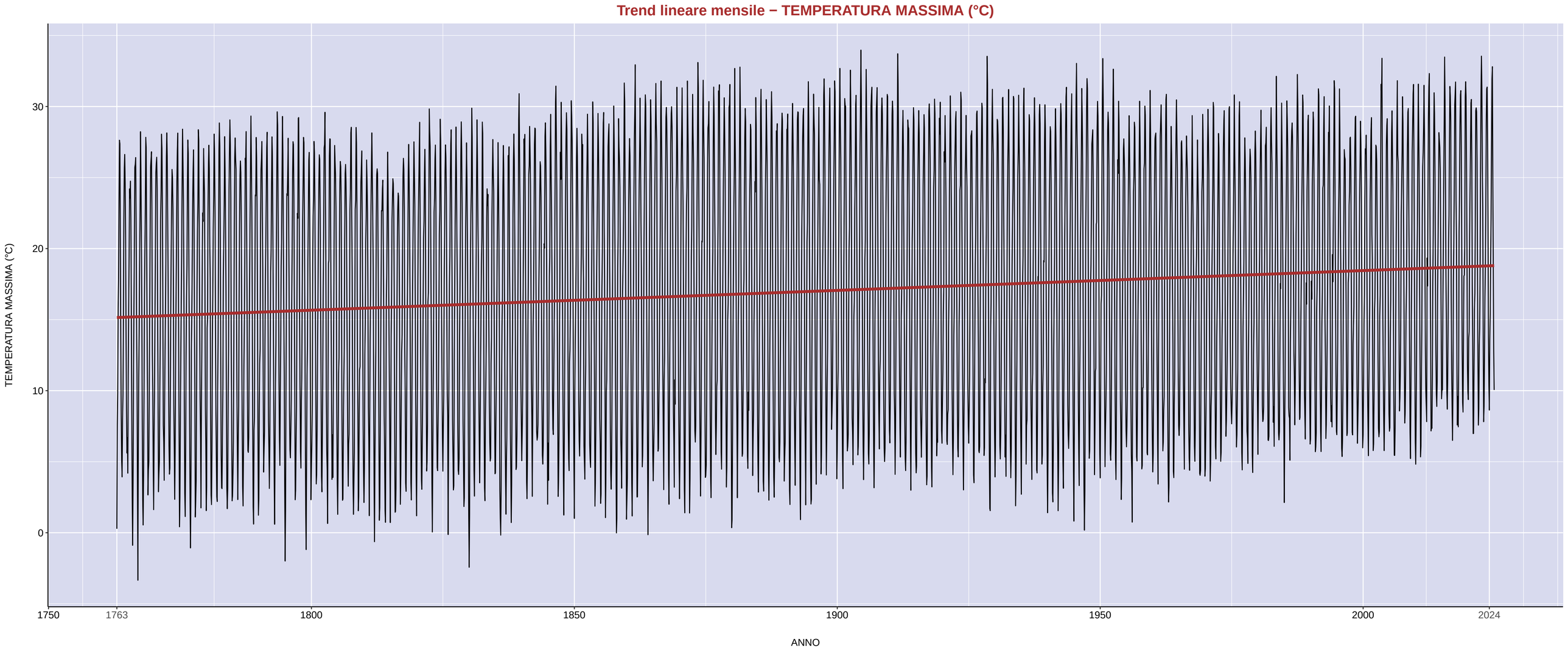

Trend polinomiale grado 2 − Temperatura Minima
TEMPERATURA MINIMA (°C)
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

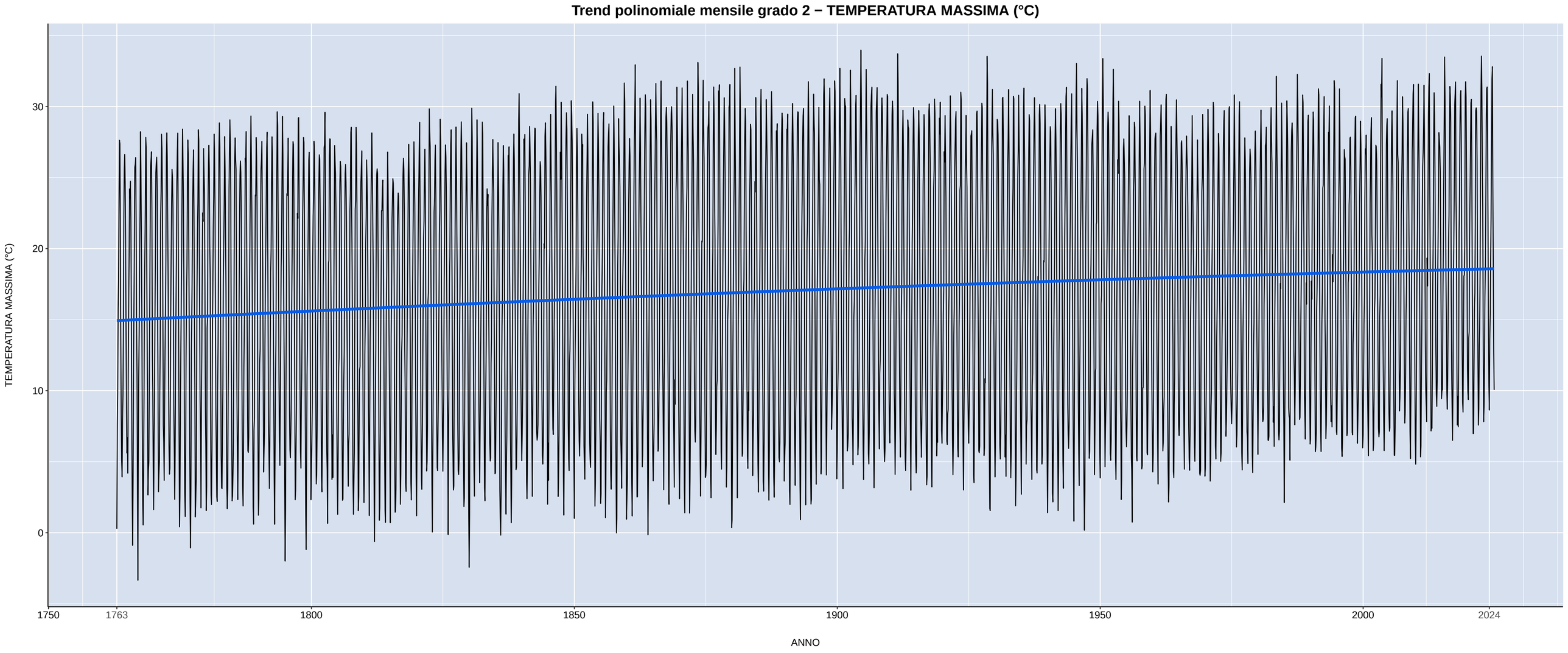

Serie giornaliera − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

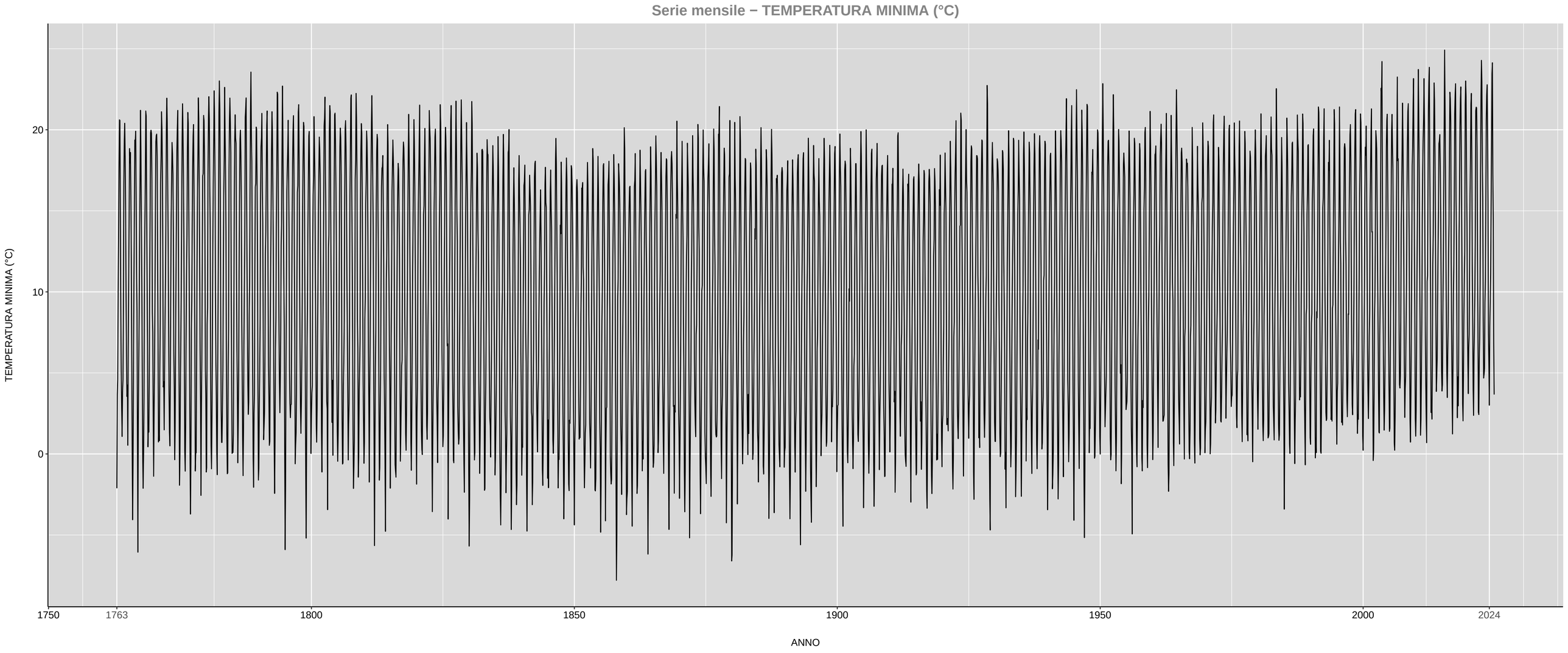

Trend lineare − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

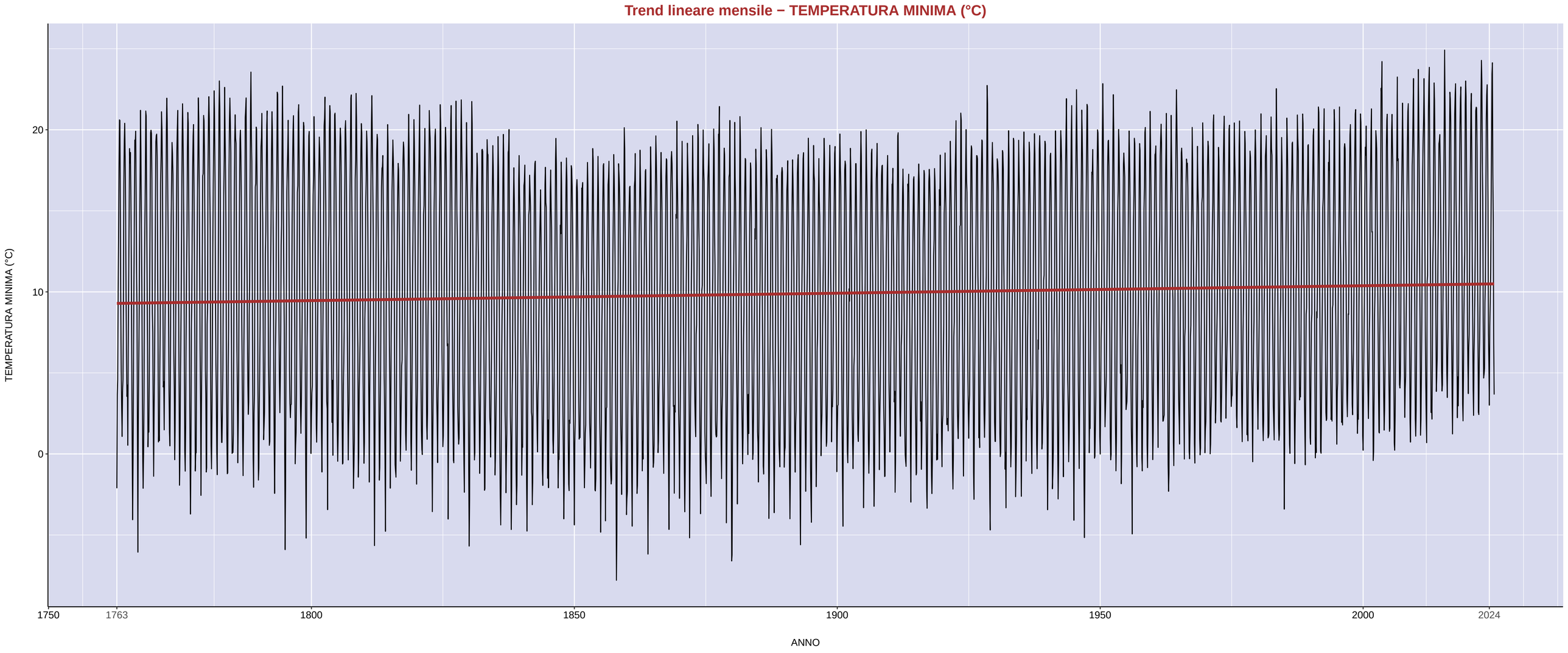

Trend polinomiale grado 2 − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
−10
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

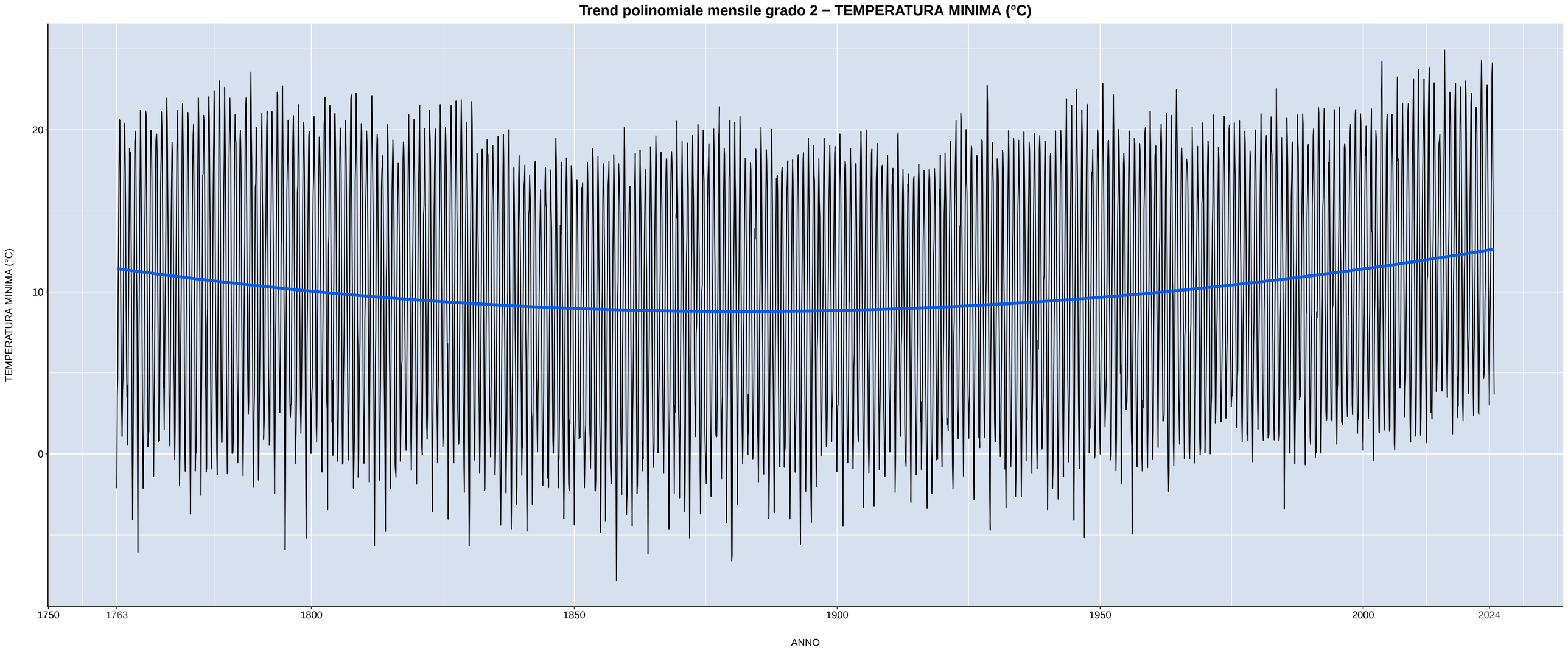

Serie mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

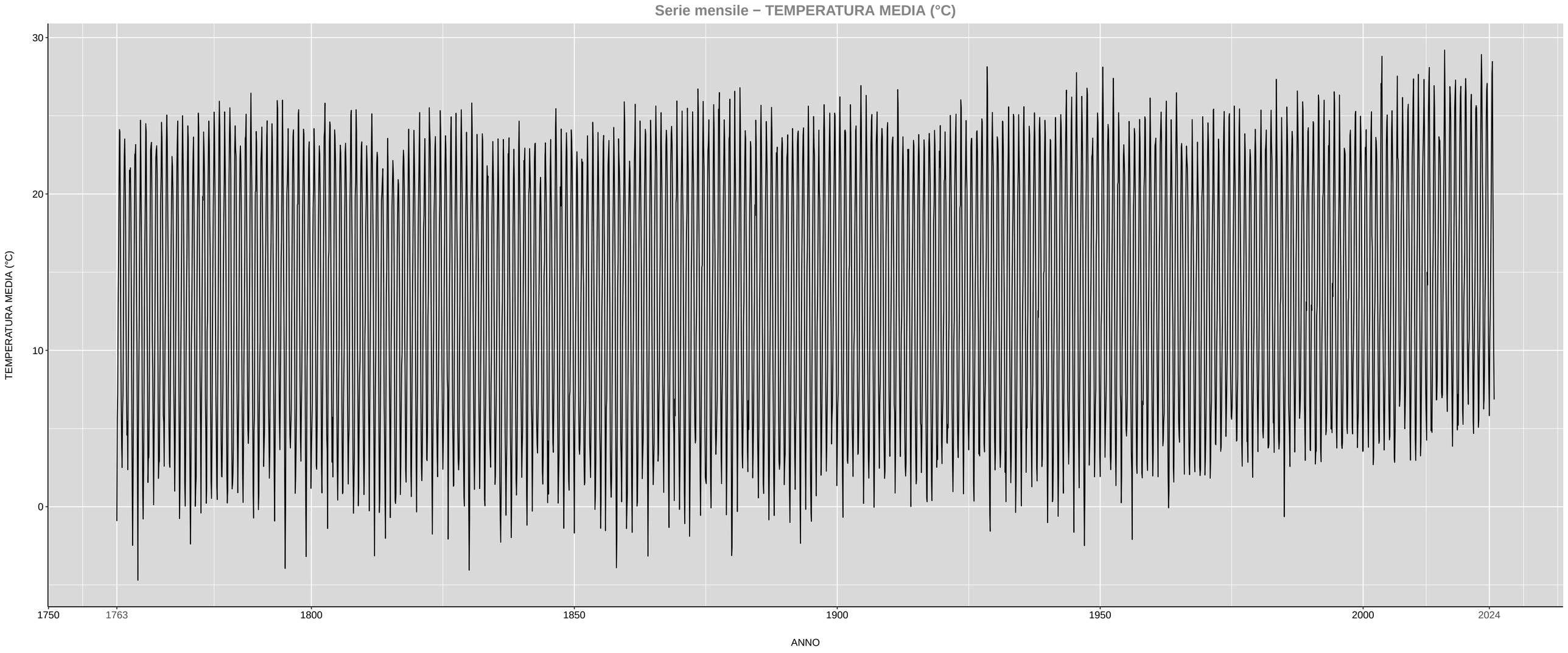

Trend lineare mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

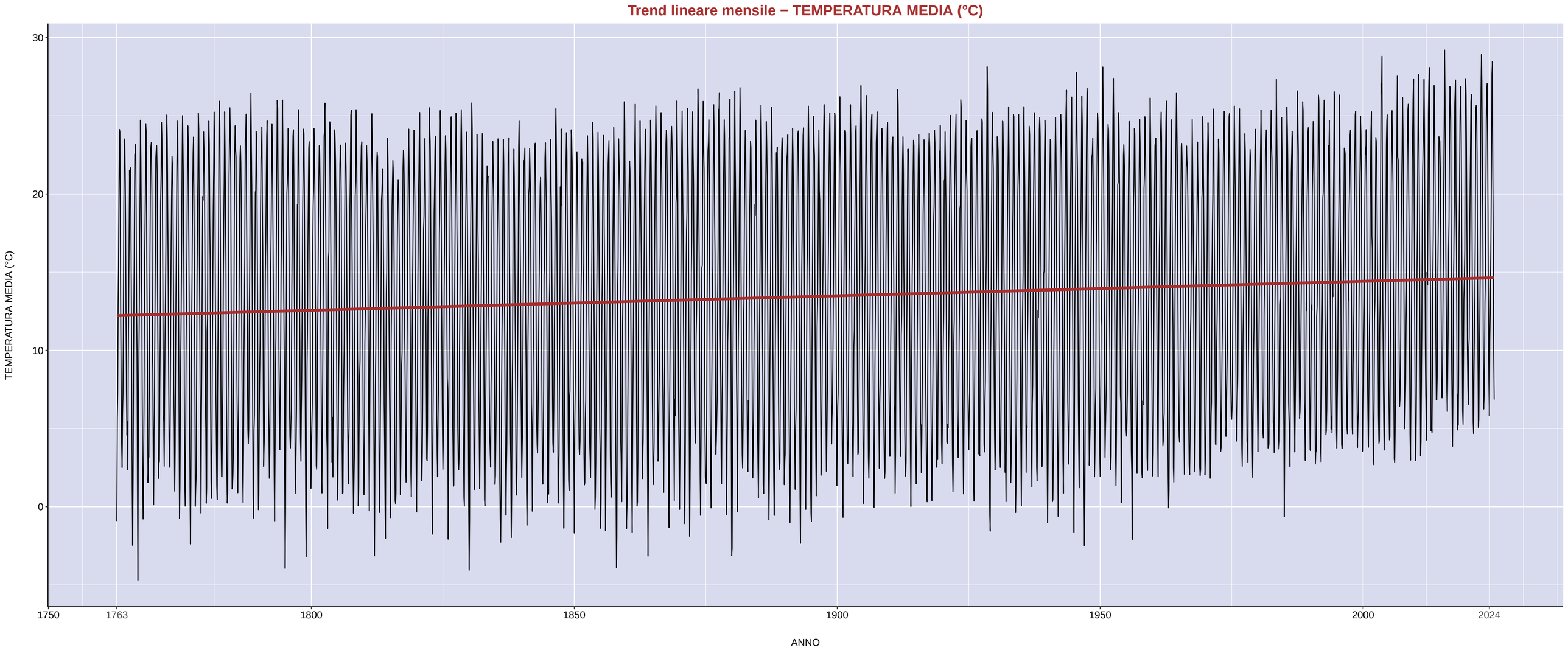

Trend polinomiale mensile grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

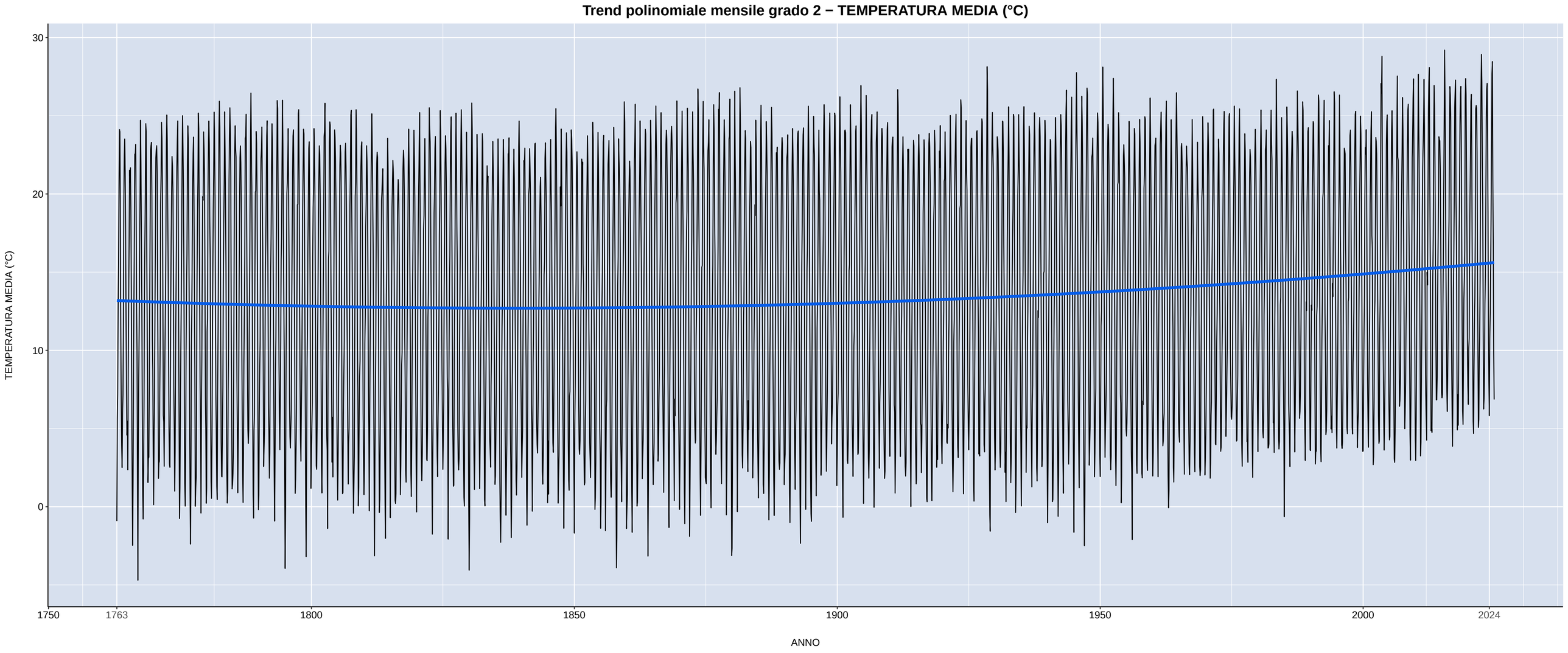

Serie mensile – TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

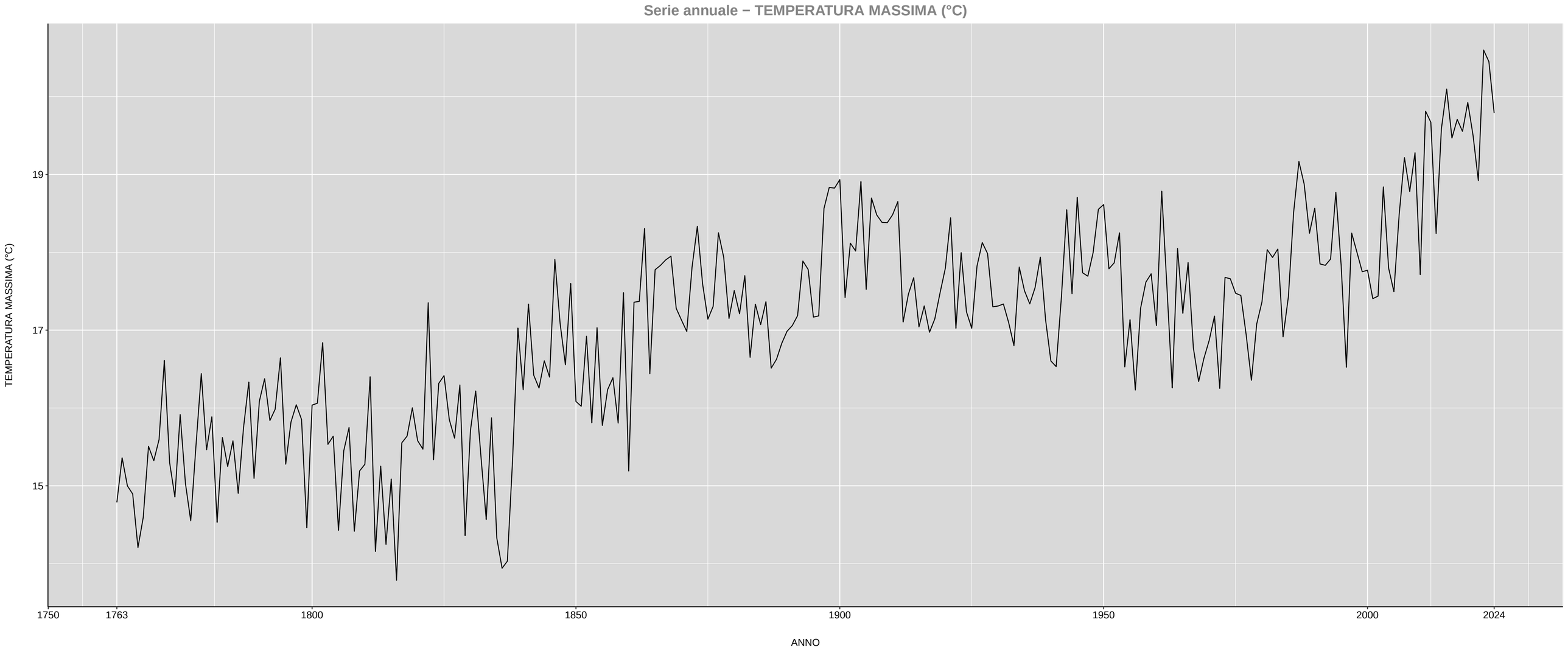

Trend lineare mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

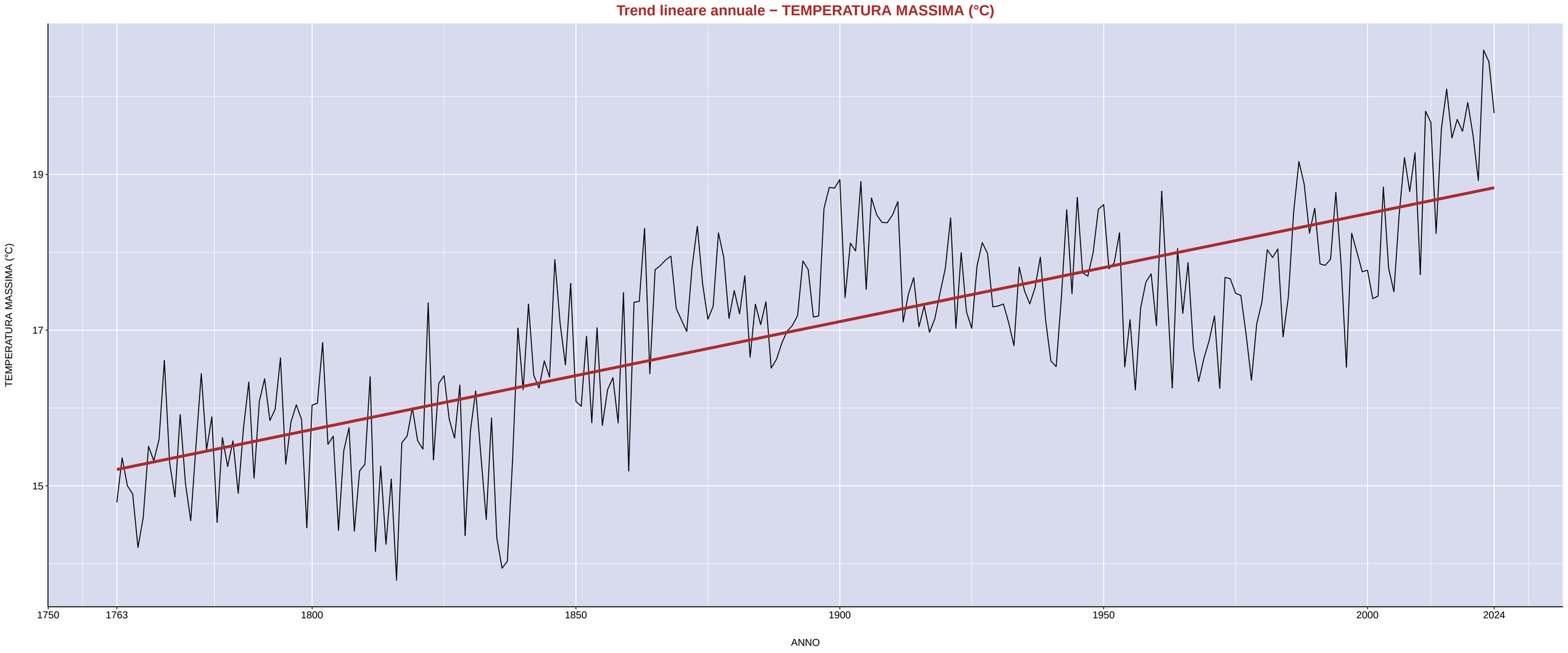

Trend polinomiale mensile grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

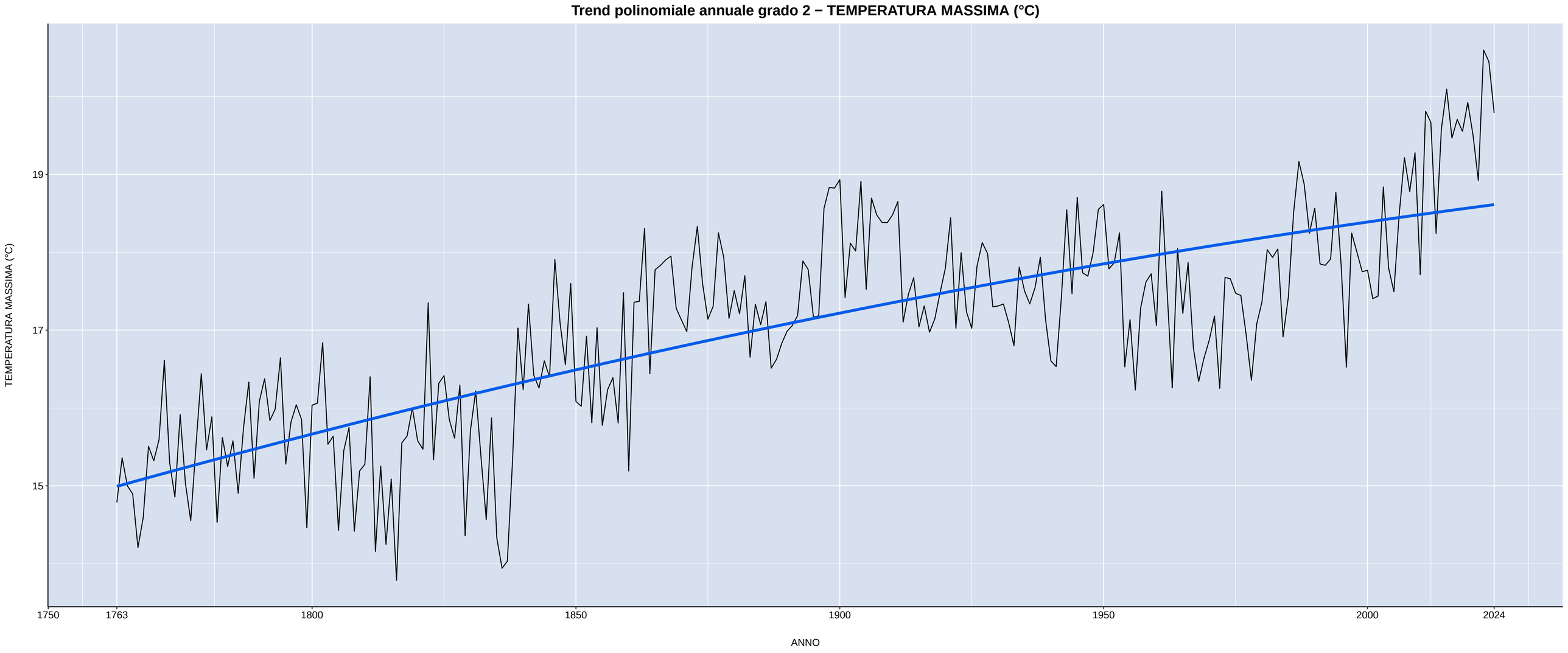

Serie mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

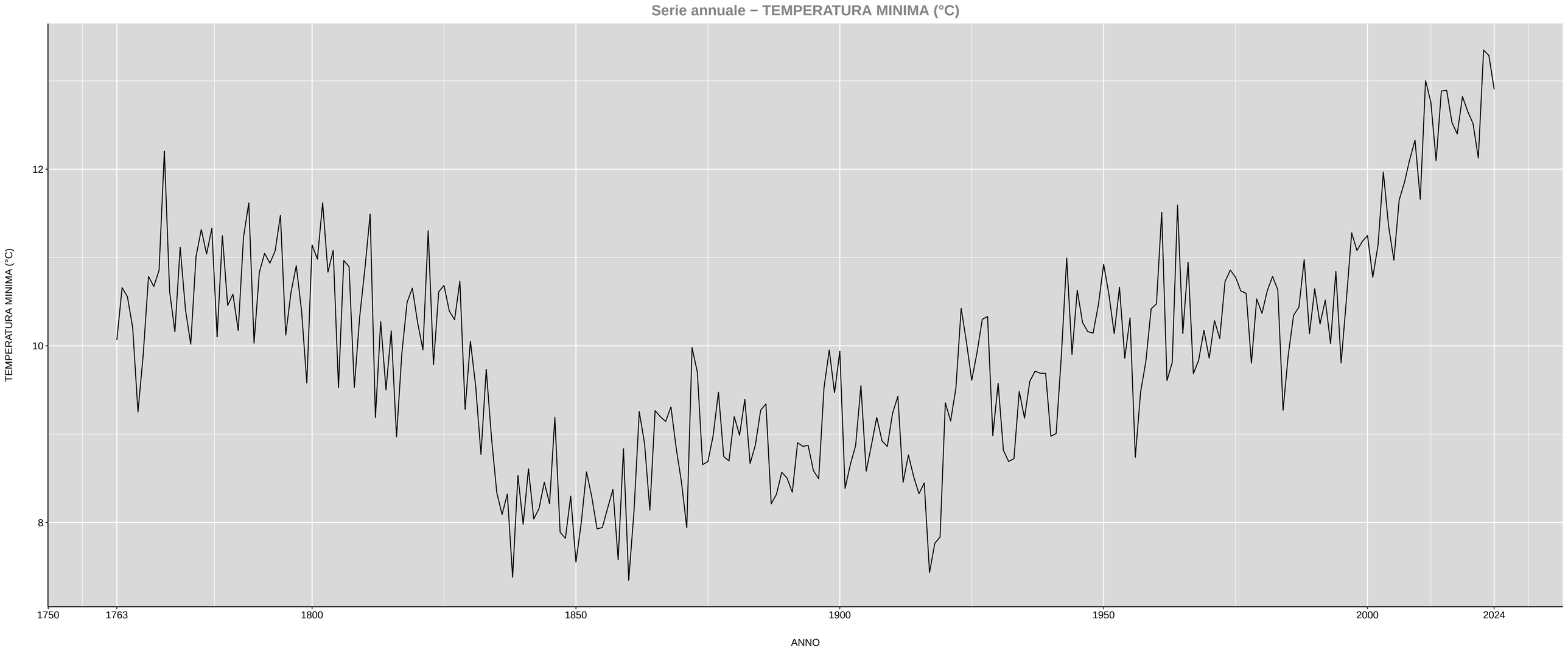

Trend lineare mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

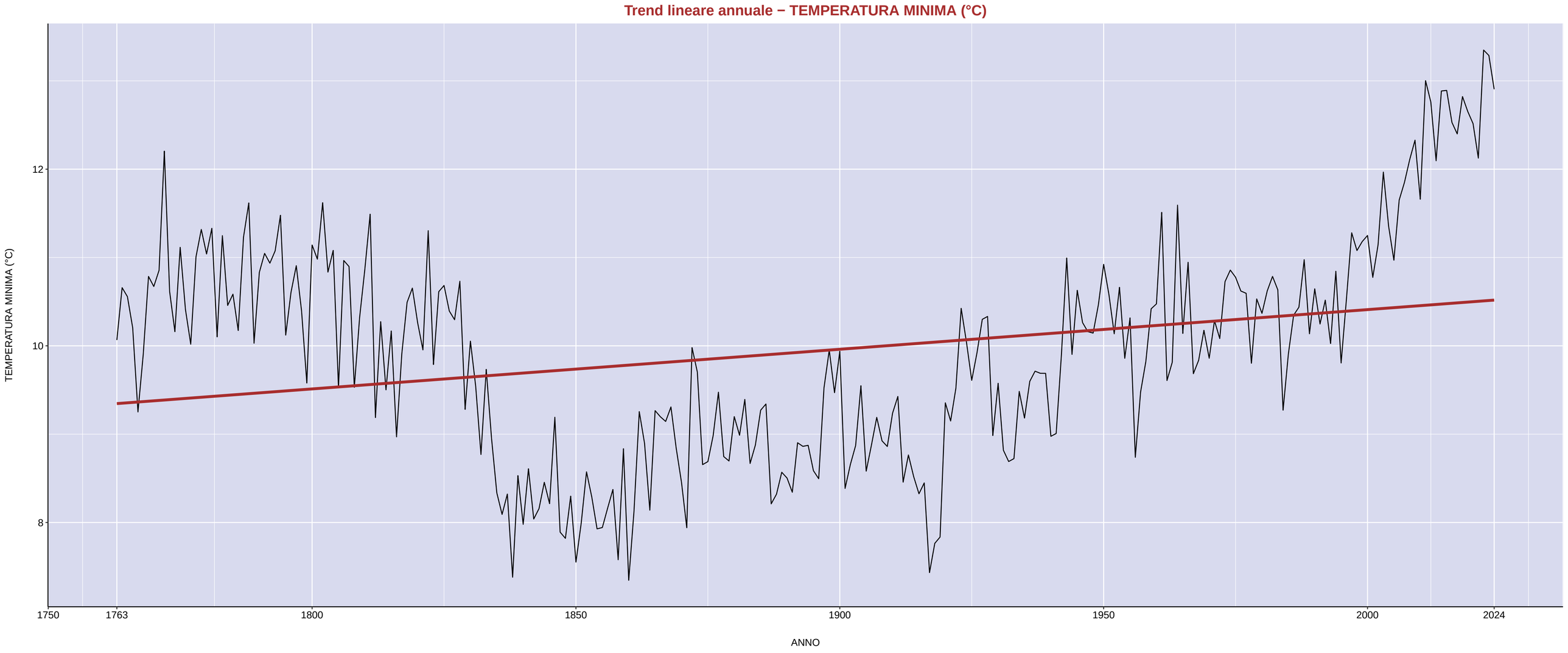

Trend polinomiale mensile grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
0
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

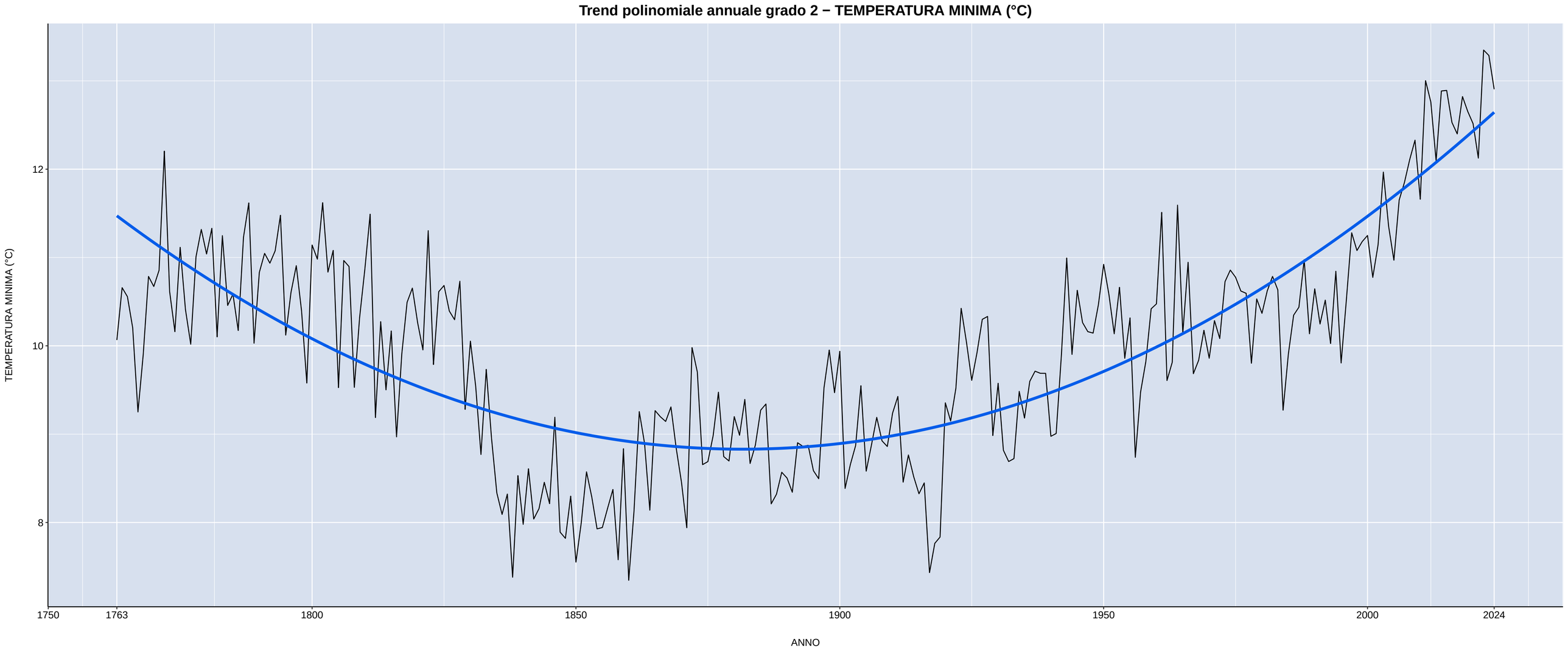

Serie annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
19
17
15
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

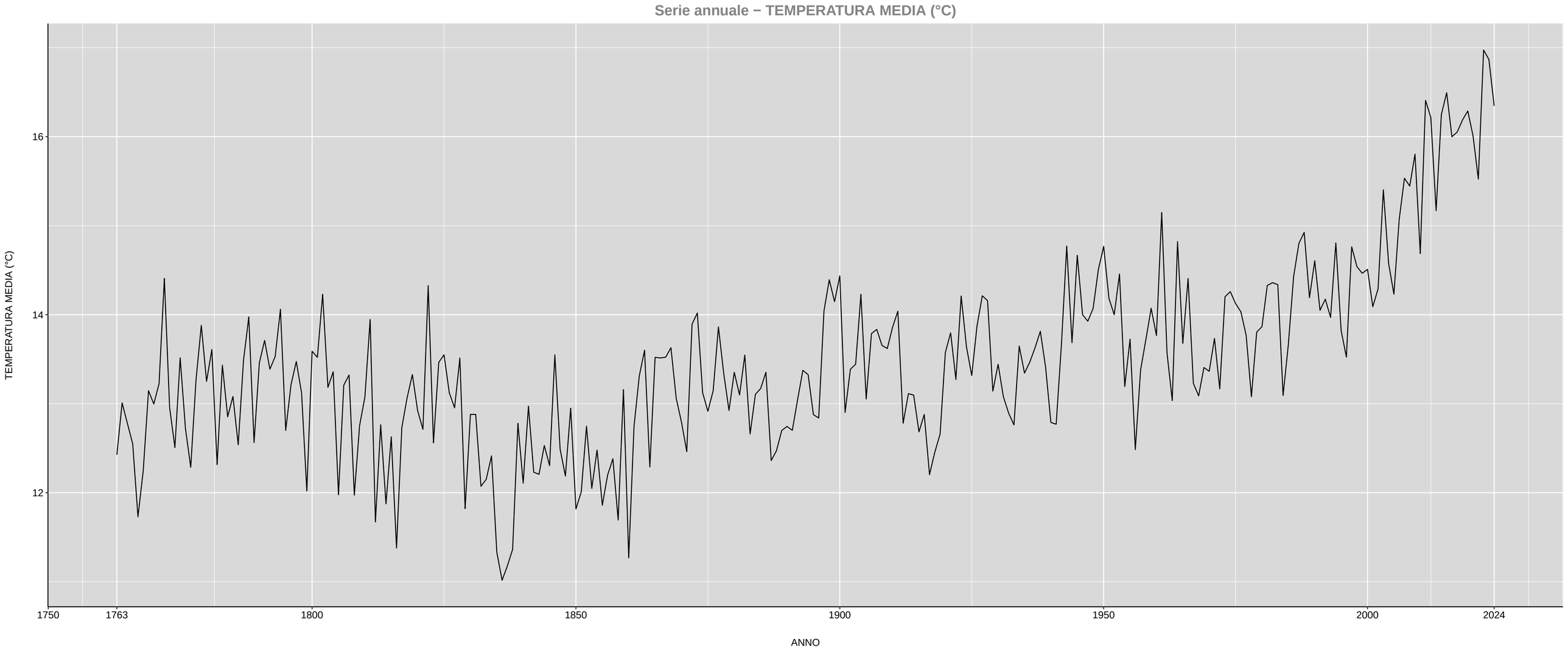

Trend lineare annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
19
17
15
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

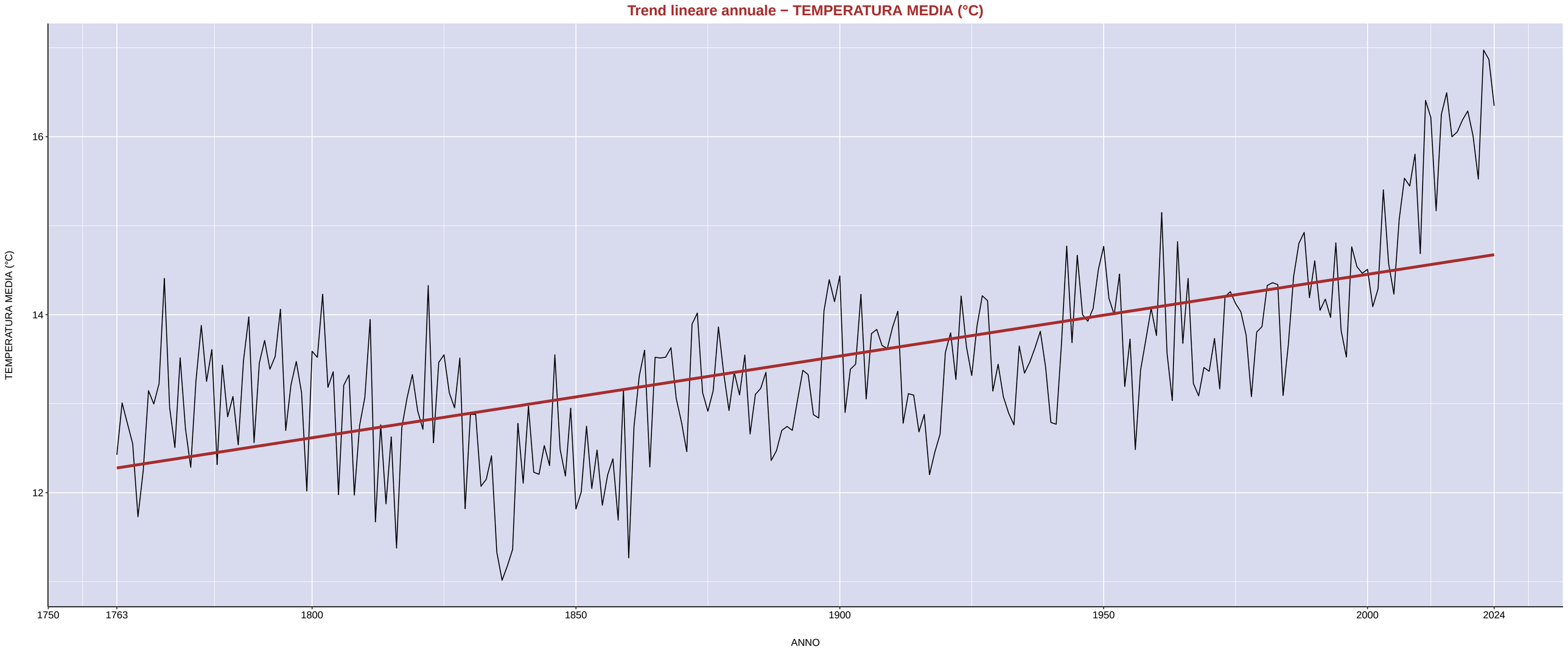

Trend polinomiale annuale grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
19
17
15
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

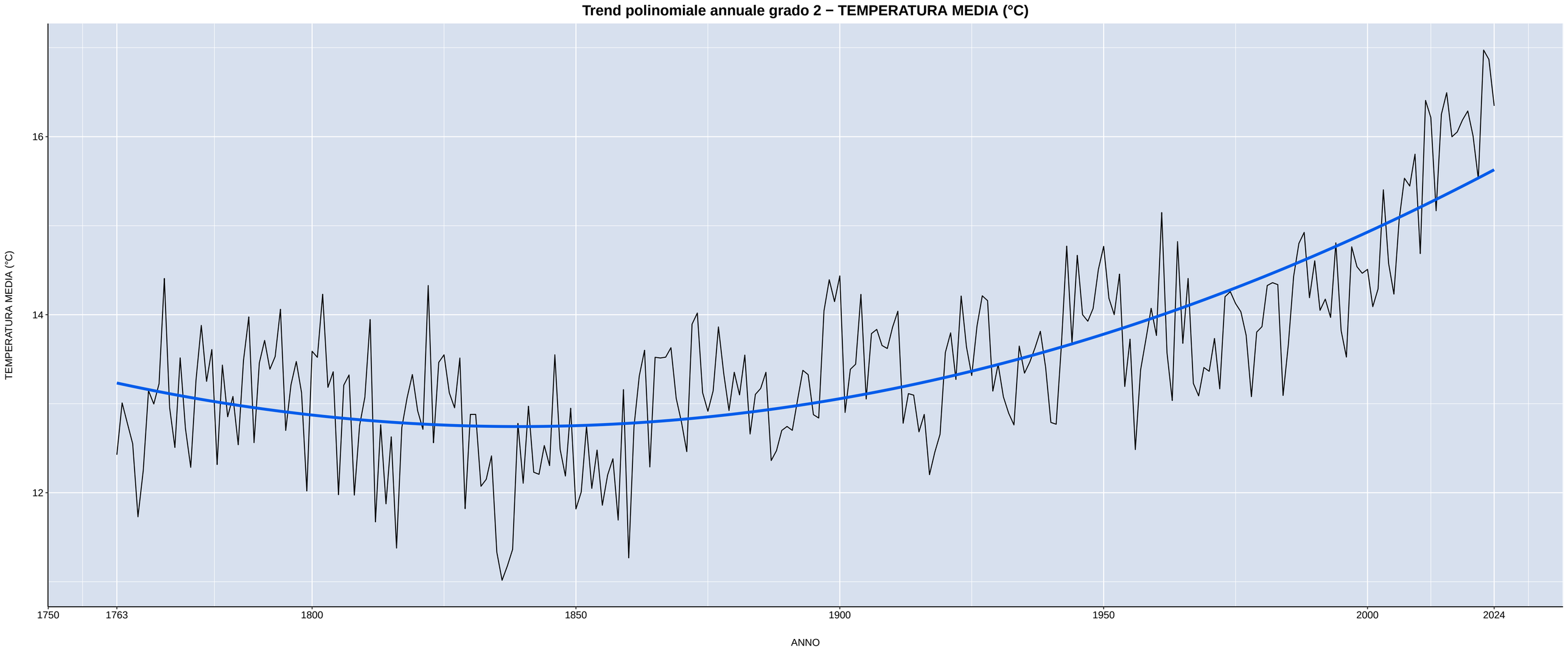

Serie annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
12
10
8
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

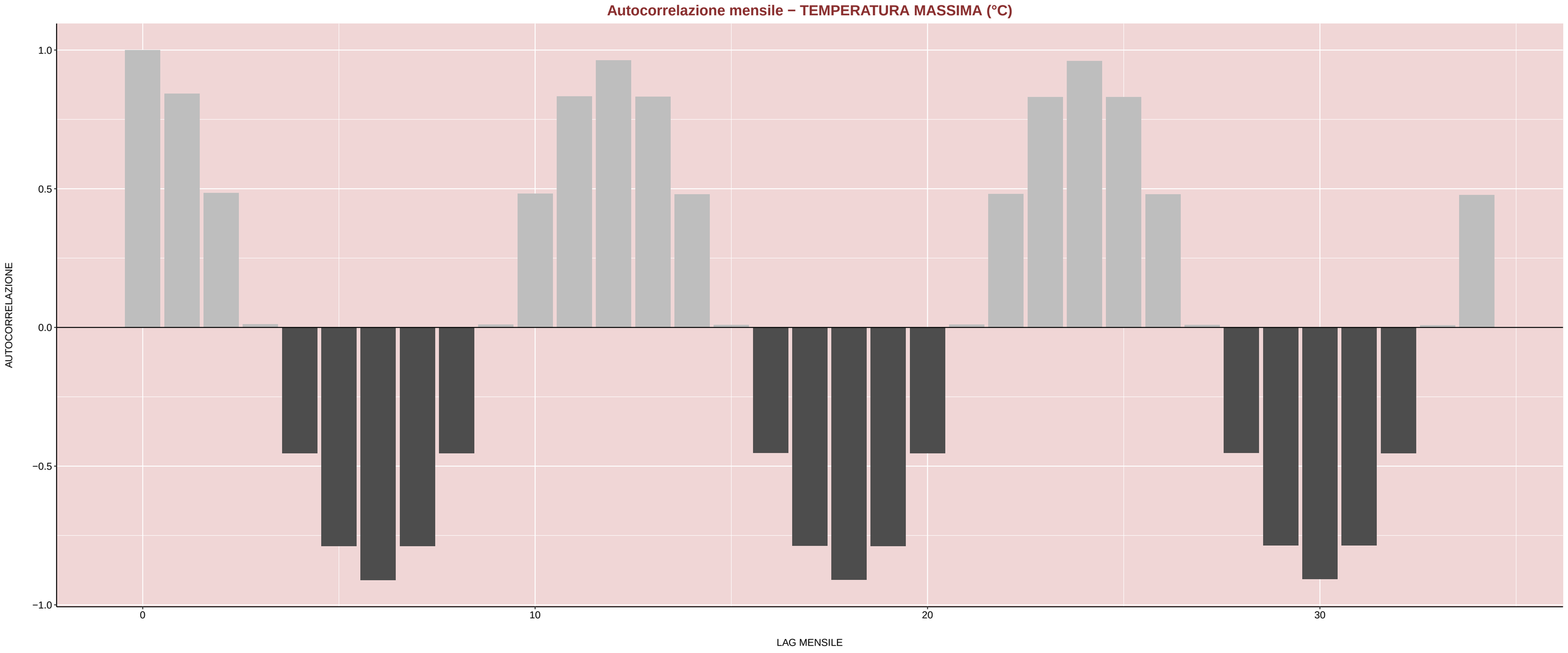

Trend lineare annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
12
10
8
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

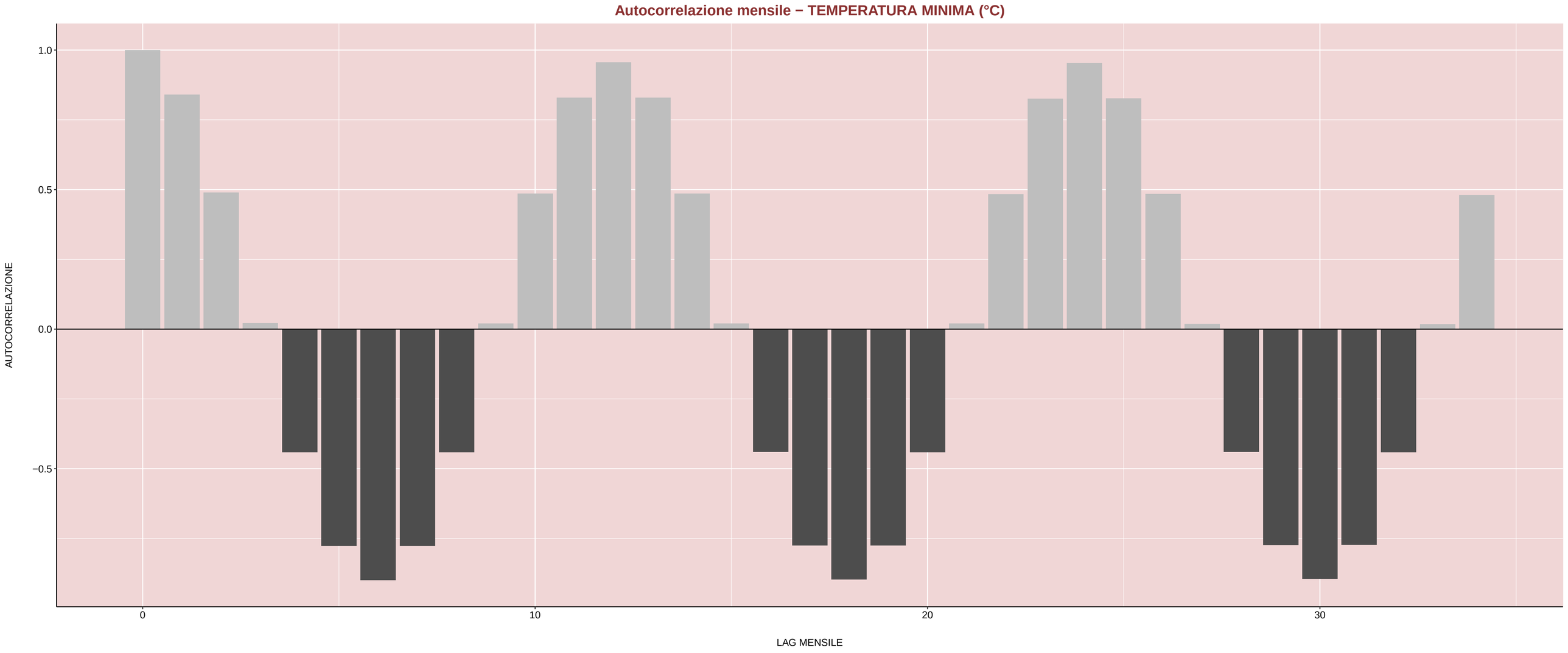

Trend polinomiale annuale grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
12
10
8
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

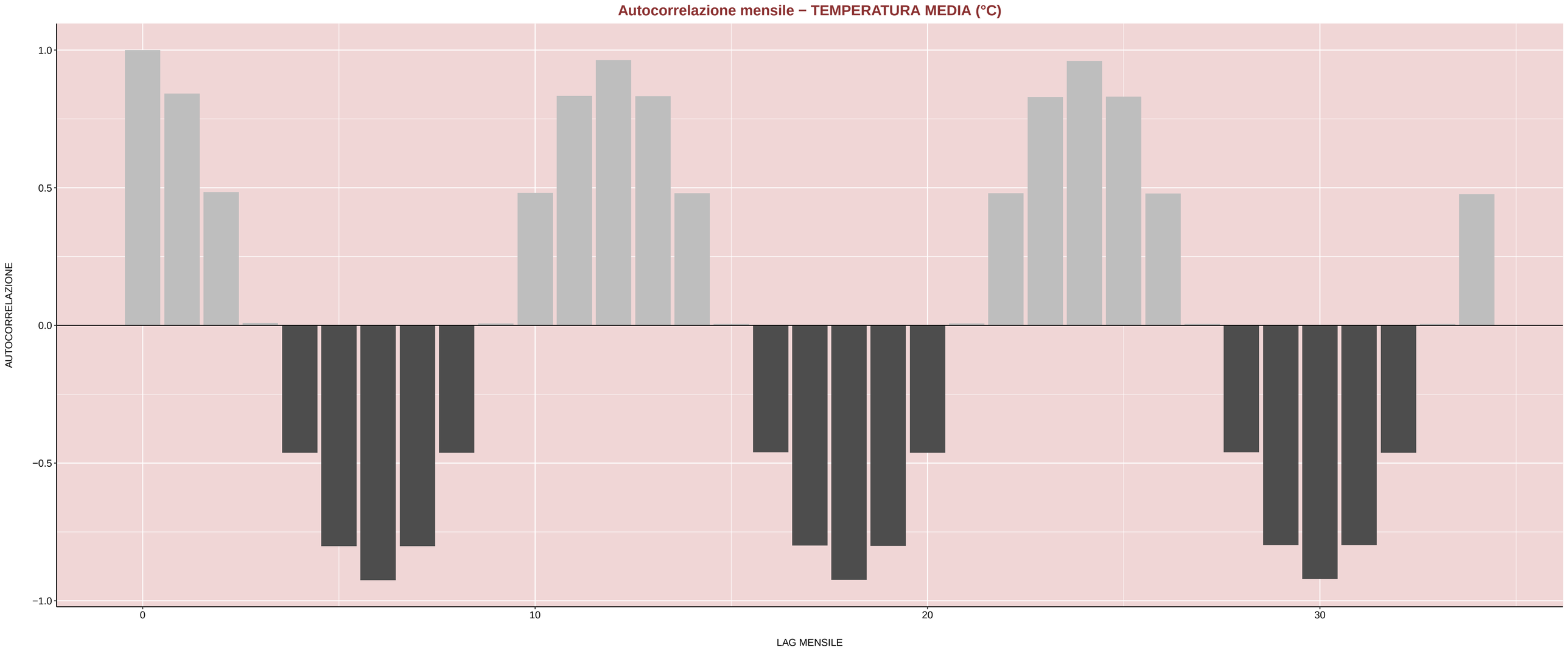

Serie annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
16
14
12
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

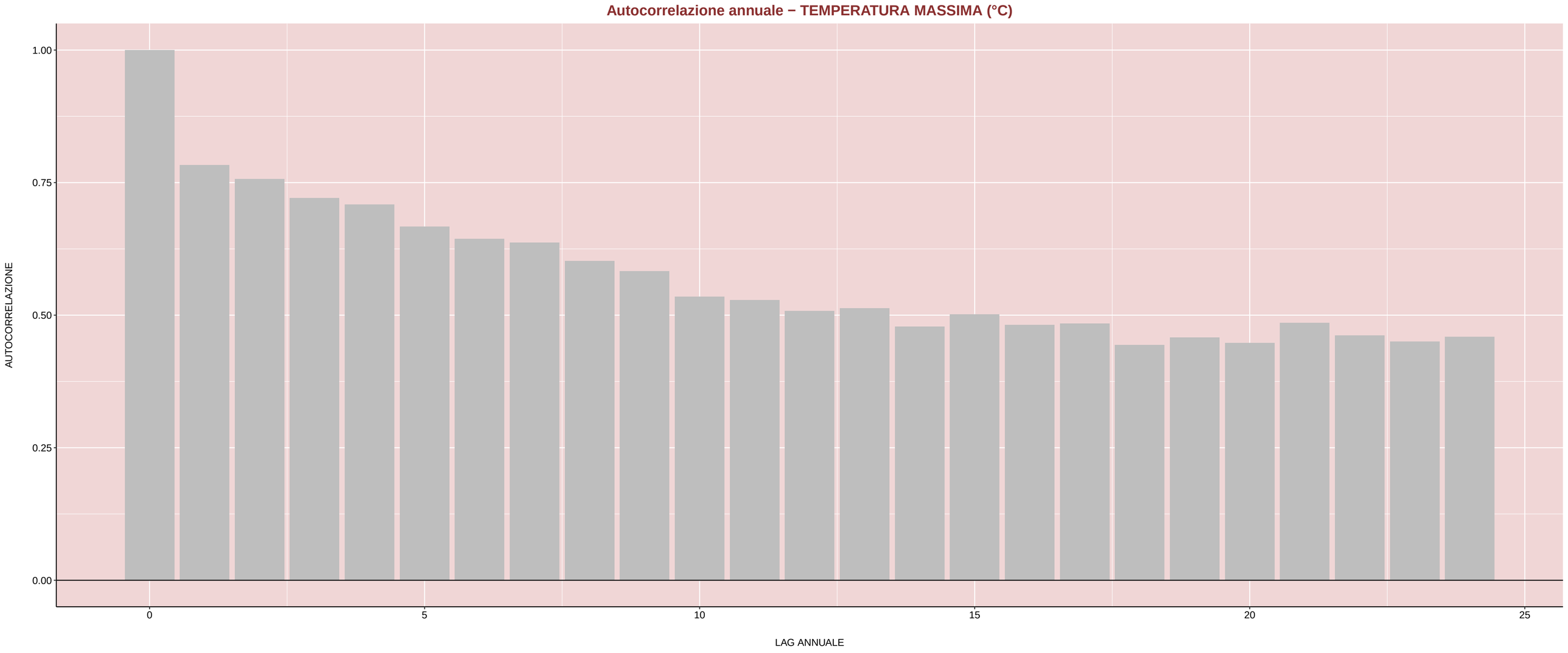

Trend lineare annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
16
14
12
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

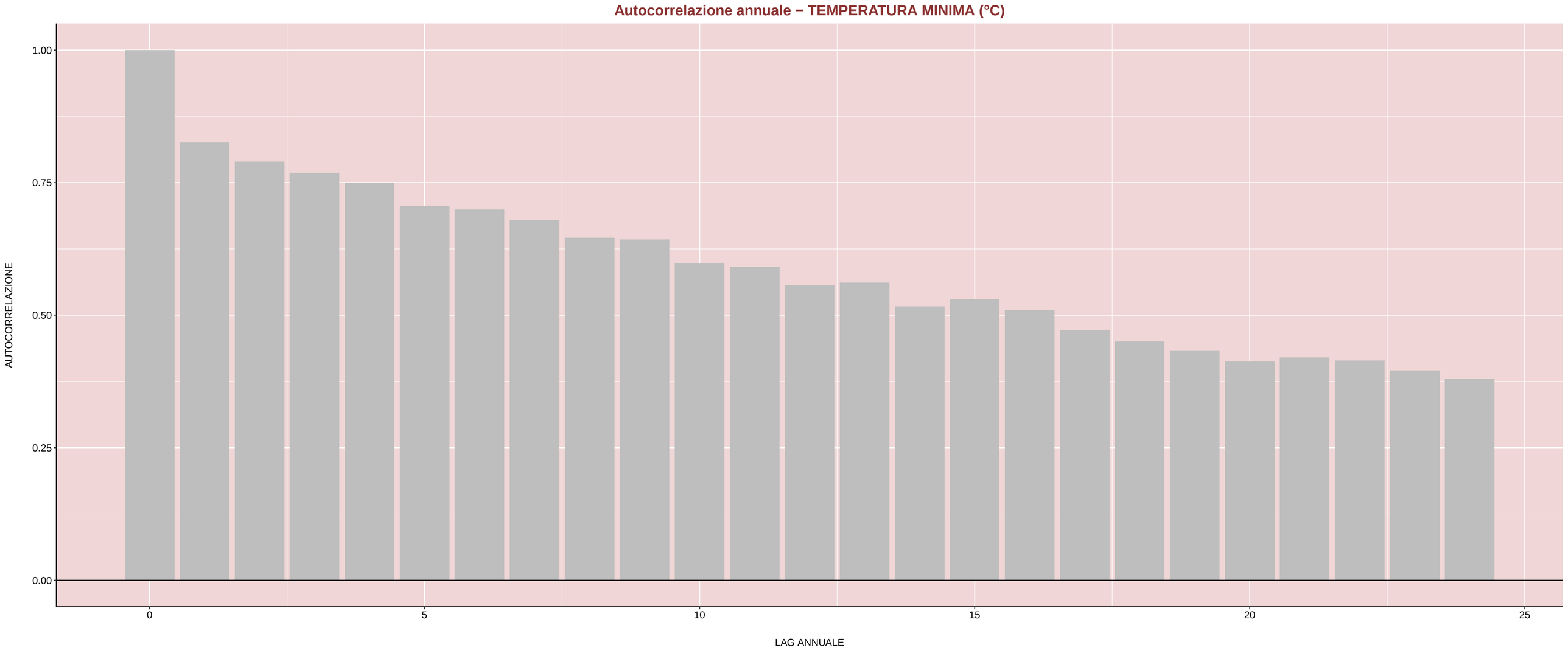

Trend polinomiale annuale grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
16
14
12
1750
1763
1800
1850
1900
1950
2000
2024
ANNO

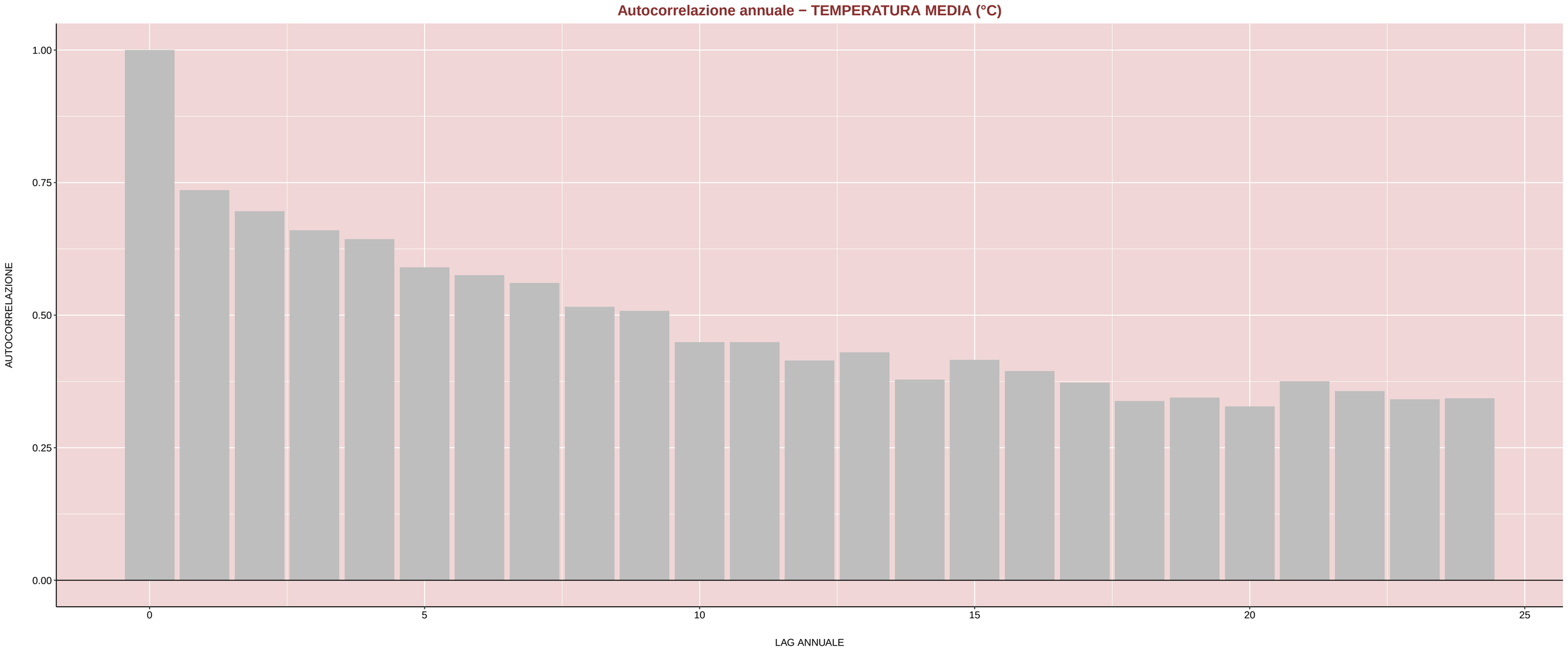

Autocorrelazione mensile − TEMPERATURA MASSIMA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
−1.0
0
10
20
30
LAG MENSILE


Autocorrelazione mensile − TEMPERATURA MINIMA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
0
10
20
30
LAG MENSILE

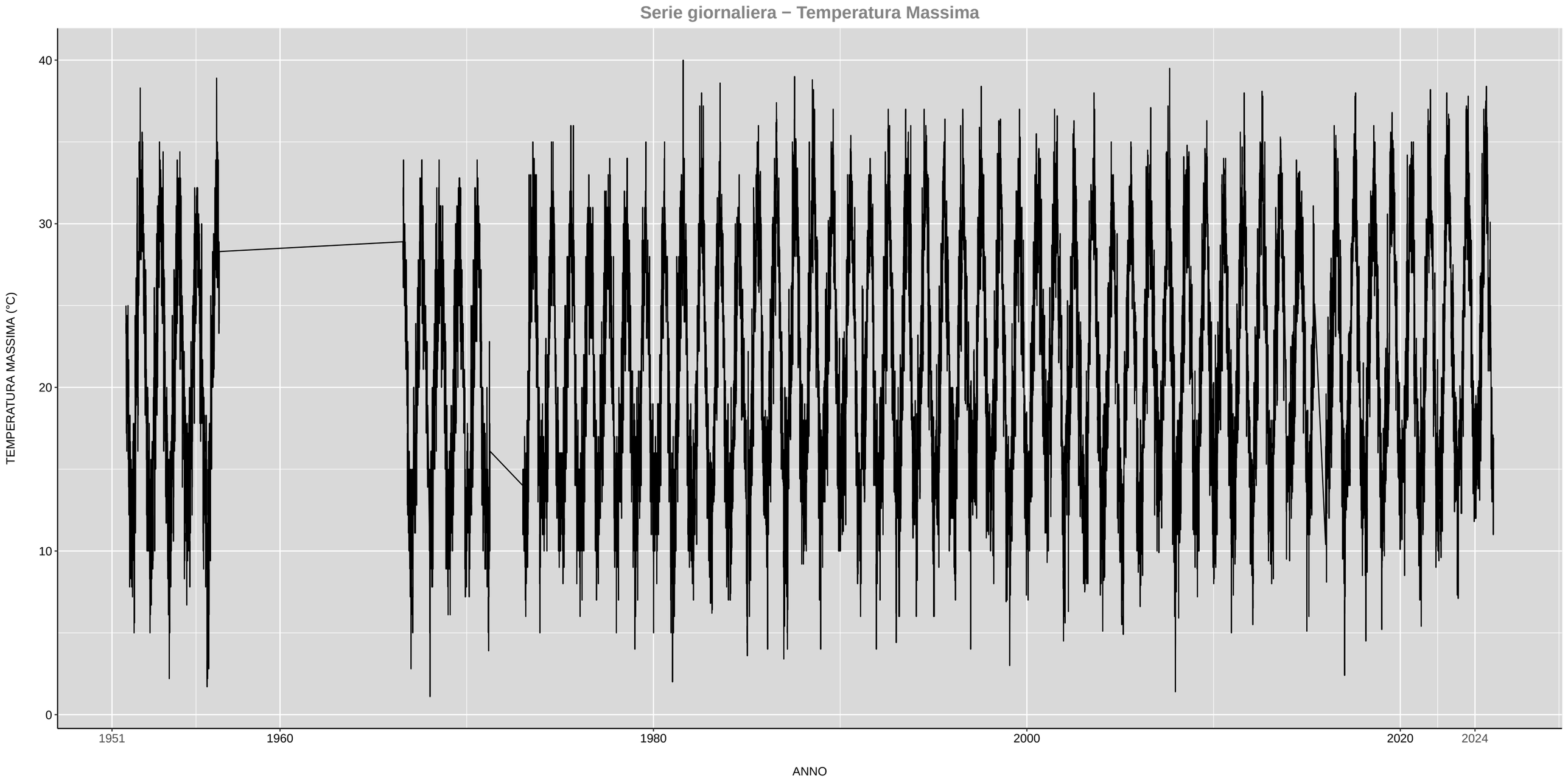

Autocorrelazione mensile − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
−1.0
0
10
20
30
LAG MENSILE

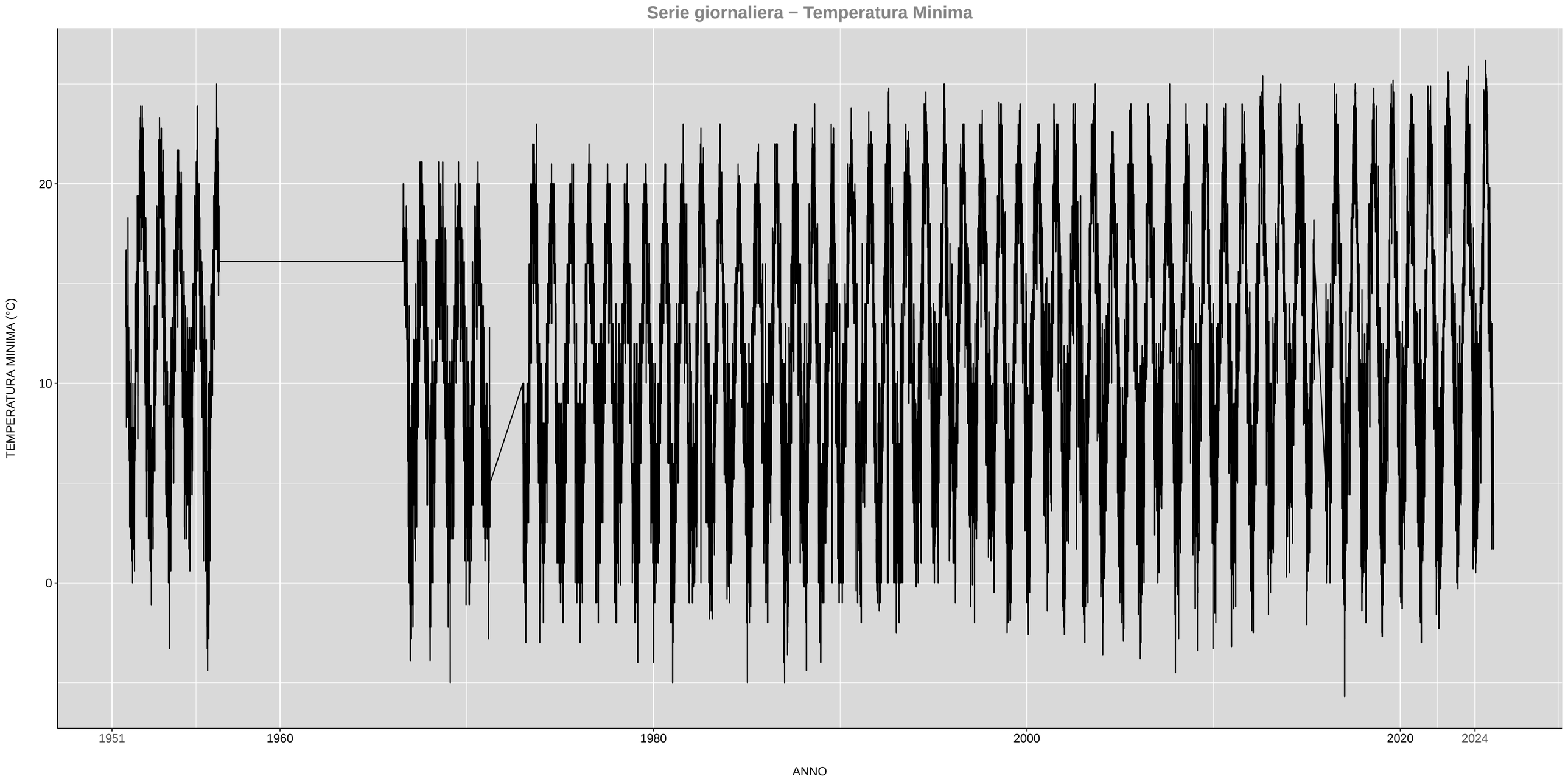

Autocorrelazione annuale − TEMPERATURA MASSIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
20
25
LAG ANNUALE

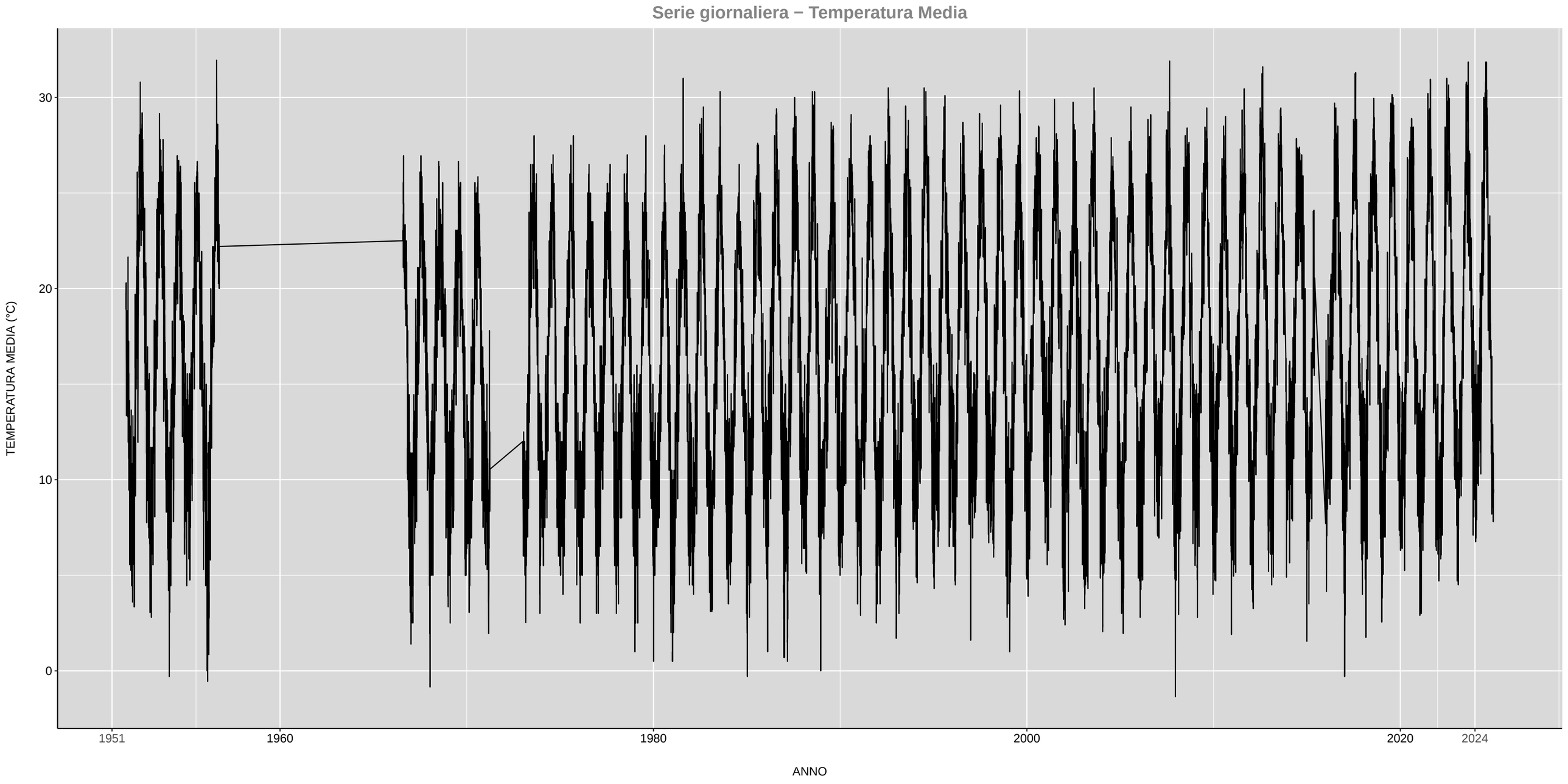

Autocorrelazione annuale − TEMPERATURA MINIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
20
25
LAG ANNUALE

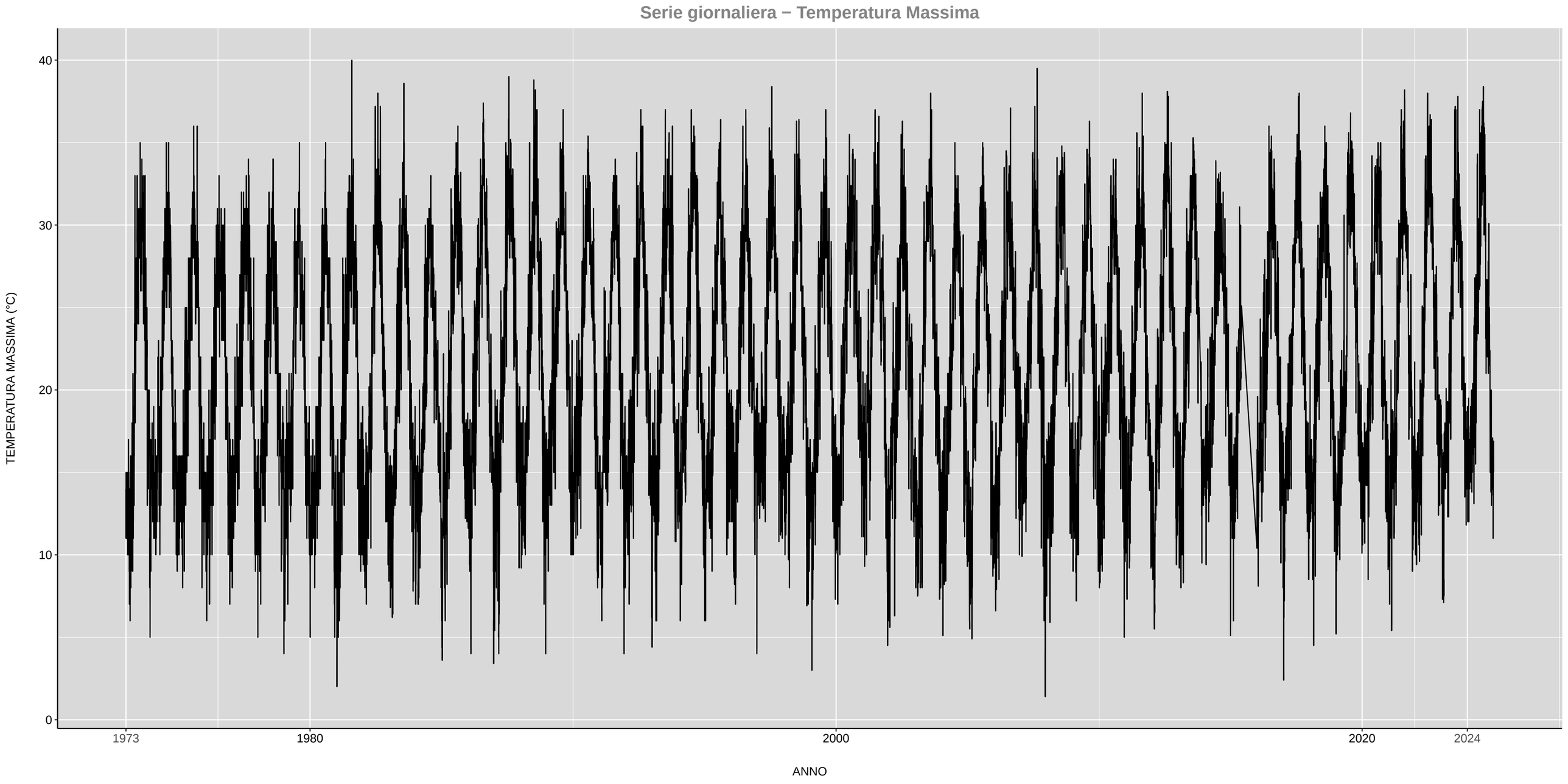

Autocorrelazione annuale − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
20
25
LAG ANNUALE

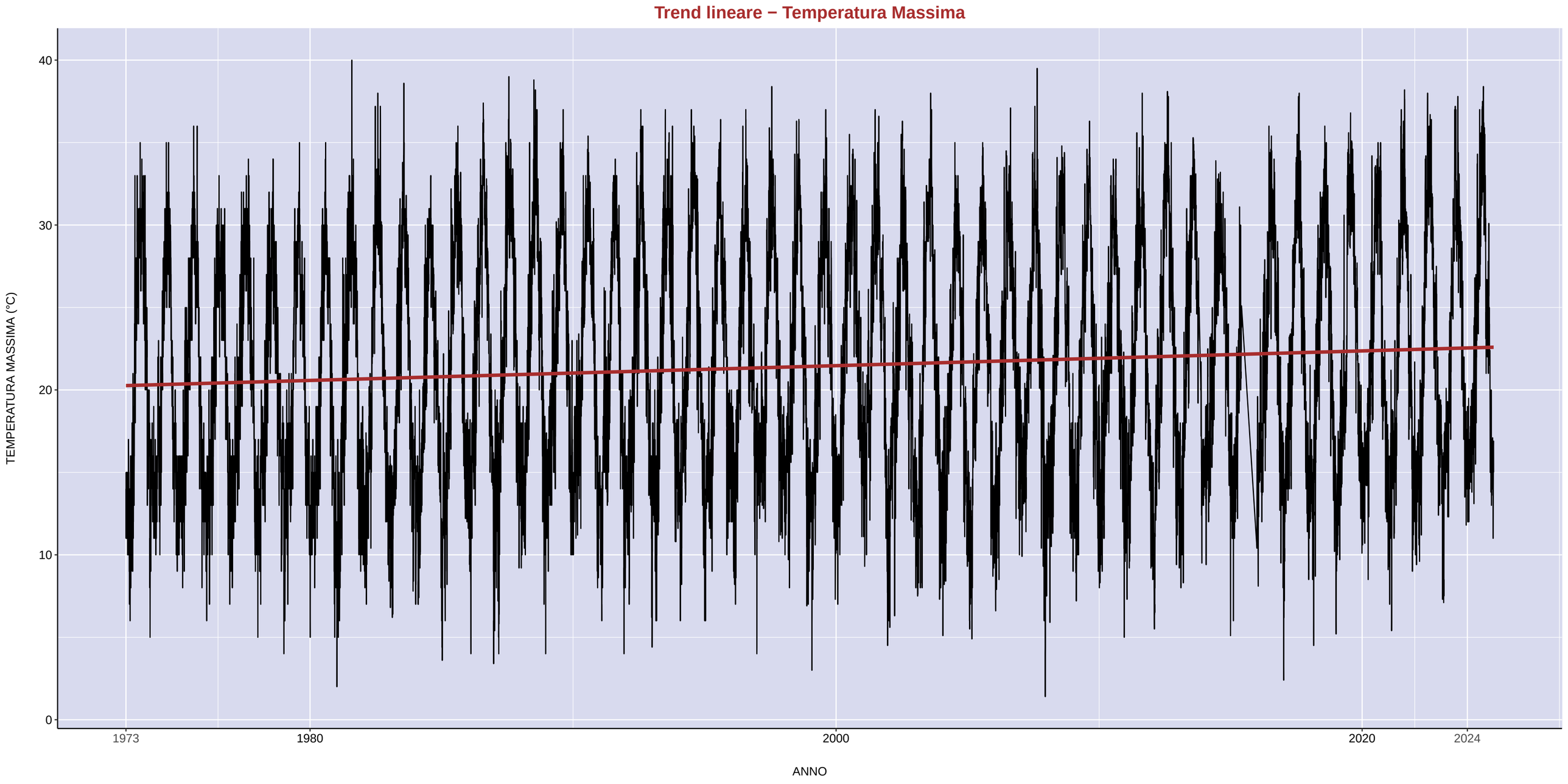
Napoli – Stazione di Napoli Capodichino

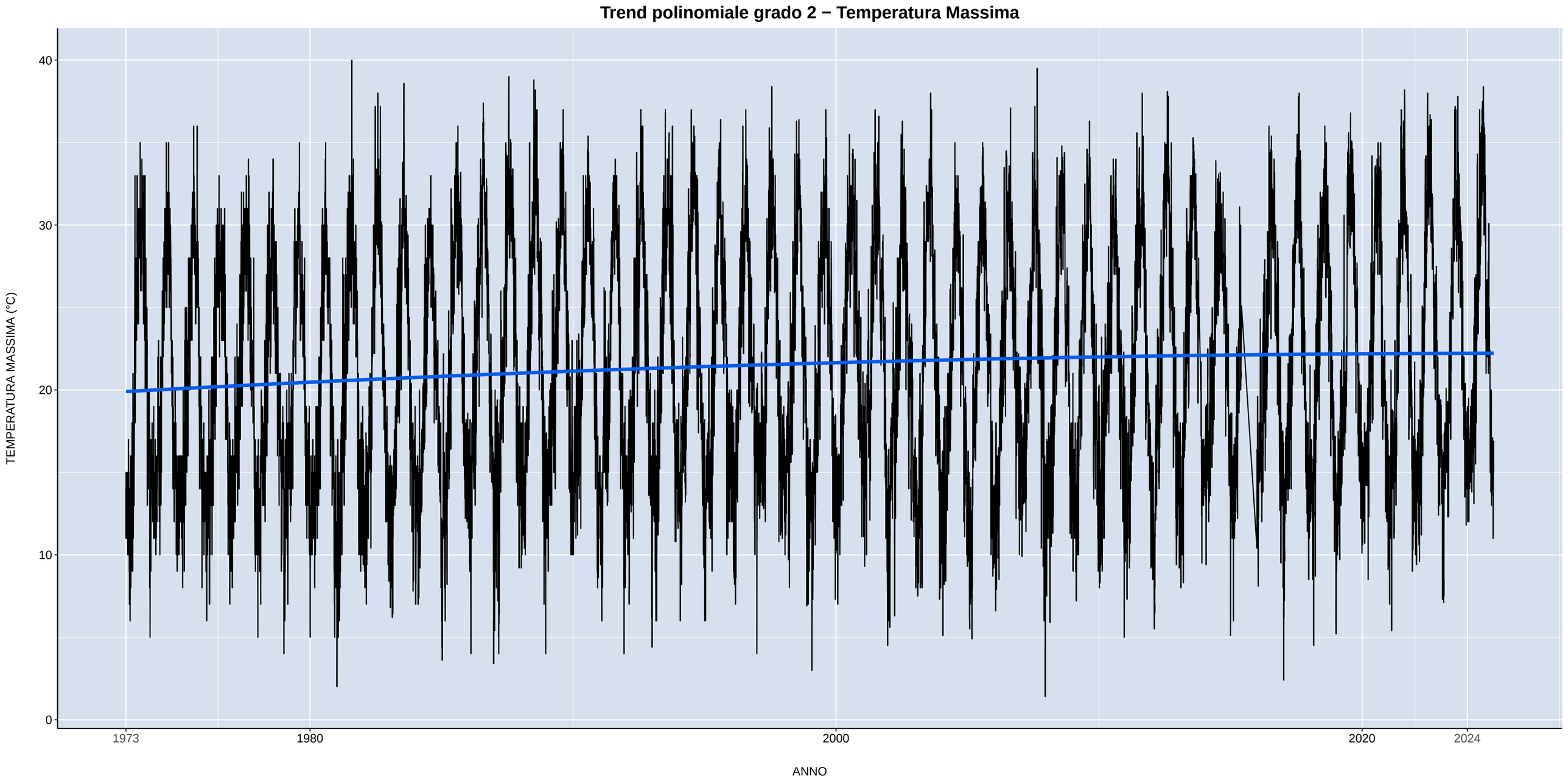

Serie giornaliera – Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1951
1960
1980
2000
2020
2024
ANNO

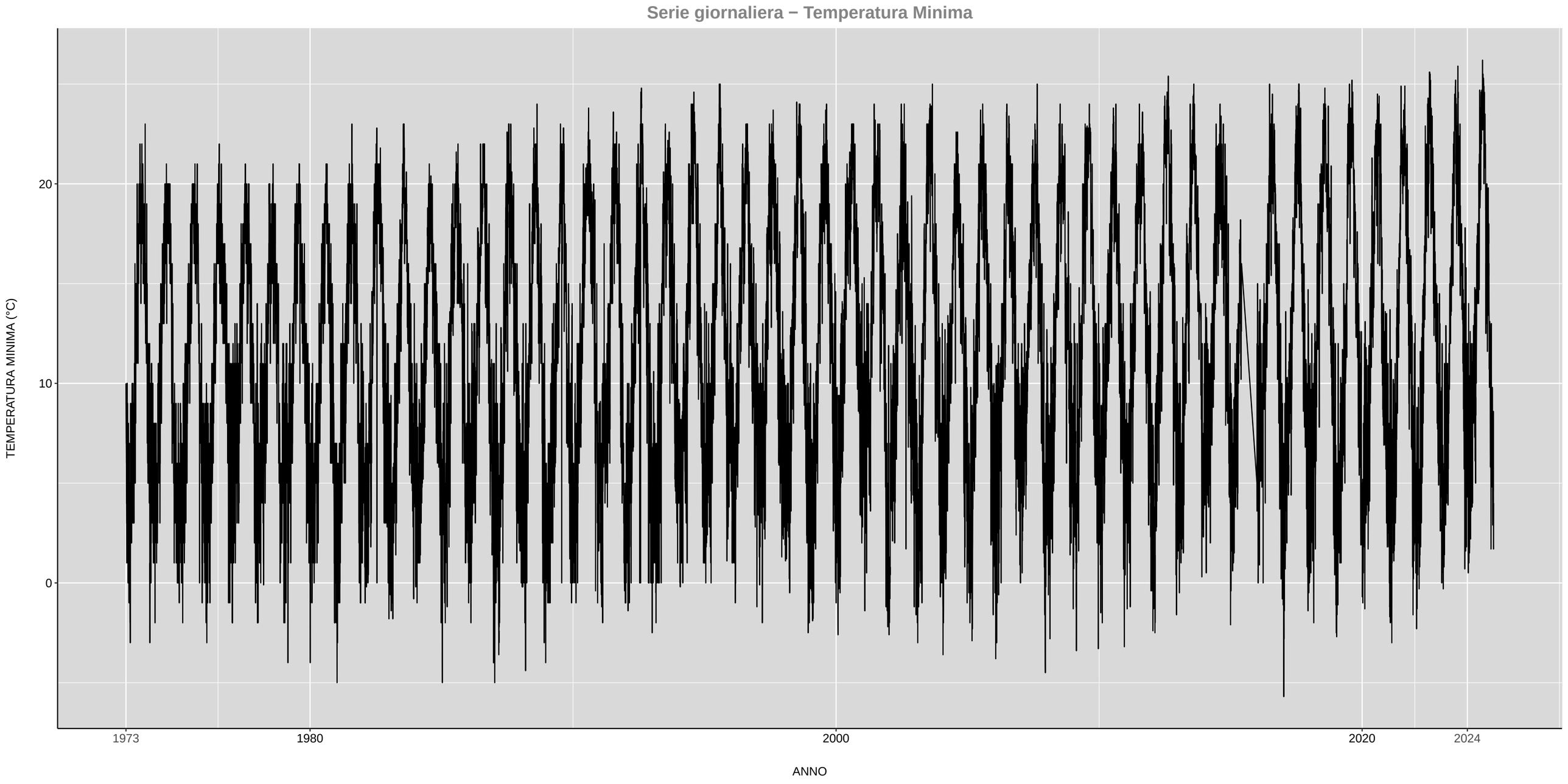

Serie giornaliera − Temperatura Minima
TEMPERATURA MINIMA (°C)
20
10
0
1951
1960
1980
2000
2020
2024
ANNO

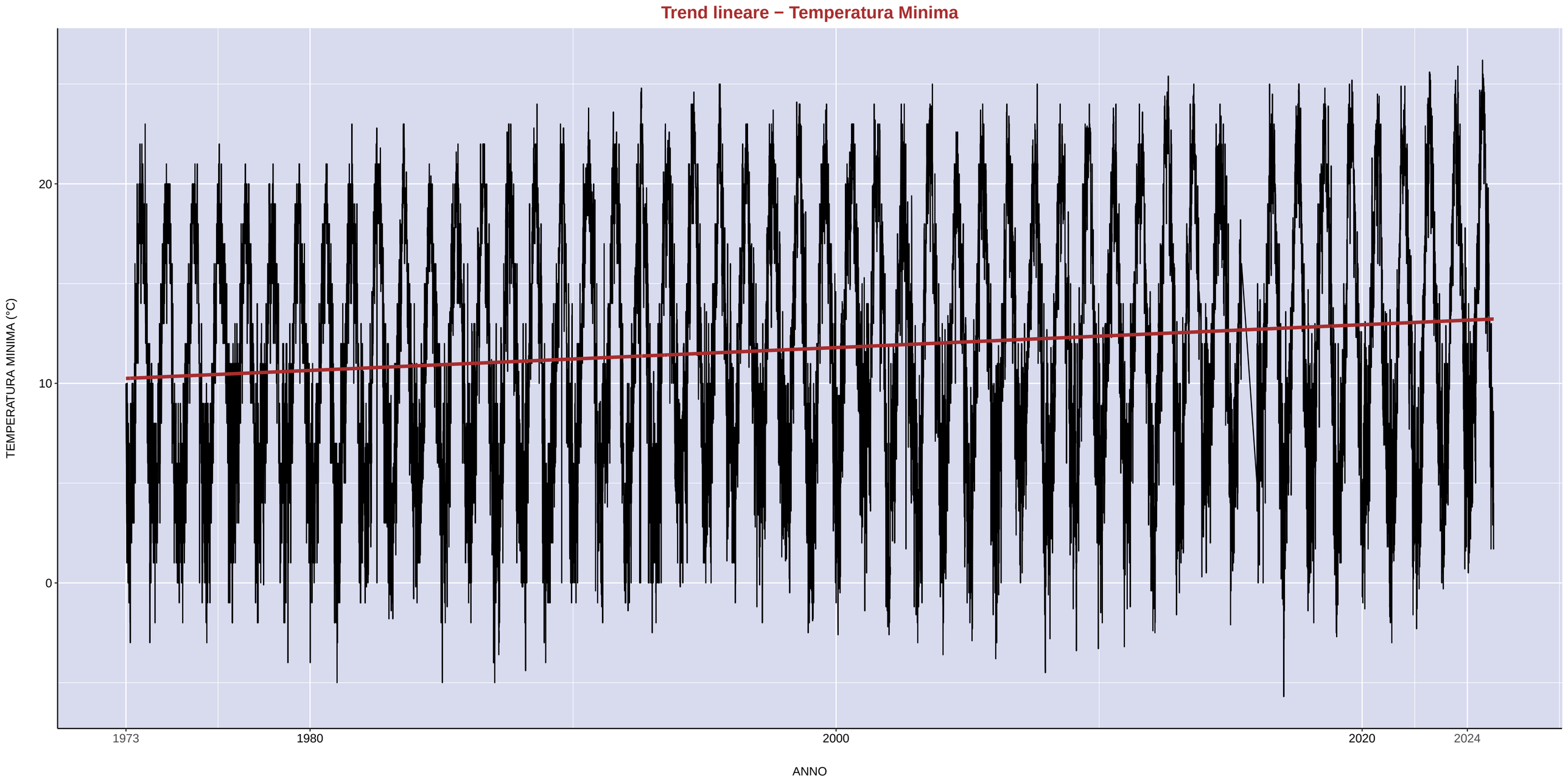

Serie giornaliera − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
1951
1960
1980
2000
2020
2024
ANNO

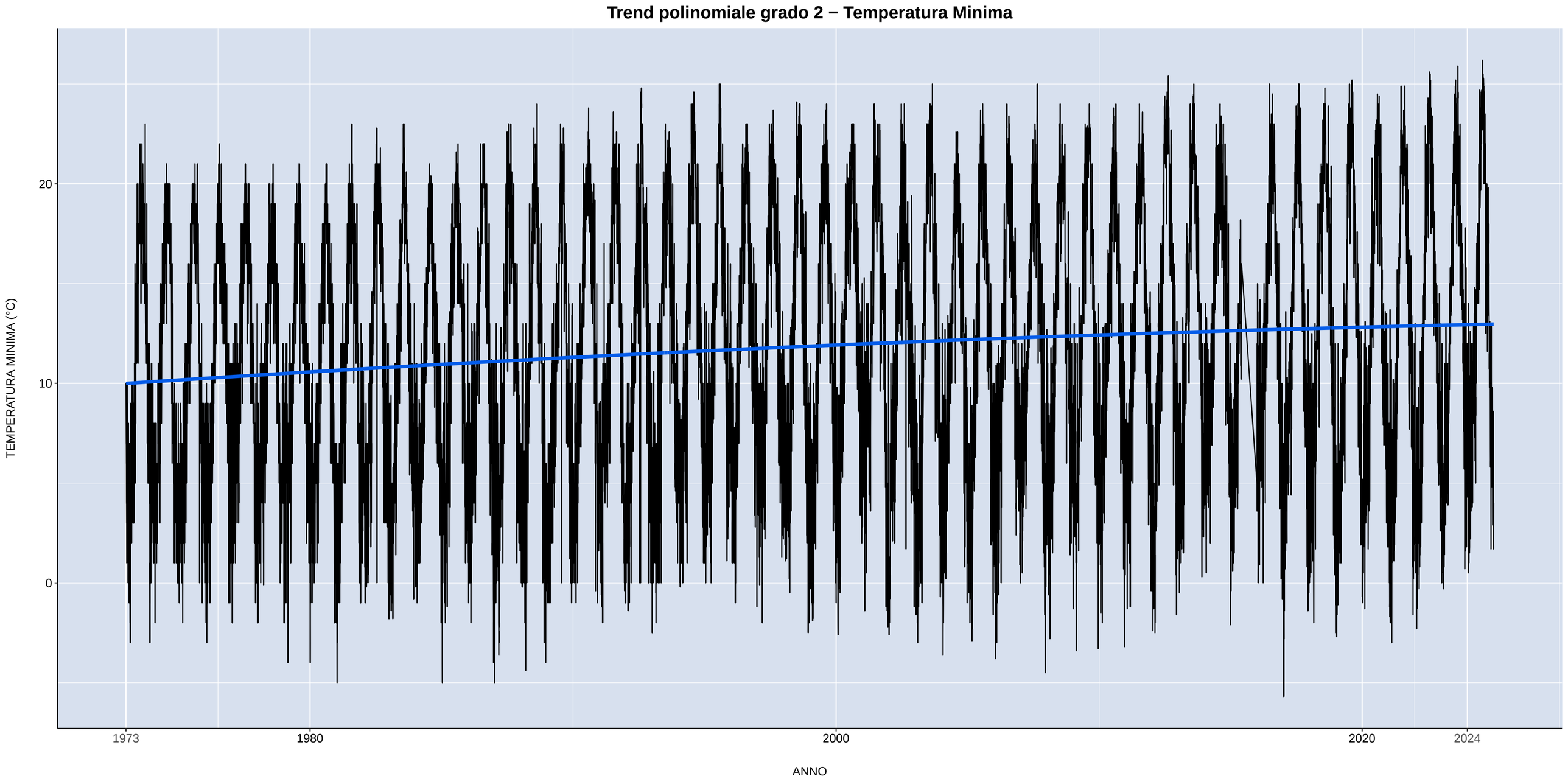

Serie giornaliera – Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1973
1980
2000
2020
2024
ANNO

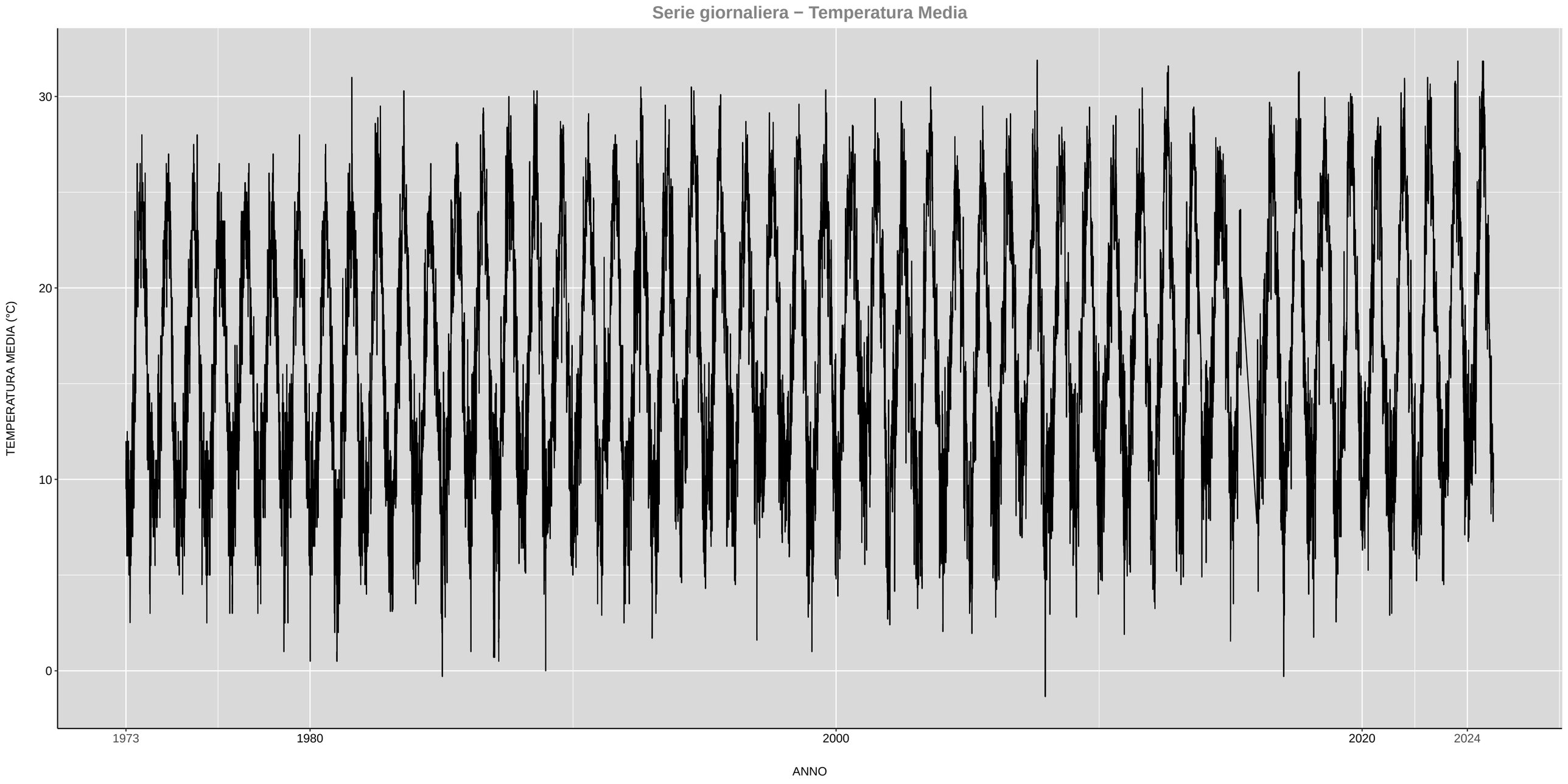

Trend lineare − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1973
1980
2000
2020
2024
ANNO

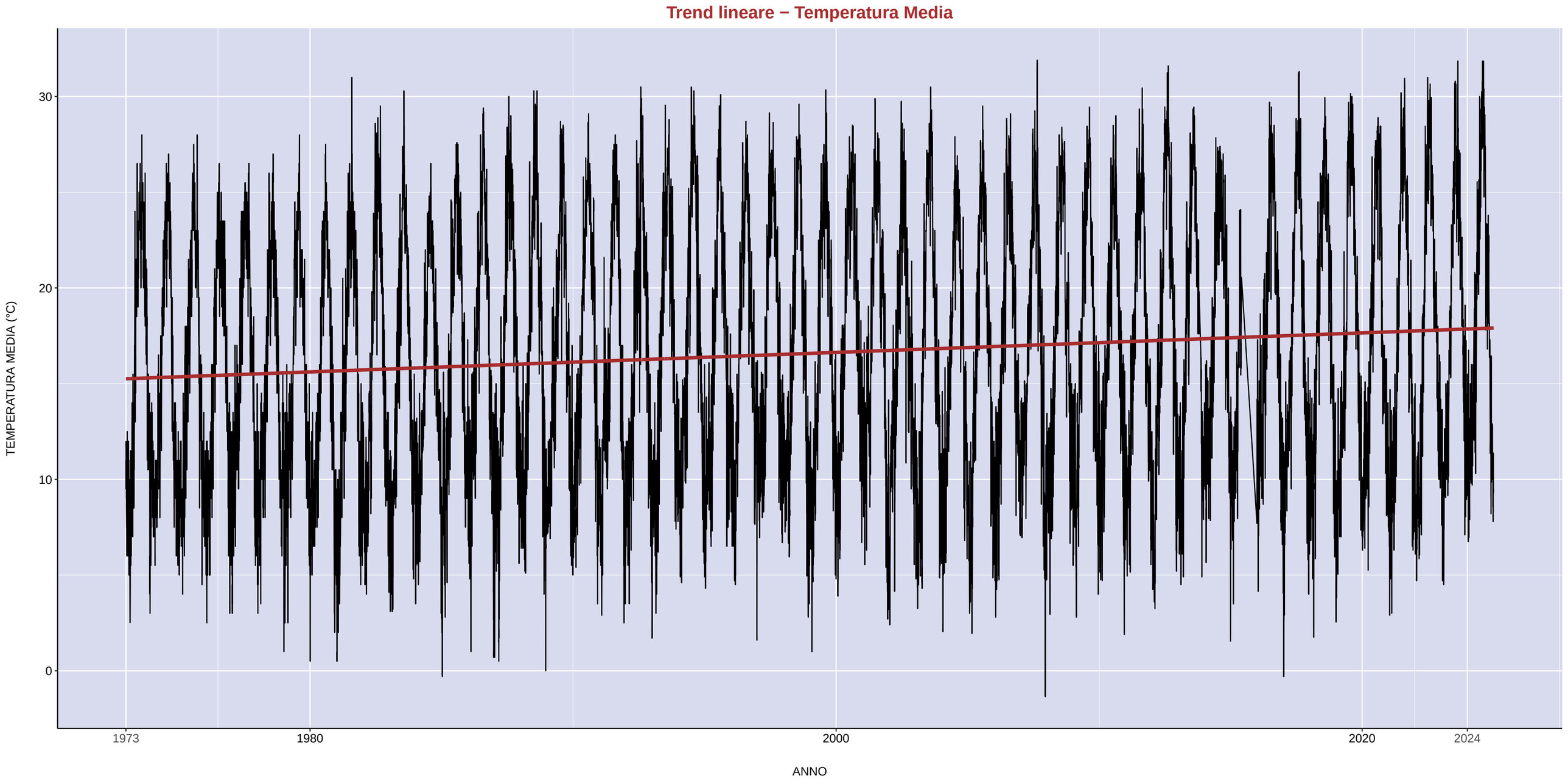

Trend polinomiale grado 2 − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1973
1980
2000
2020
2024
ANNO

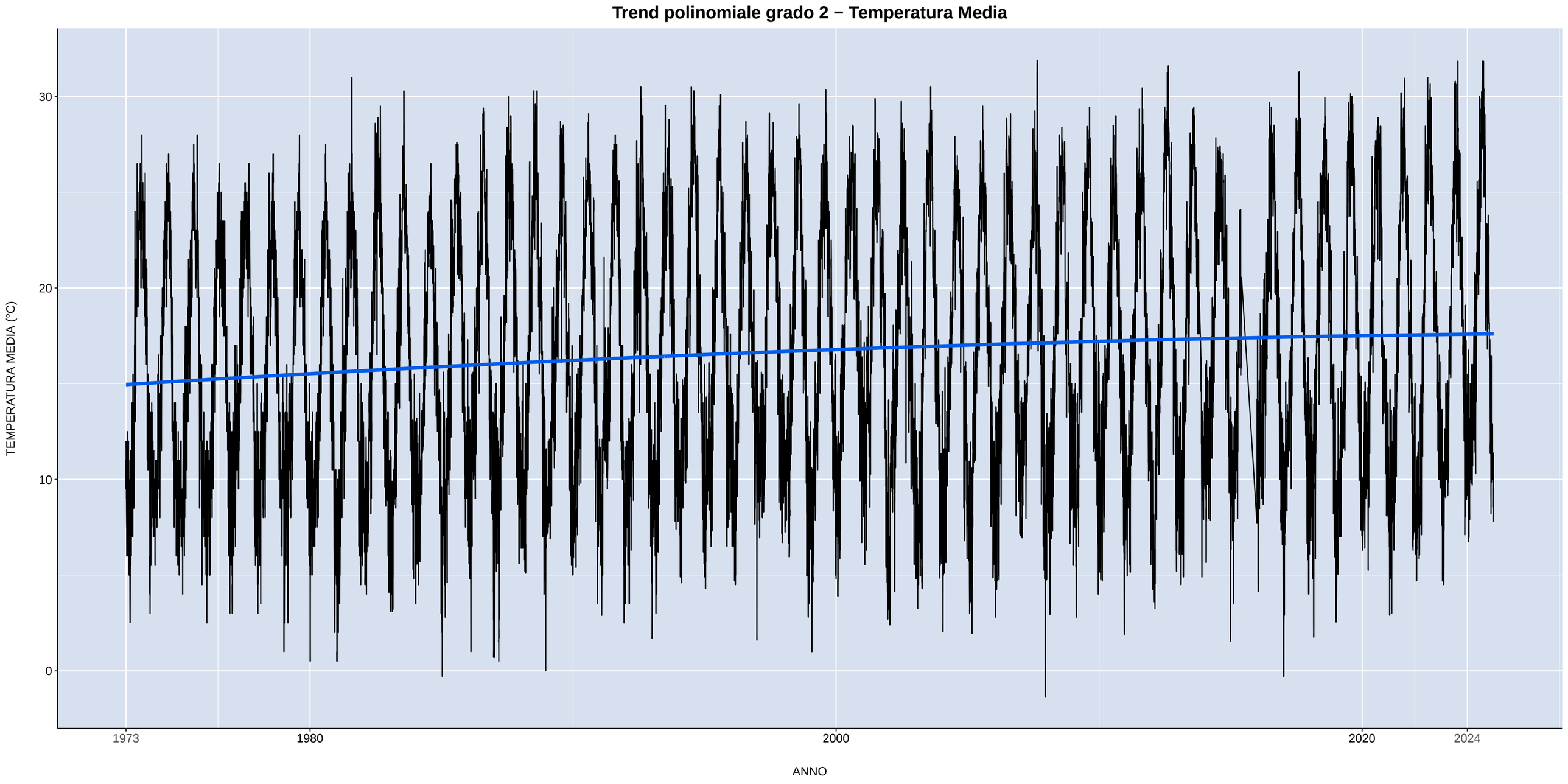

Serie giornaliera − Temperatura Minima
TEMPERATURA MINIMA (°C)
20
10
0
1973
1980
2000
2020
2024
ANNO

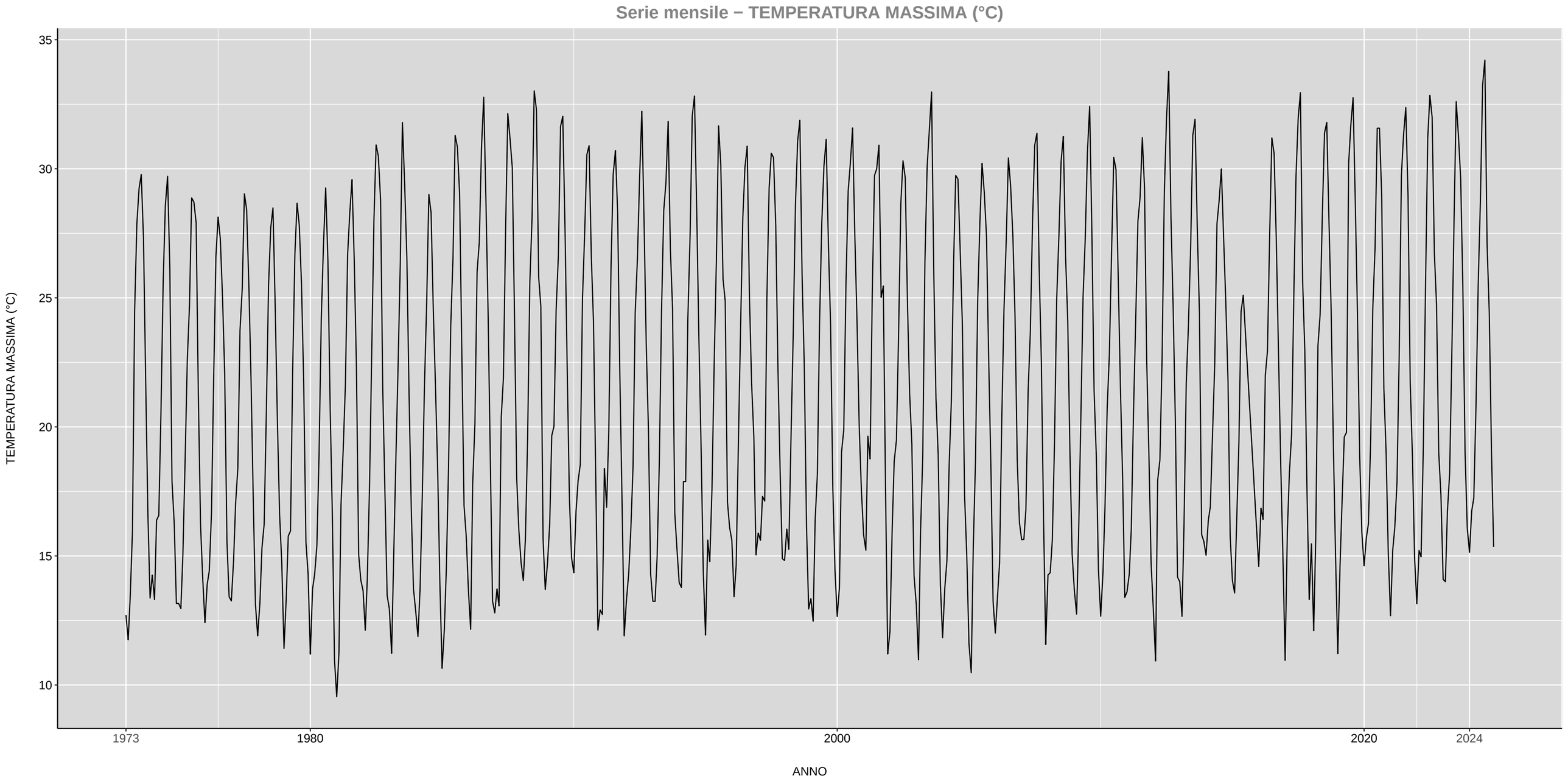

Trend lineare − Temperatura Minima
TEMPERATURA MINIMA (°C)
20
10
0
1973
1980
2000
2020
2024
ANNO

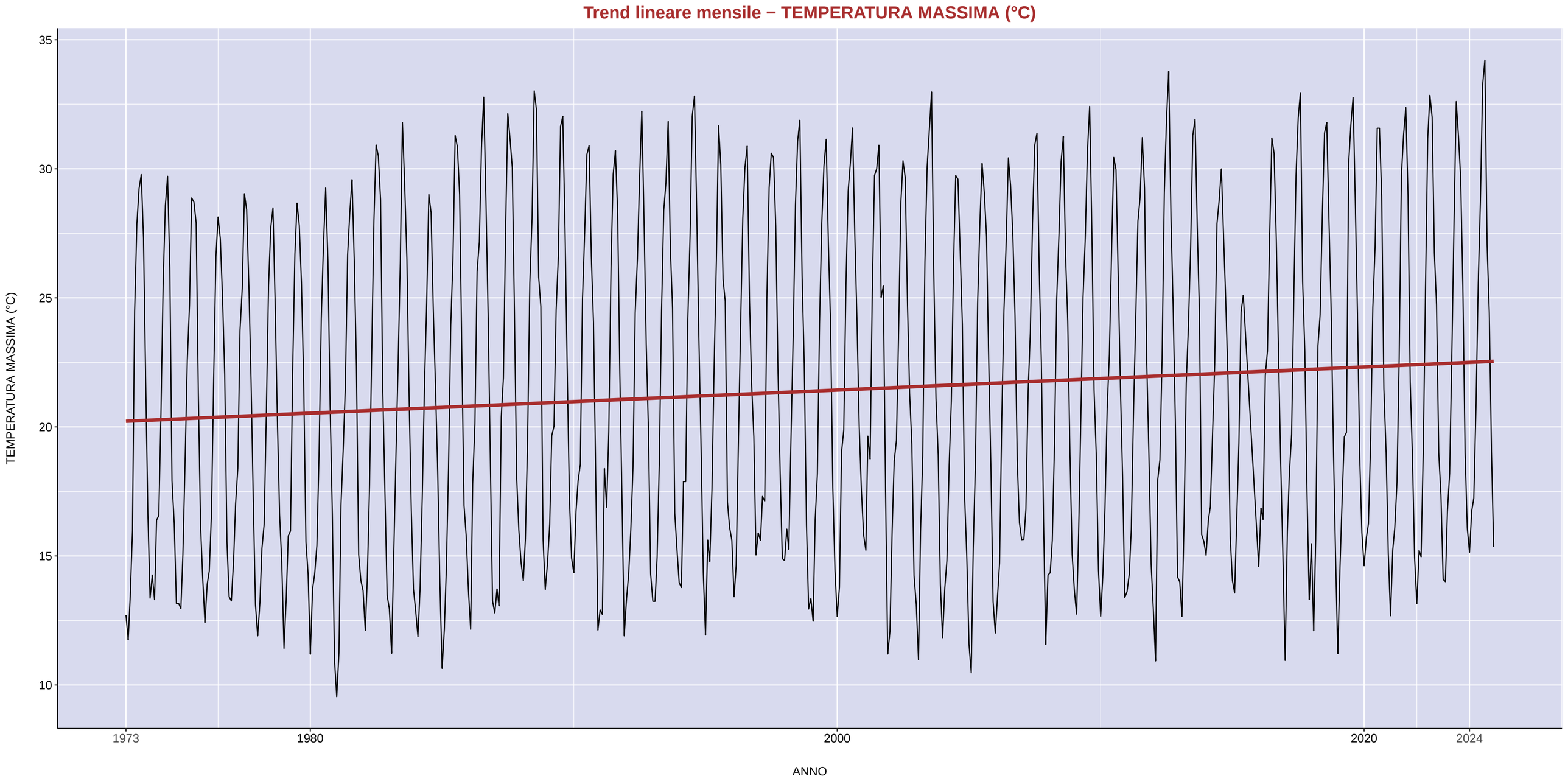

Trend polinomiale grado 2 − Temperatura Minima
TEMPERATURA MINIMA (°C)
20
10
0
1973
1980
2000
2020
2024
ANNO

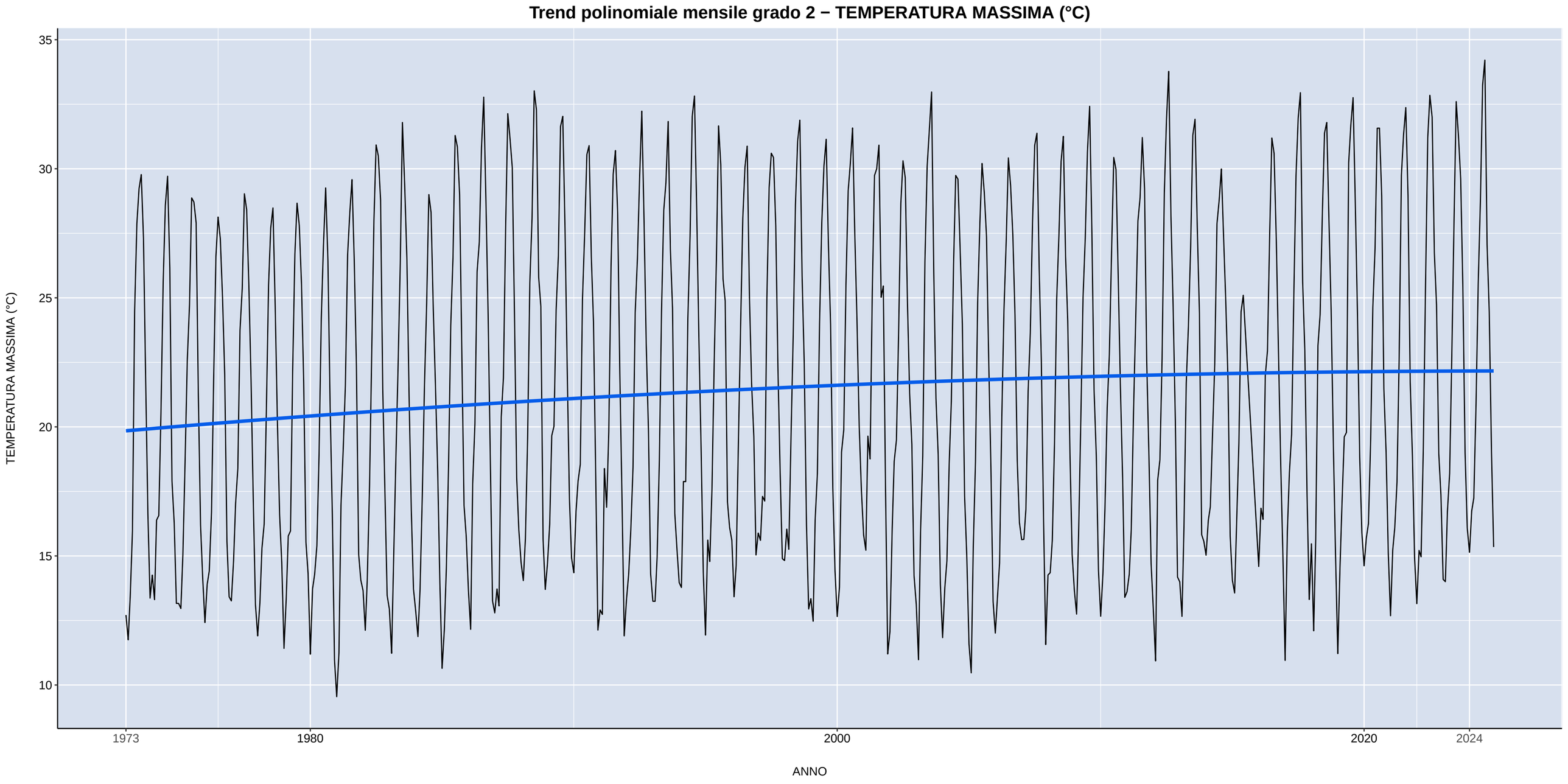

Serie giornaliera − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
1973
1980
2000
2020
2024
ANNO

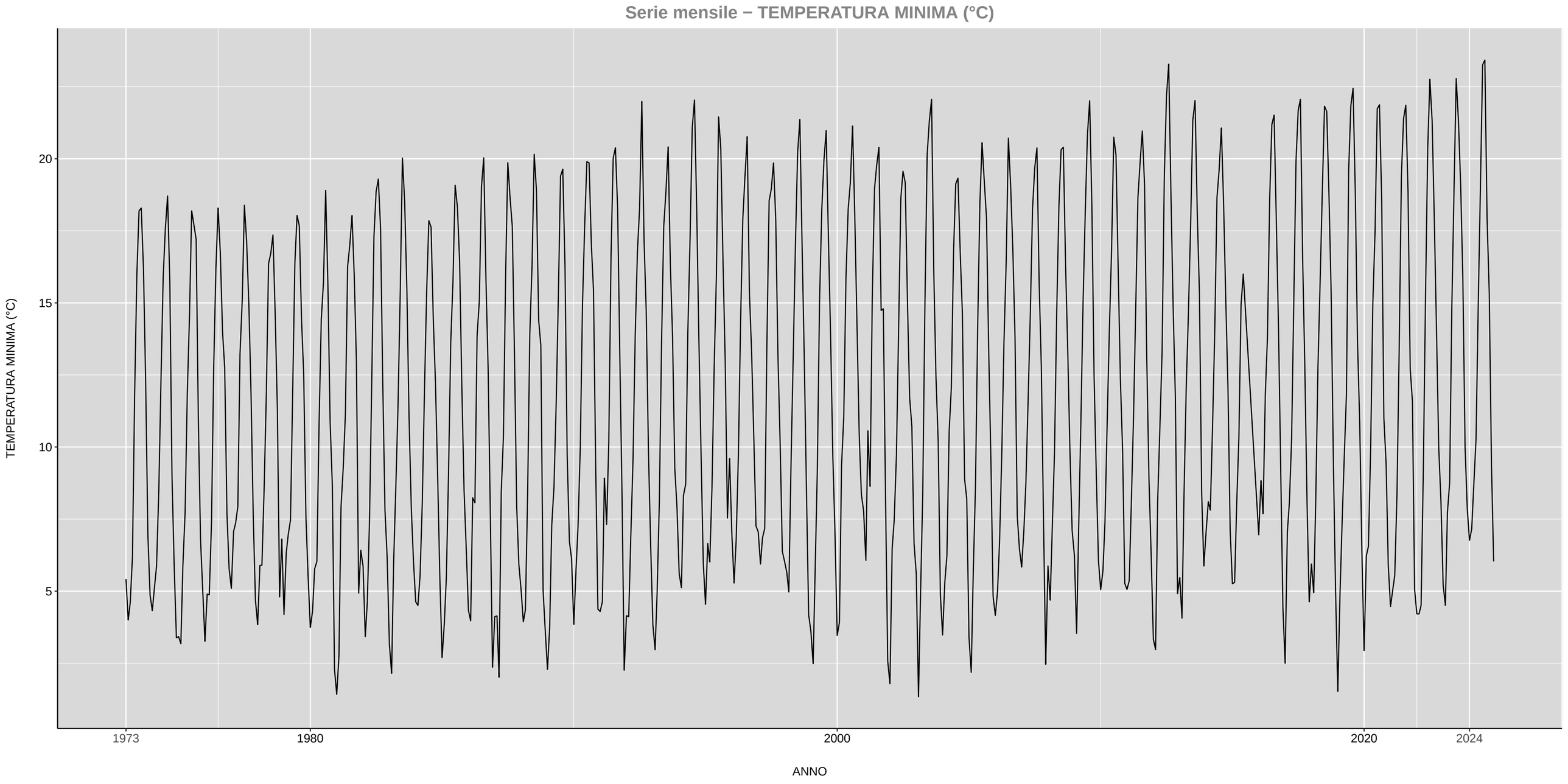

Trend lineare − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
1973
1980
2000
2020
2024
ANNO

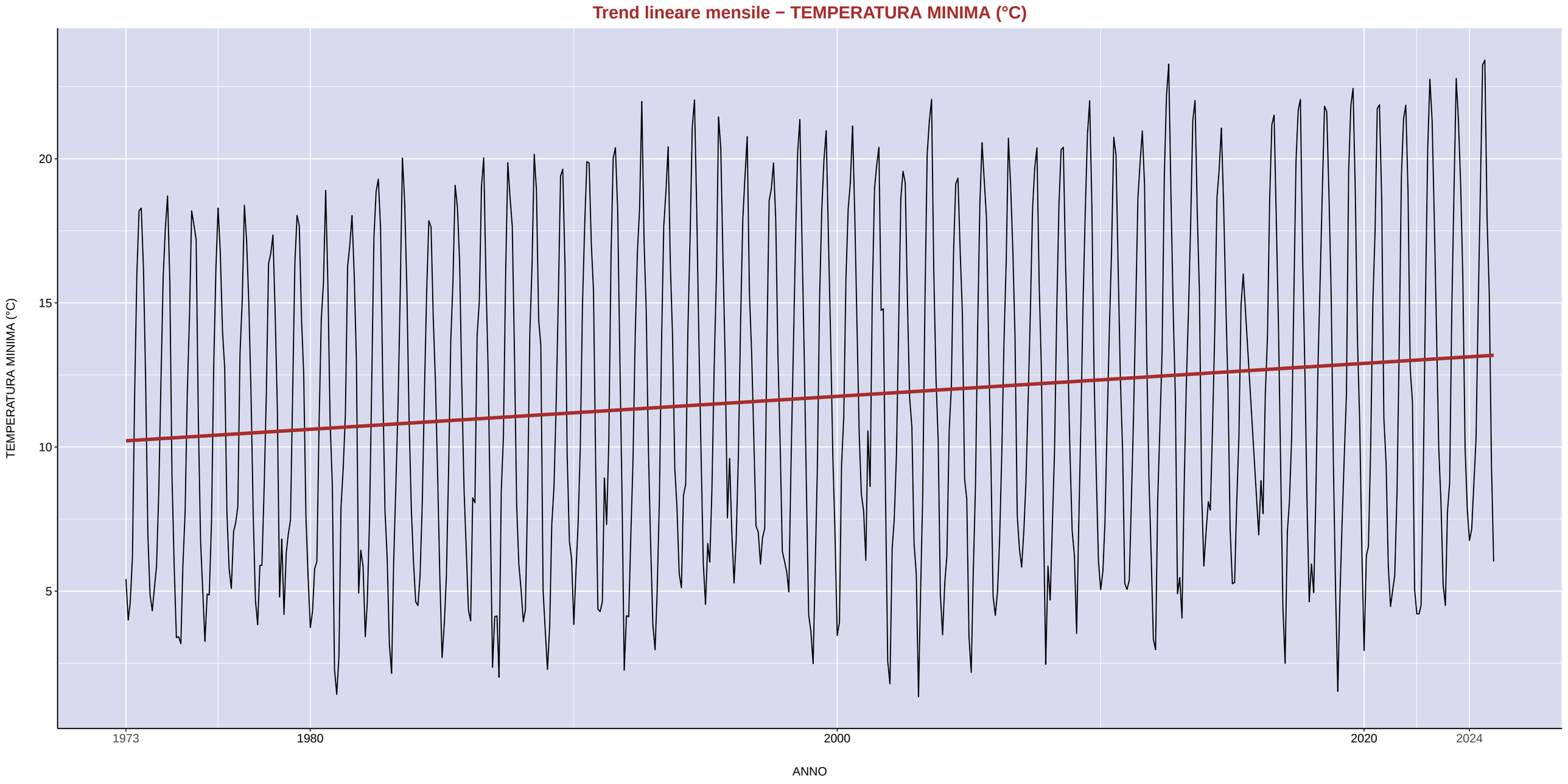

Trend polinomiale grado 2 − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
1973
1980
2000
2020
2024
ANNO

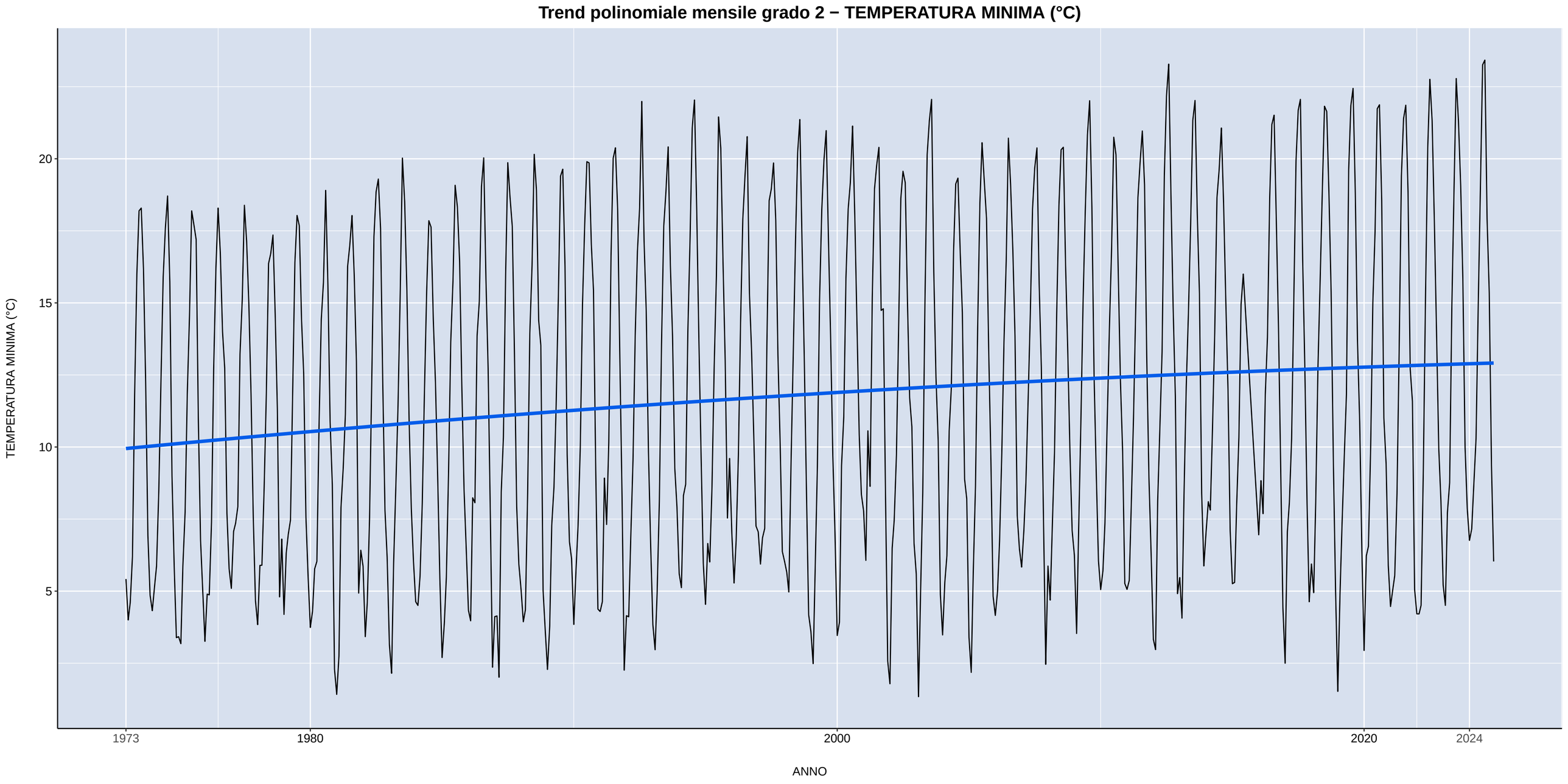

Serie mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
35
30
25
20
15
10
1973
1980
2000
2020
2024
ANNO

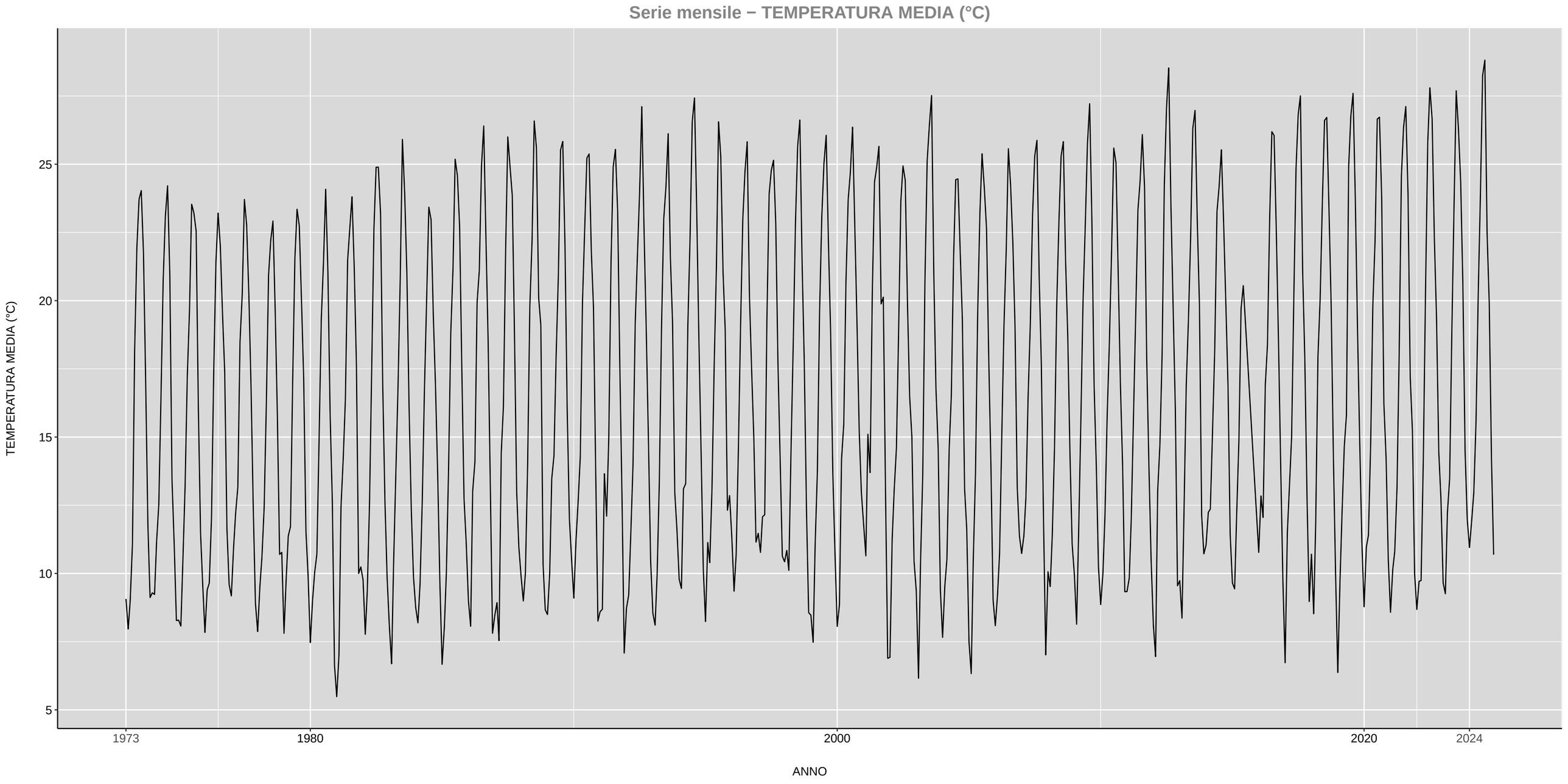

Trend lineare mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
35
30
25
20
15
10
1973
1980
2000
2020
2024
ANNO

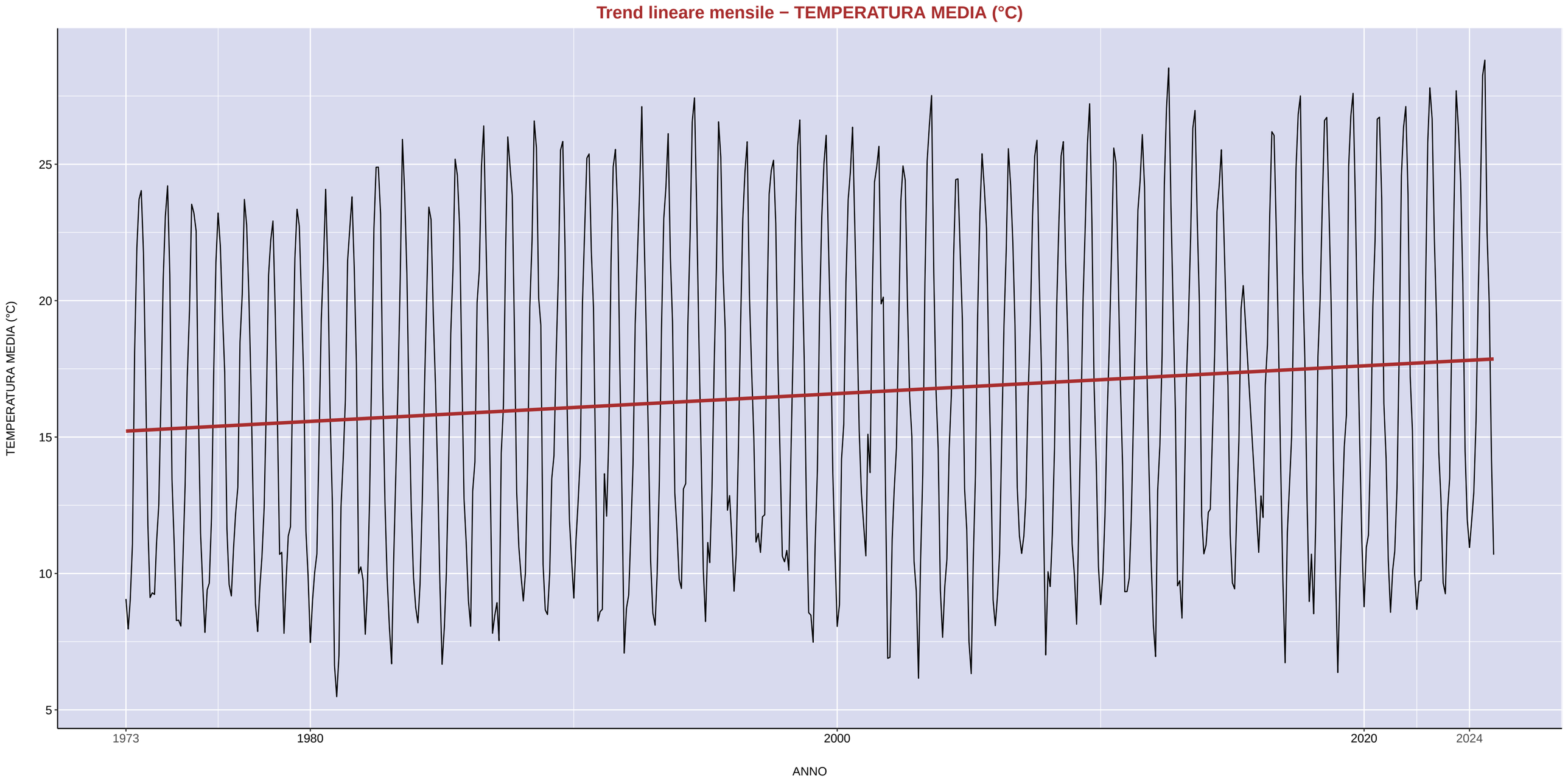

Trend polinomiale mensile grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
35
30
25
20
15
10
1973
1980
2000
2020
2024
ANNO

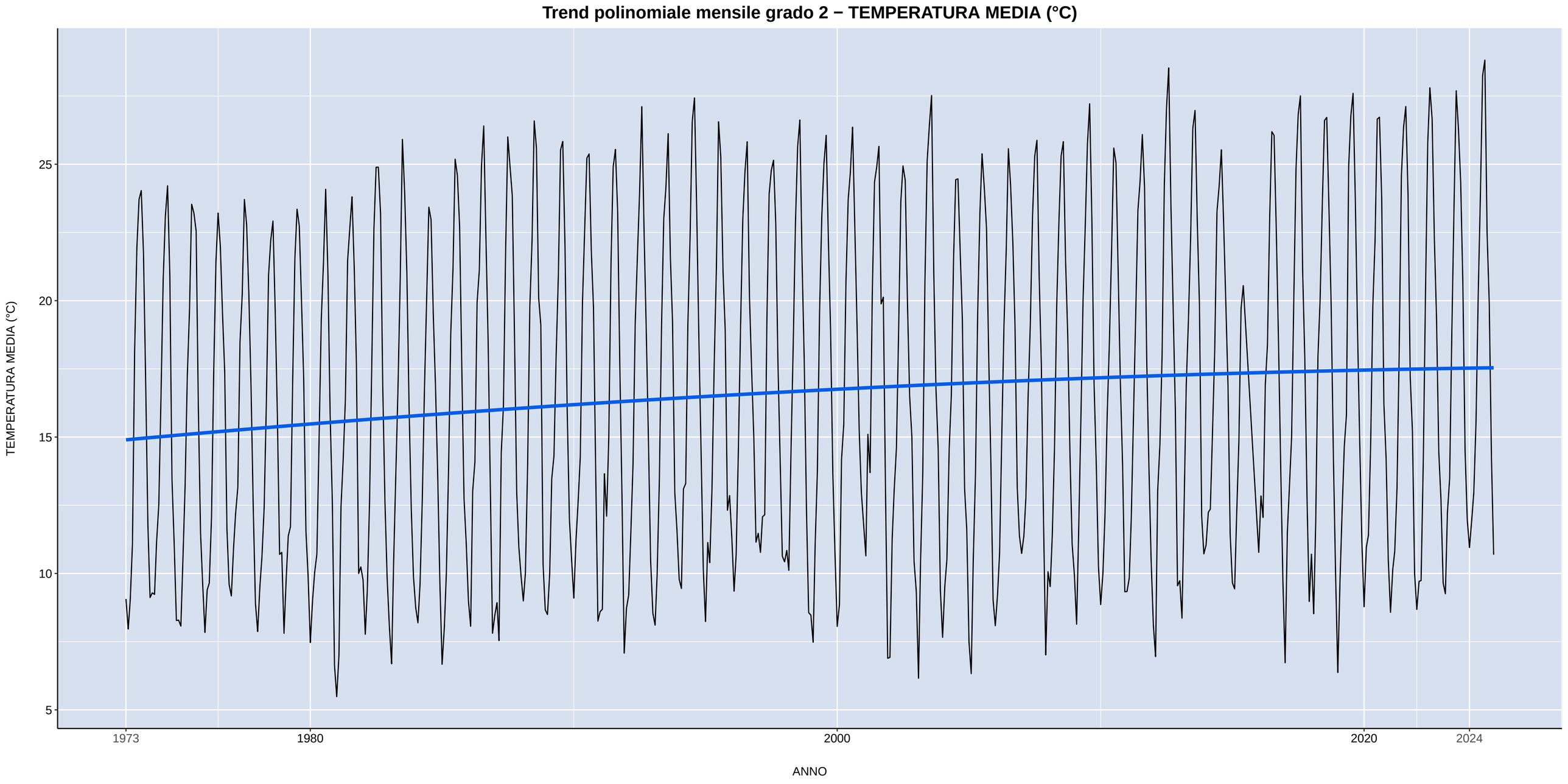

Serie mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
15
10
5
1973
1980
2000
2020
2024
ANNO

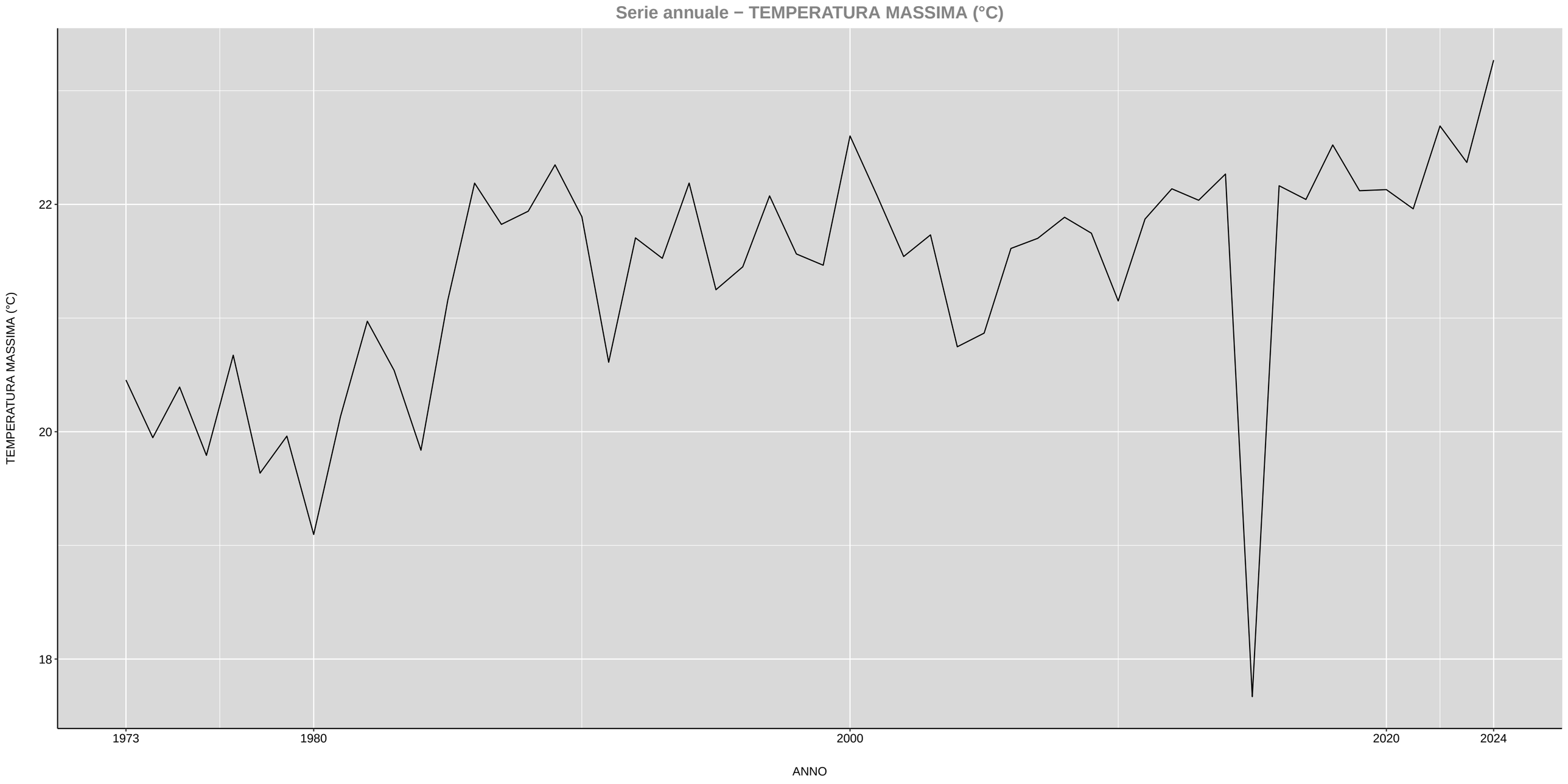

Trend lineare mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
15
10
5
1973
1980
2000
2020
2024
ANNO

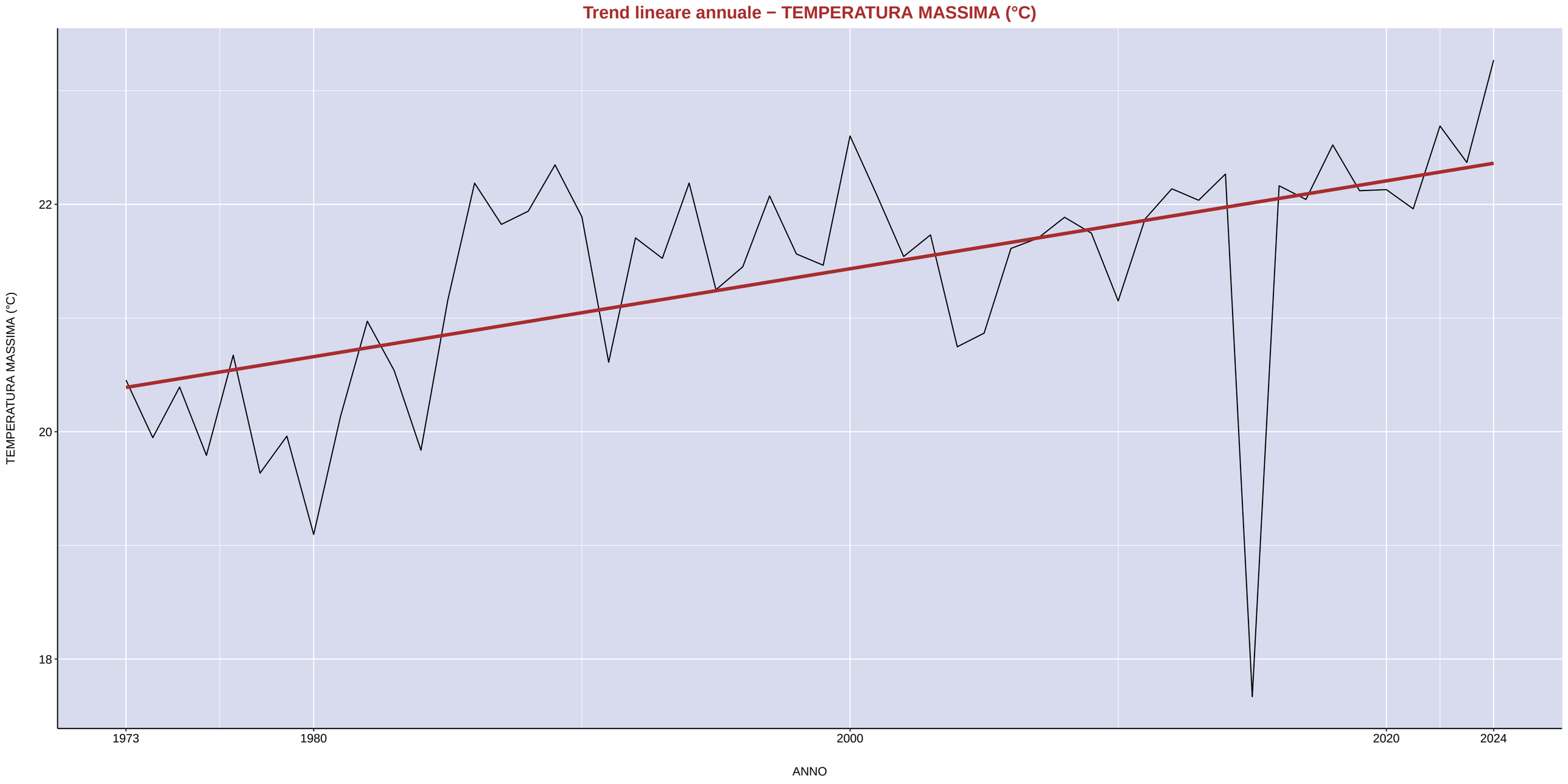

Trend polinomiale mensile grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
15
10
5
1973
1980
2000
2020
2024
ANNO

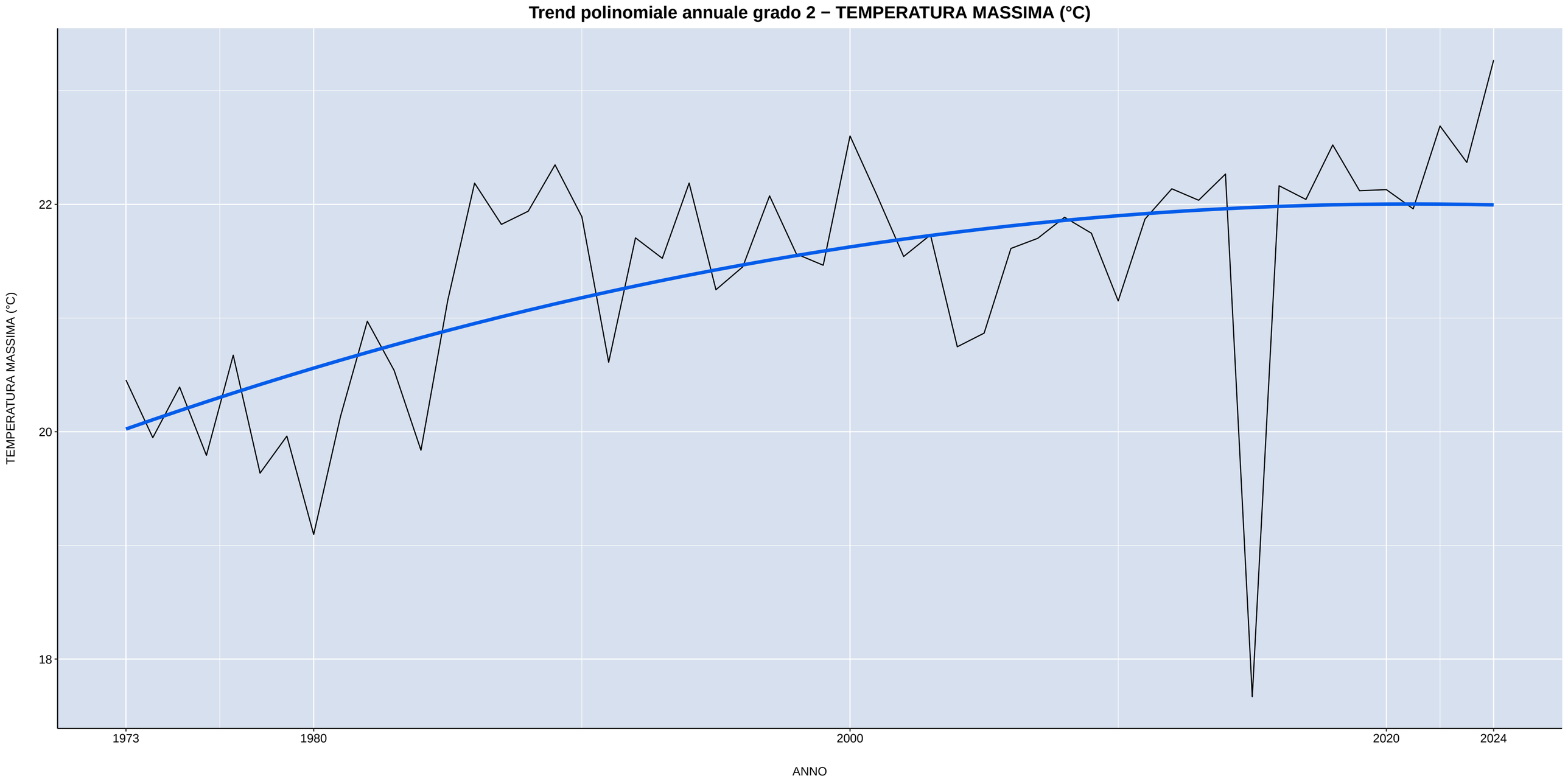

Serie mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
25
20
15
10
5
1973
1980
2000
2020
2024
ANNO

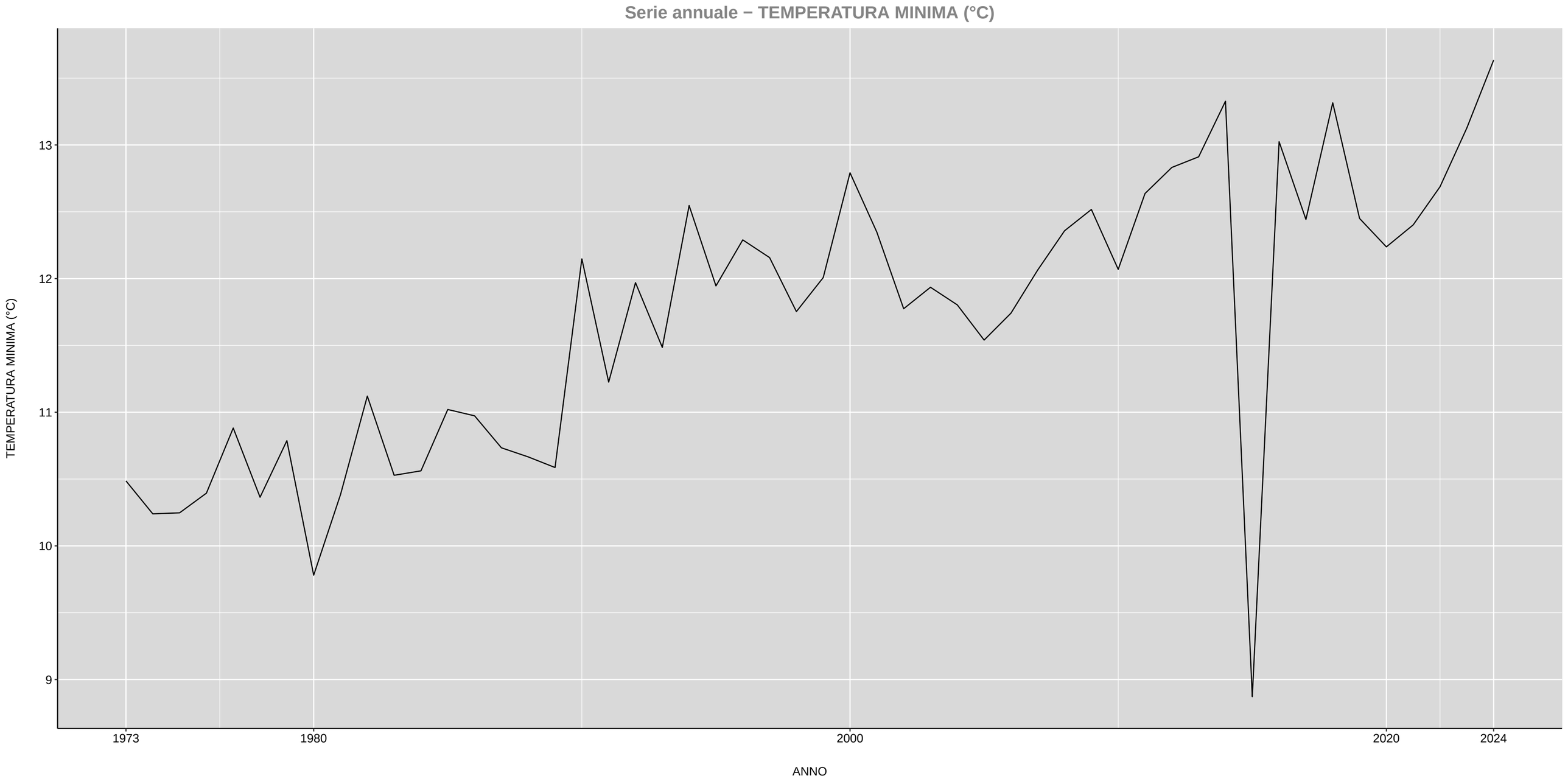

Trend lineare mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
25
20
15
10
5
1973
1980
2000
2020
2024
ANNO

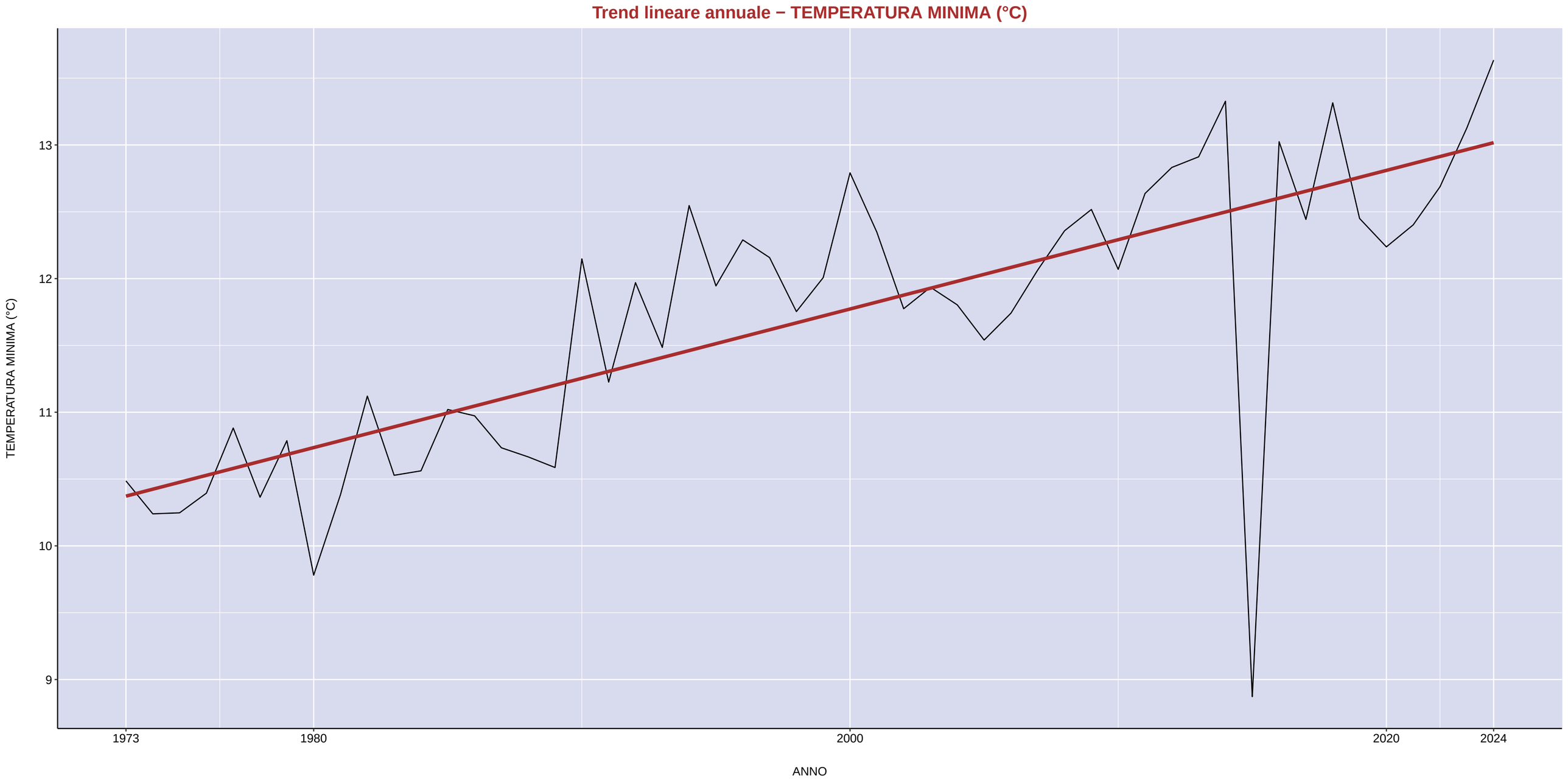

Trend polinomiale mensile grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
25
20
15
10
5
1973
1980
2000
2020
2024
ANNO

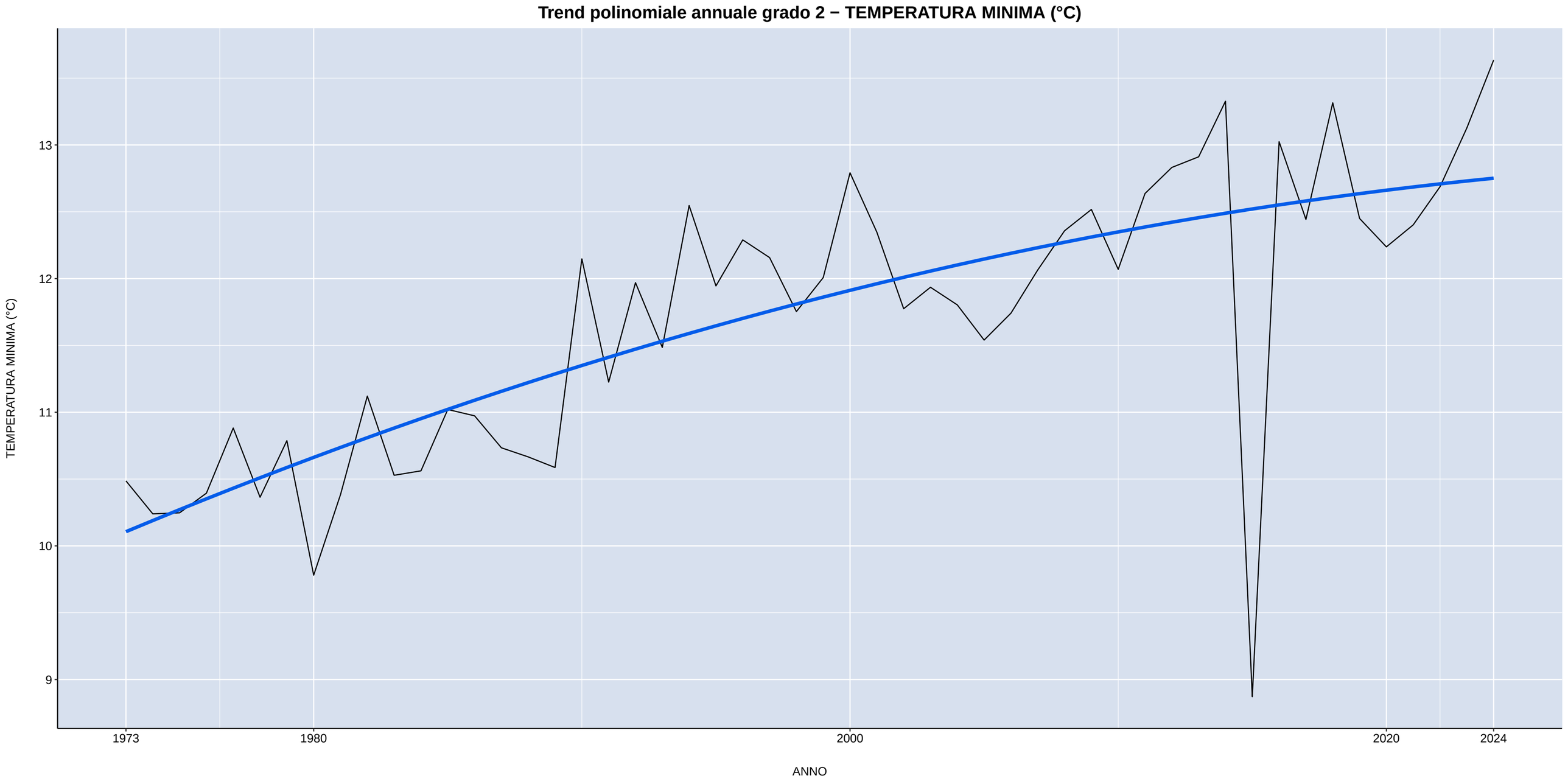

Serie annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
22
20
18
1973
1980
2000
2020
2024
ANNO

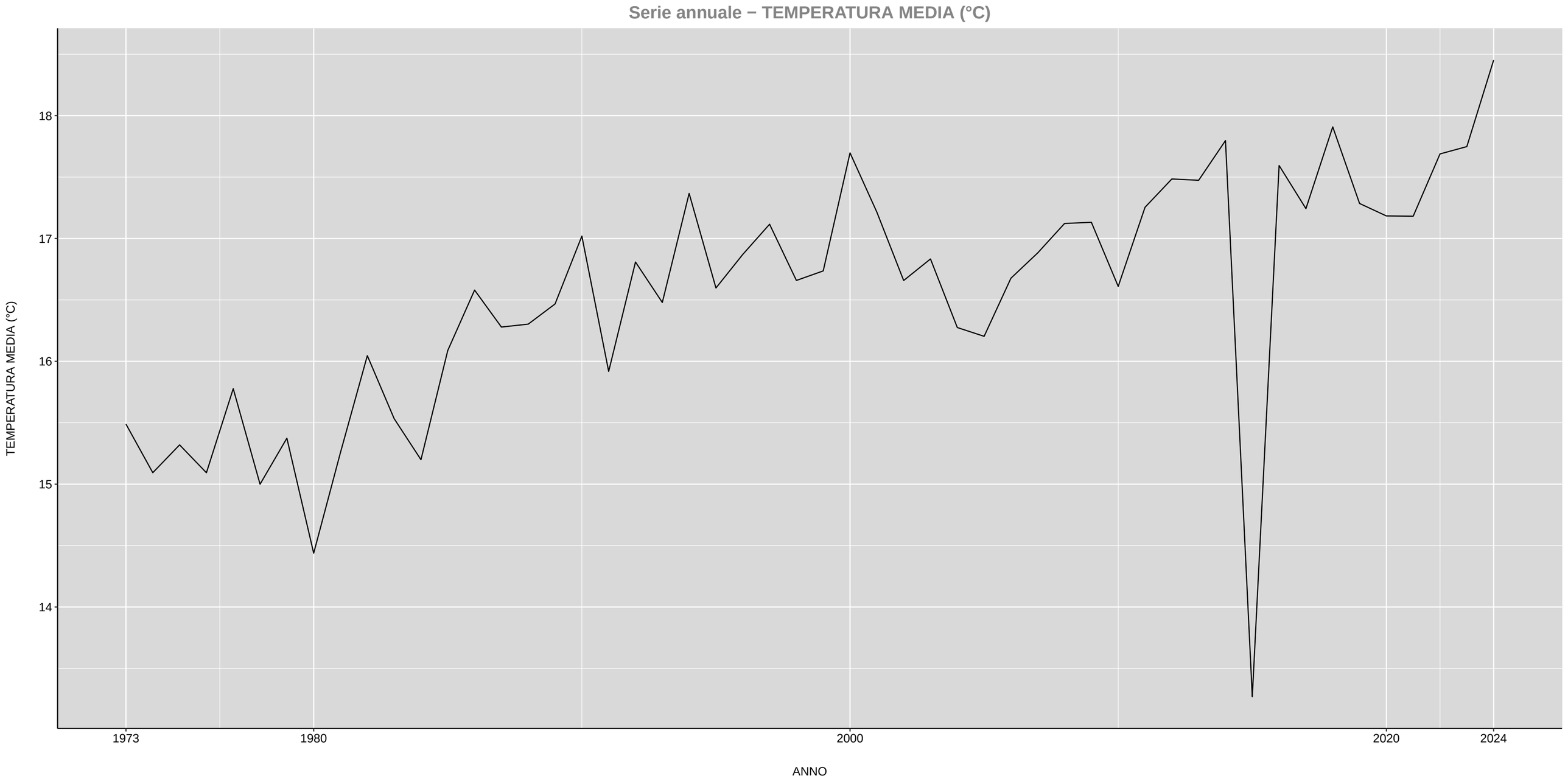

Trend lineare annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
22
20
18
1973
1980
2000
2020
2024
ANNO

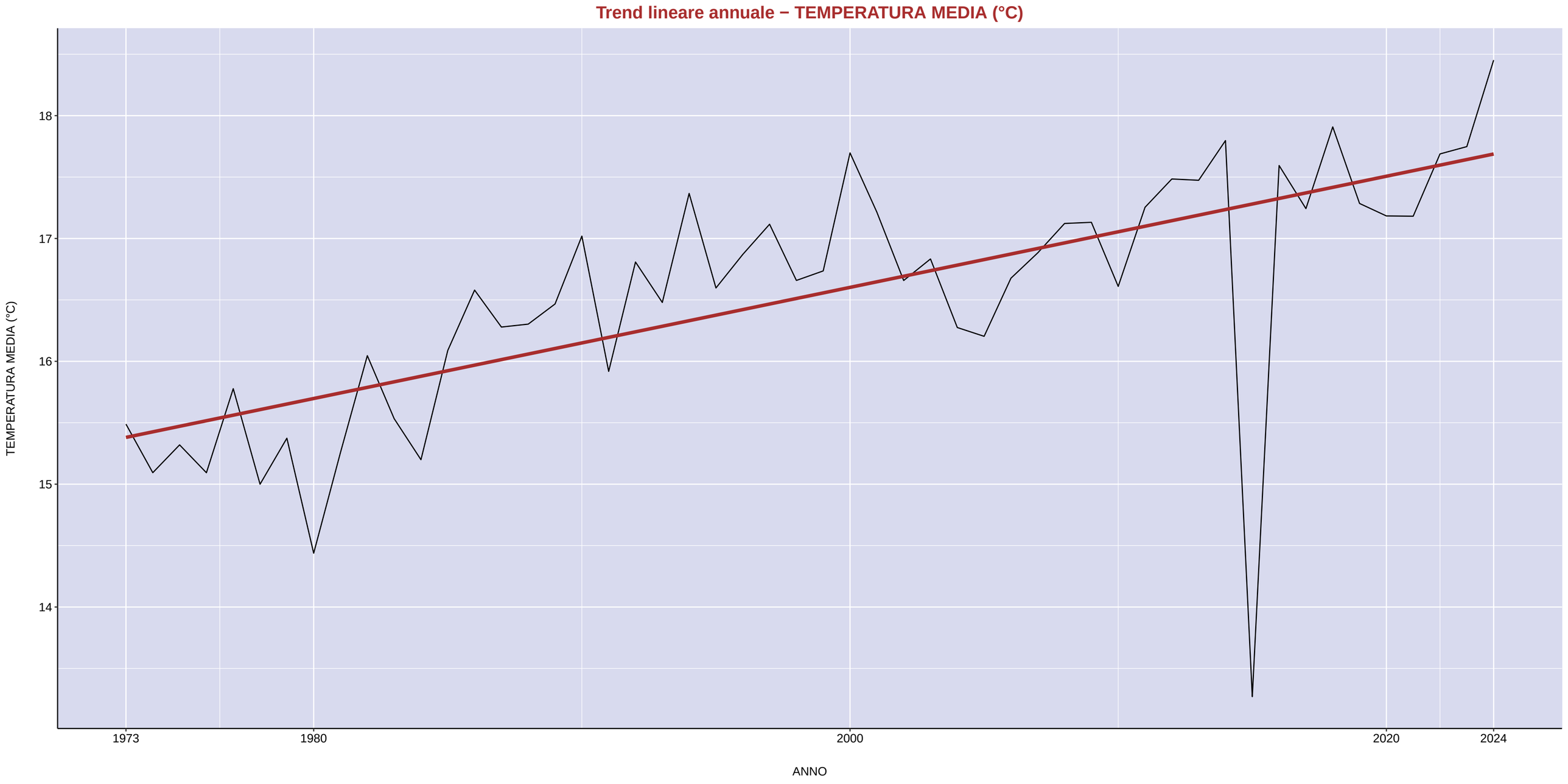

Trend polinomiale annuale grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
22
20
18
1973
1980
2000
2020
2024
ANNO

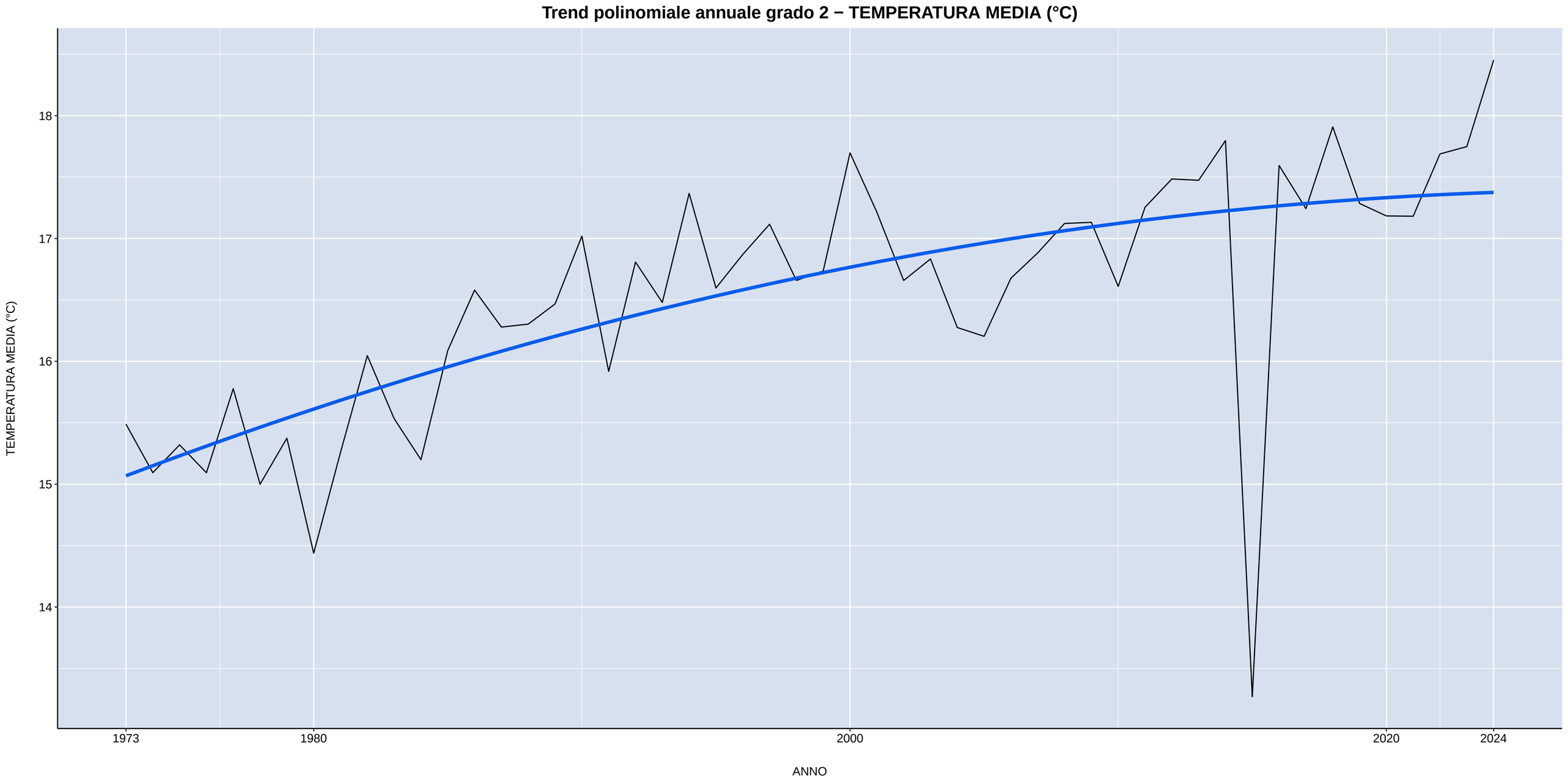

Serie annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
13
12
11
10
9
1973
1980
2000
2020
2024
ANNO

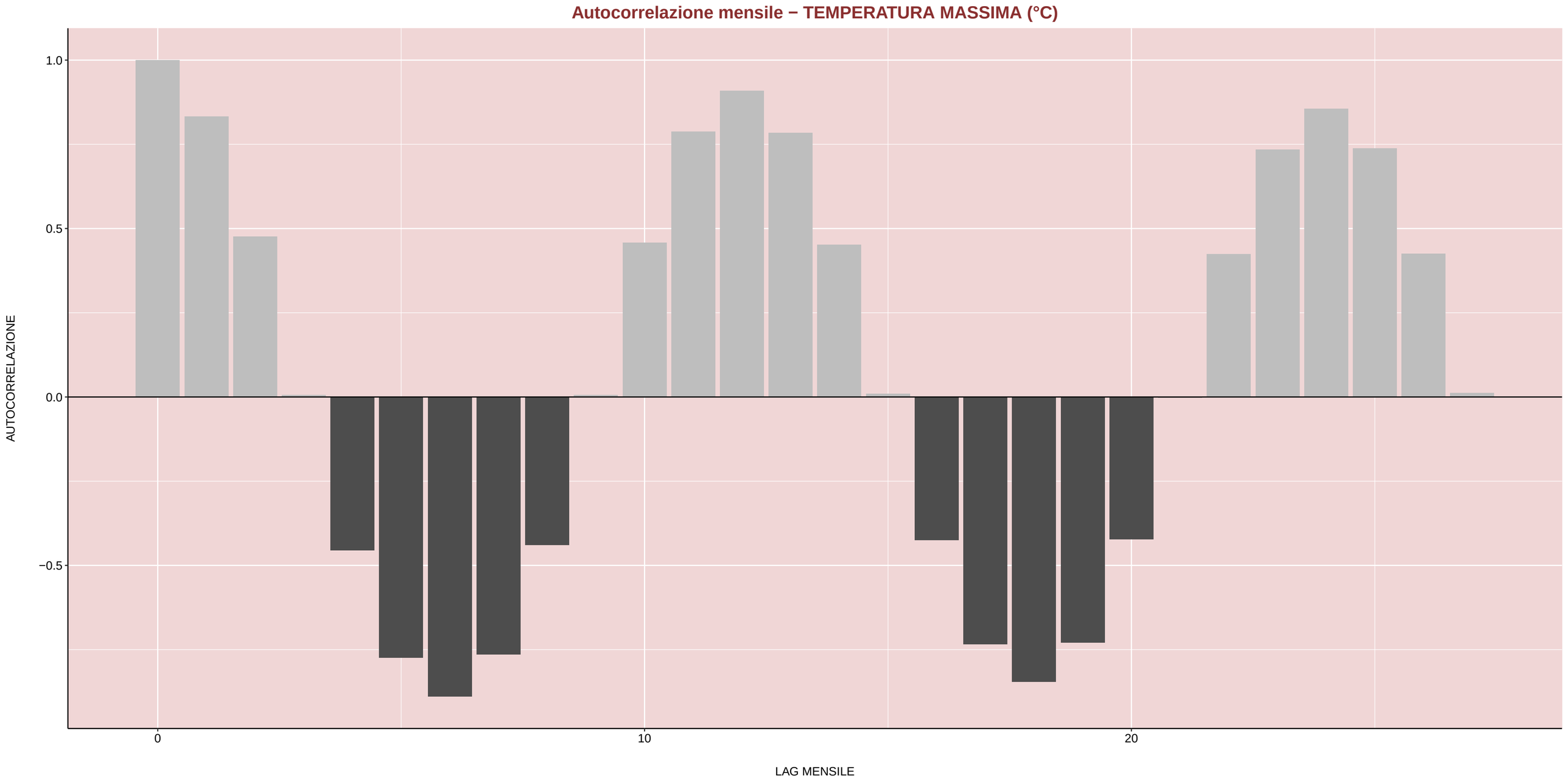

Trend lineare annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
13
12
11
10
9
1973
1980
2000
2020
2024
ANNO

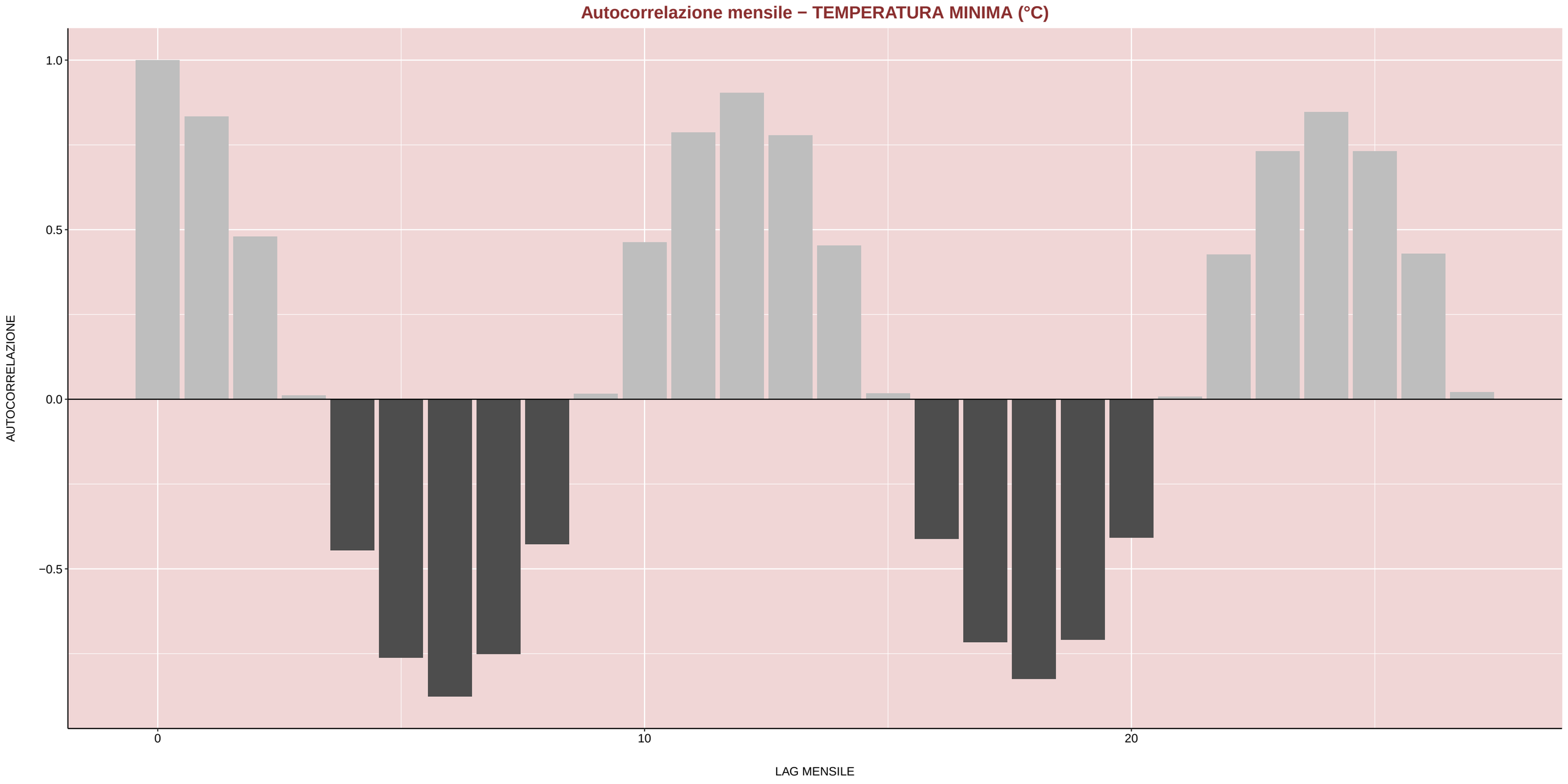

Trend polinomiale annuale grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
13
12
11
10
9
1973
1980
2000
2020
2024
ANNO

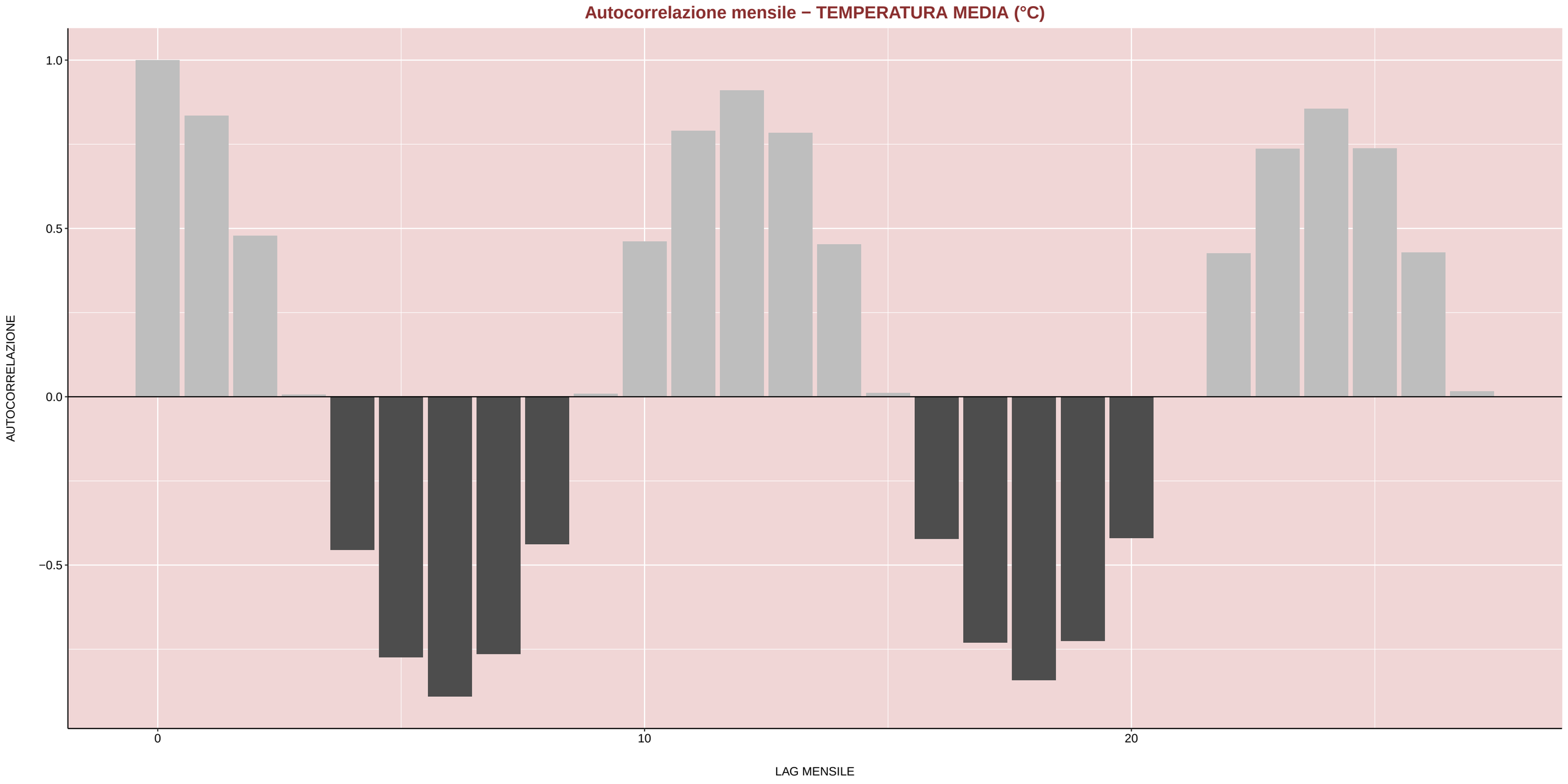

Serie annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
18
17
16
15
14
1973
1980
2000
2020
2024
ANNO

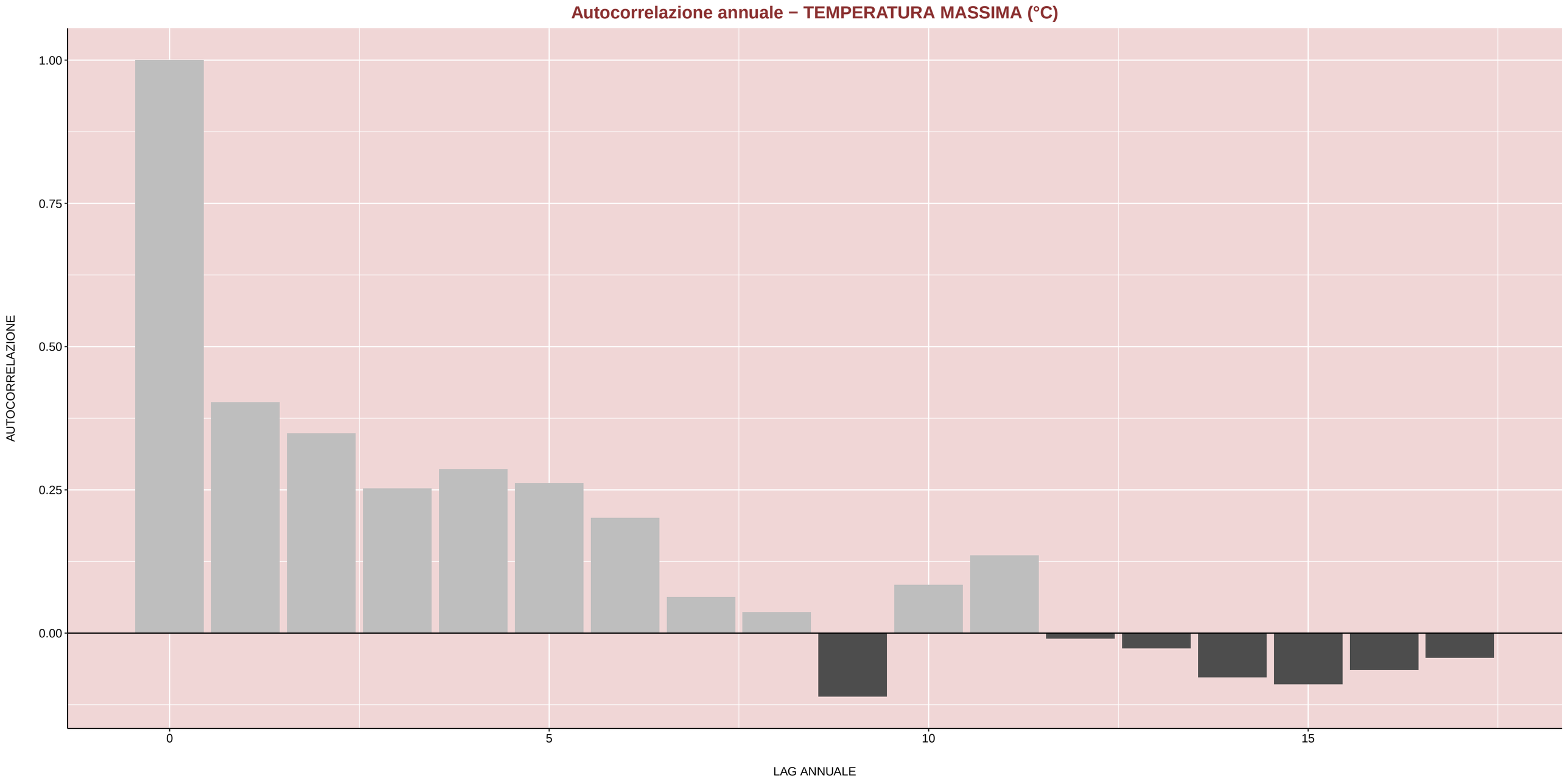

Trend lineare annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
18
17
16
15
14
1973
1980
2000
2020
2024
ANNO

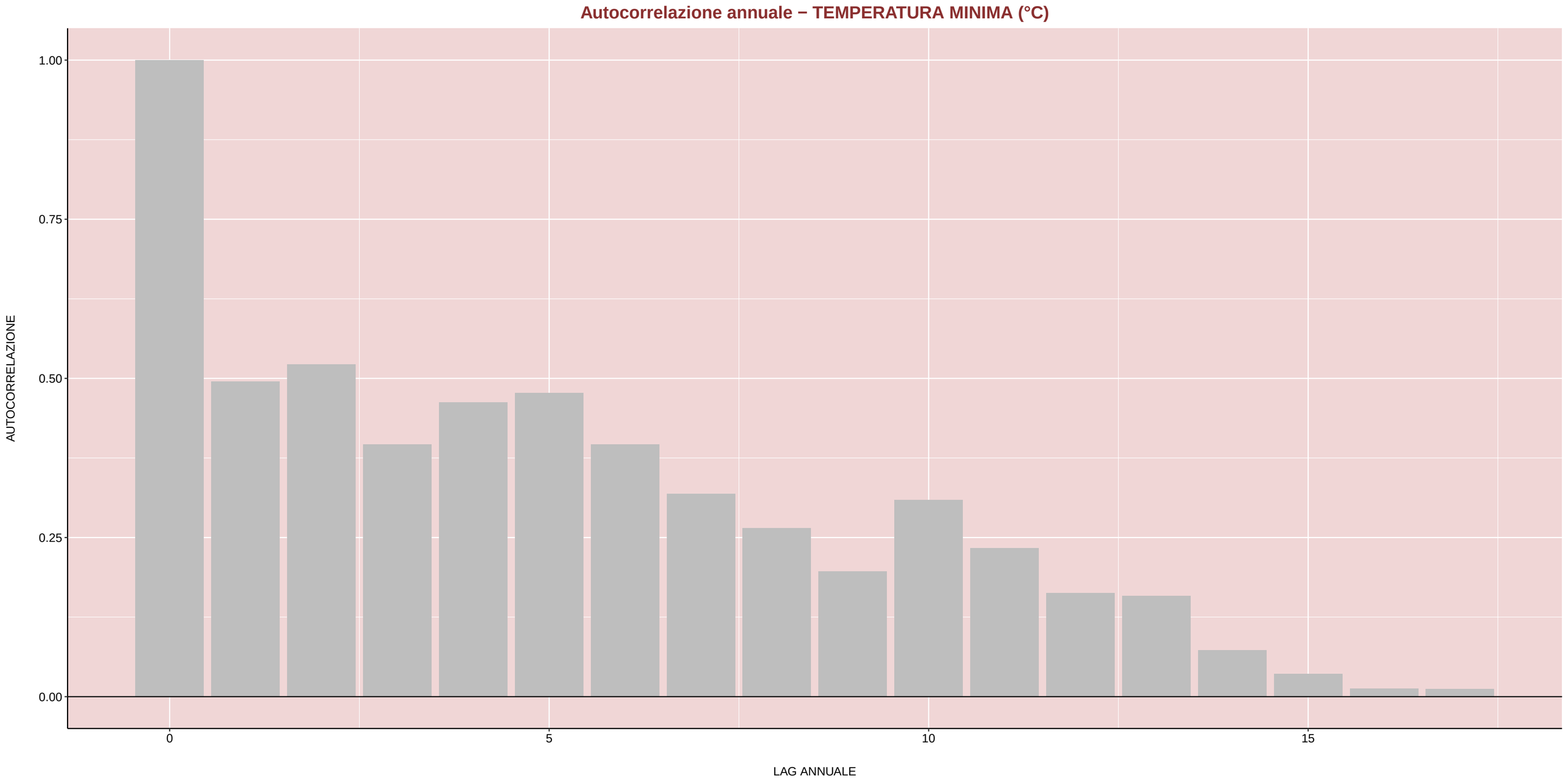

Trend polinomiale annuale grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
18
17
16
15
14
1973
1980
2000
2020
2024
ANNO

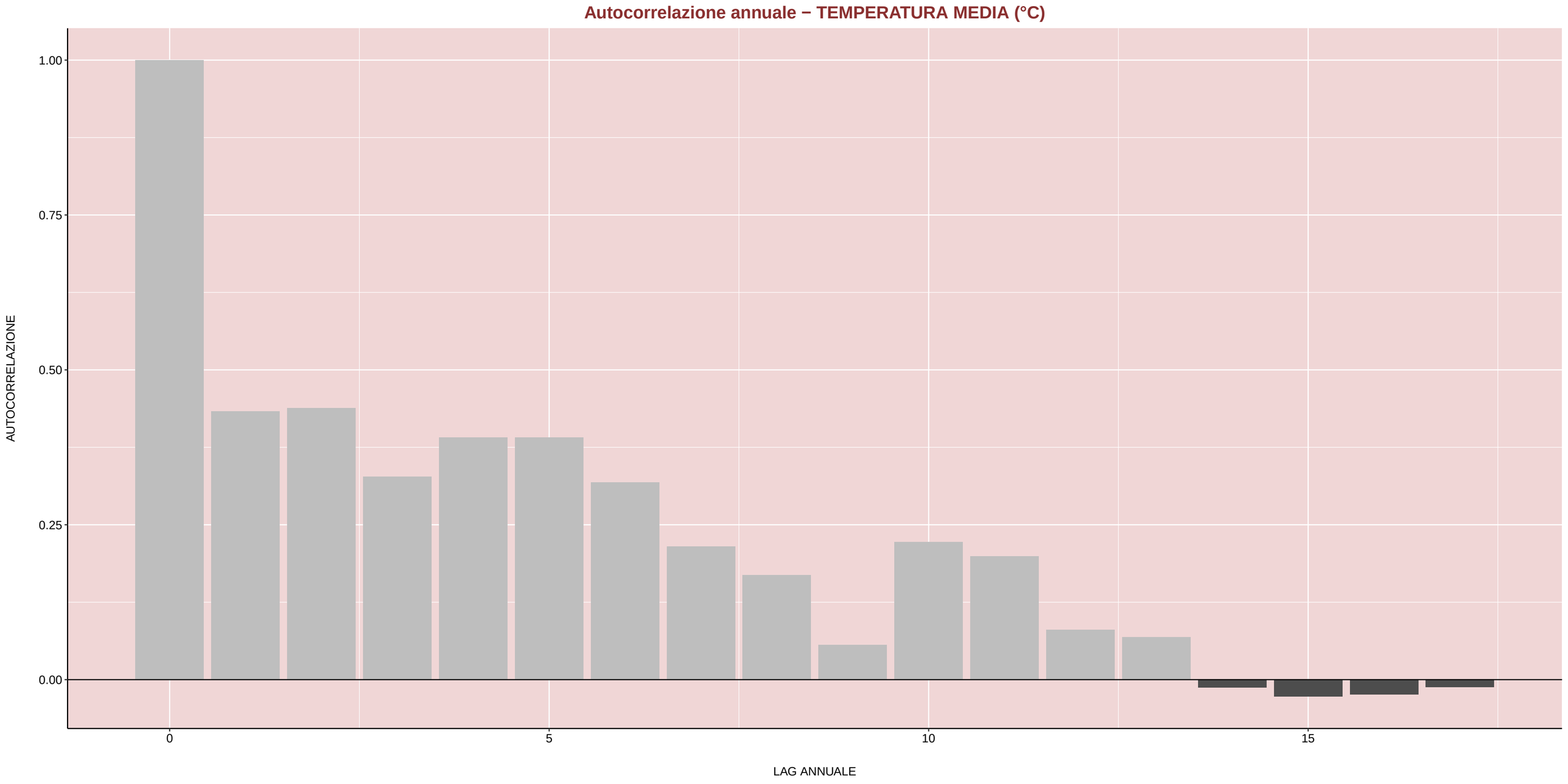

Autocorrelazione mensile − TEMPERATURA MASSIMA (°C)
1.0
0.5
0.0
−0.5
AUTOCORRELAZIONE
0
10
20
LAG MENSILE


Autocorrelazione mensile − TEMPERATURA MINIMA (°C)
1.0
0.5
0.0
−0.5
AUTOCORRELAZIONE
0
10
20
LAG MENSILE

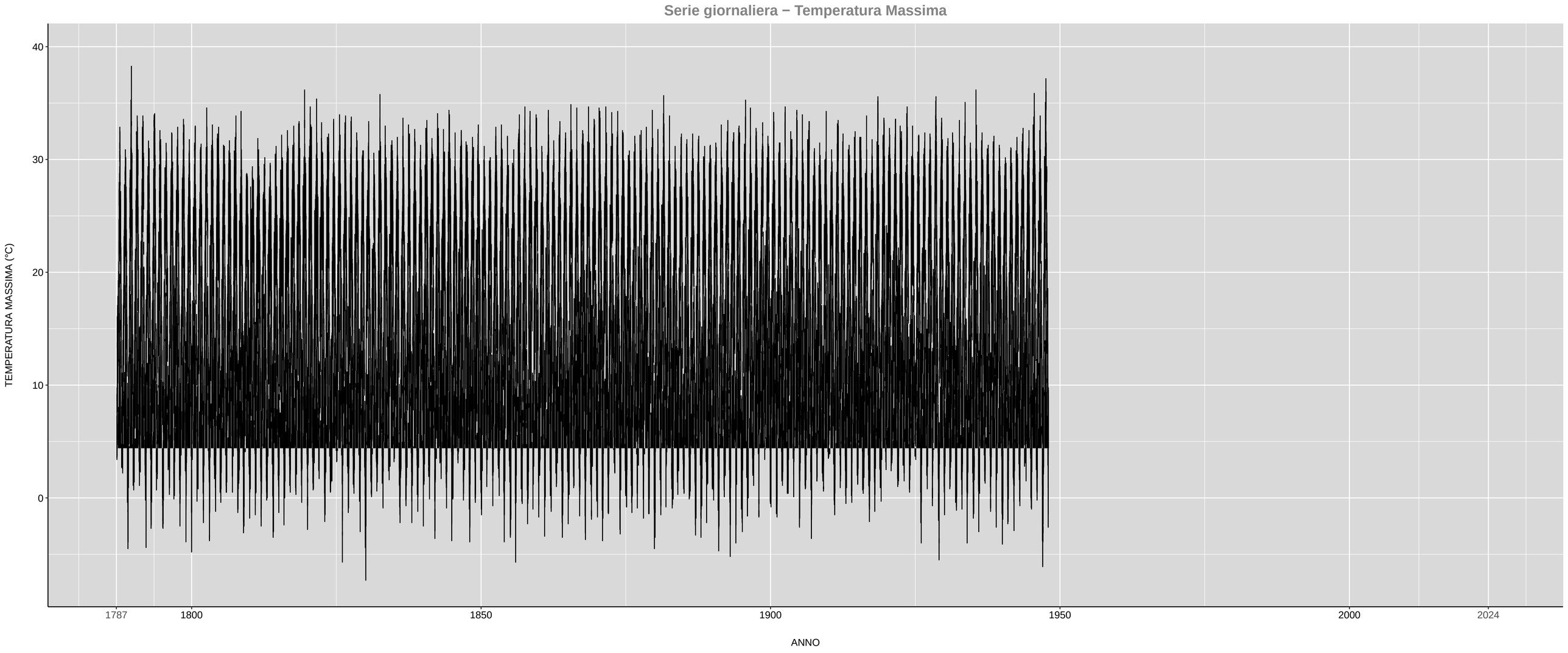

Autocorrelazione mensile − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
0
10
20
LAG MENSILE

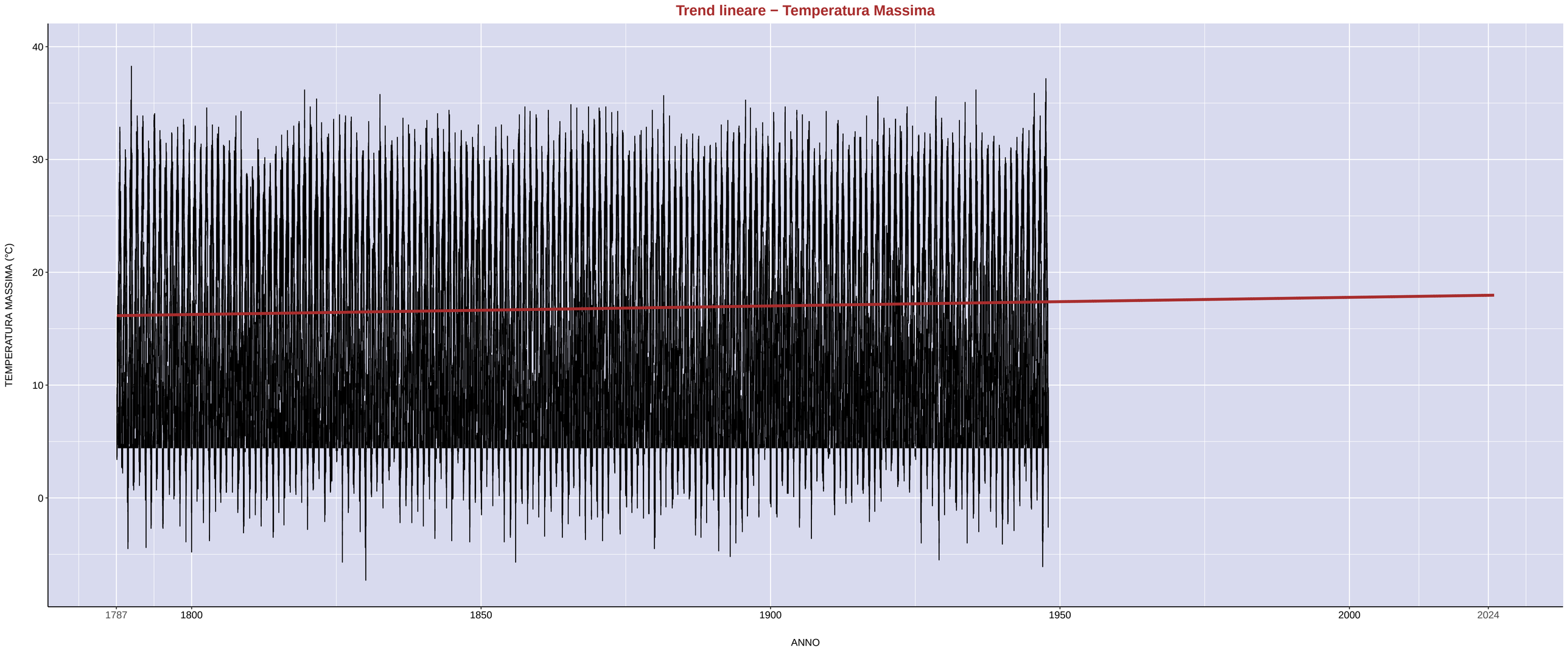

Autocorrelazione annuale − TEMPERATURA MASSIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
LAG ANNUALE

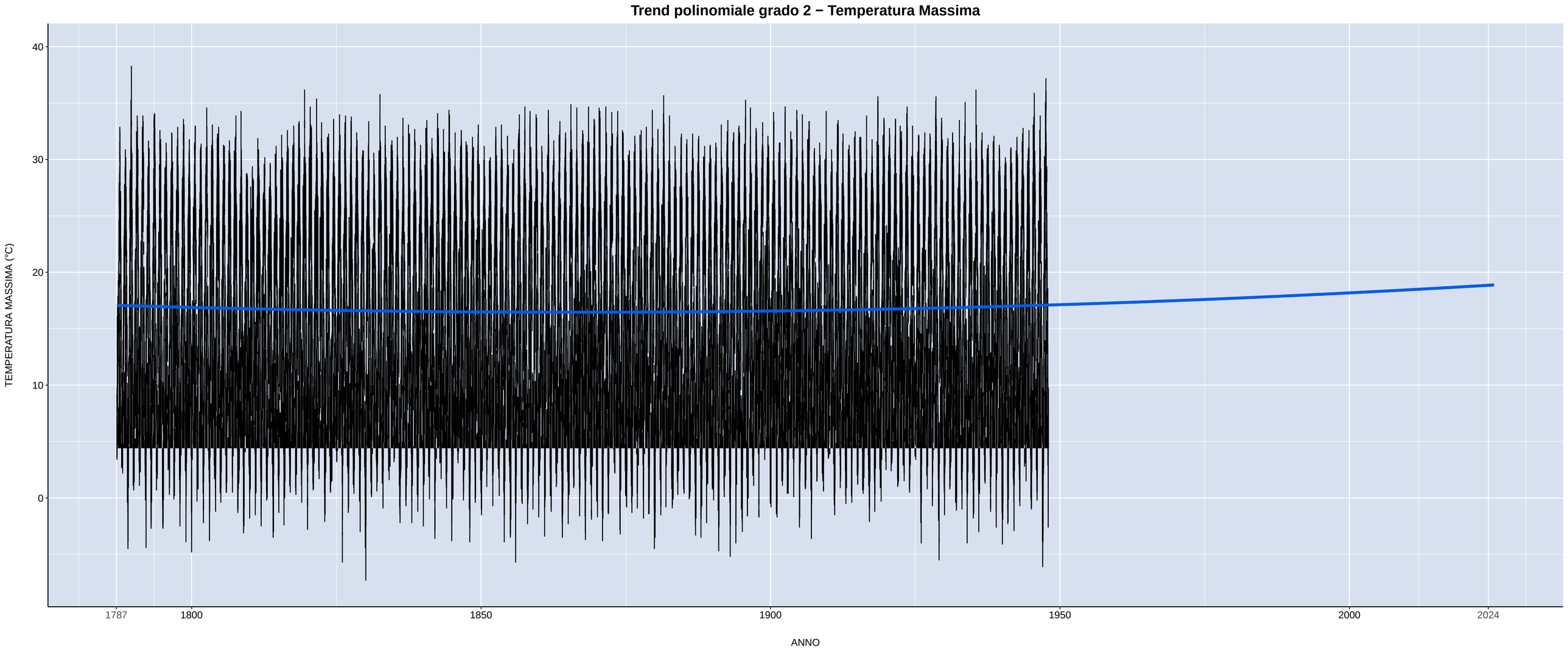

Autocorrelazione annuale − TEMPERATURA MINIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
LAG ANNUALE

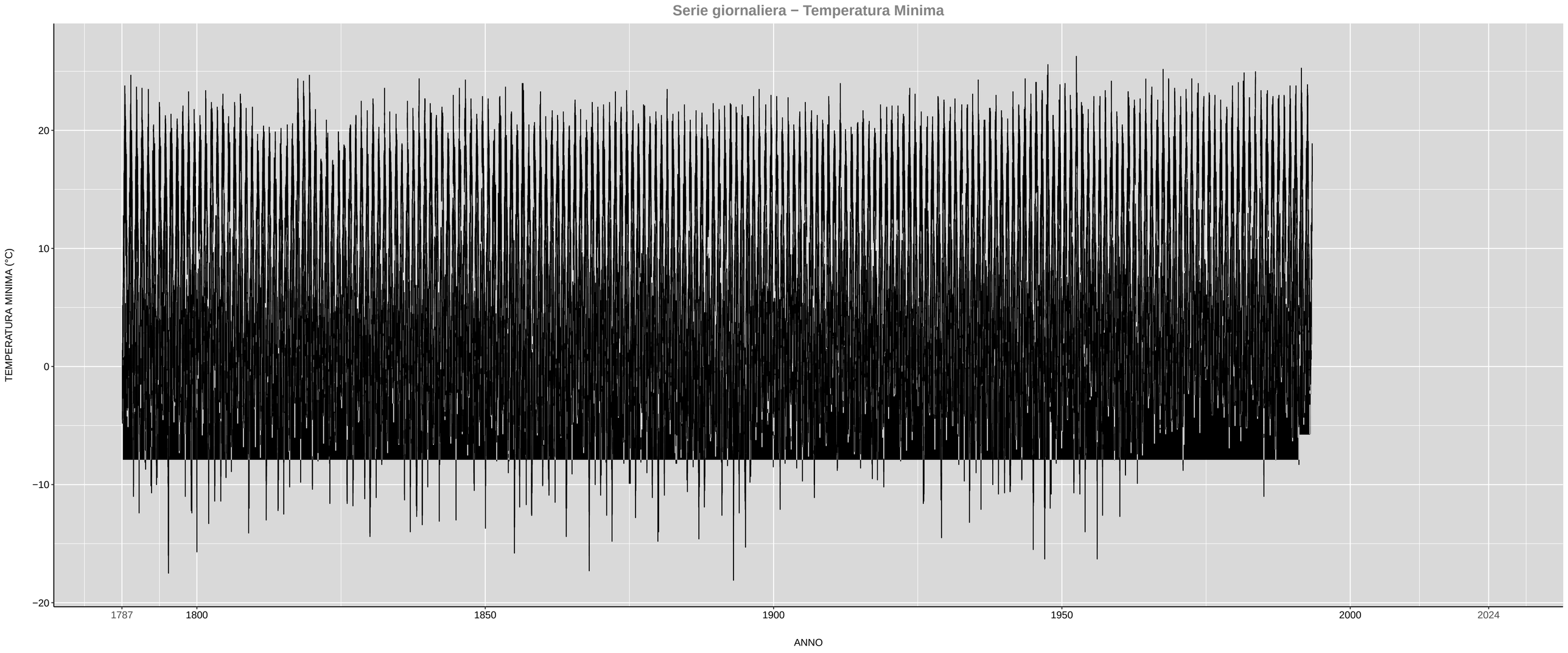

Autocorrelazione annuale − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
LAG ANNUALE

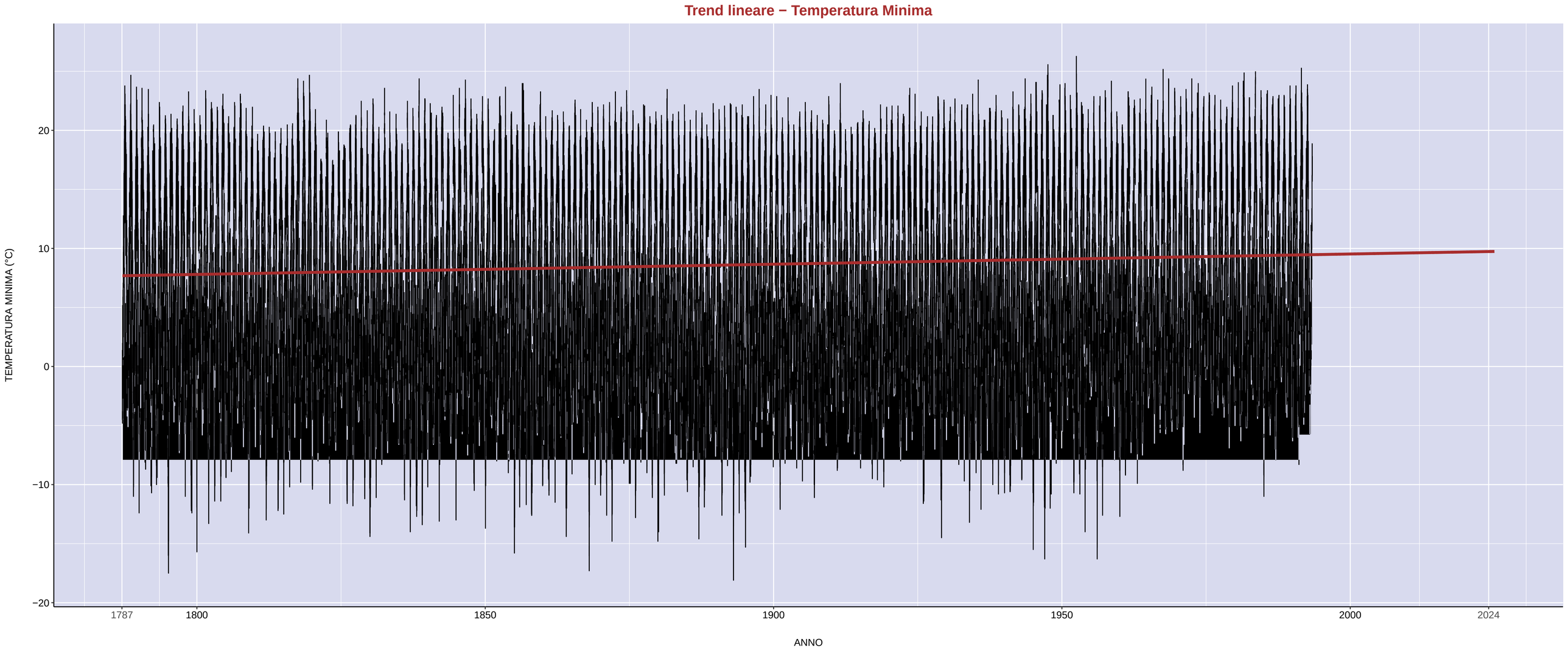
Torino – Stazione di Torino Giardini Reali

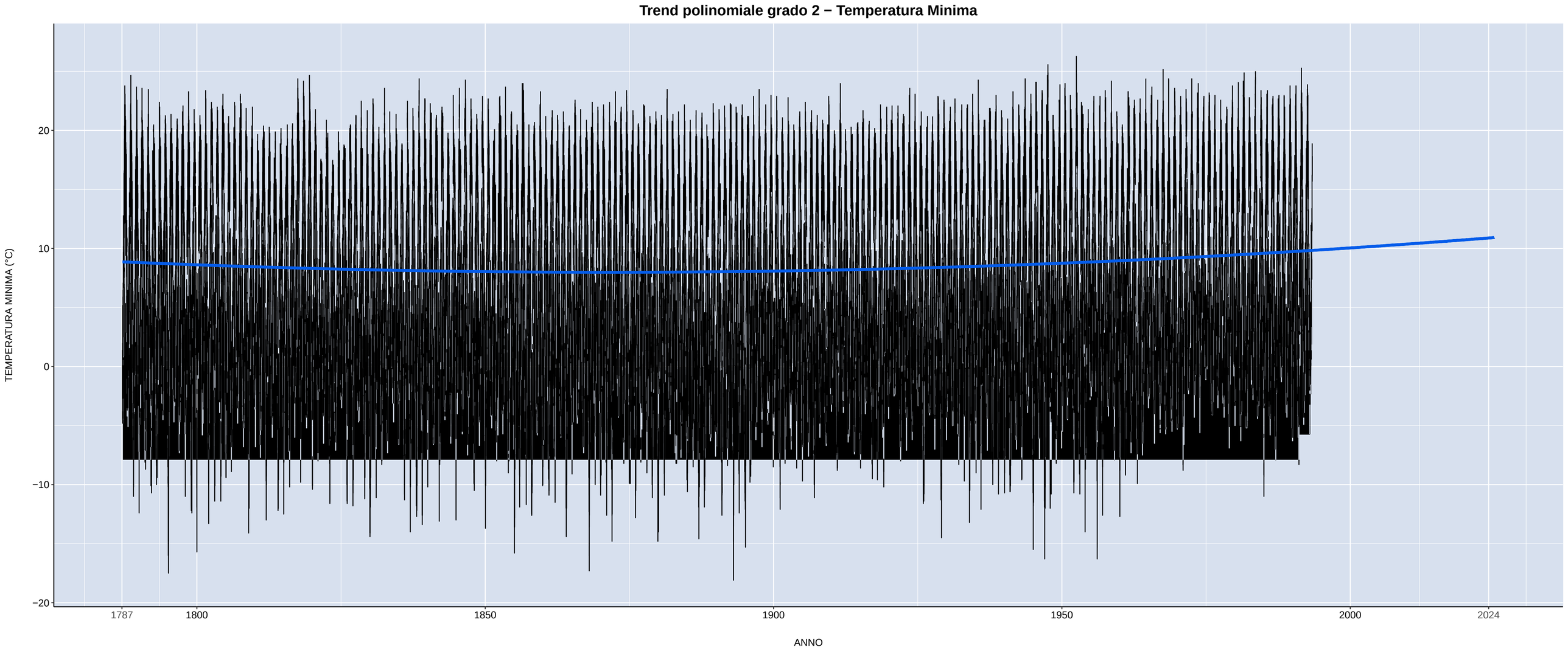

Serie giornaliera − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

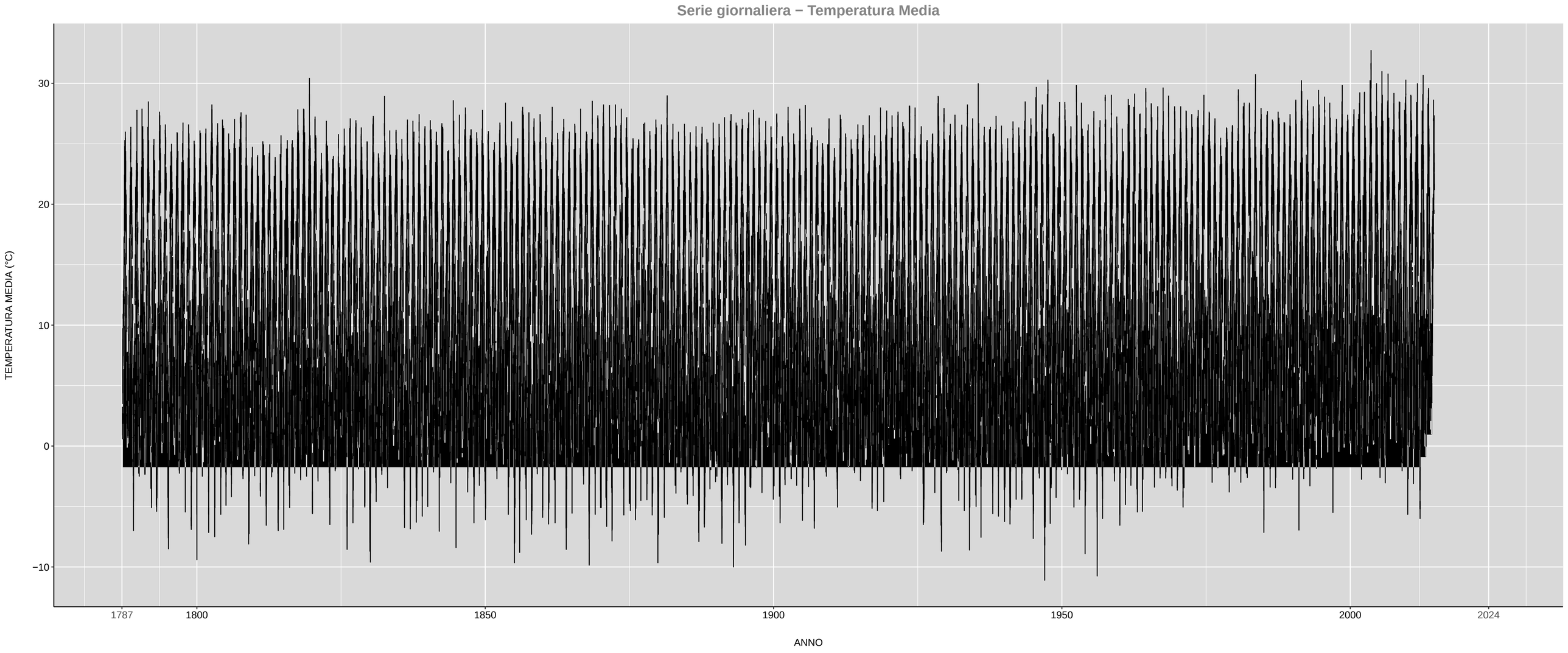

Trend lineare − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

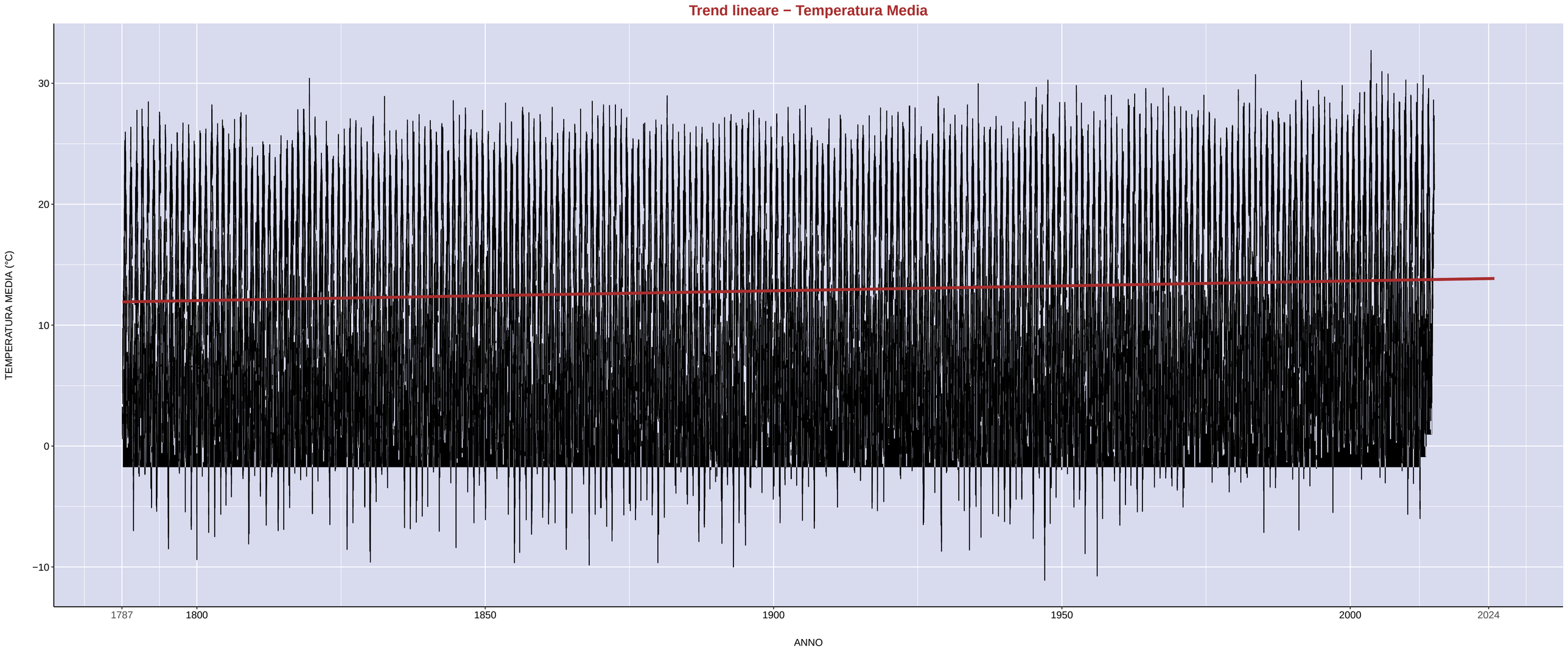

Trend polinomiale grado 2 − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

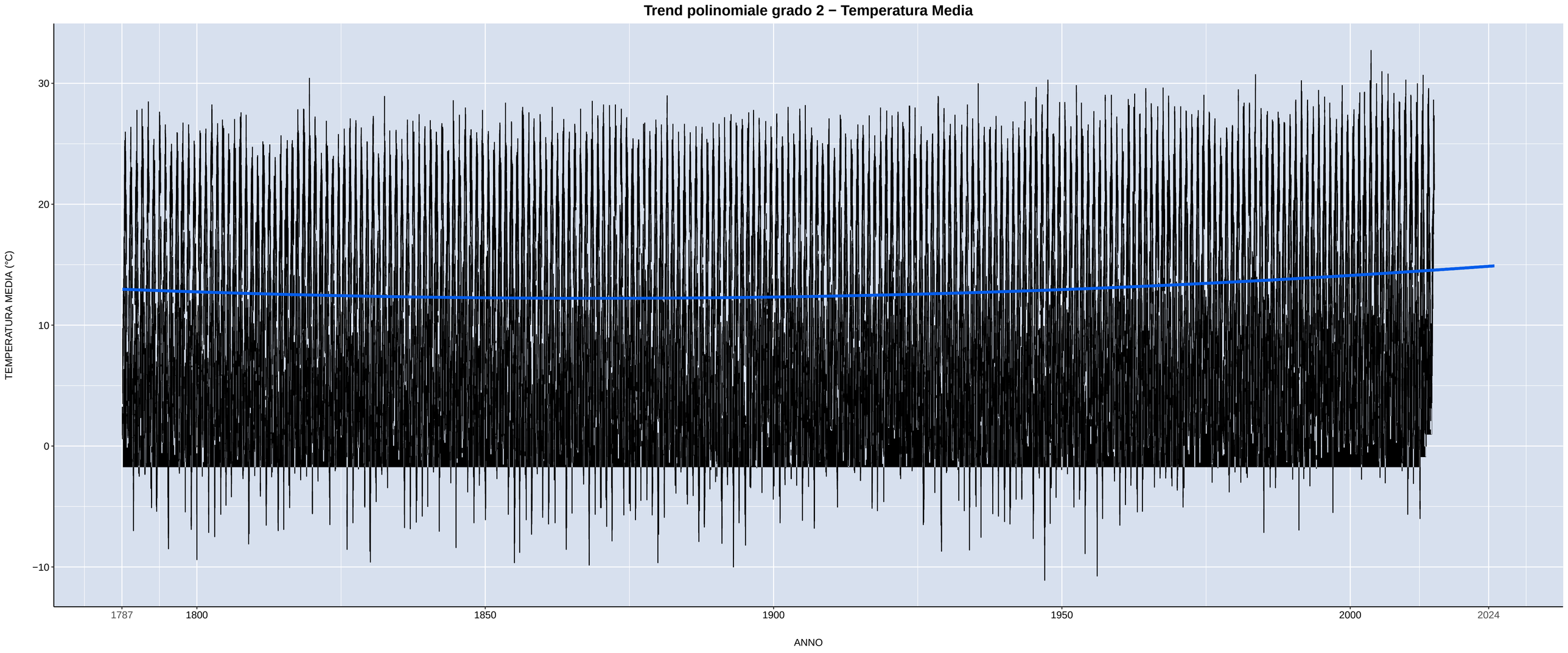

Serie giornaliera − Temperatura Minima
TEMPERATURA MINIMA (°C)
20
10
0
−10
−20
1787
1800
1850
1900
1950
2000
2024
ANNO

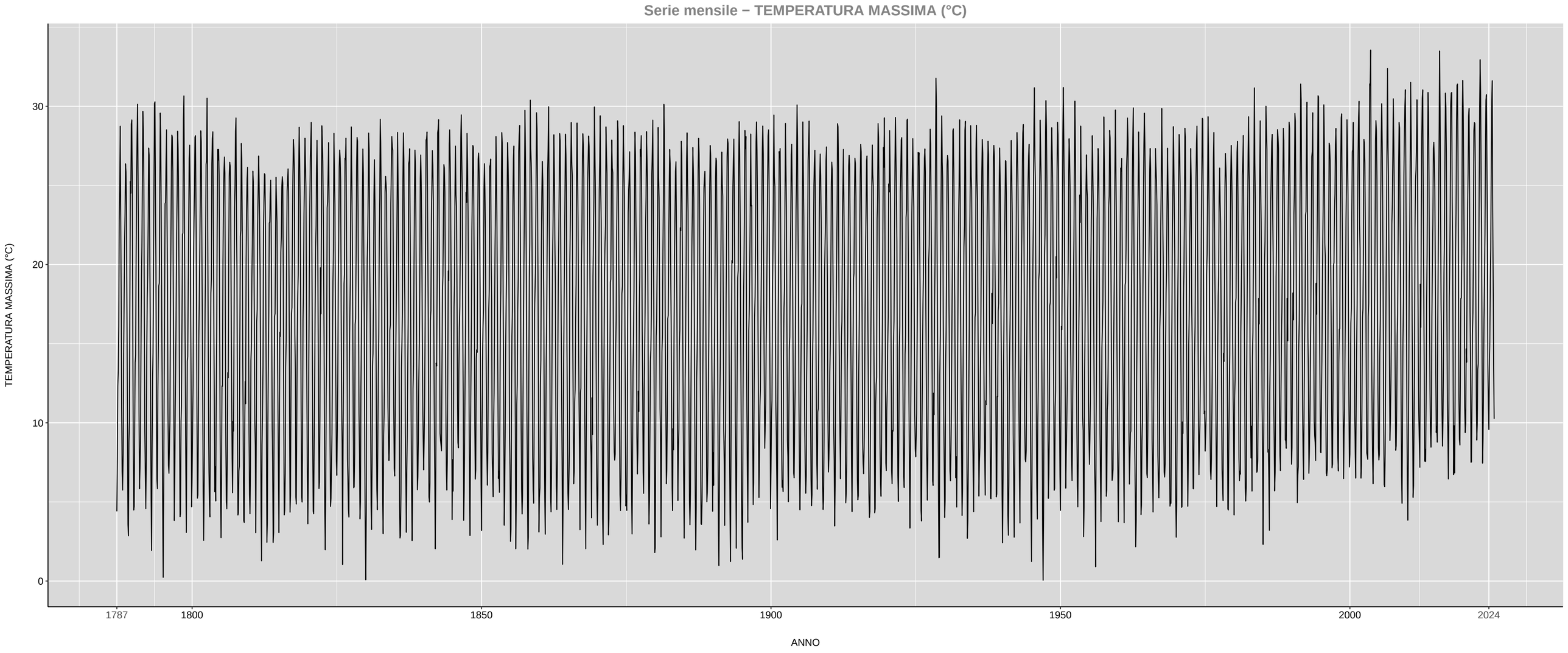

Trend lineare − Temperatura Minima
TEMPERATURA MINIMA (°C)
20
10
0
−10
−20
1787
1800
1850
1900
1950
2000
2024
ANNO

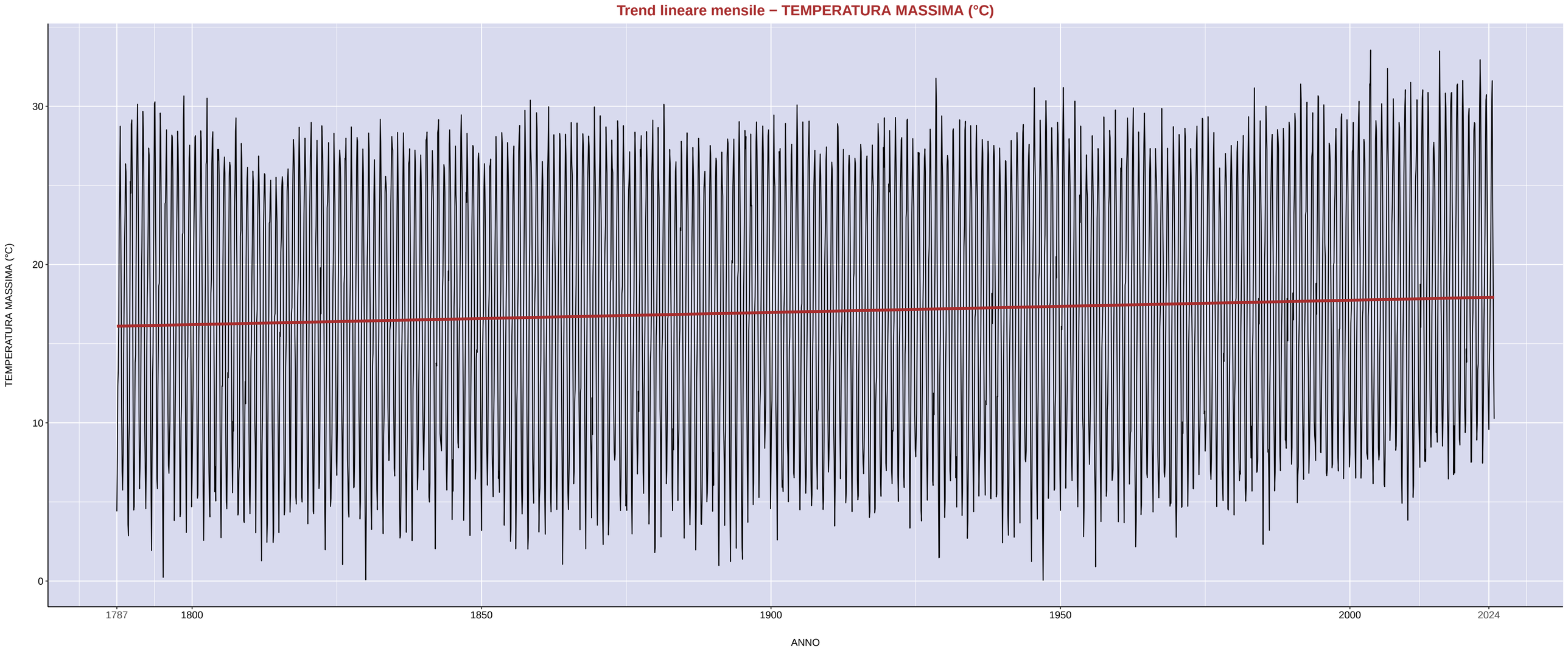

Trend polinomiale grado 2 − Temperatura Minima
TEMPERATURA MINIMA (°C)
20
10
0
−10
−20
1787
1800
1850
1900
1950
2000
2024
ANNO

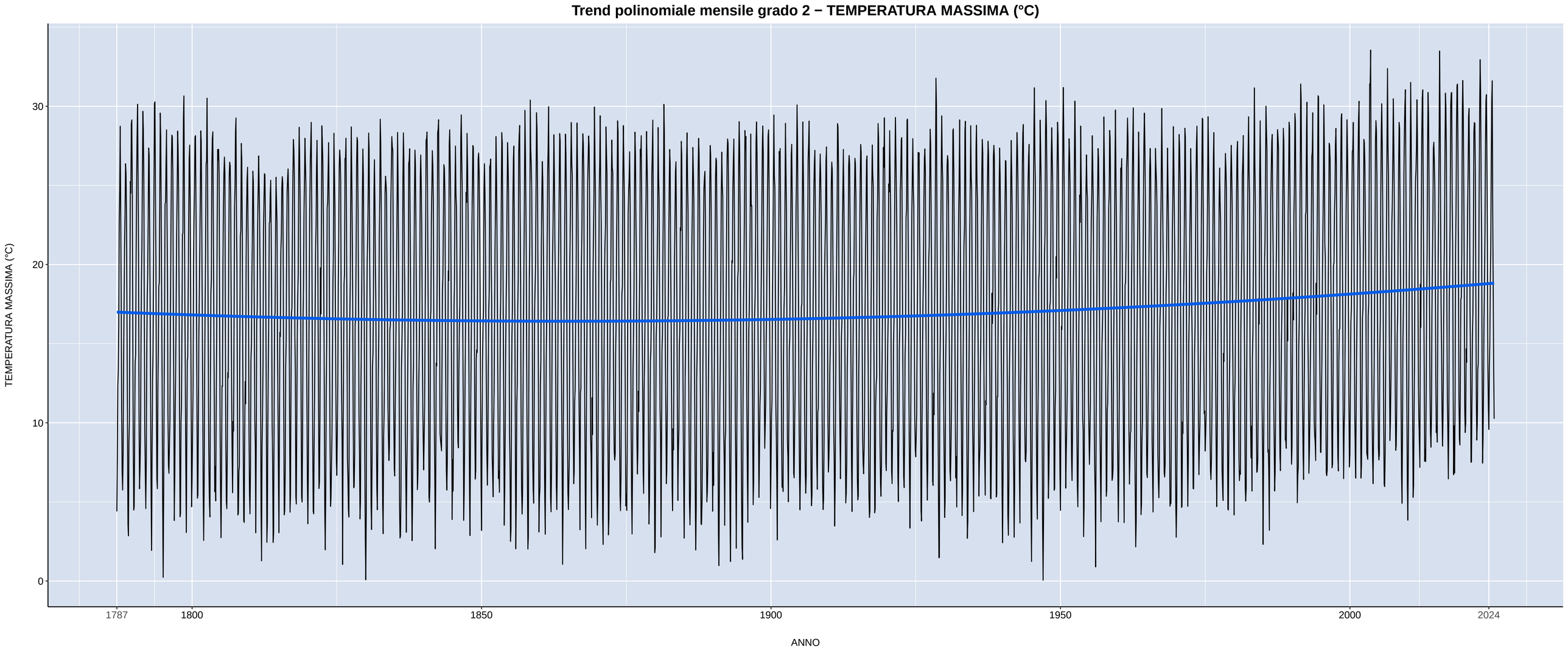

Serie giornaliera − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
−10
1787
1800
1850
1900
1950
2000
2024
ANNO

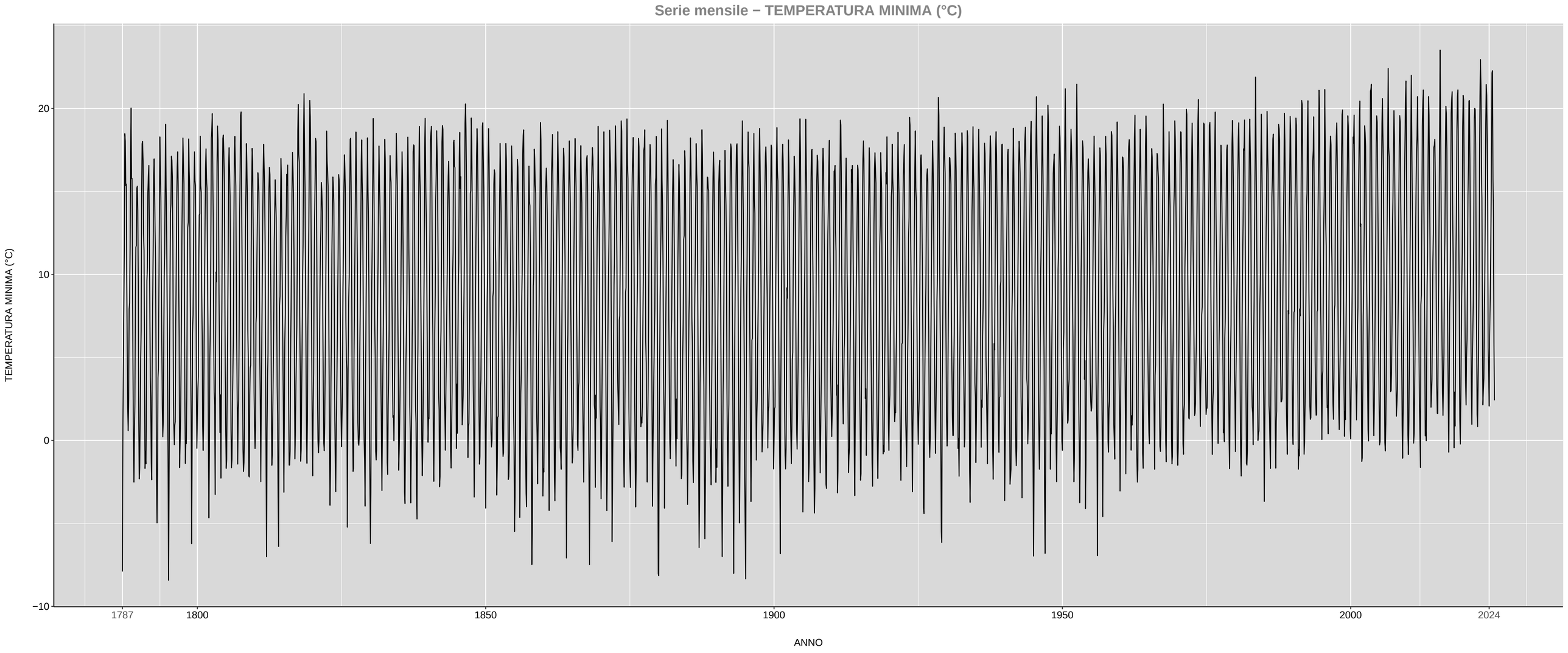

Trend lineare − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
−10
1787
1800
1850
1900
1950
2000
2024
ANNO

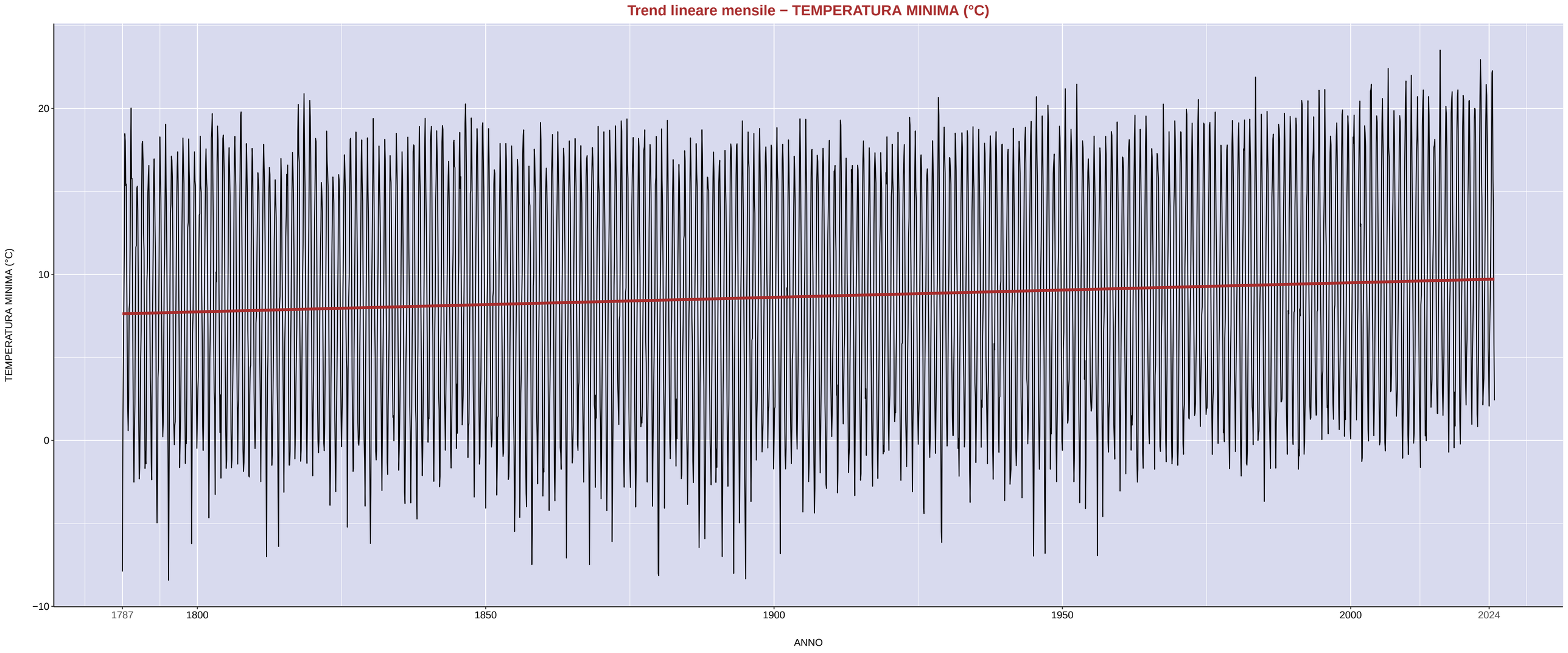

Trend polinomiale grado 2 − Temperatura Media
TEMPERATURA MEDIA (°C)
30
20
10
0
−10
1787
1800
1850
1900
1950
2000
2024
ANNO

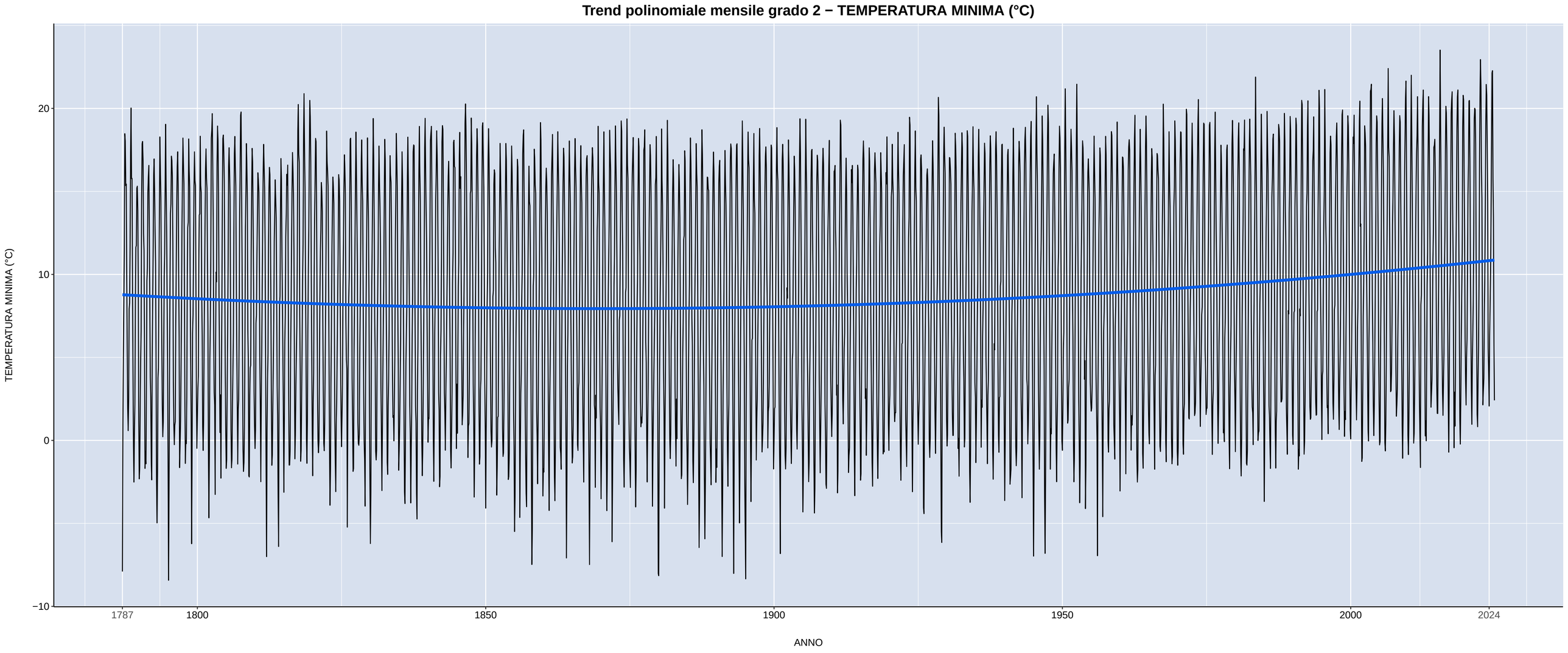

Serie mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

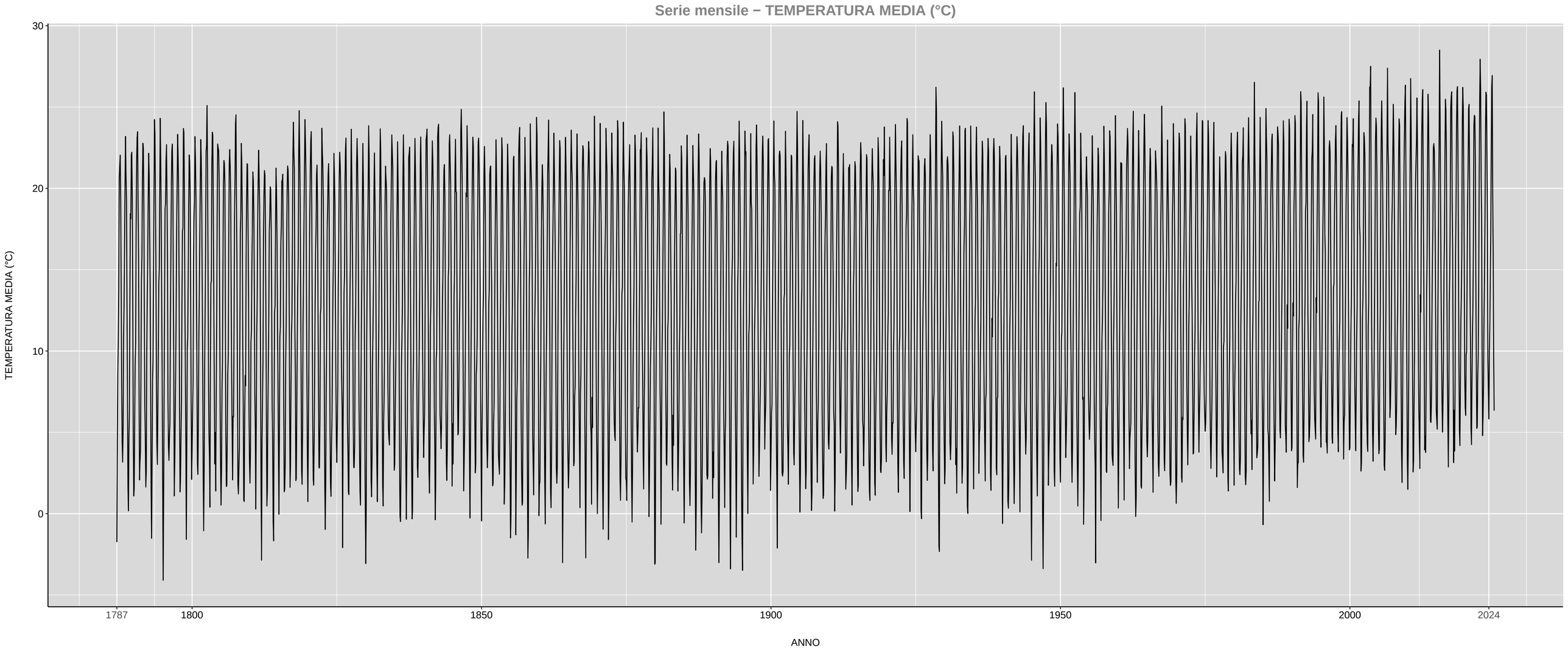

Trend lineare mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

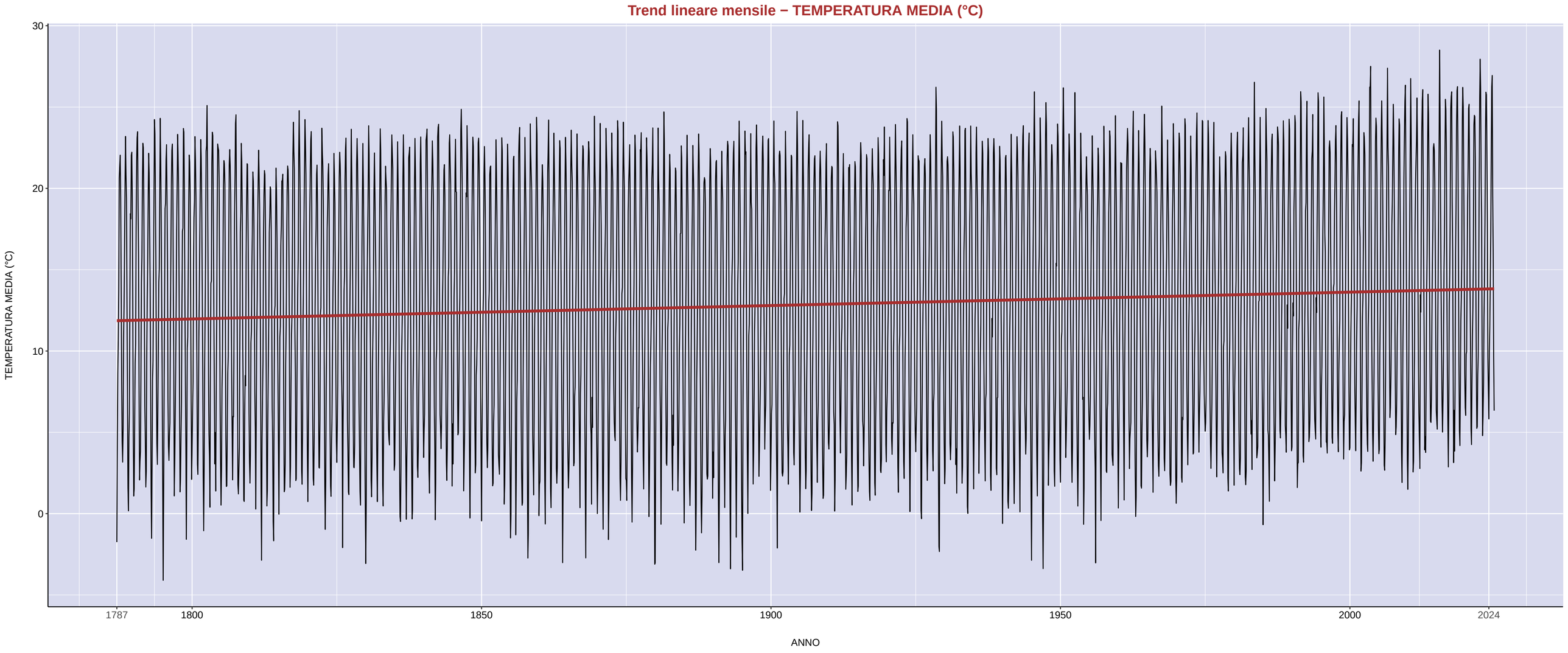

Trend polinomiale mensile grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

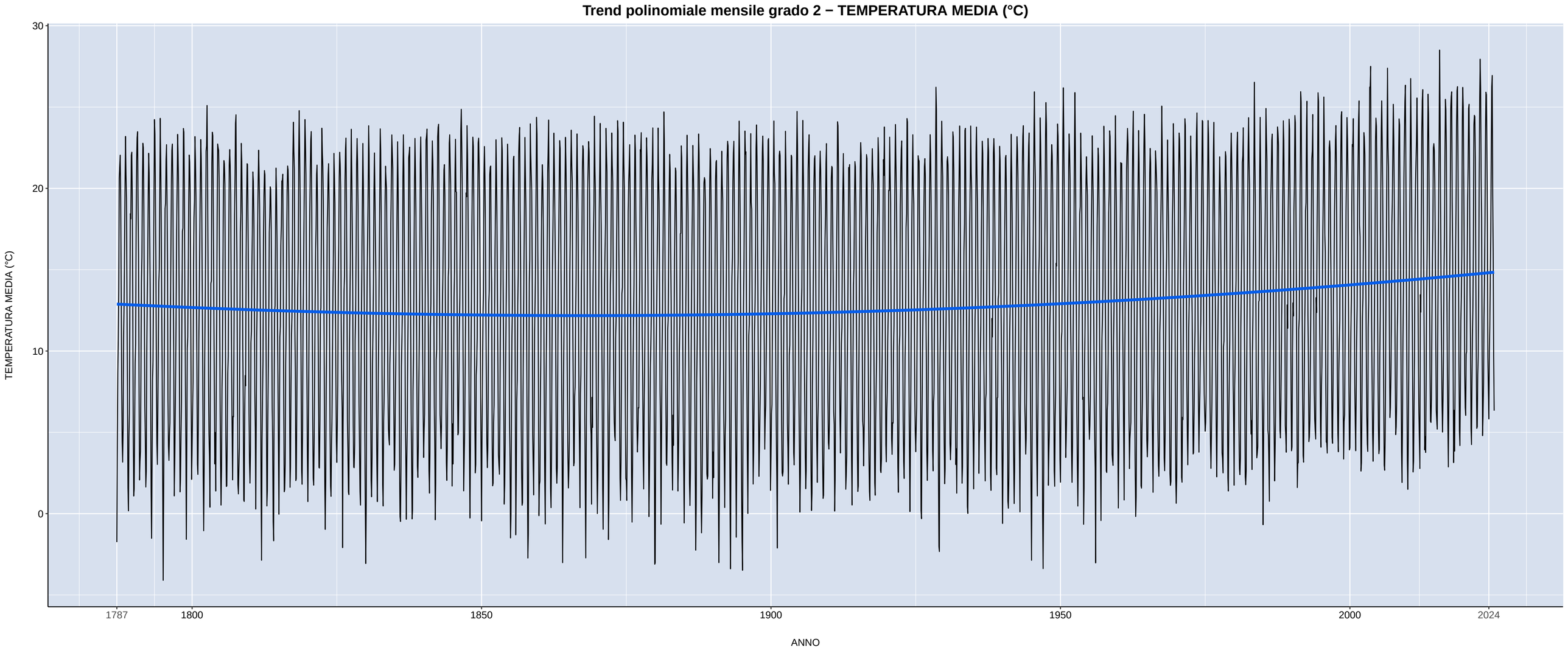

Serie mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
10
0
−10
1787
1800
1850
1900
1950
2000
2024
ANNO

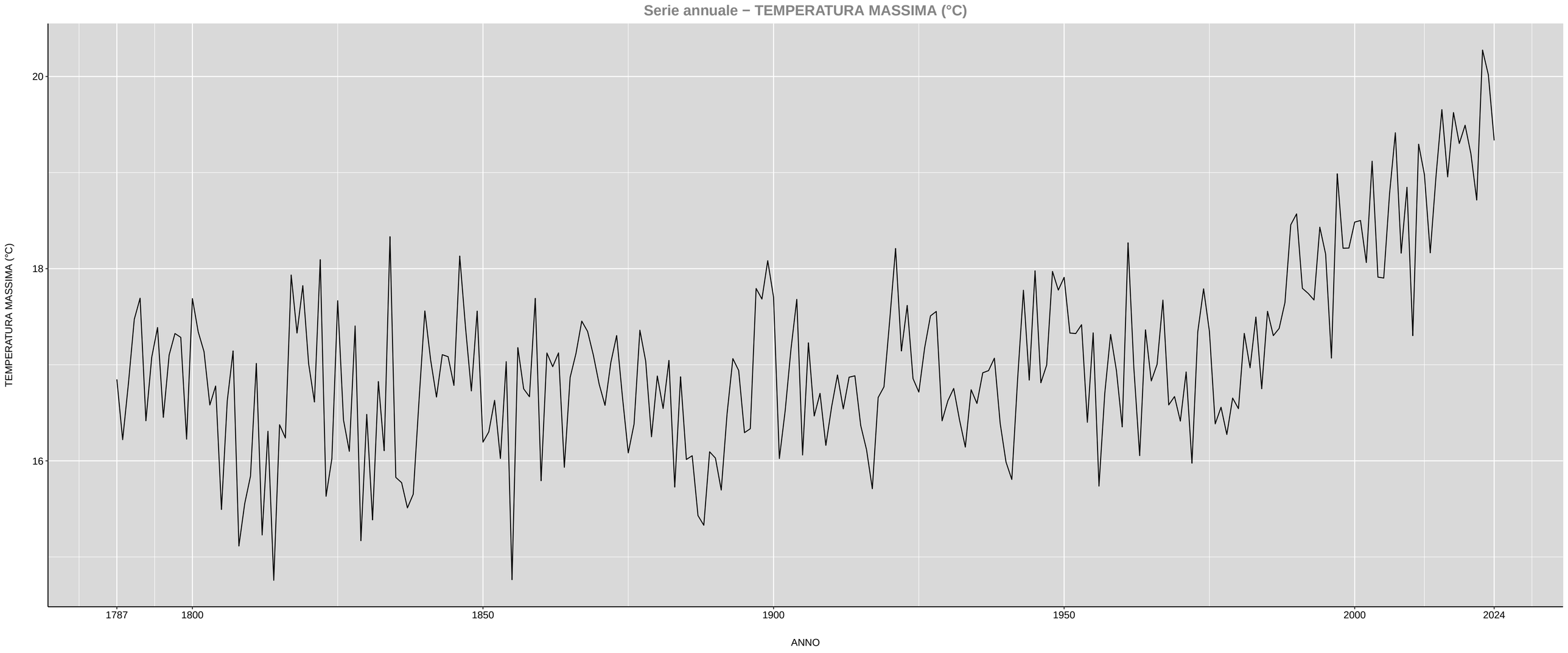

Trend lineare mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
10
0
−10
1787
1800
1850
1900
1950
2000
2024
ANNO

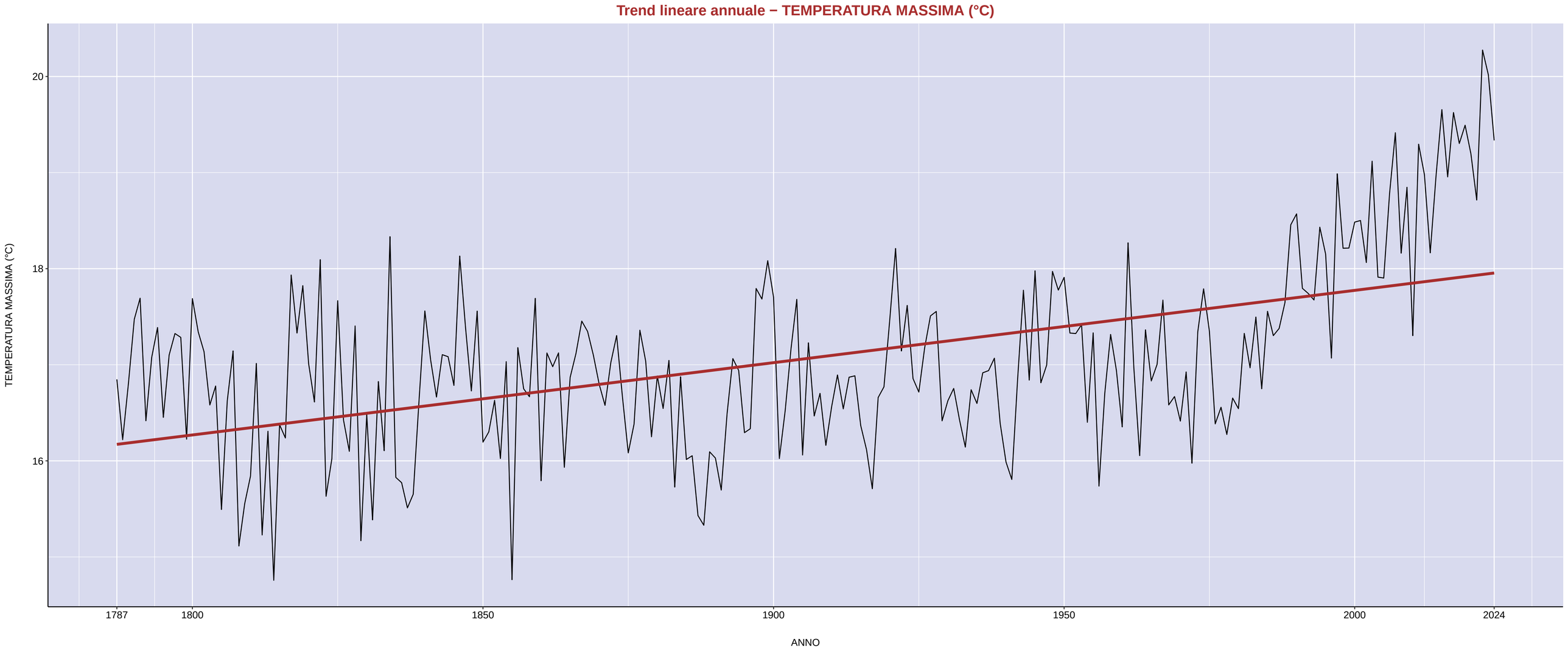

Trend polinomiale mensile grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
20
10
0
−10
1787
1800
1850
1900
1950
2000
2024
ANNO

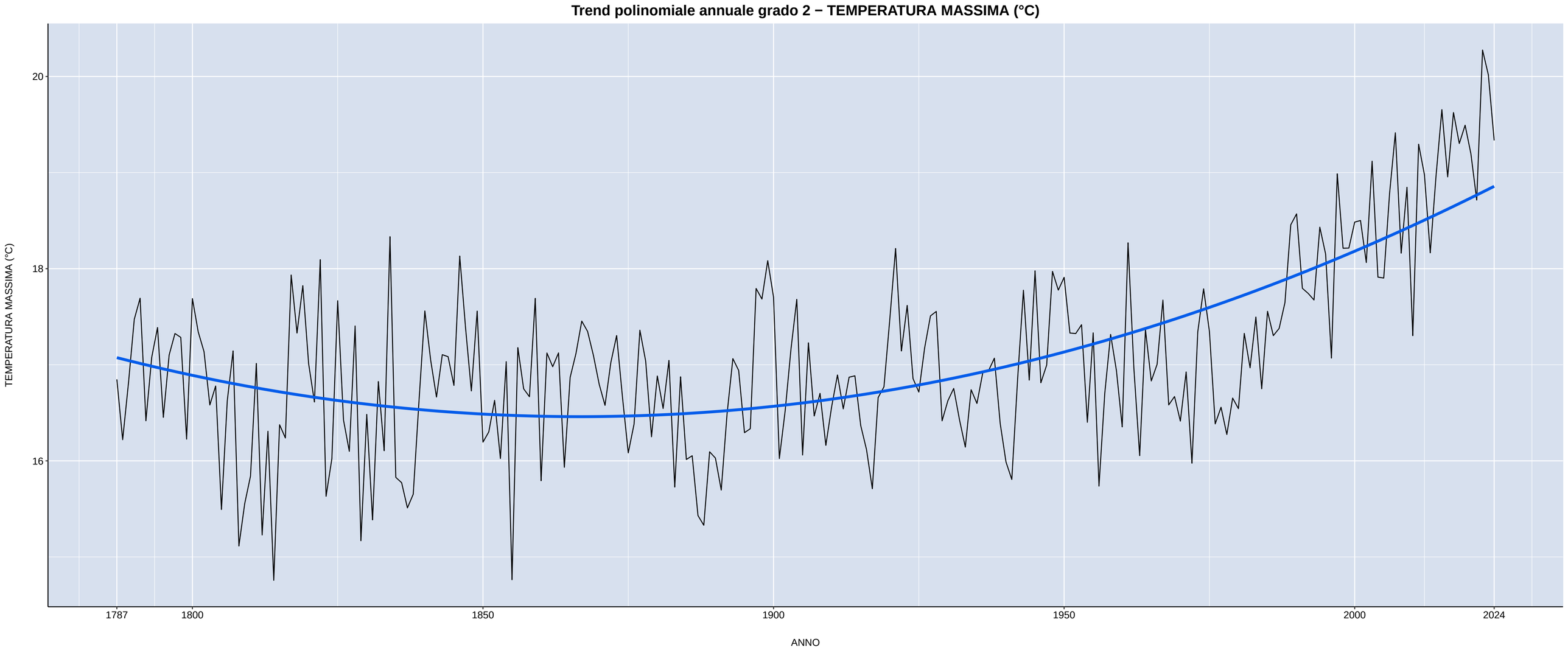

Serie mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

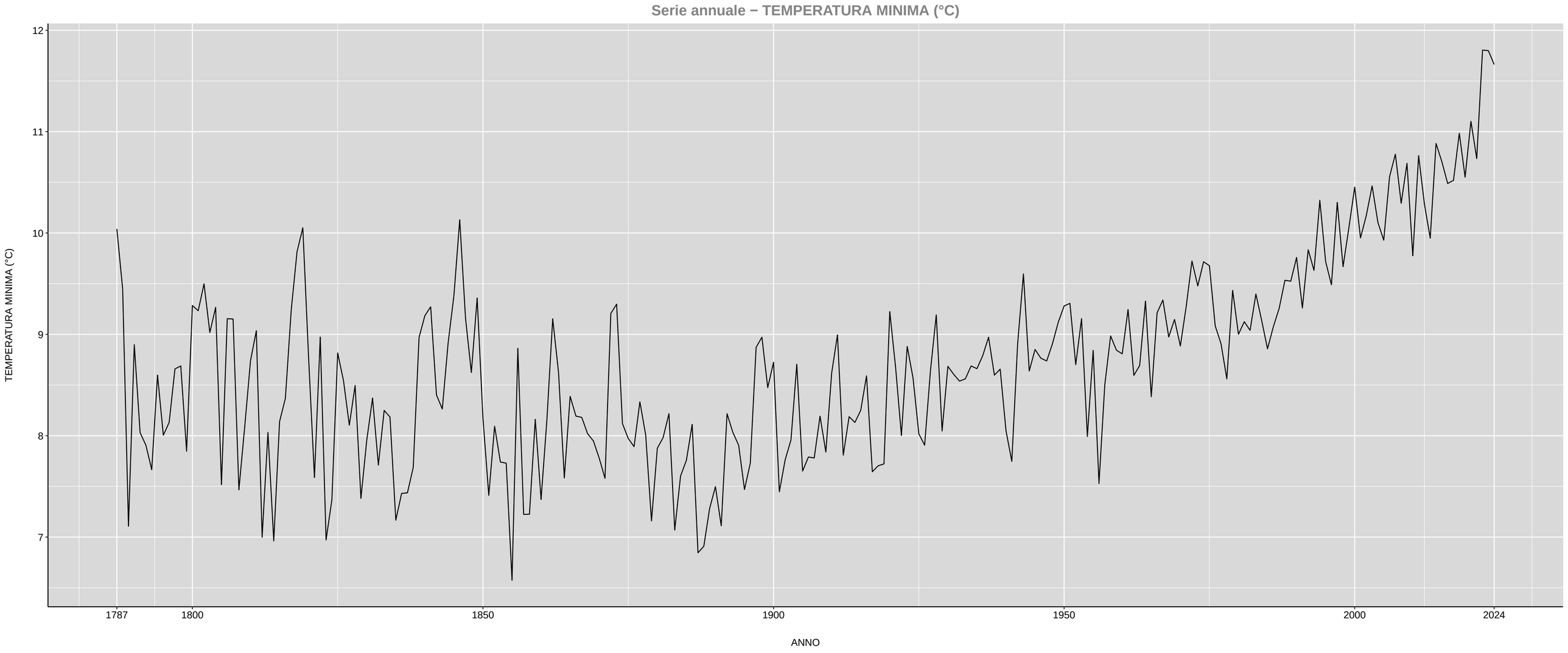

Trend lineare mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

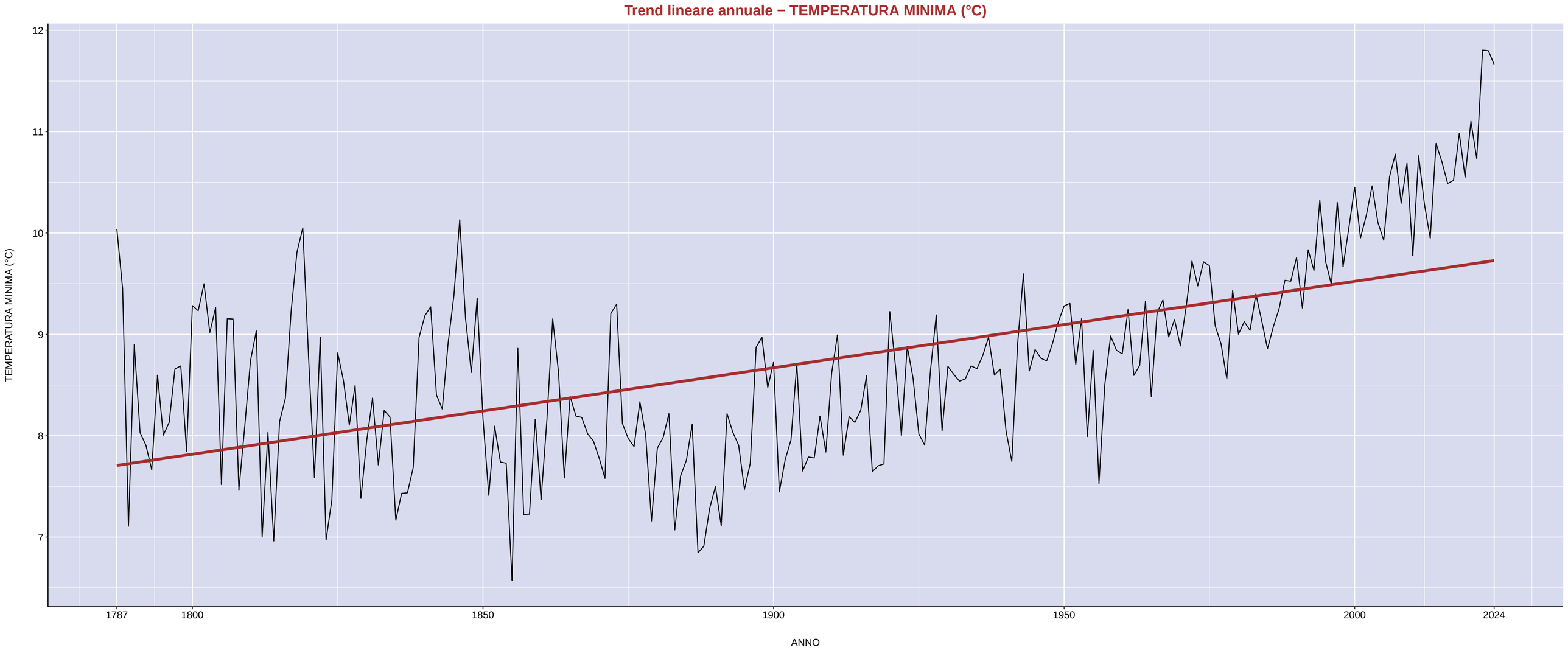

Trend polinomiale mensile grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
20
10
0
1787
1800
1850
1900
1950
2000
2024
ANNO

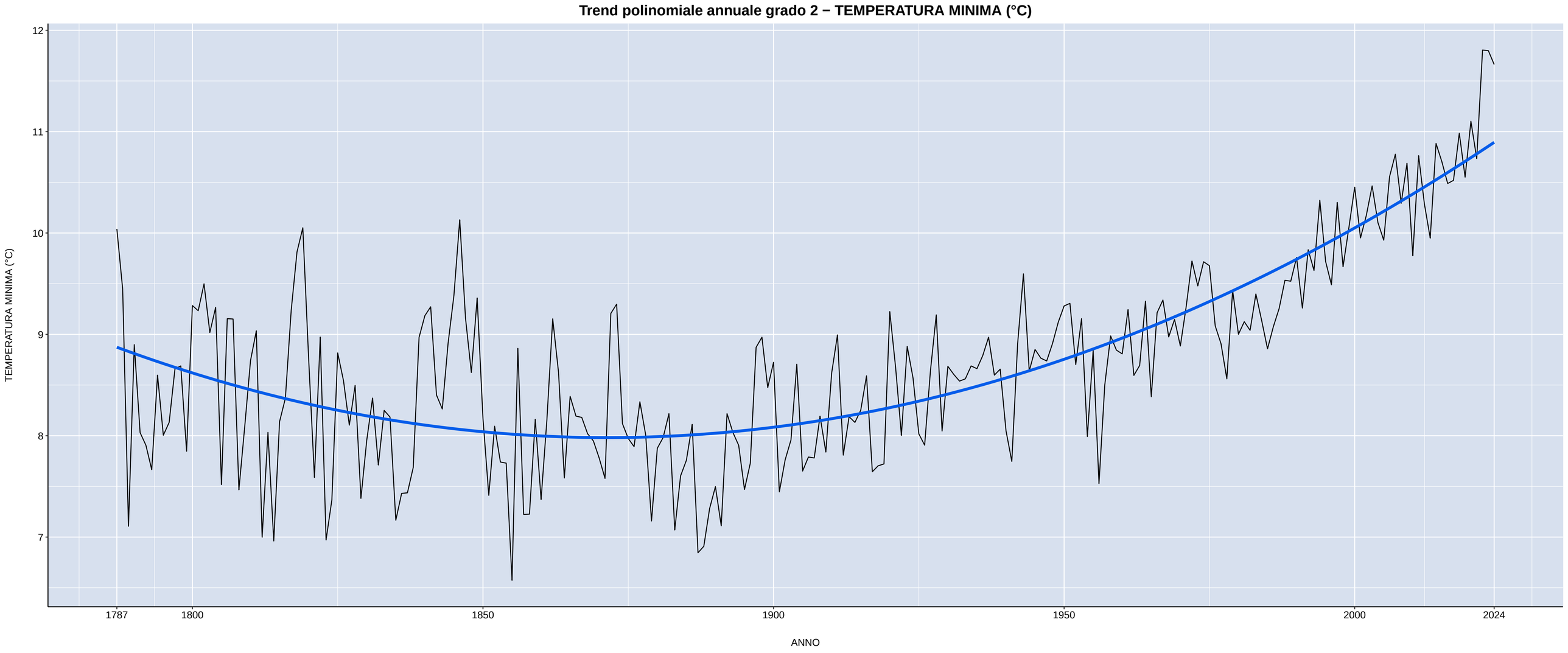

Serie annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
20
18
16
1787
1800
1850
1900
1950
2000
2024
ANNO

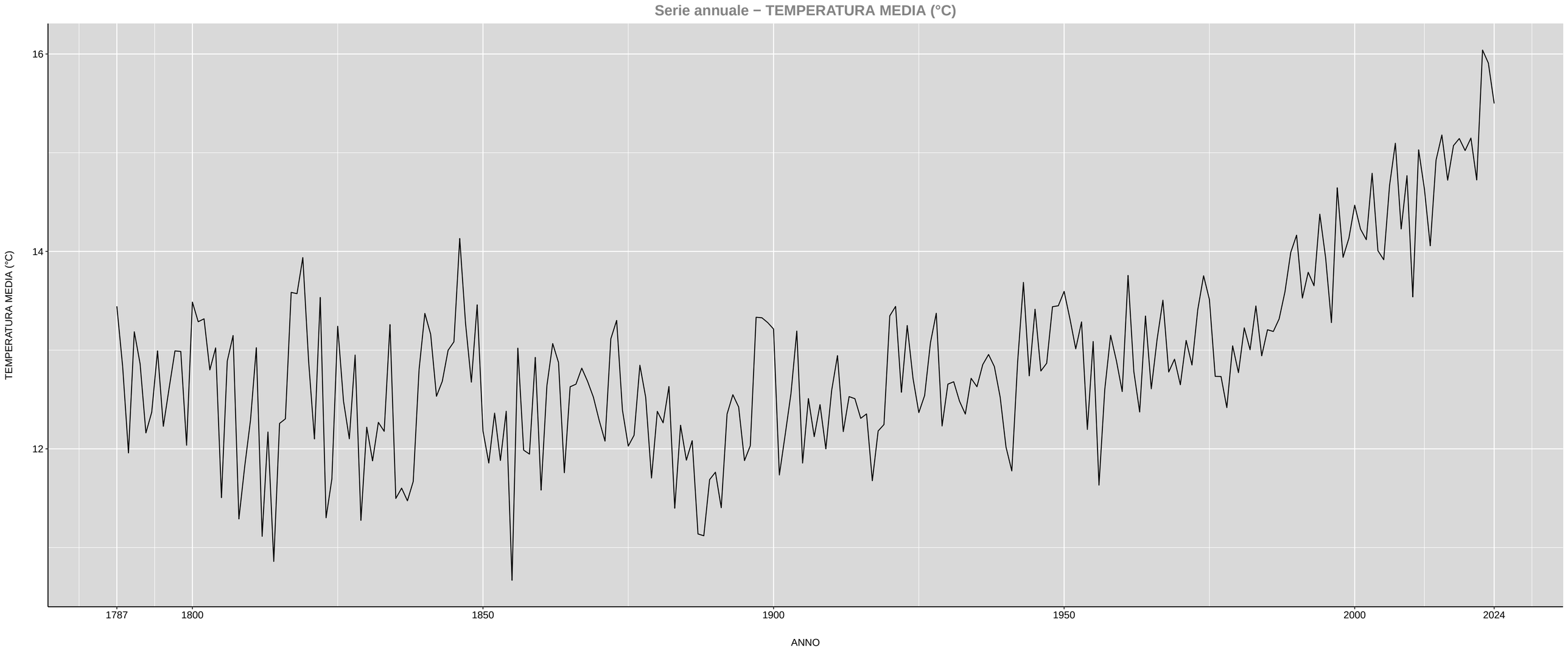

Trend lineare annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
20
18
16
1787
1800
1850
1900
1950
2000
2024
ANNO

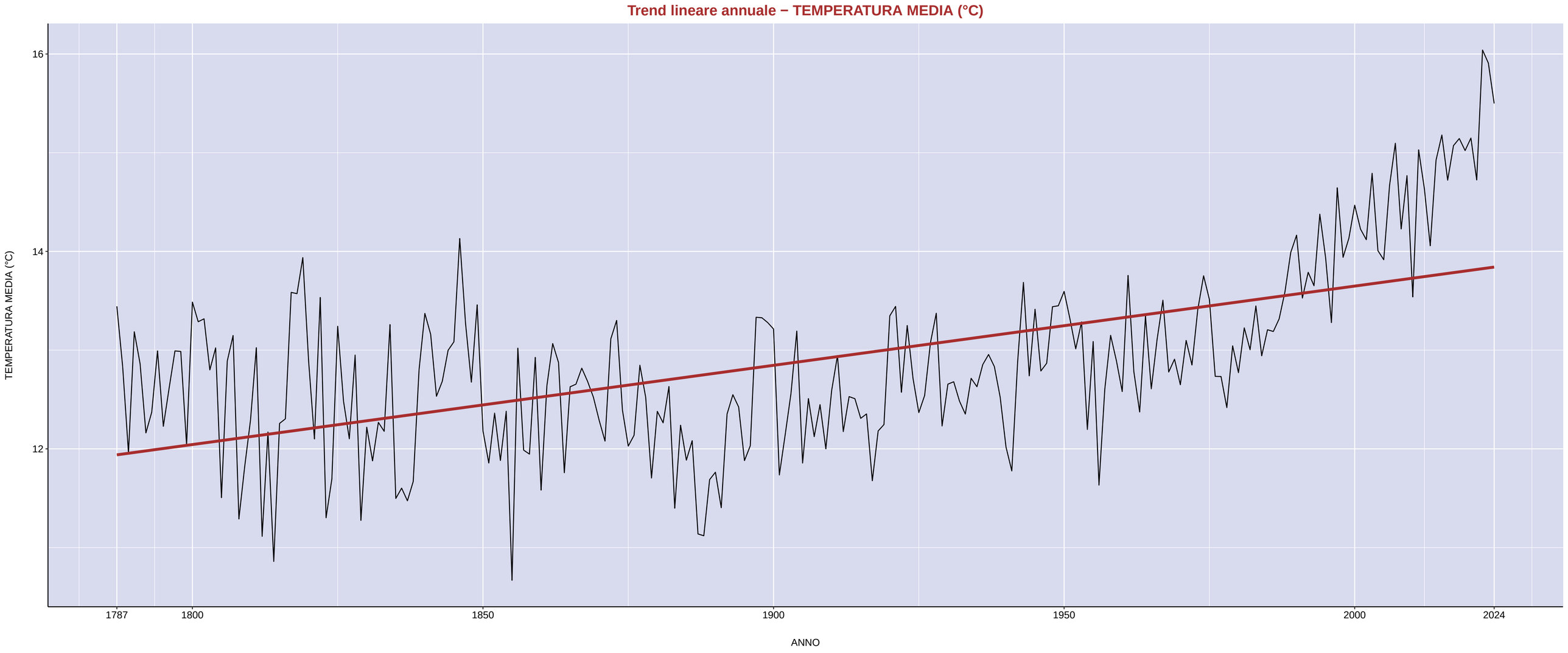

Trend polinomiale annuale grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
20
18
16
1787
1800
1850
1900
1950
2000
2024
ANNO

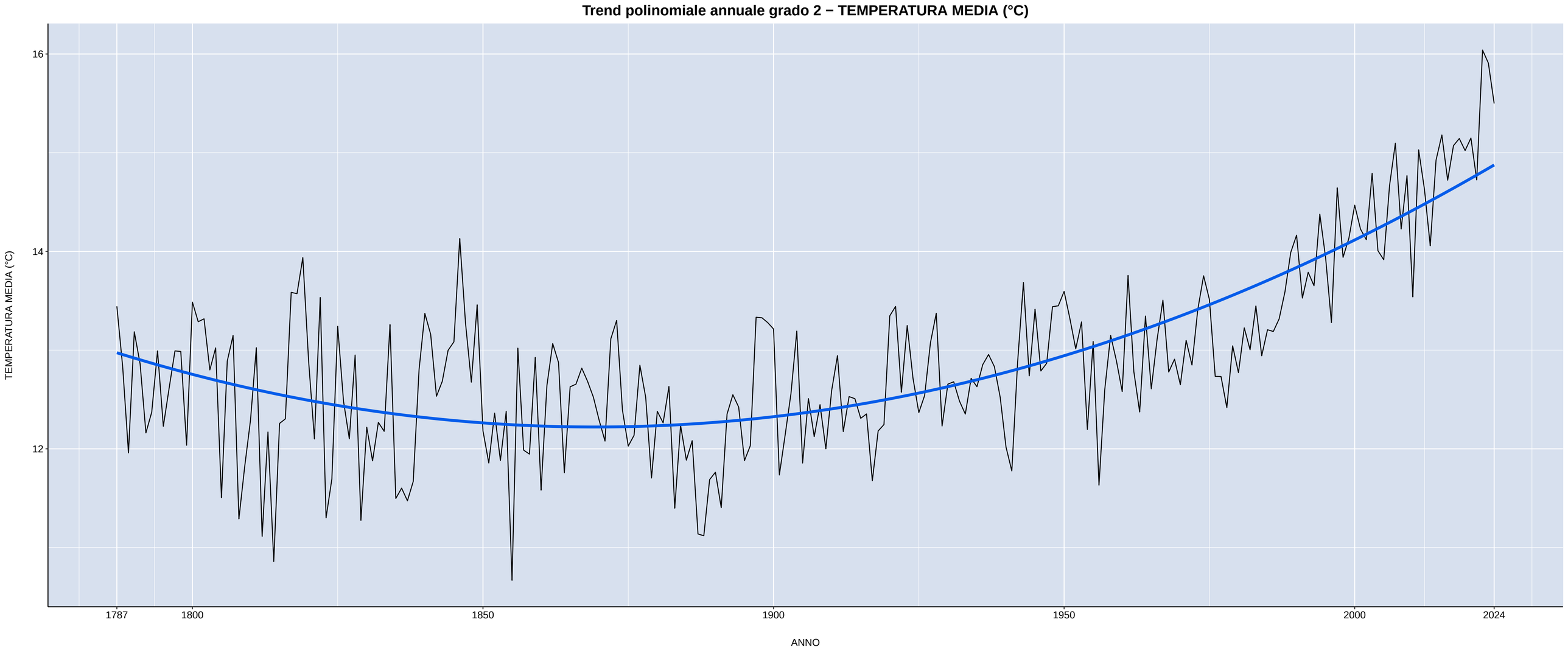

Serie annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
12
11
10
9
8
7
1787
1800
1850
1900
1950
2000
2024
ANNO

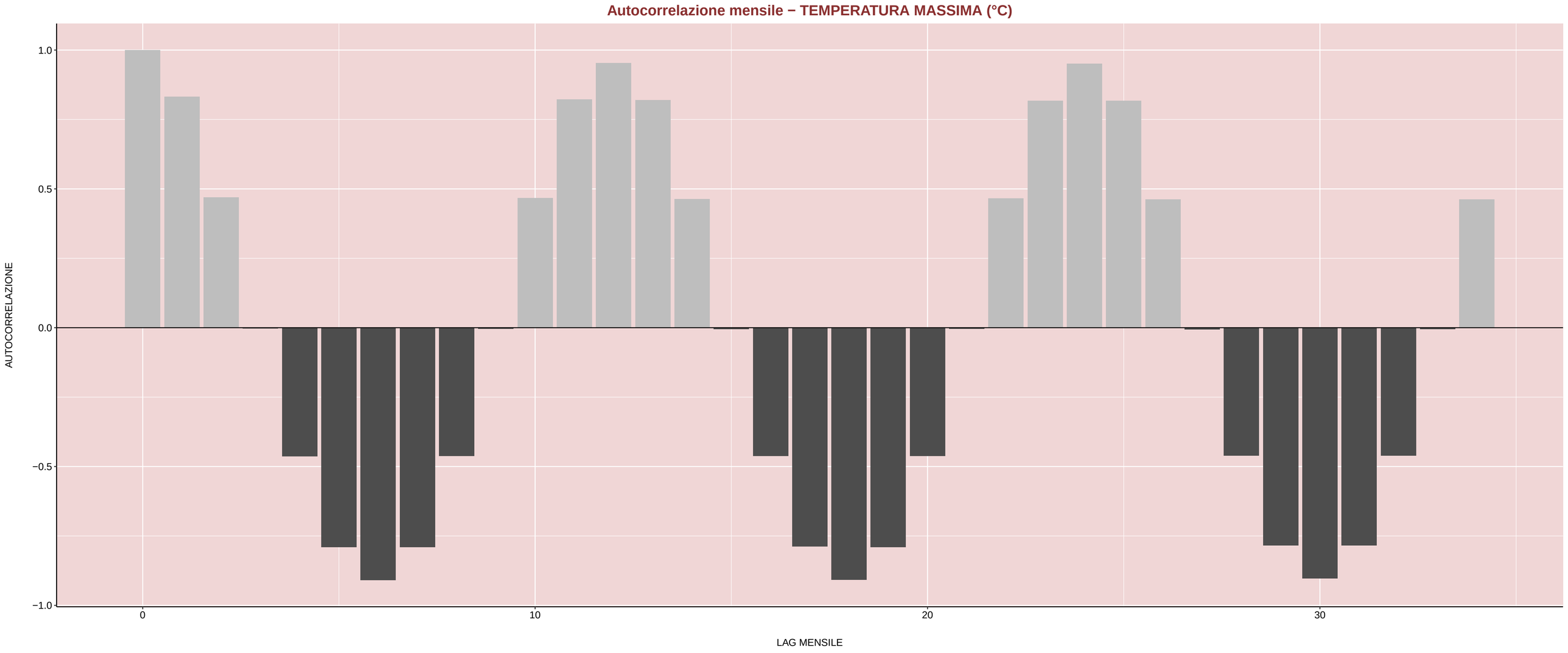

Trend lineare annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
12
11
10
9
8
7
1787
1800
1850
1900
1950
2000
2024
ANNO

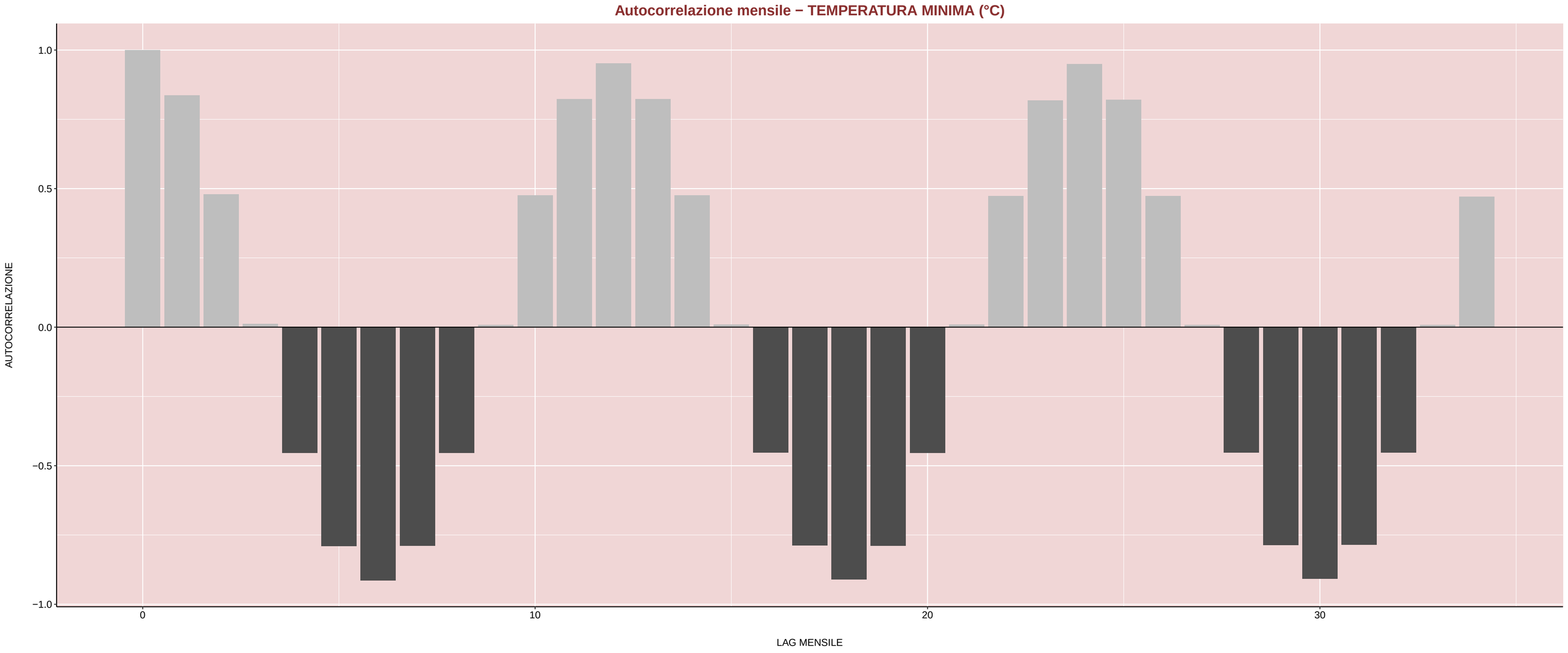

Trend polinomiale annuale grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
12
11
10
9
8
7
1787
1800
1850
1900
1950
2000
2024
ANNO

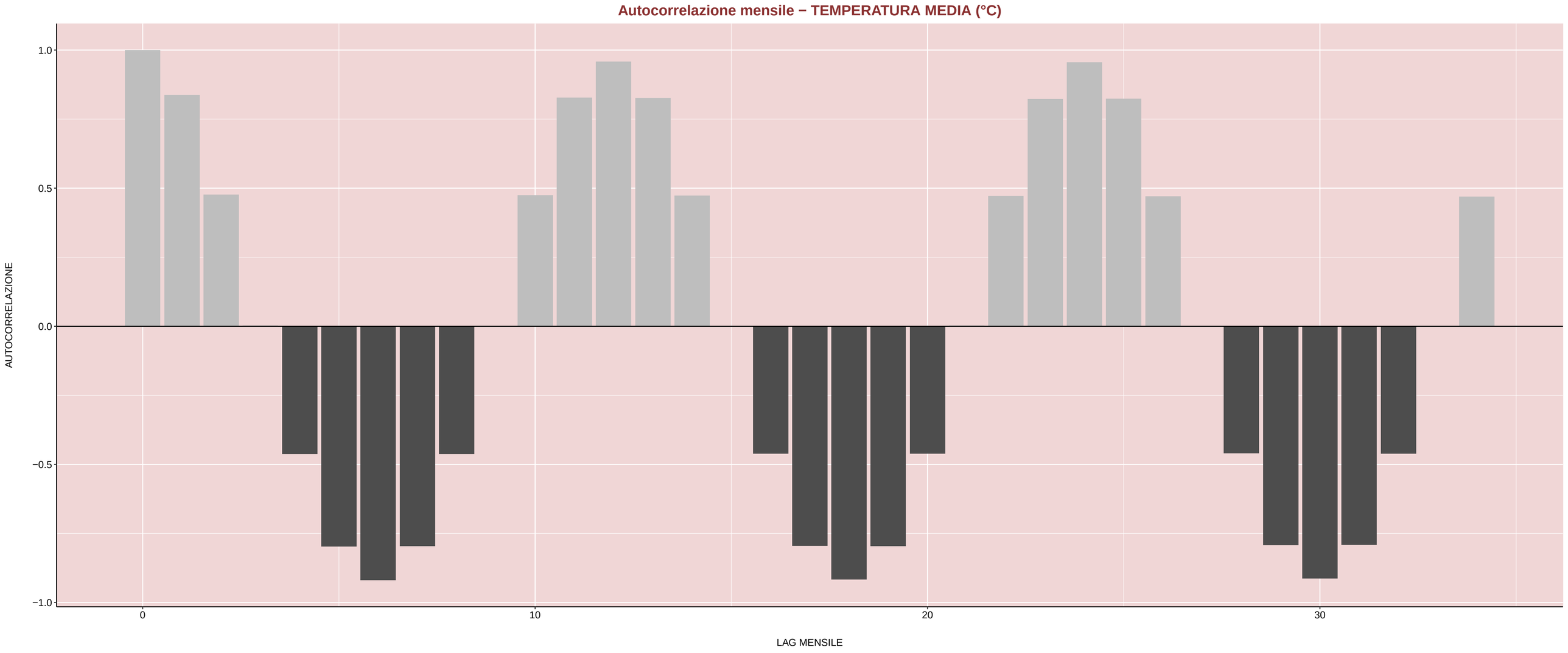

Serie annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
16
14
12
1787
1800
1850
1900
1950
2000
2024
ANNO

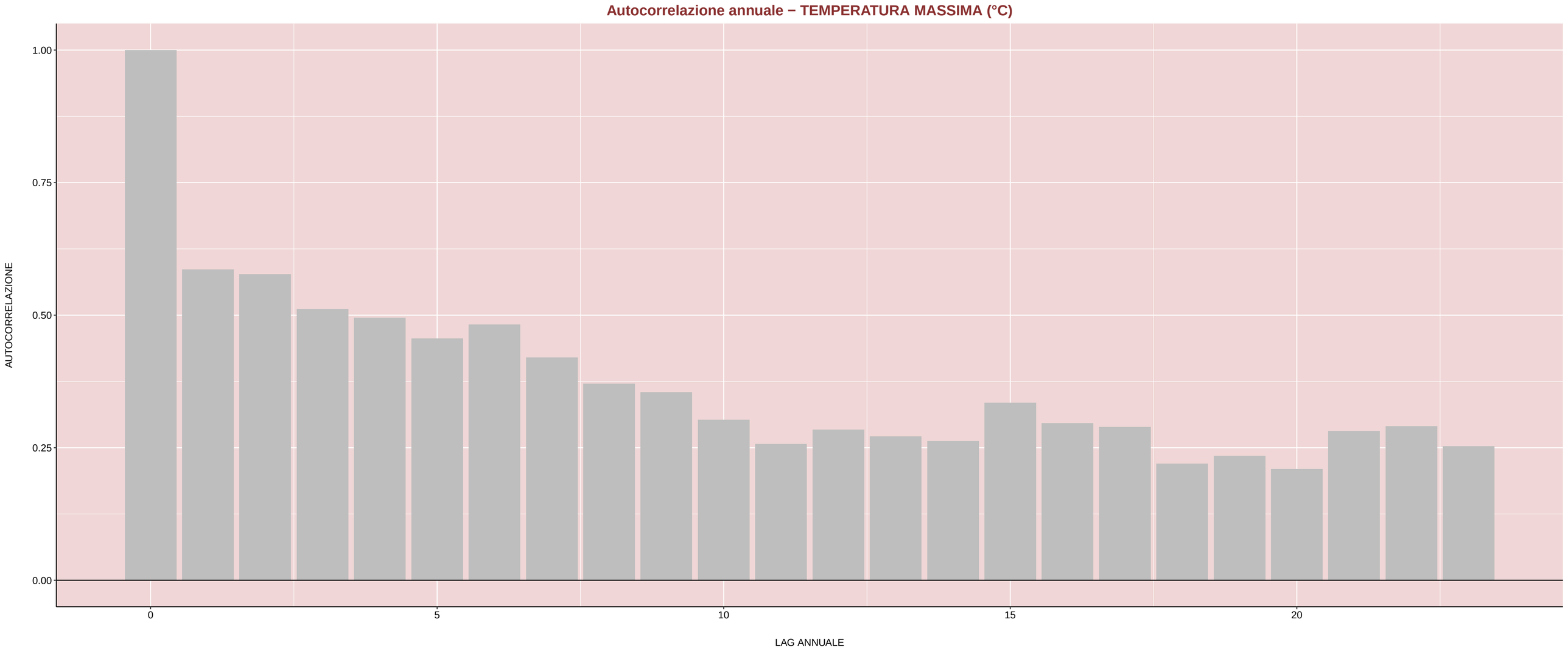

Trend lineare annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
16
14
12
1787
1800
1850
1900
1950
2000
2024
ANNO

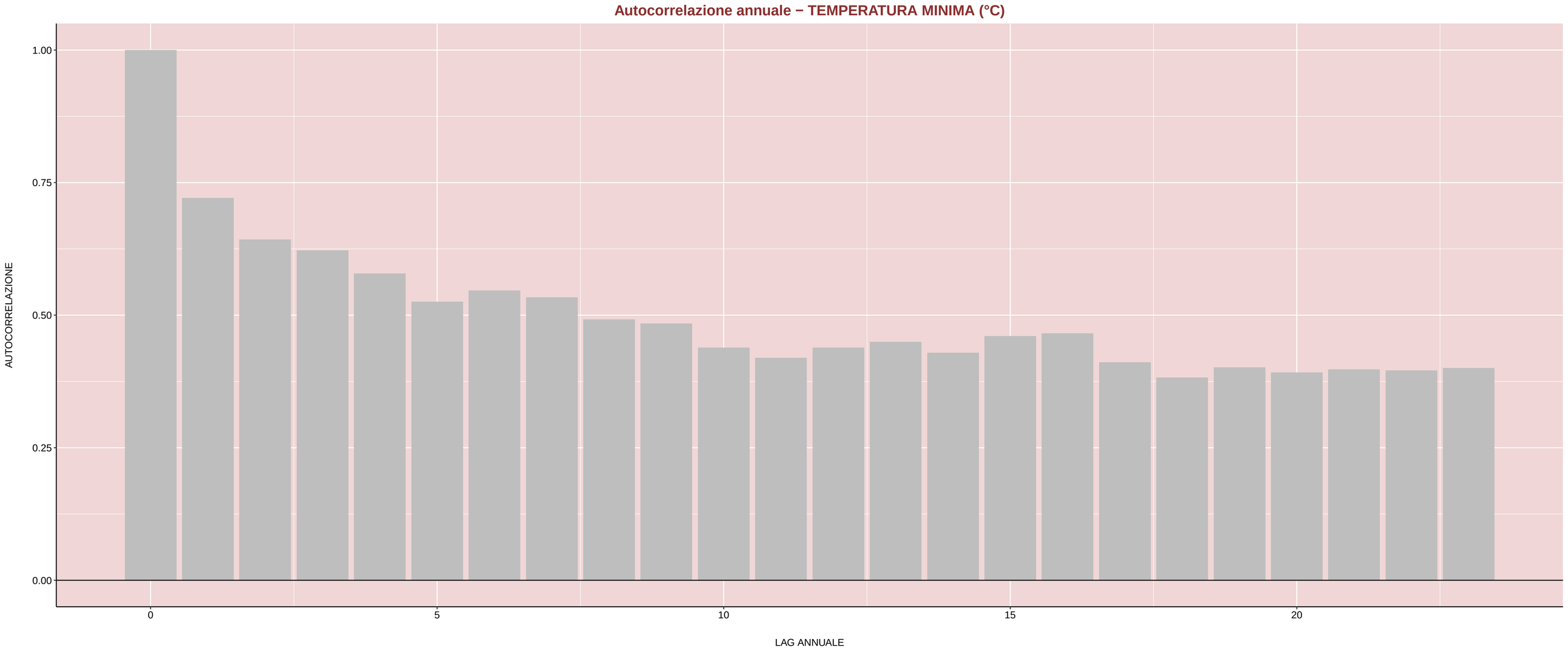

Trend polinomiale annuale grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
16
14
12
1787
1800
1850
1900
1950
2000
2024
ANNO

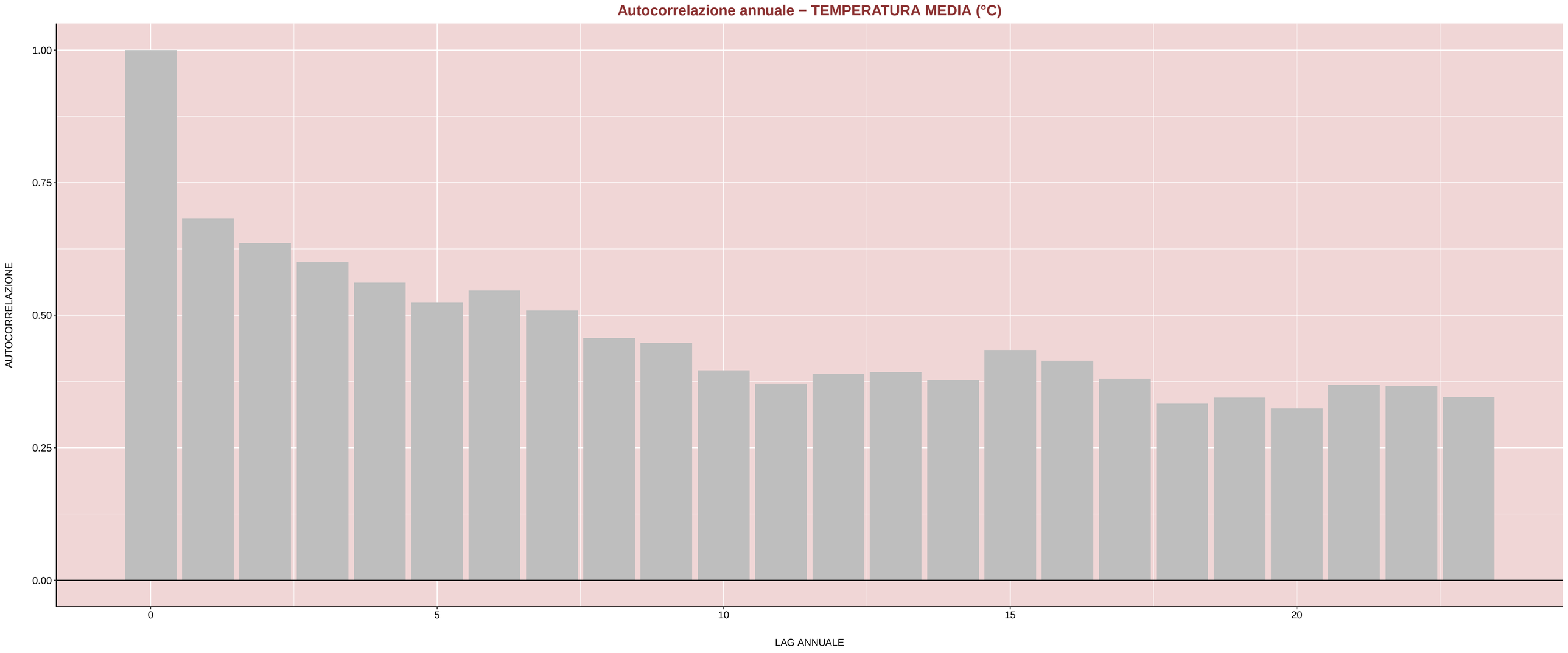

Autocorrelazione mensile − TEMPERATURA MASSIMA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
−1.0
0
10
20
30
LAG MENSILE


Autocorrelazione mensile − TEMPERATURA MINIMA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
−1.0
0
10
20
30
LAG MENSILE

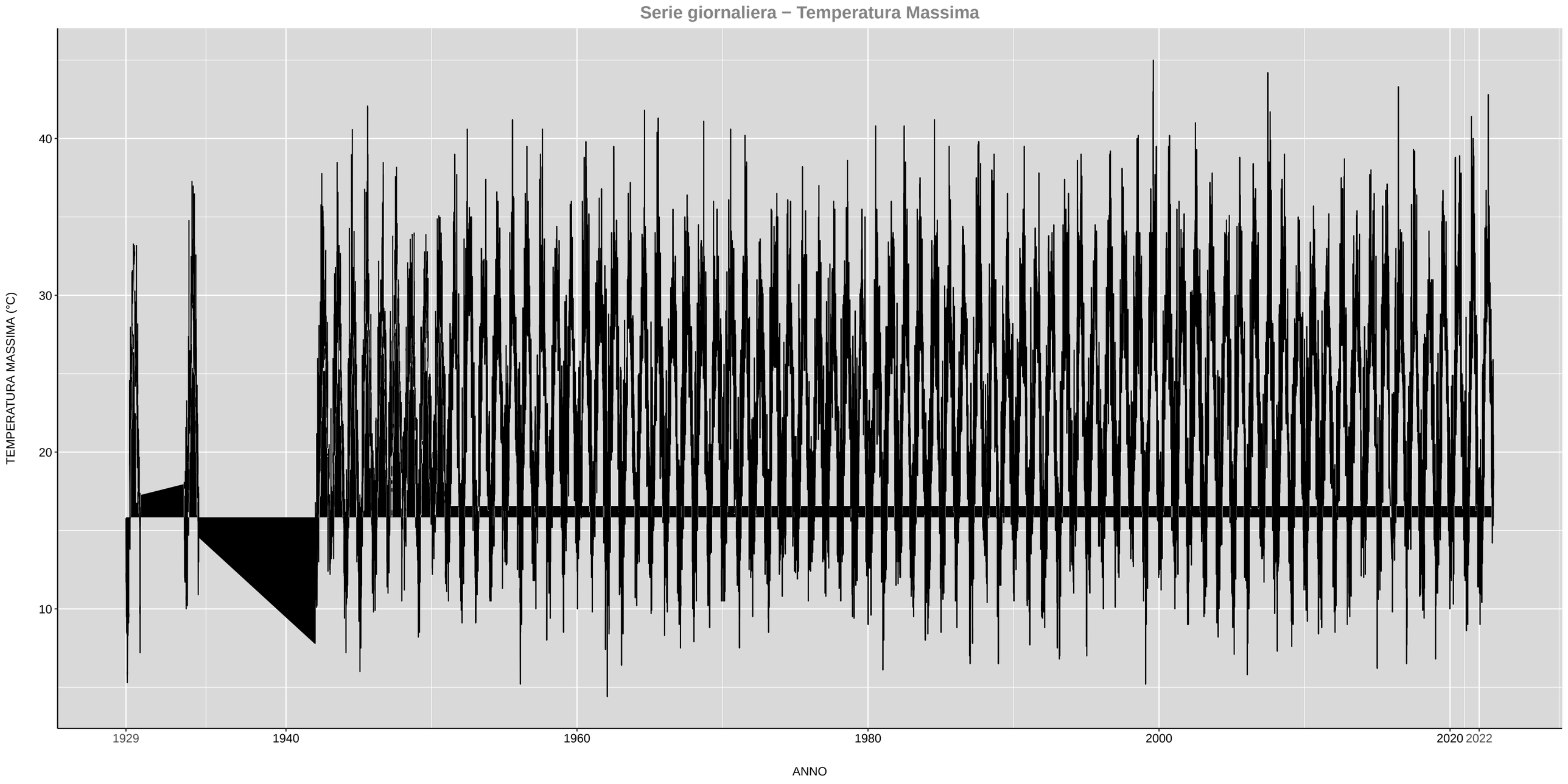

Autocorrelazione mensile − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
−1.0
0
10
20
30
LAG MENSILE

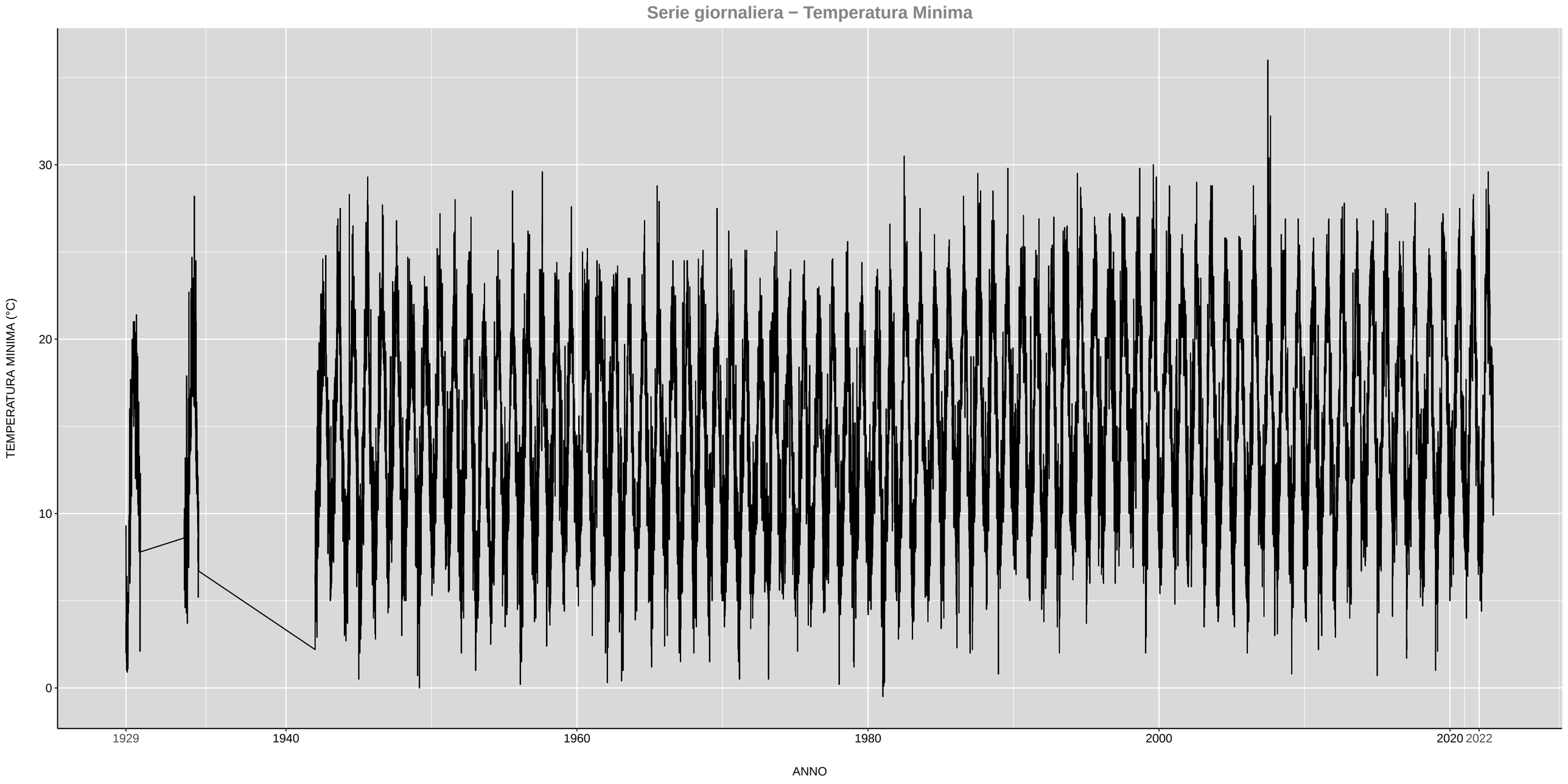

Autocorrelazione annuale − TEMPERATURA MASSIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
20
LAG ANNUALE

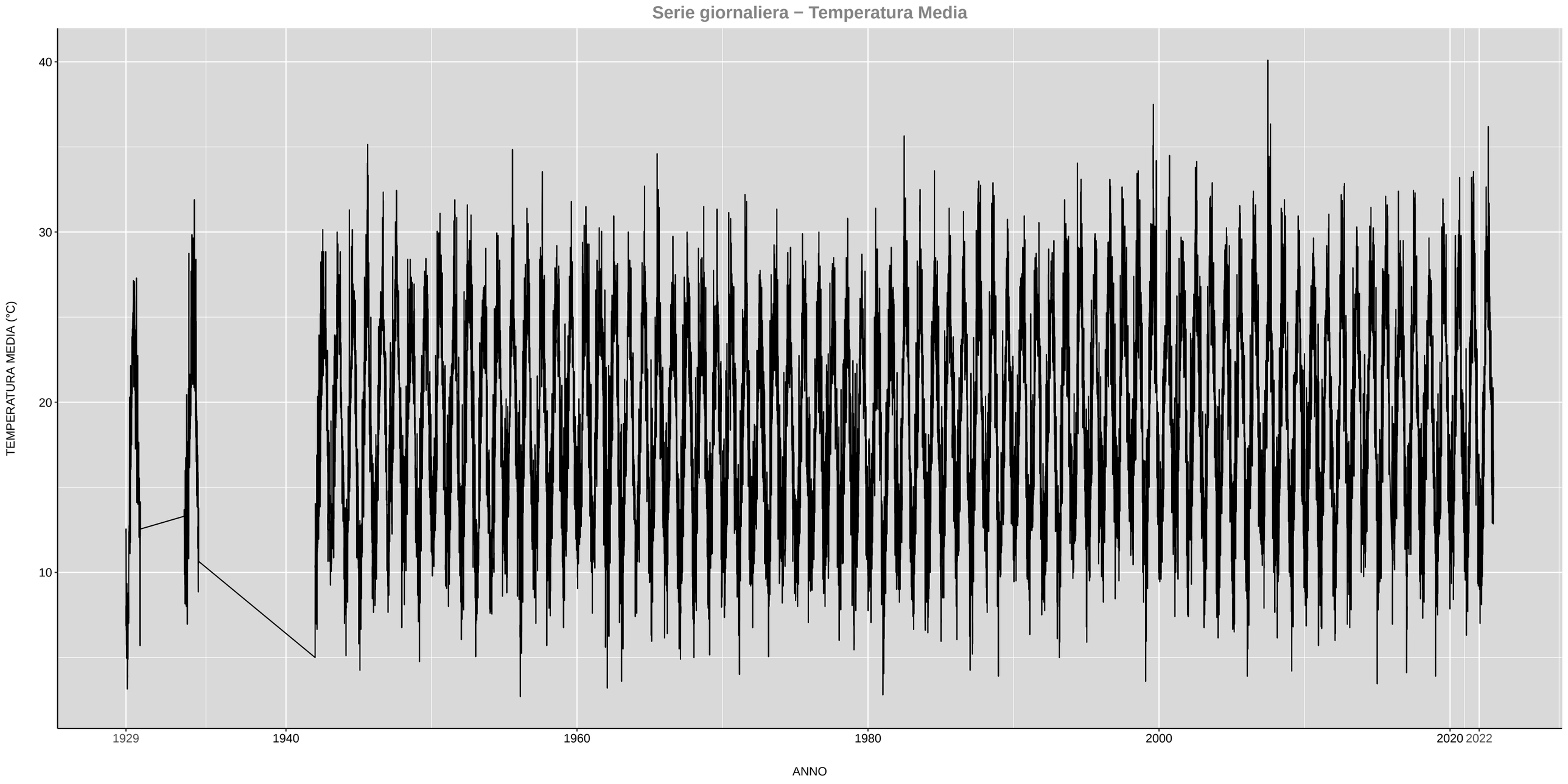

Autocorrelazione annuale − TEMPERATURA MINIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
20
LAG ANNUALE

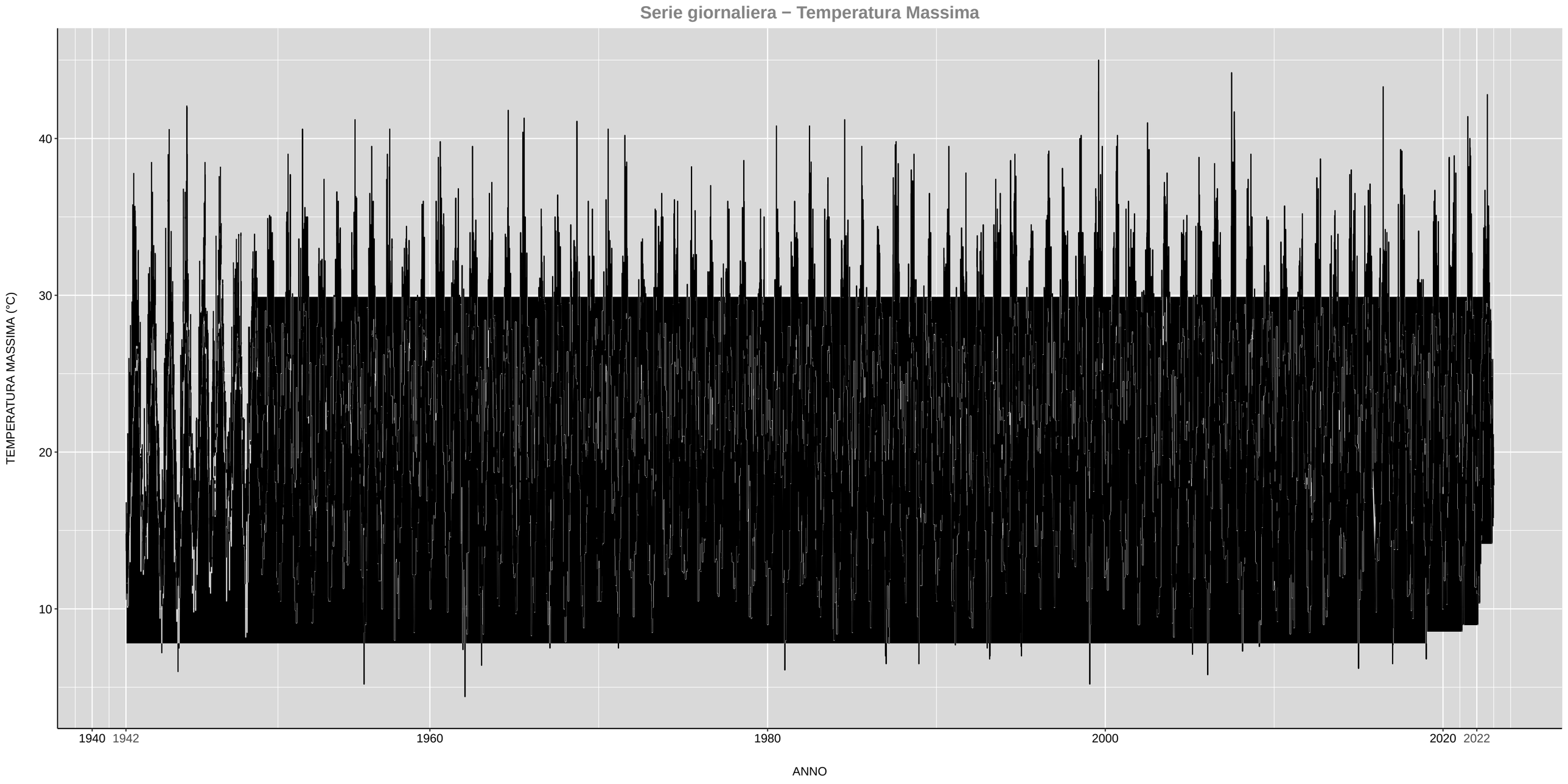

Autocorrelazione annuale − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
20
LAG ANNUALE

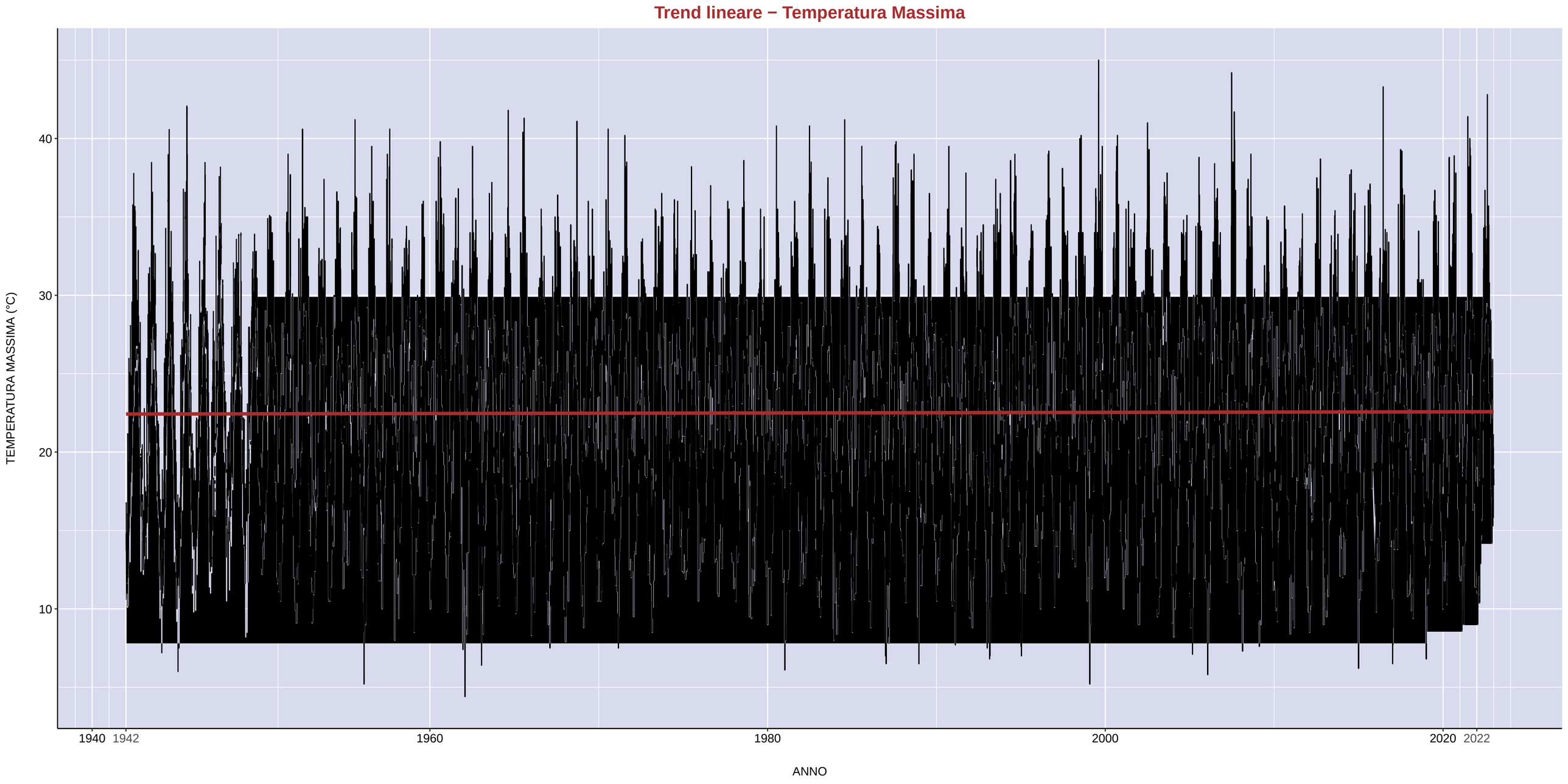
Palermo – Stazione di Palermo Osservatorio Astronomico

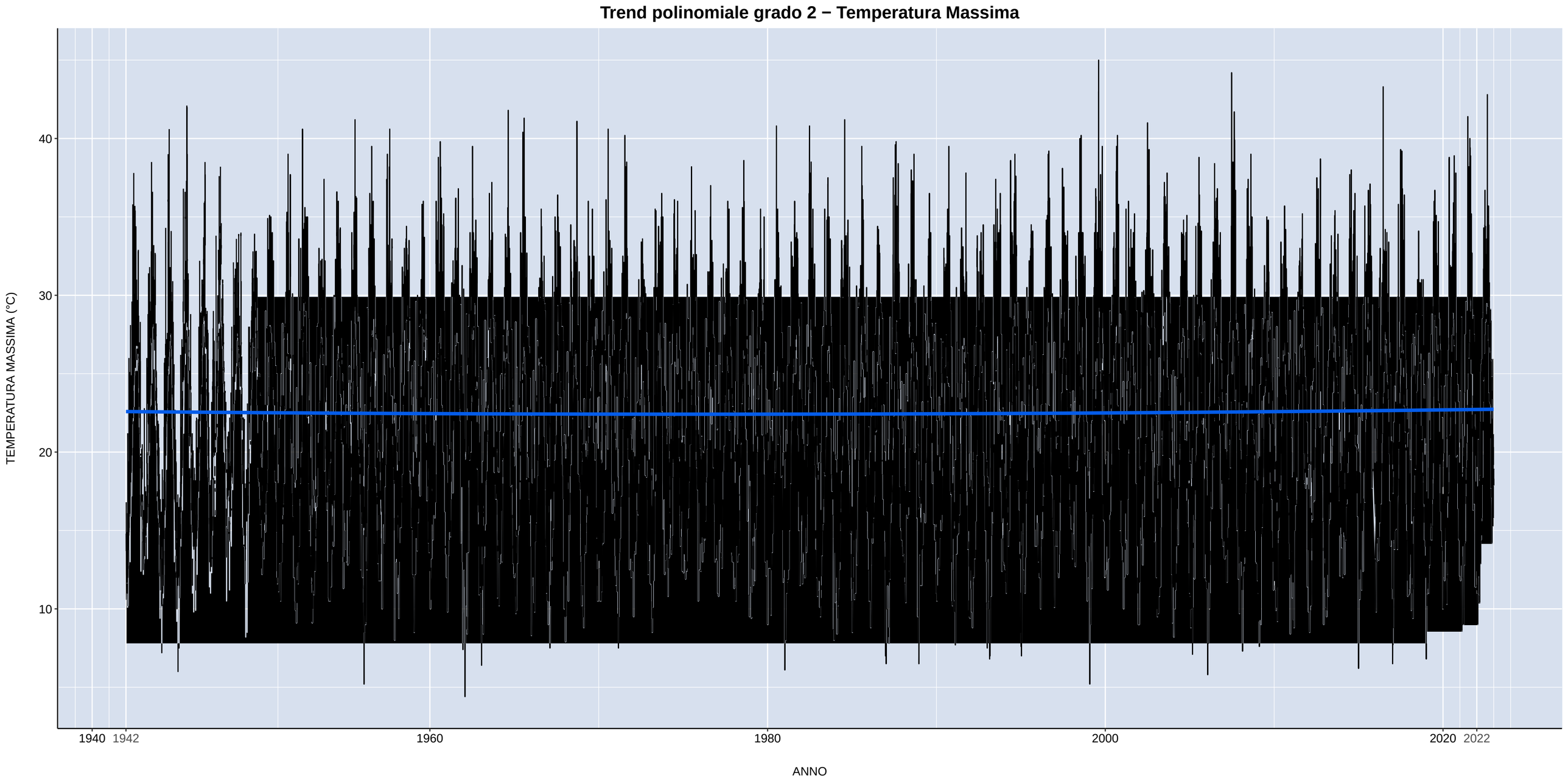

Serie giornaliera – Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
1929
1940
1960
1980
2000
2020
2022
ANNO

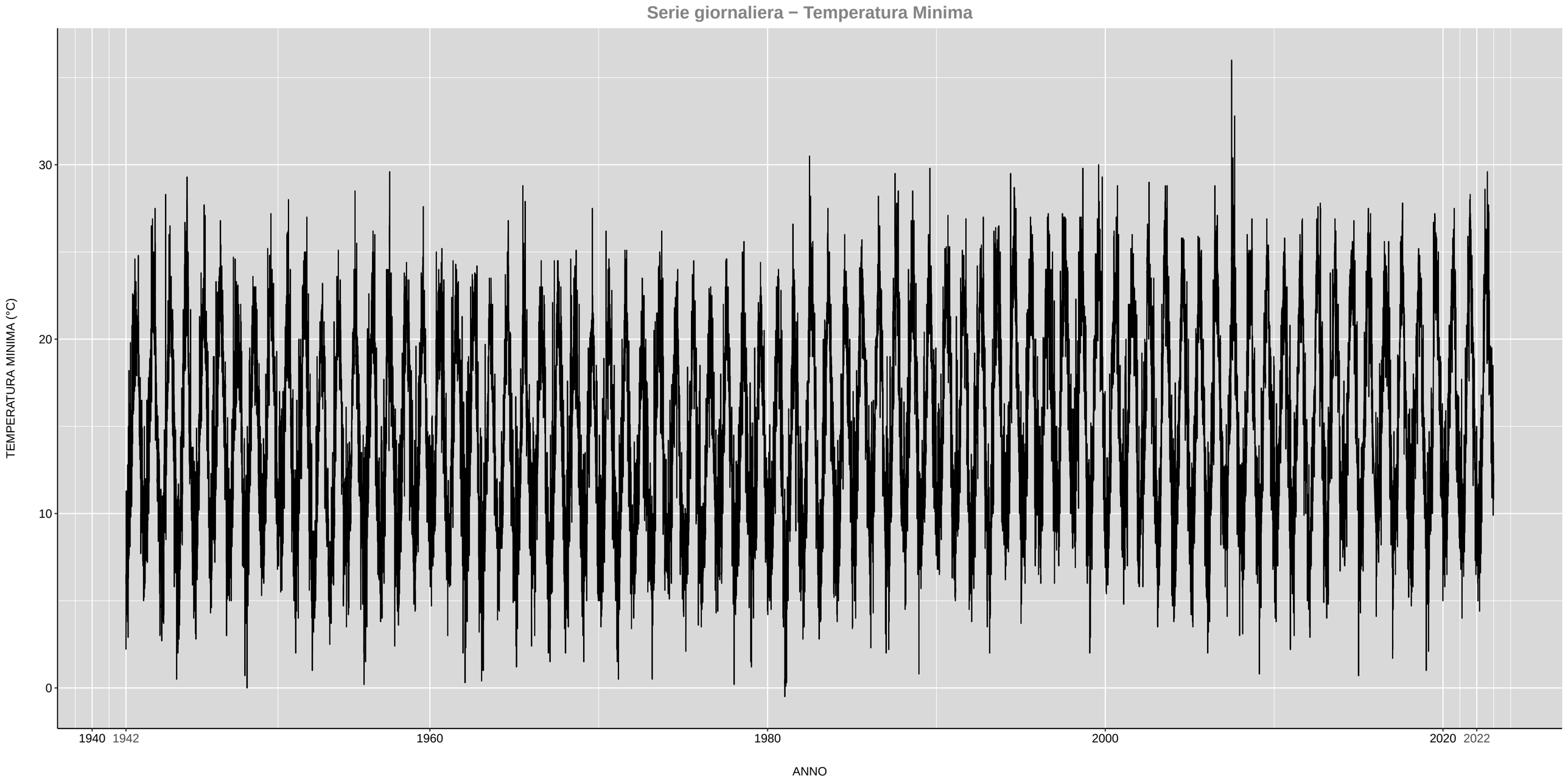

Serie giornaliera − Temperatura Minima
TEMPERATURA MINIMA (°C)
30
20
10
0
1929
1940
1960
1980
2000
2020
2022
ANNO

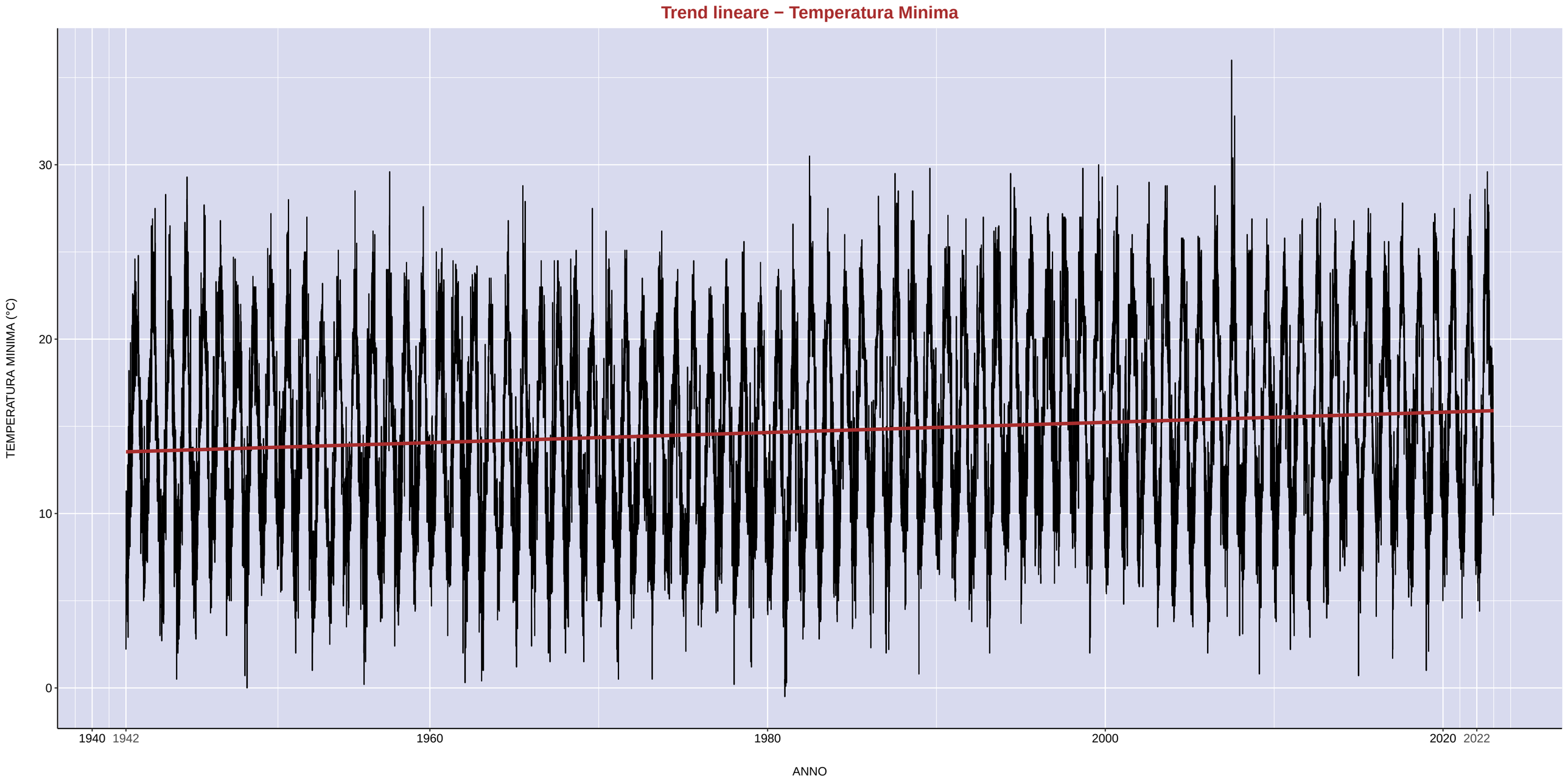

Serie giornaliera − Temperatura Media
TEMPERATURA MEDIA (°C)
40
30
20
10
1929
1940
1960
1980
2000
2020
2022
ANNO

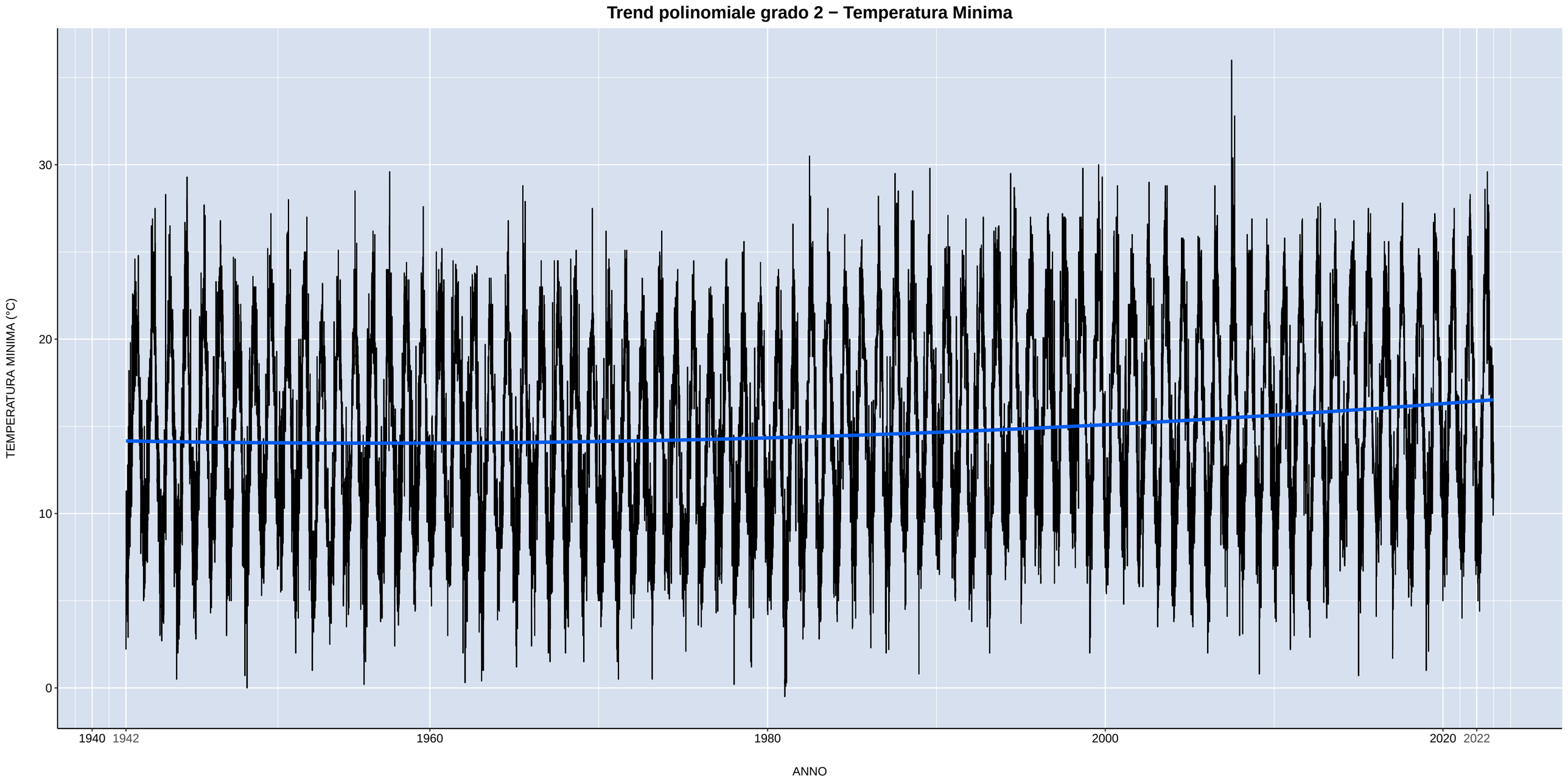

Serie giornaliera – Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
1940
1942
1960
1980
2000
2020
2022
ANNO

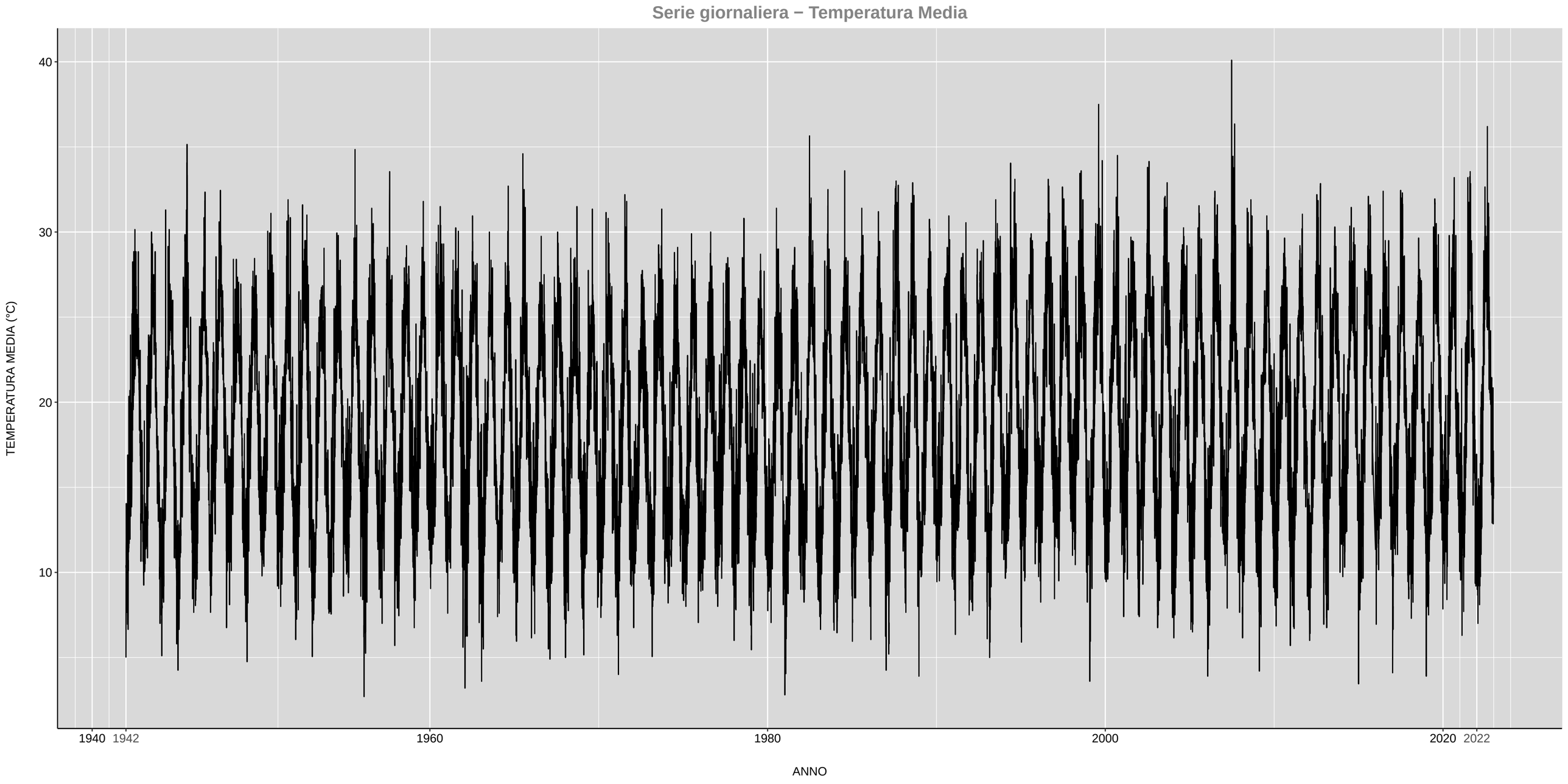

Trend lineare − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
1940
1942
1960
1980
2000
2020
2022
ANNO

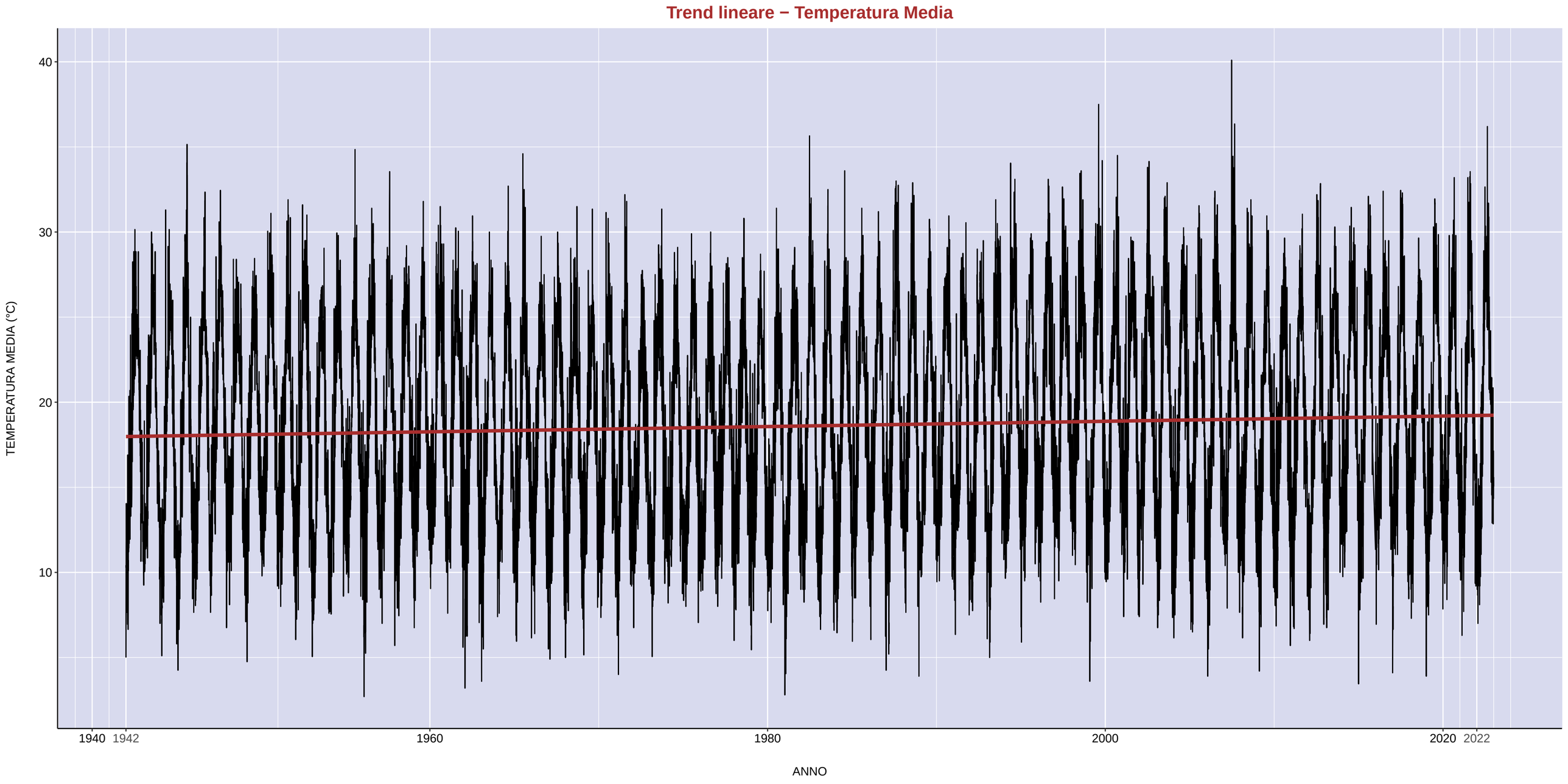

Trend polinomiale grado 2 − Temperatura Massima
TEMPERATURA MASSIMA (°C)
40
30
20
10
1940
1942
1960
1980
2000
2020
2022
ANNO

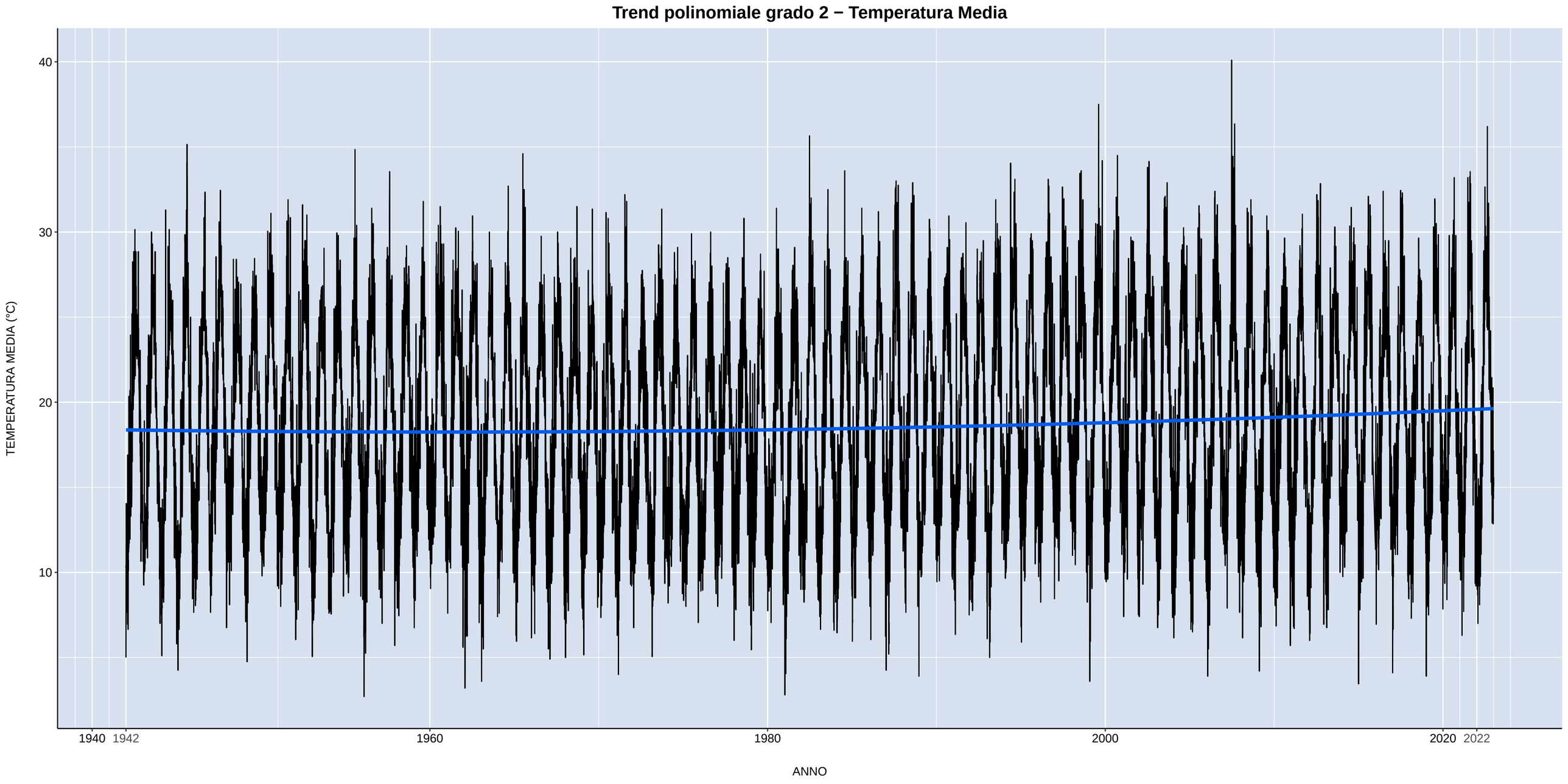

Serie giornaliera − Temperatura Minima
TEMPERATURA MINIMA (°C)
30
20
10
0
1940
1942
1960
1980
2000
2020
2022
ANNO

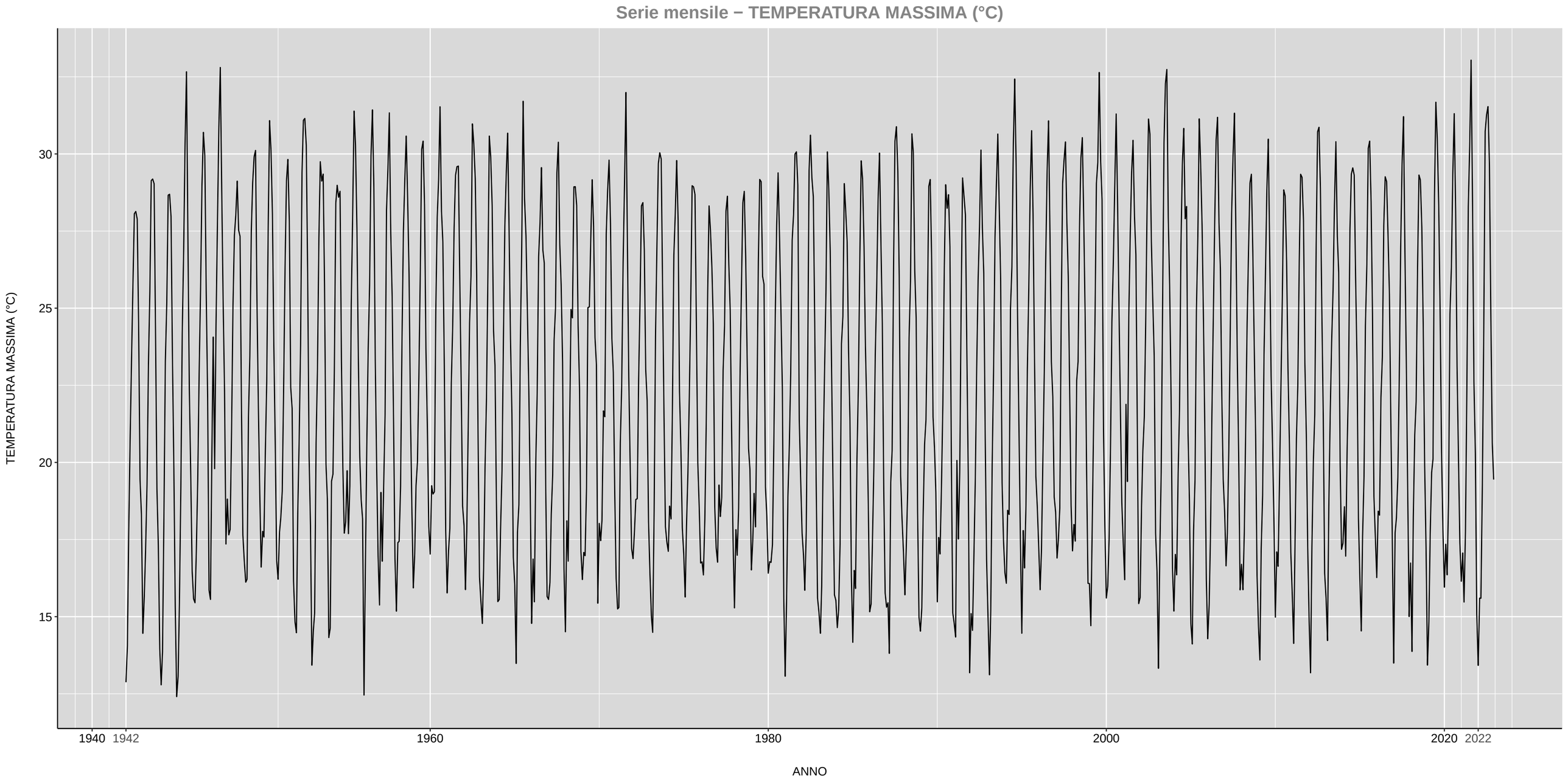

Trend lineare – Temperatura Minima
TEMPERATURA MINIMA (°C)
30
20
10
0
1940
1942
1960
1980
2000
2020
2022
ANNO

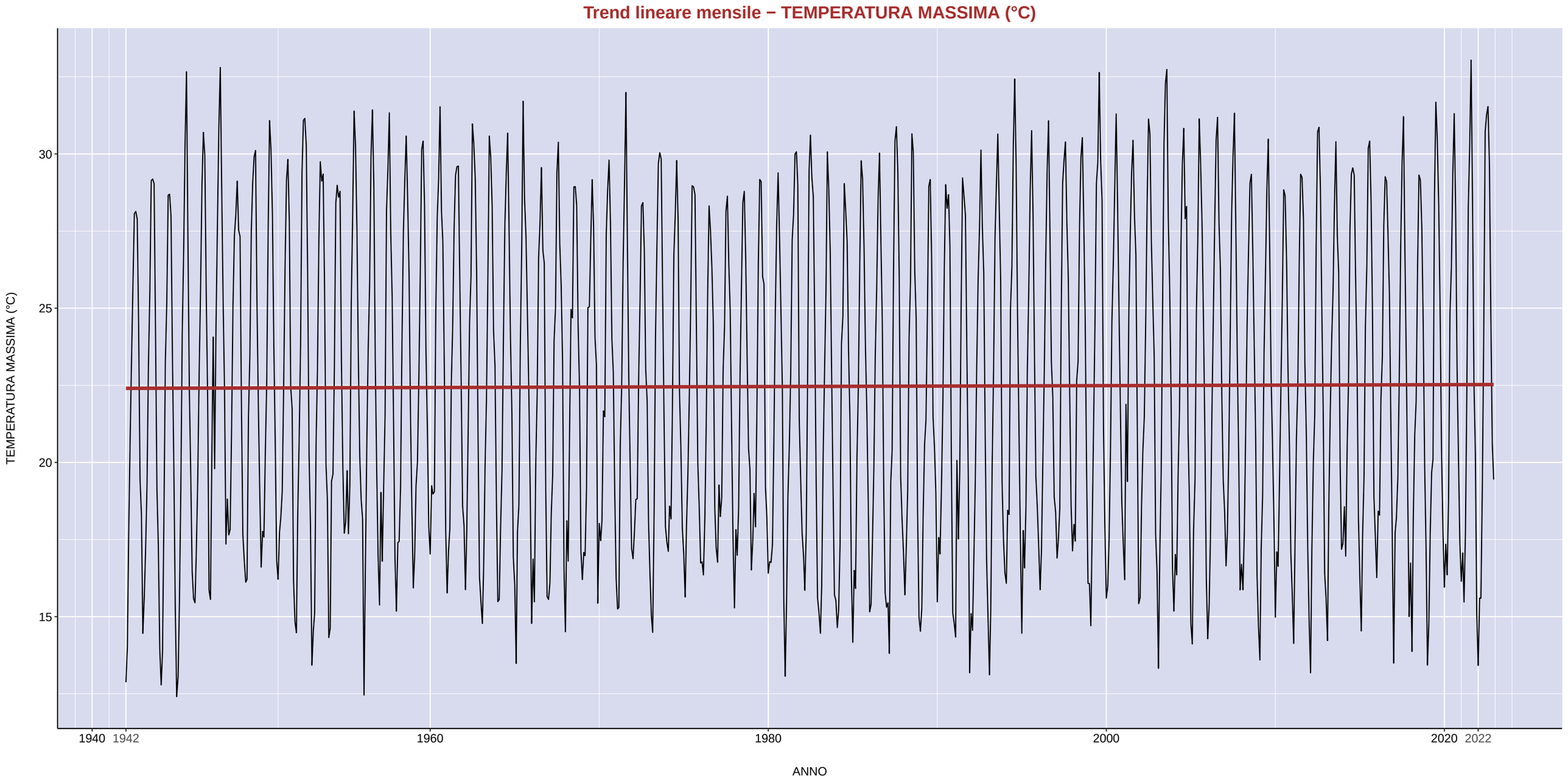

Trend polinomiale grado 2 − Temperatura Minima
TEMPERATURA MINIMA (°C)
30
20
10
0
1940
1942
1960
1980
2000
2020
2022
ANNO

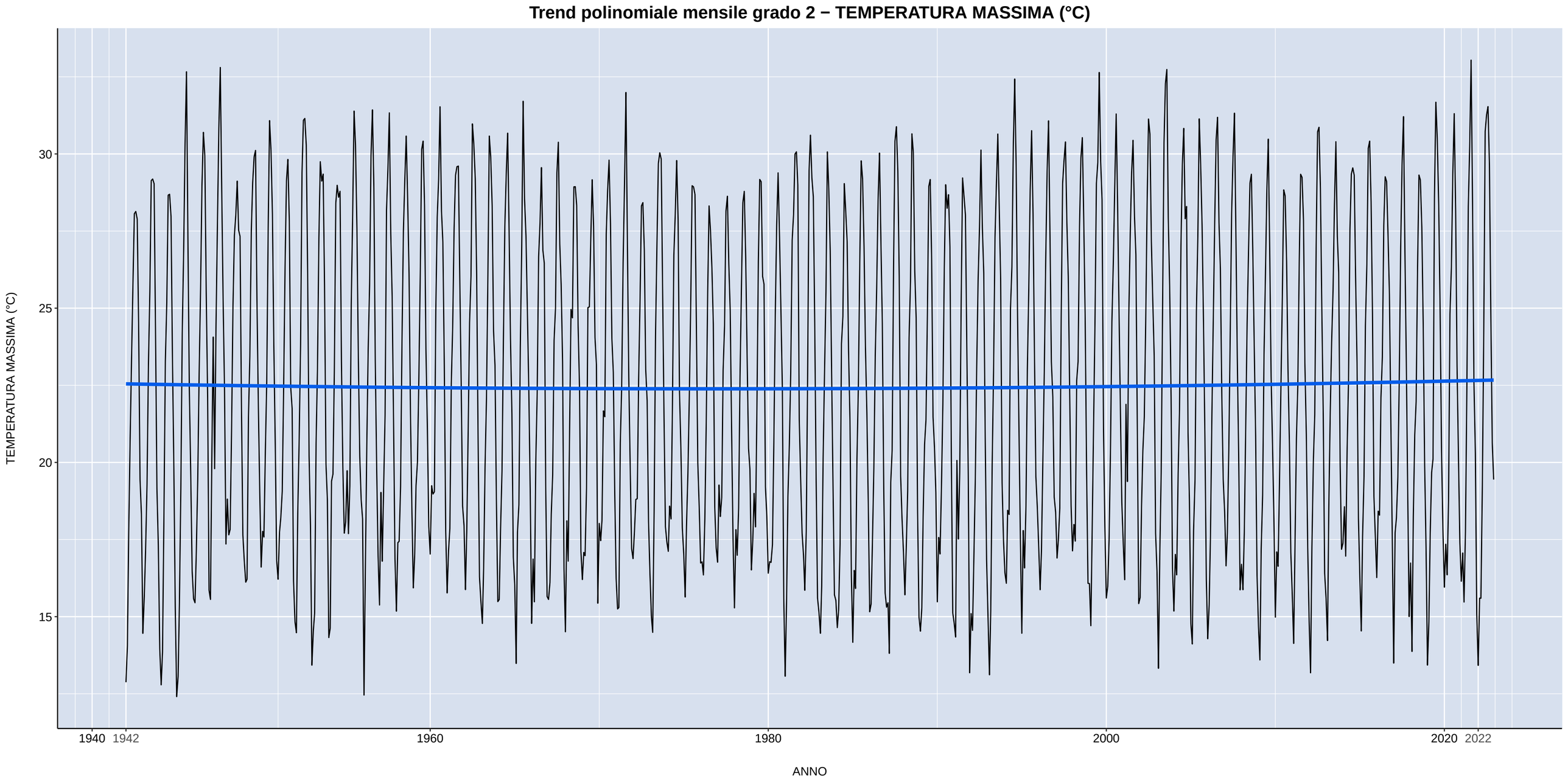

Serie giornaliera − Temperatura Media
TEMPERATURA MEDIA (°C)
40
30
20
10
1940
1942
1960
1980
2000
2020
2022
ANNO

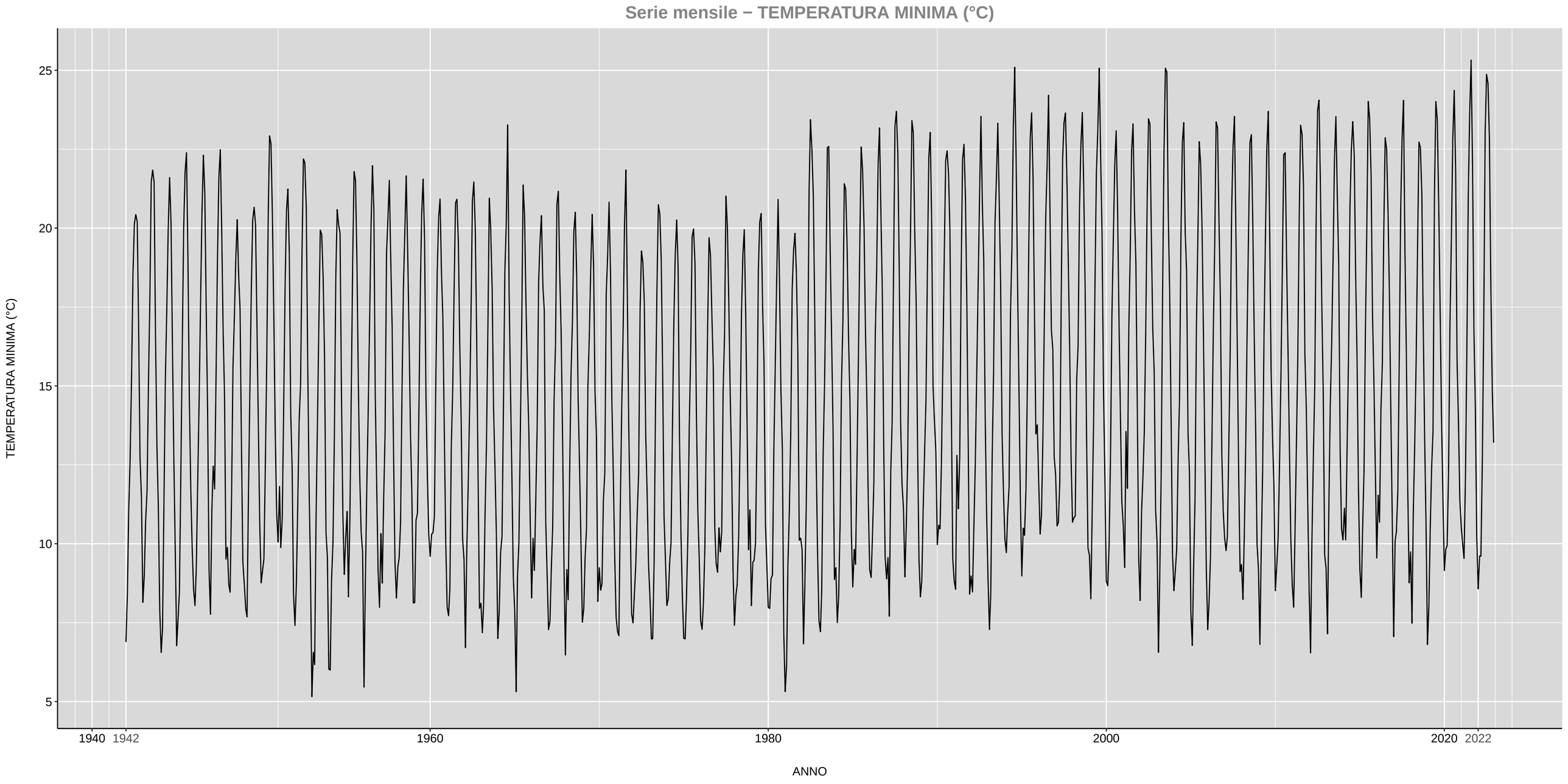

Trend lineare − Temperatura Media
TEMPERATURA MEDIA (°C)
40
30
20
10
1940
1942
1960
1980
2000
2020
2022
ANNO

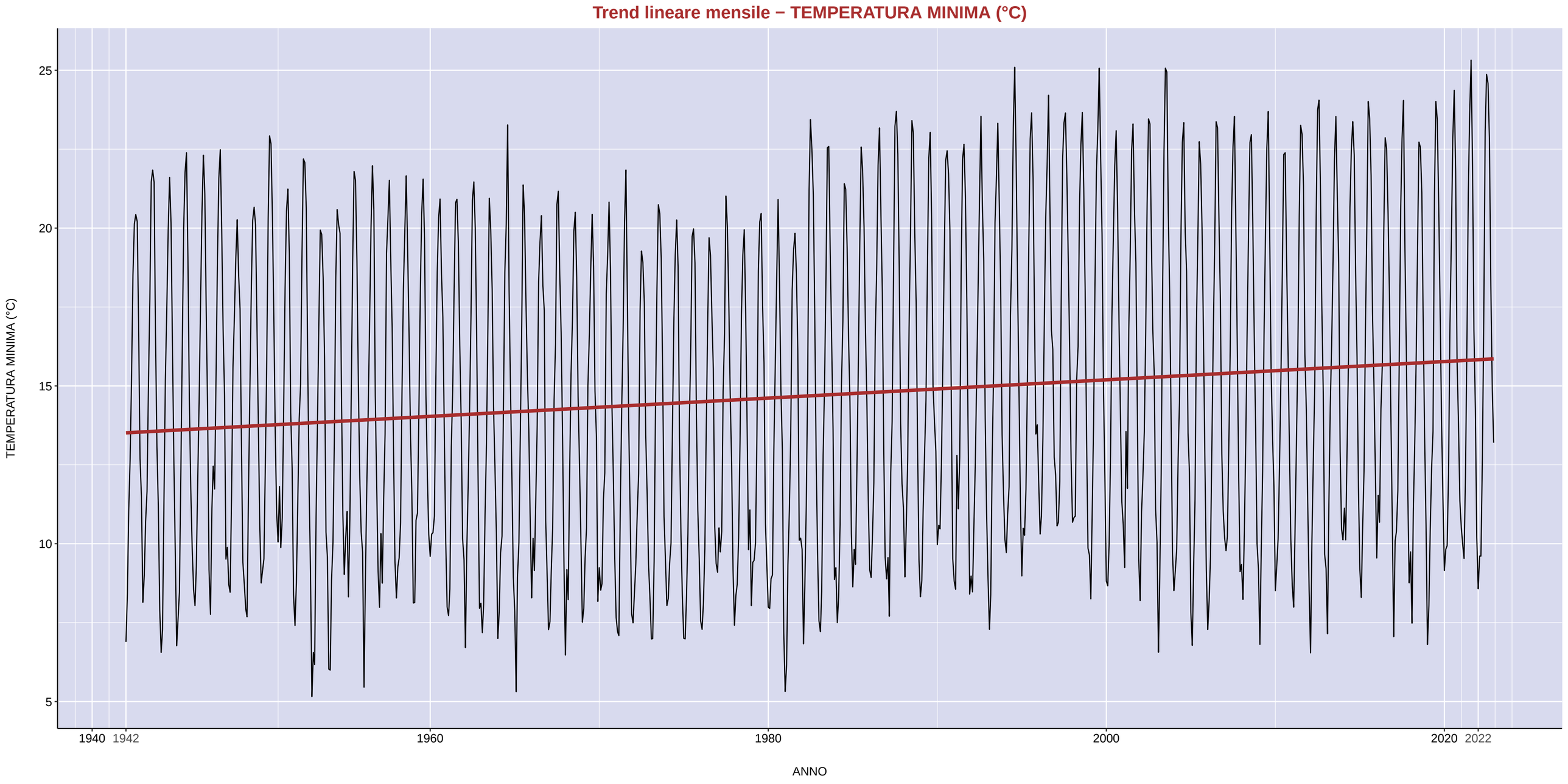

Trend polinomiale grado 2 − Temperatura Media
TEMPERATURA MEDIA (°C)
40
30
20
10
1940
1942
1960
1980
2000
2020
2022
ANNO

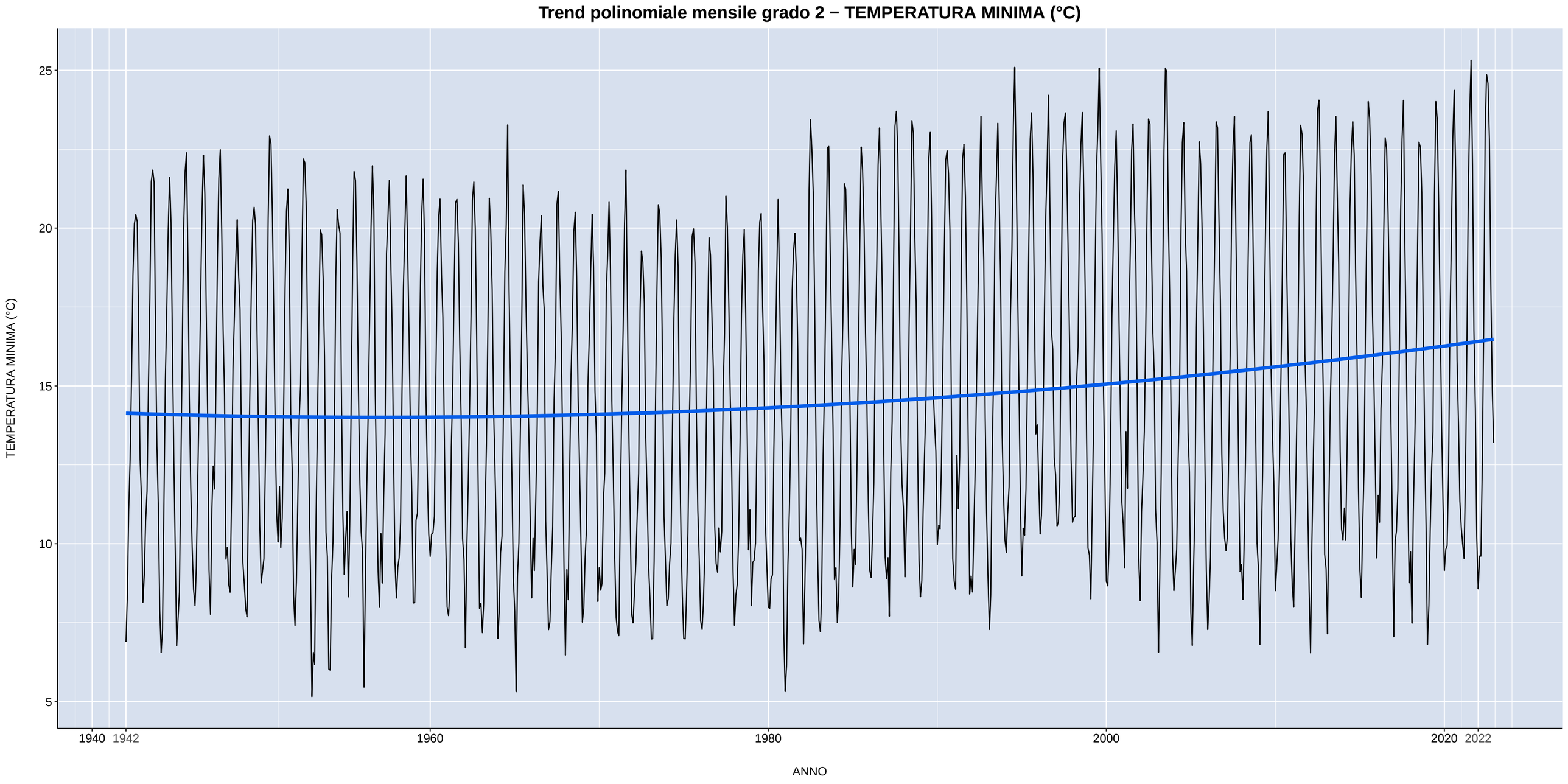

Serie mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
25
20
15
1940
1942
1960
1980
2000
2020
2022
ANNO

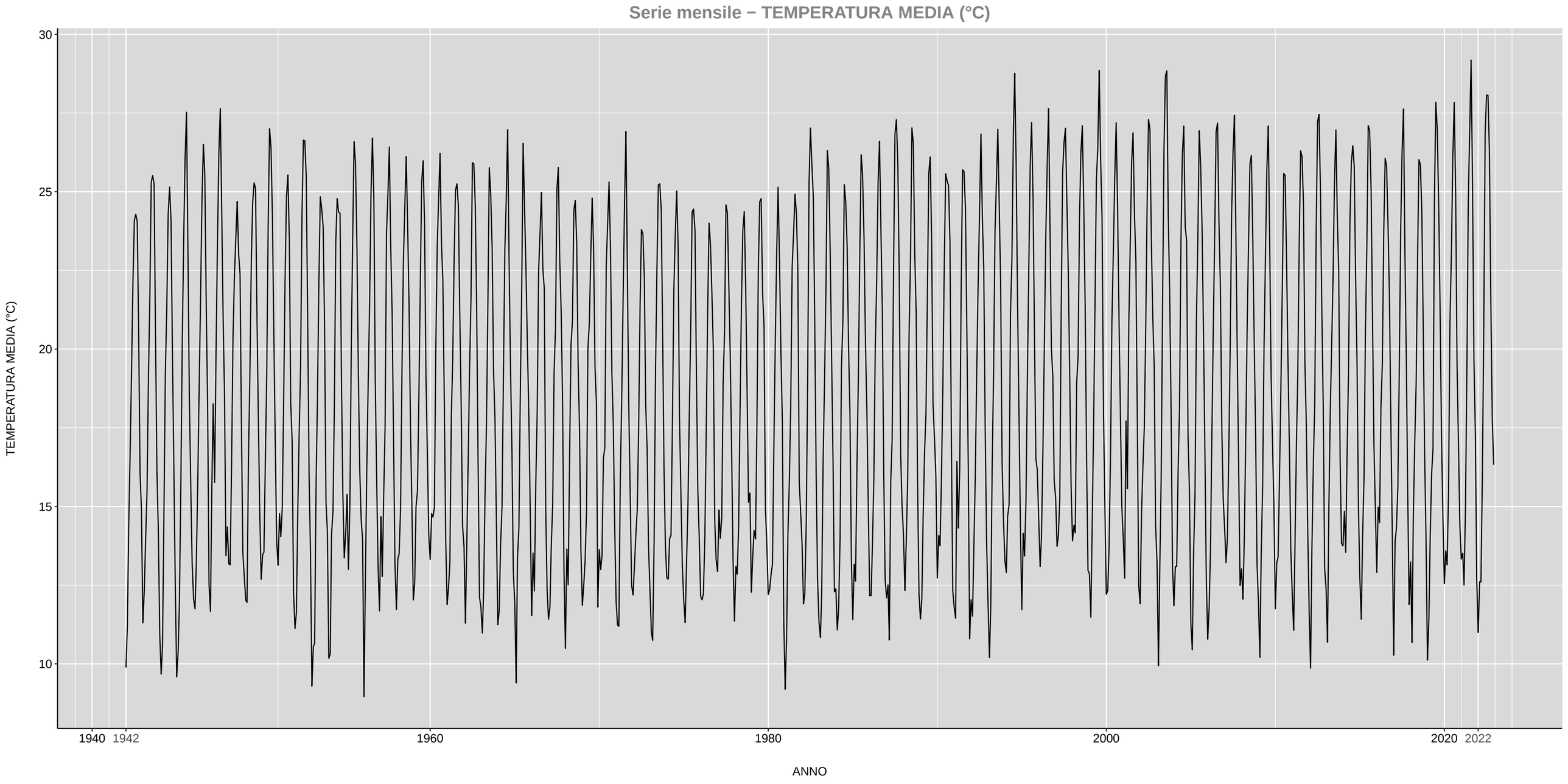

Trend lineare mensile − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
25
20
15
1940
1942
1960
1980
2000
2020
2022
ANNO

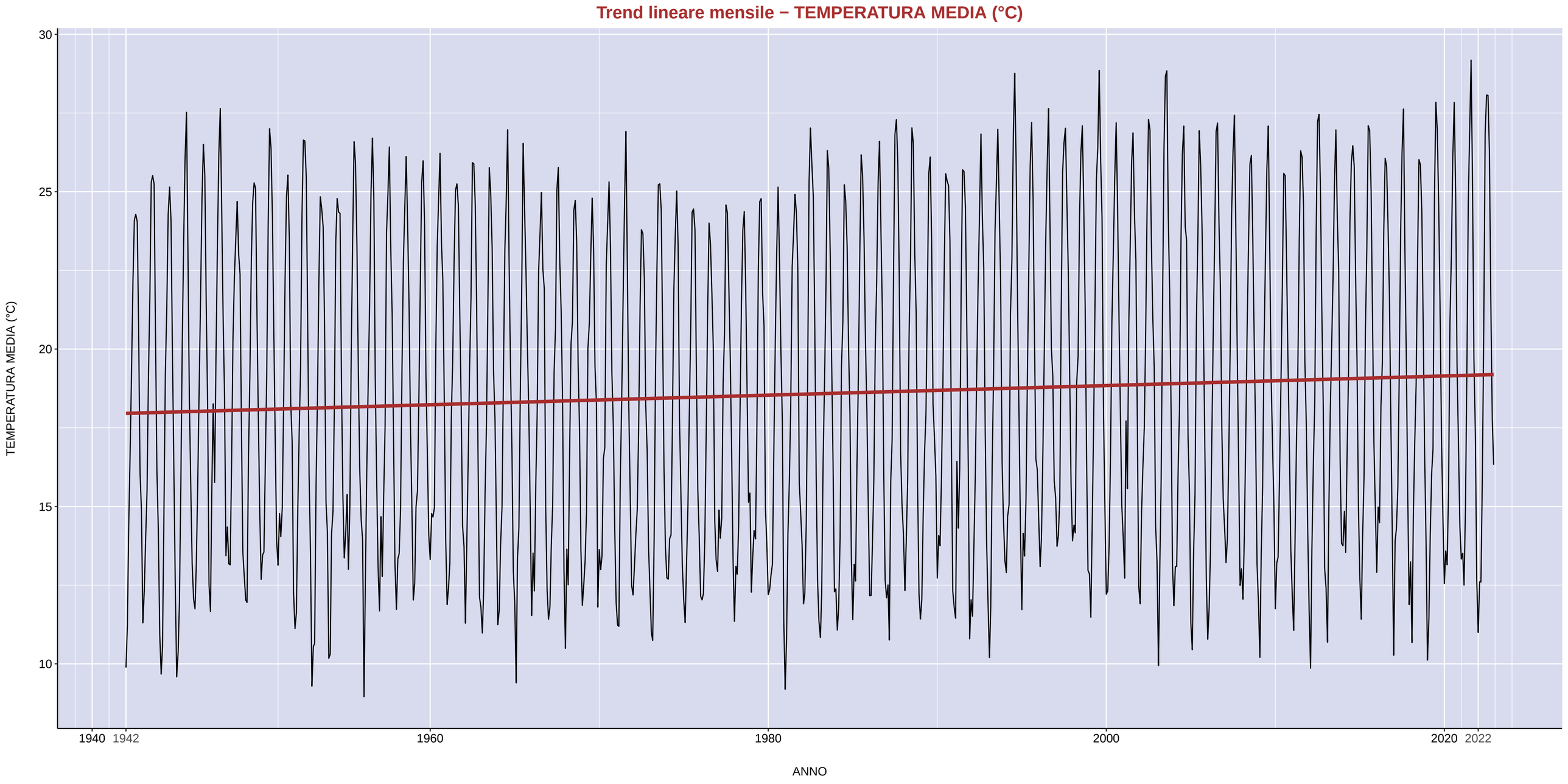

Trend polinomiale mensile grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
30
25
20
15
1940
1942
1960
1980
2000
2020
2022
ANNO

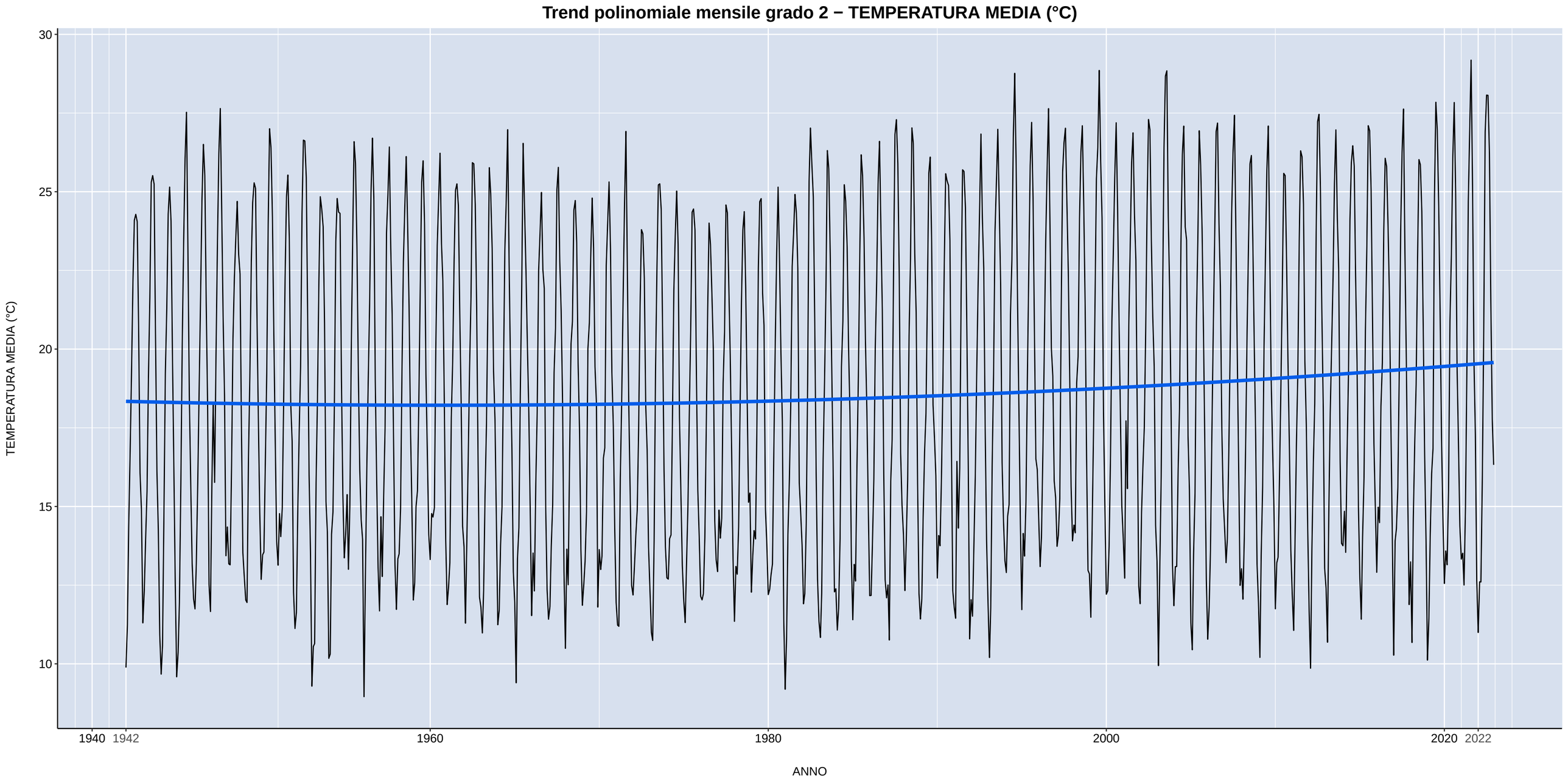

Serie mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
25
20
15
10
5
1940
1942
1960
1980
2000
2020
2022
ANNO

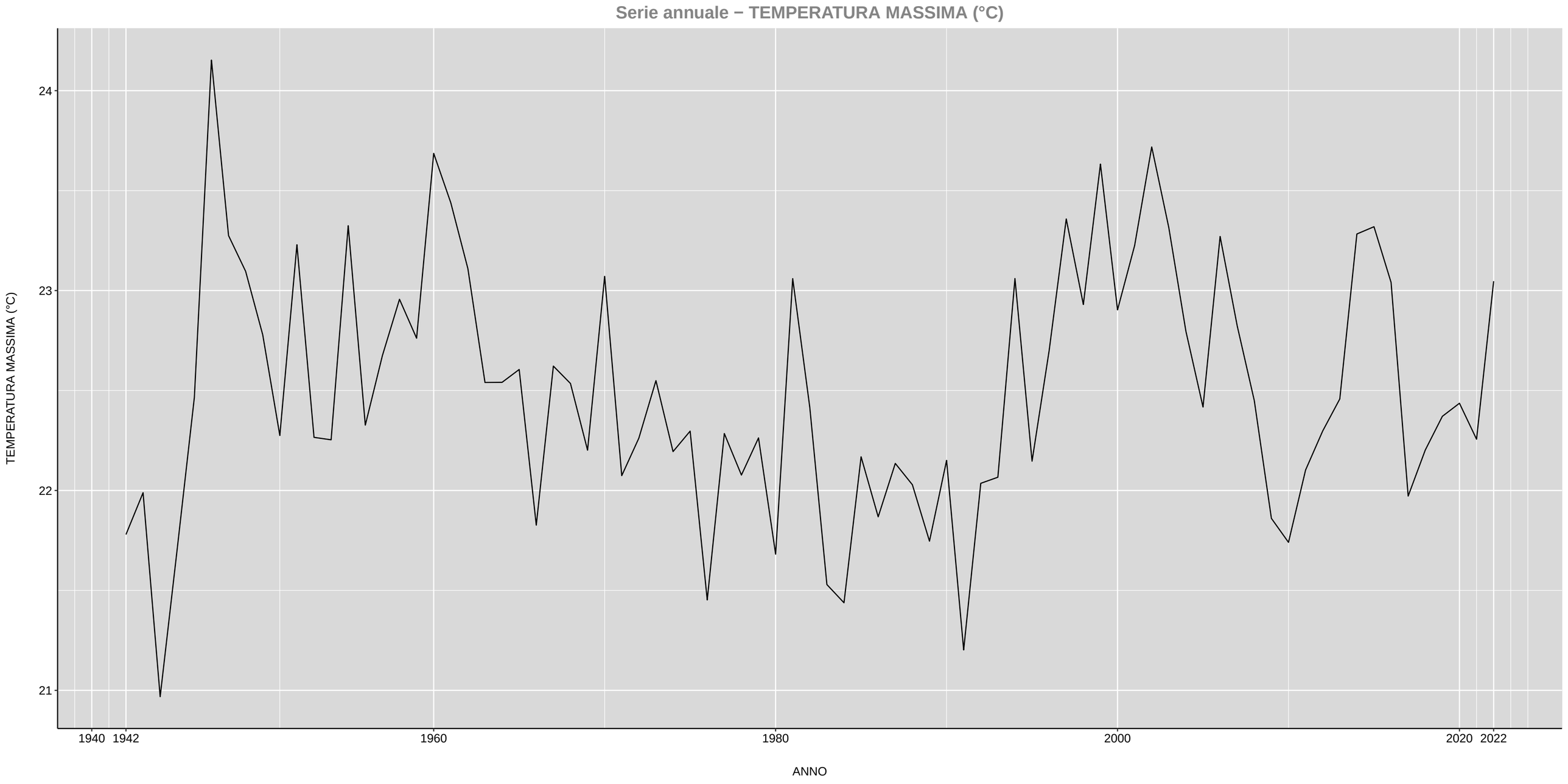

Trend lineare mensile − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
25
20
15
10
5
1940
1942
1960
1980
2000
2020
2022
ANNO

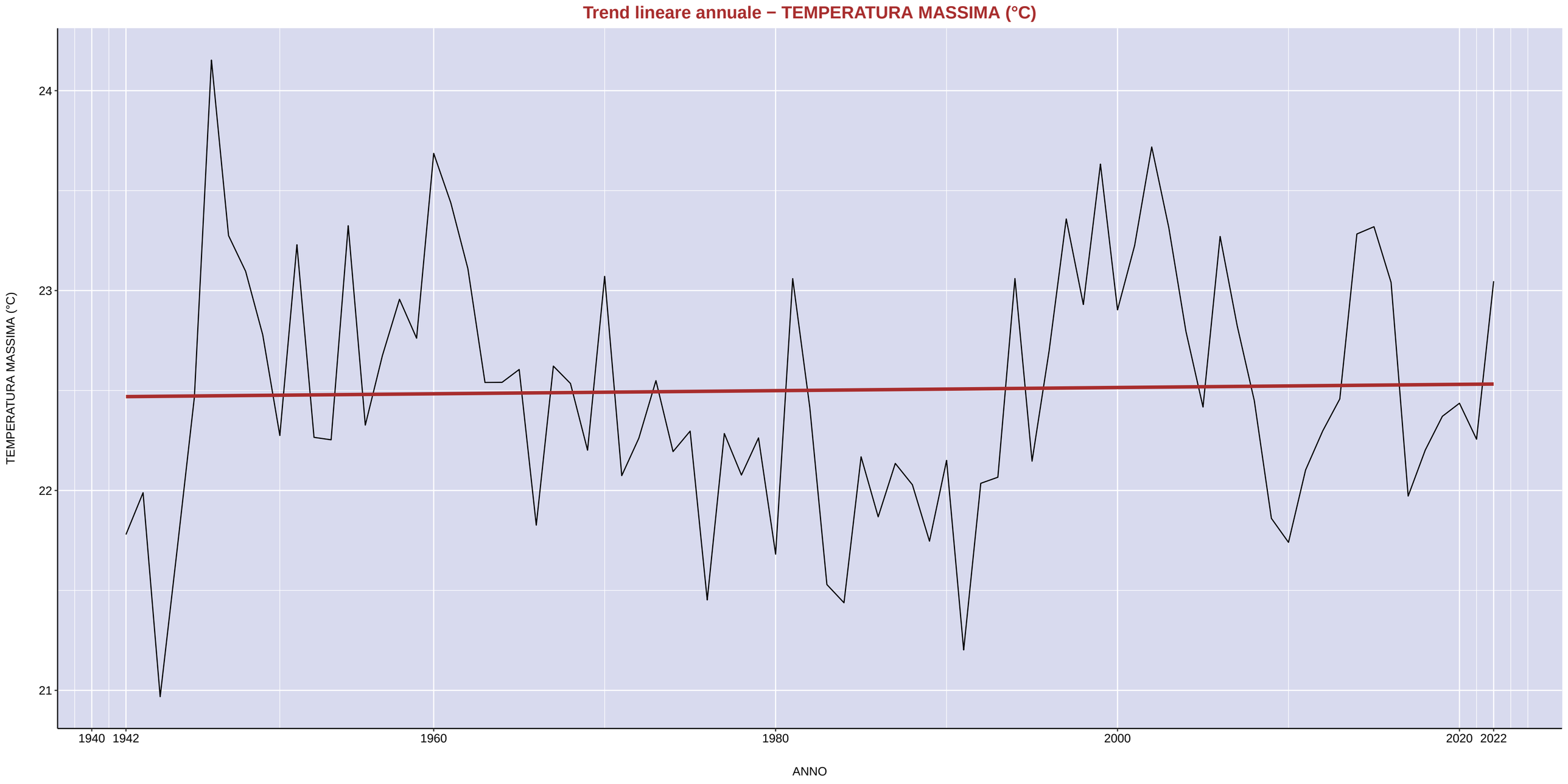

Trend polinomiale mensile grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
25
20
15
10
5
1940
1942
1960
1980
2000
2020
2022
ANNO

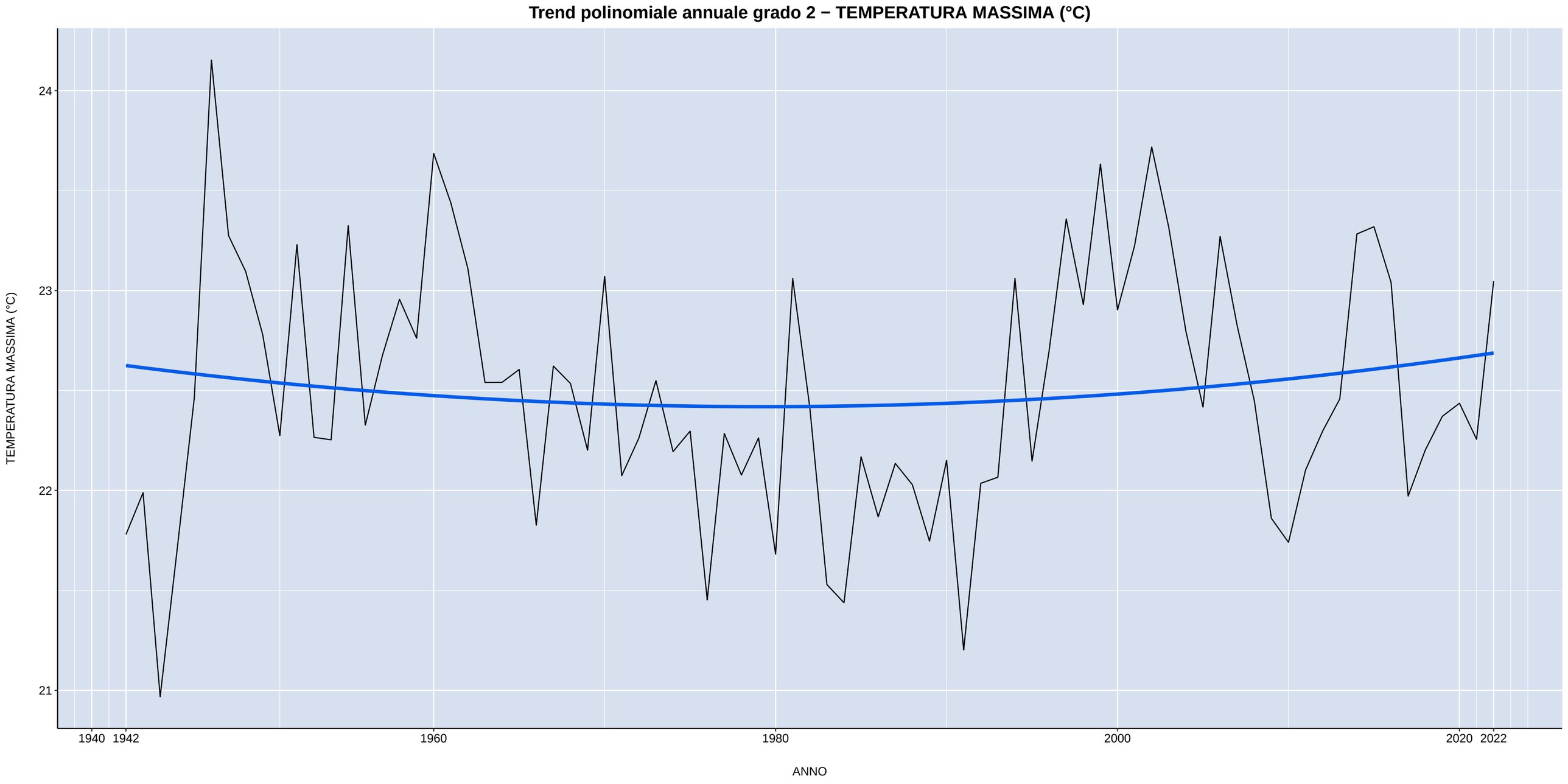

Serie mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
25
20
15
10
1940
1942
1960
1980
2000
2020
2022
ANNO


Trend lineare mensile − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
30
25
20
15
10
1940
1942
1960
1980
2000
2020
2022
ANNO

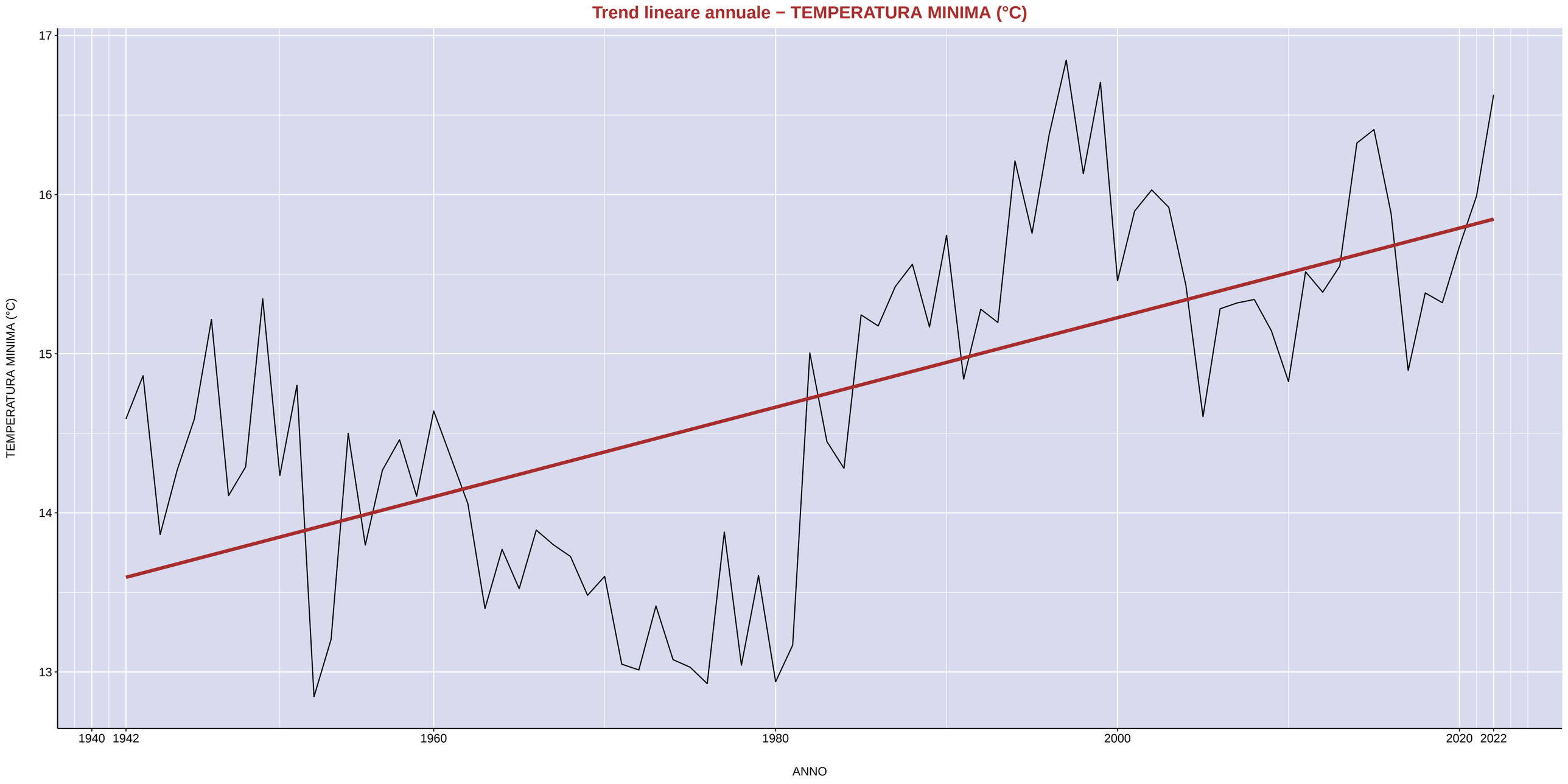

Trend polinomiale mensile grado 2 − TEMPERATURA MEDIA (°C)
30
25
20
15
10
TEMPERATURA MEDIA (°C)
1940
1942
1960
1980
2000
2020
2022
ANNO

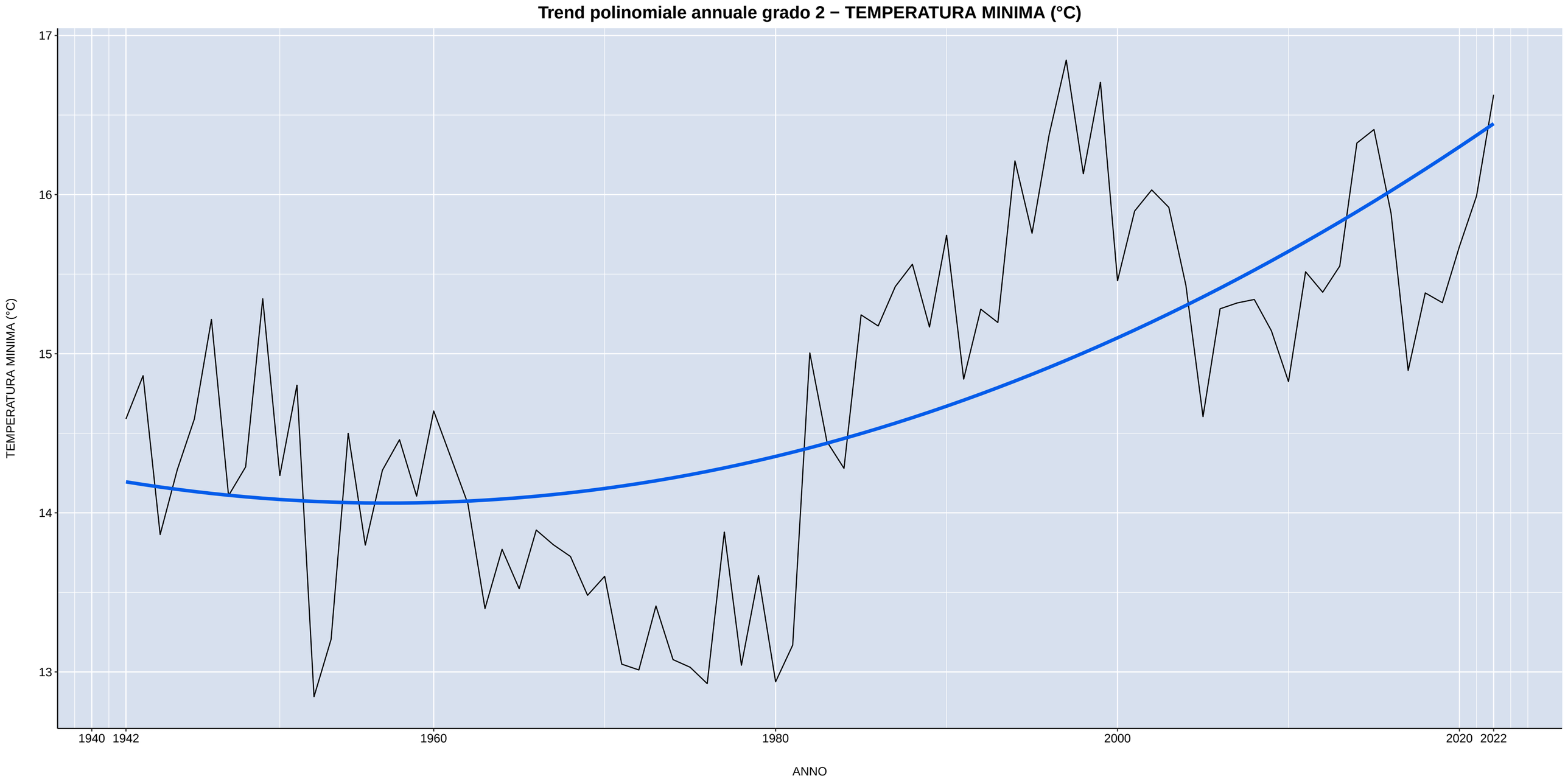

Serie annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
24
23
22
21
1940
1942
1960
1980
2000
2020
2022
ANNO

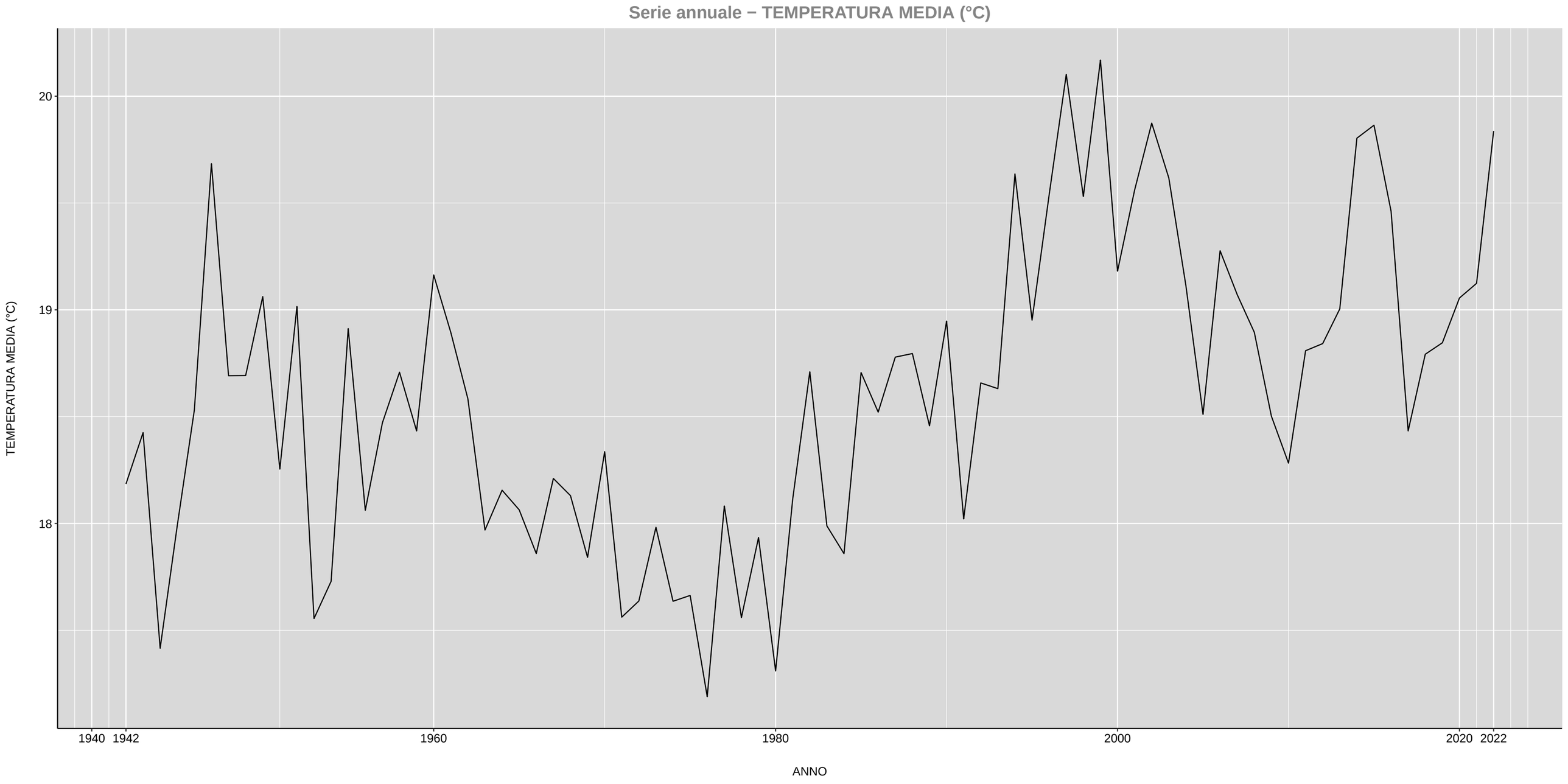

Trend lineare annuale − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
24
23
22
21
1940
1942
1960
1980
2000
2020
2022
ANNO

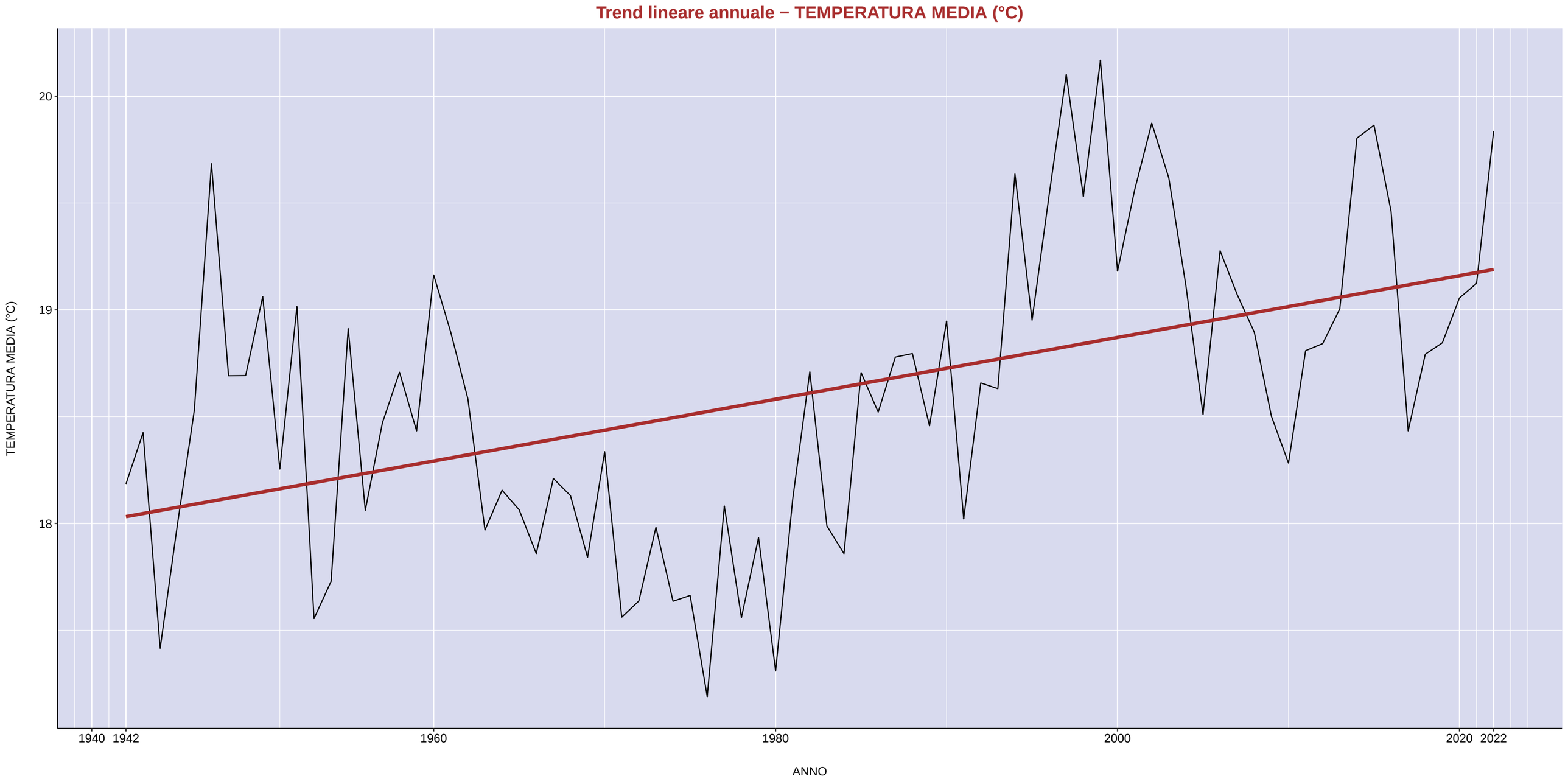

Trend polinomiale annuale grado 2 − TEMPERATURA MASSIMA (°C)
TEMPERATURA MASSIMA (°C)
24
23
22
21
1940
1942
1960
1980
2000
2020
2022
ANNO

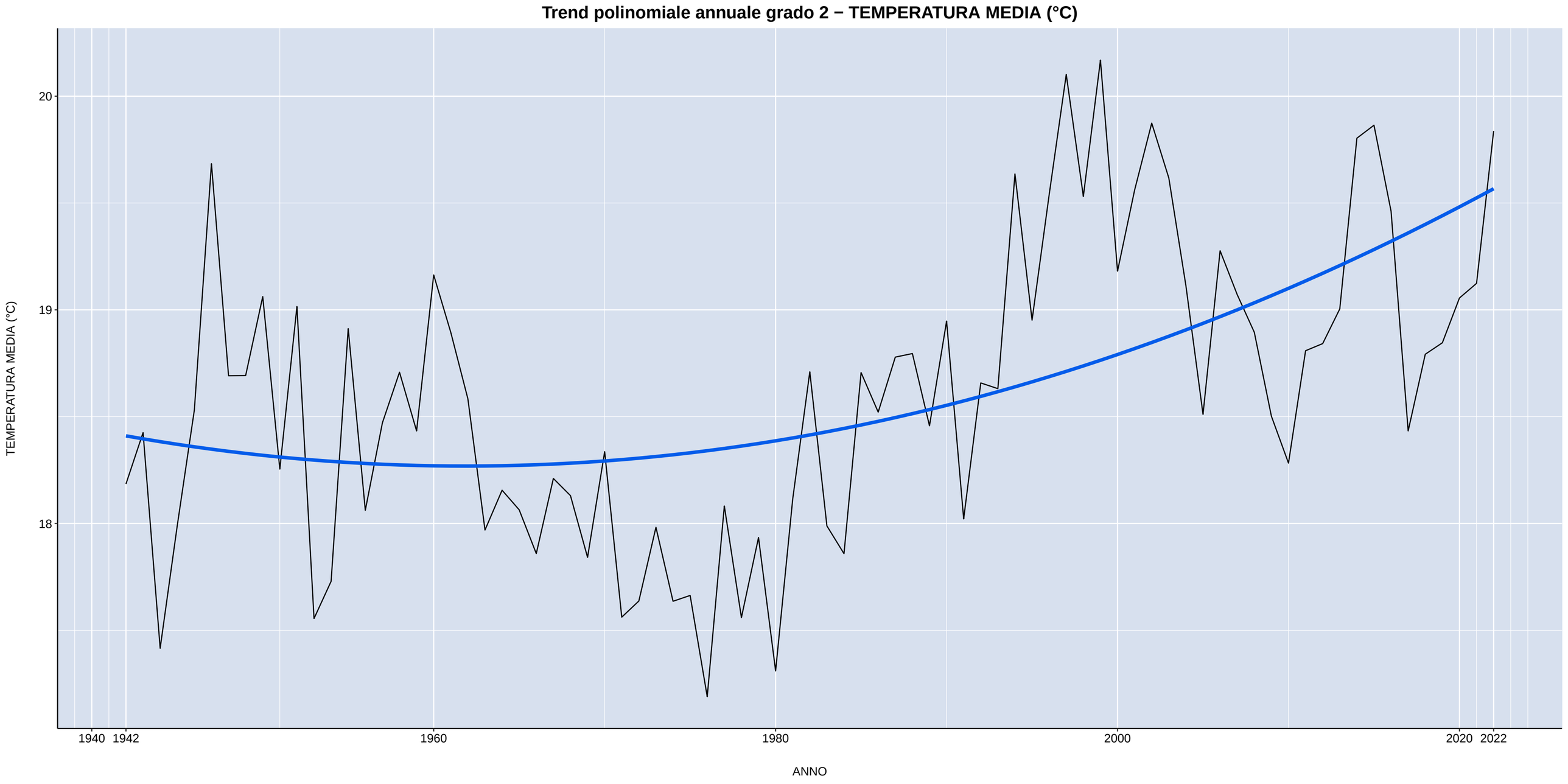
Serie annuale − TEMPERATURA MINIMA (°C)
17
16
15
14
13
TEMPERATURA MINIMA (°C)
1940
1942
1960
1980
2000
2020
2022
ANNO

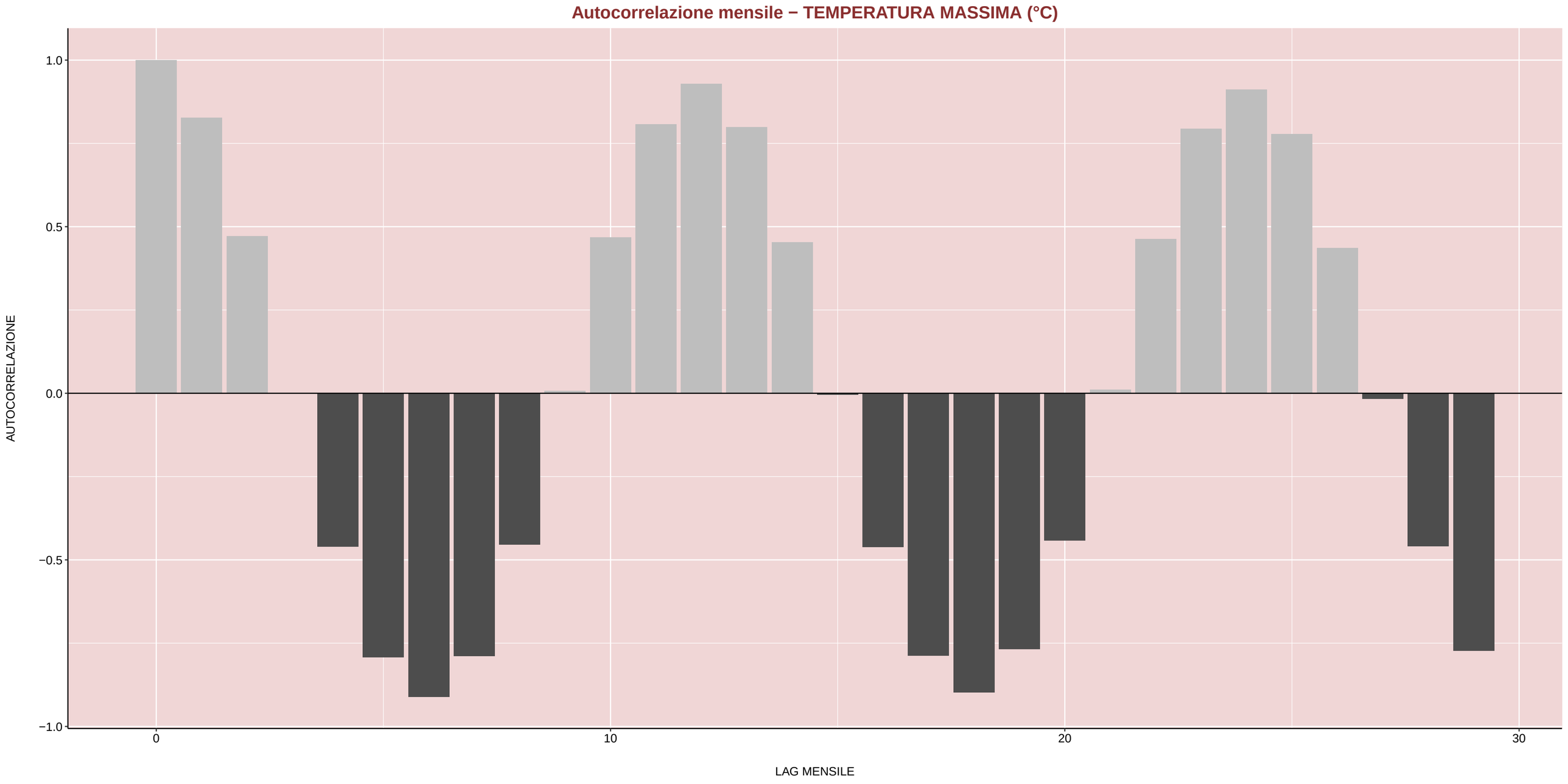

Trend lineare annuale − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
17
16
15
14
13
1940
1942
1960
1980
2000
2020
2022
ANNO

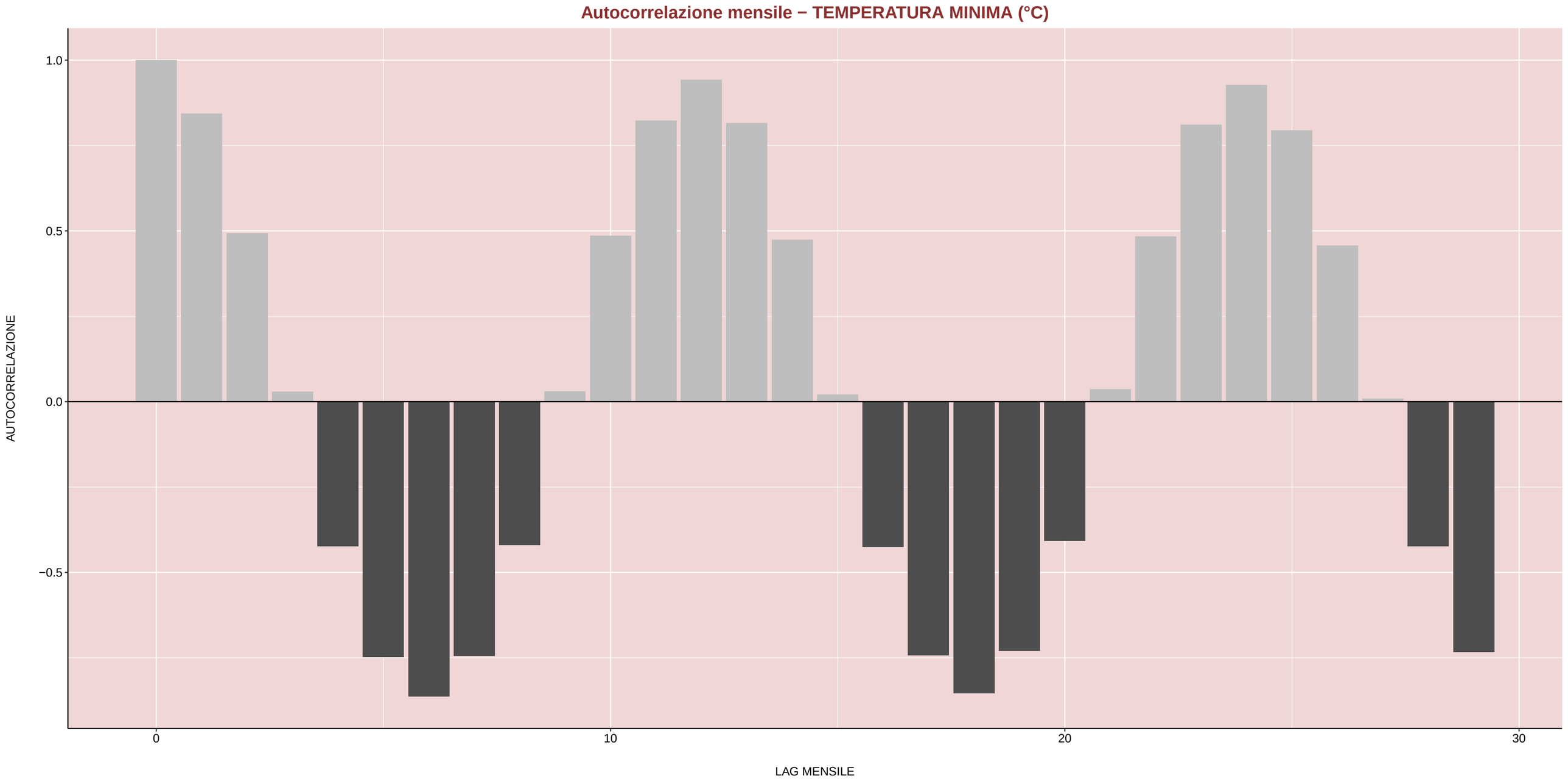

Trend polinomiale annuale grado 2 − TEMPERATURA MINIMA (°C)
TEMPERATURA MINIMA (°C)
17
16
15
14
13
1940
1942
1960
1980
2000
2020
2022
ANNO

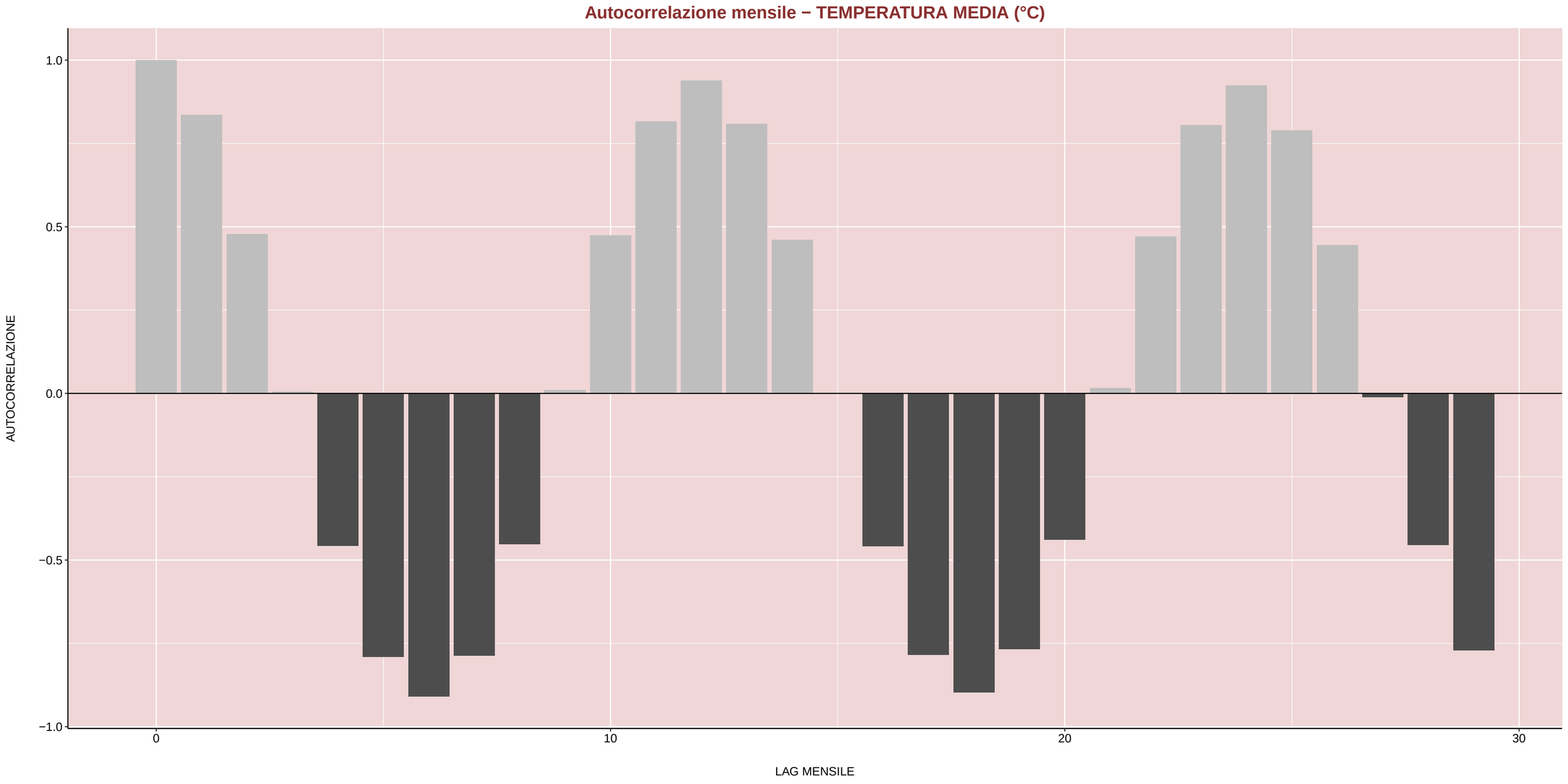

Serie annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
20
19
18
1940
1942
1960
1980
2000
2020
2022
ANNO

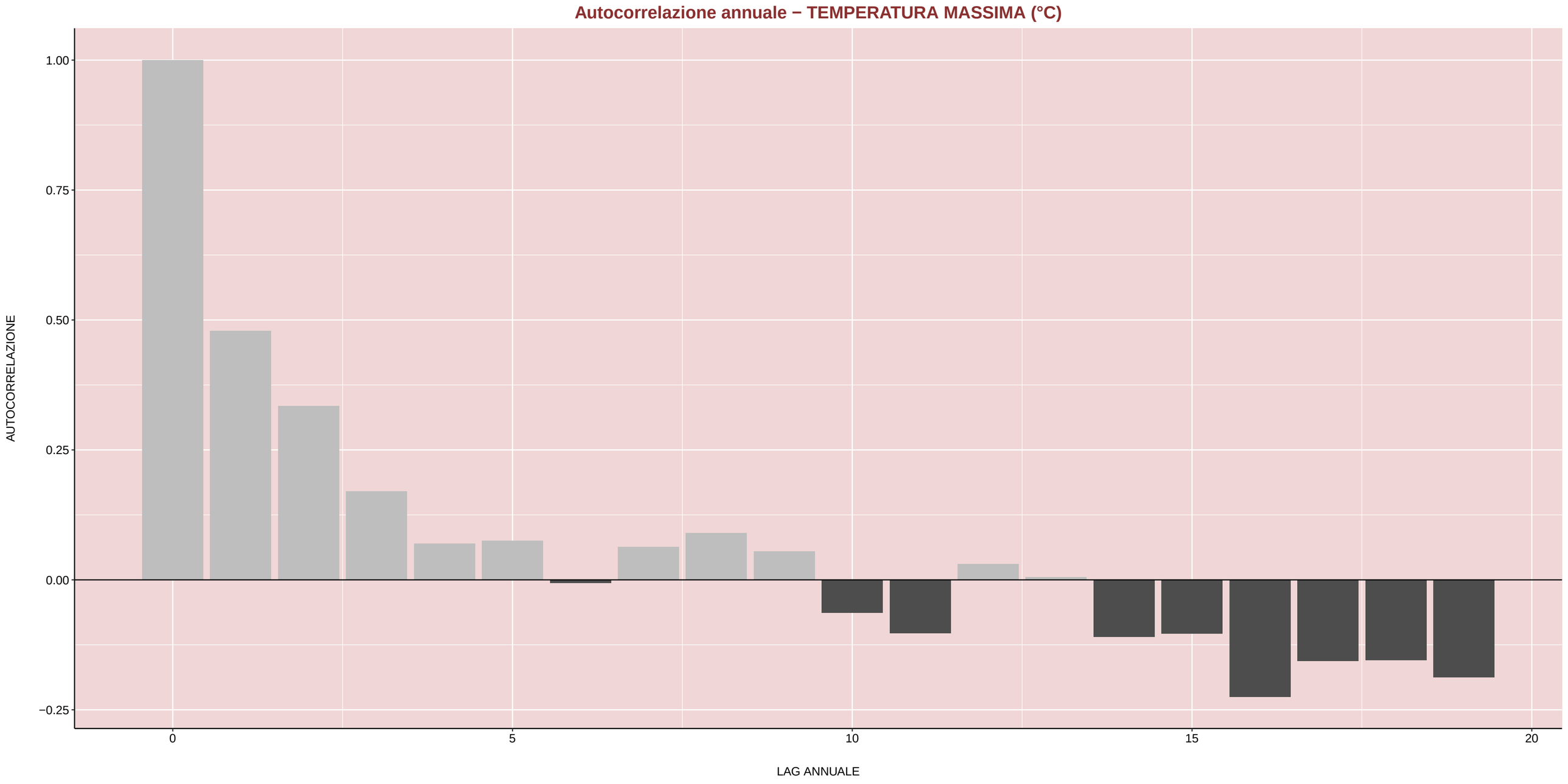

Trend lineare annuale − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
20
19
18
1940
1942
1960
1980
2000
2020
2022
ANNO

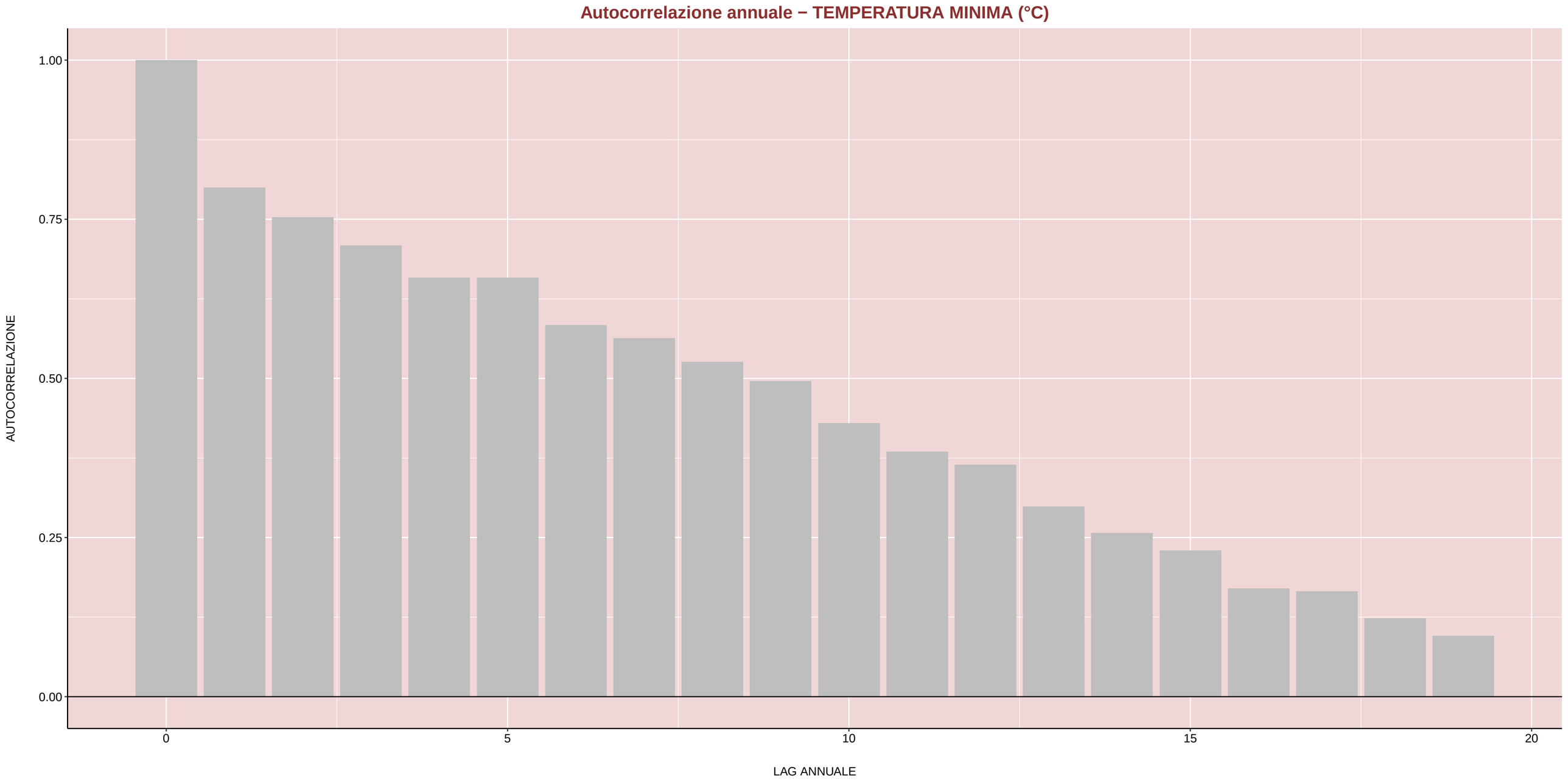

Trend polinomiale annuale grado 2 − TEMPERATURA MEDIA (°C)
TEMPERATURA MEDIA (°C)
20
19
18
1940
1942
1960
1980
2000
2020
2022
ANNO

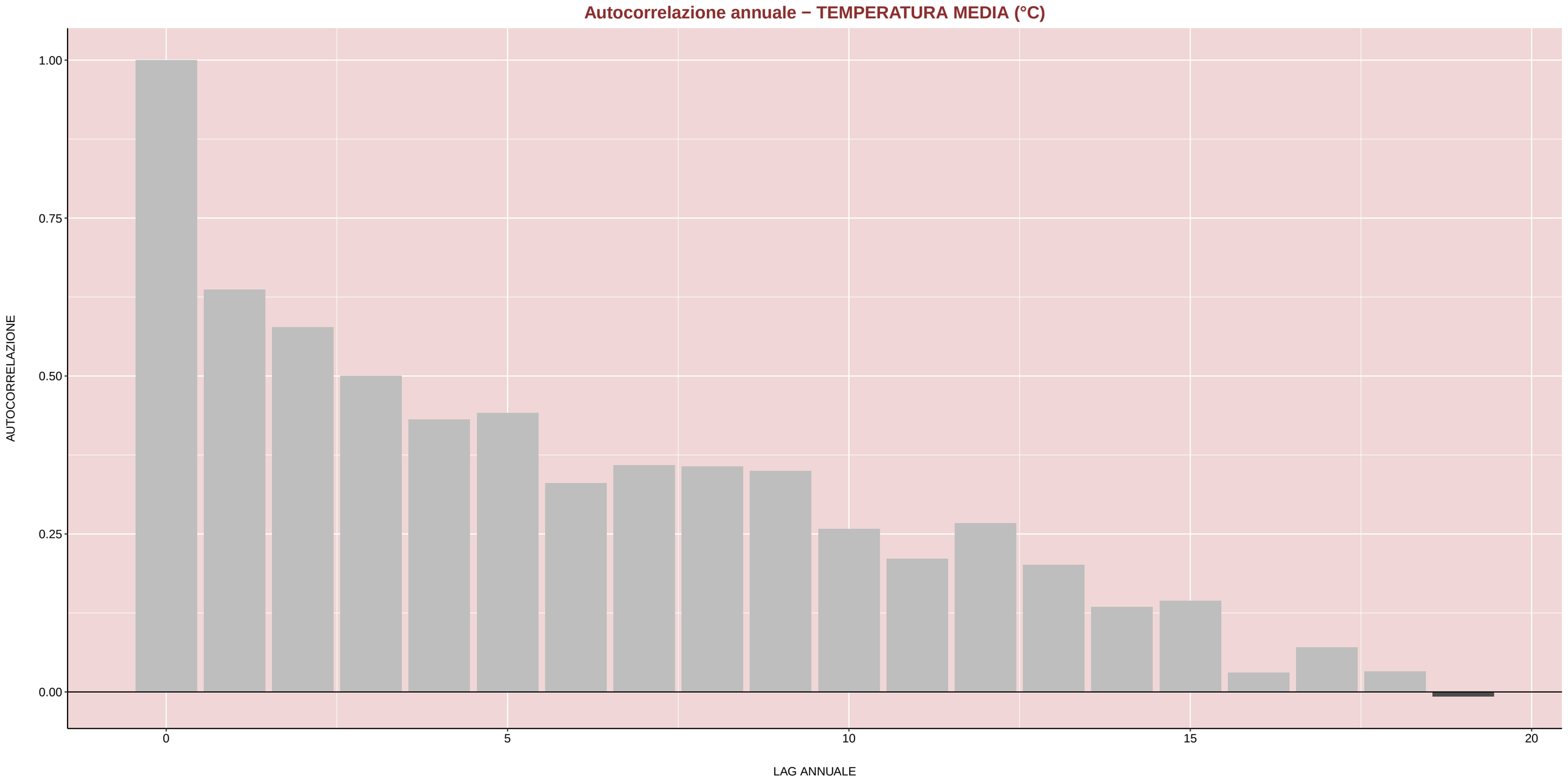

Autocorrelazione mensile − TEMPERATURA MASSIMA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
−1.0
0
10
20
30
LAG MENSILE


Autocorrelazione mensile − TEMPERATURA MINIMA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
0
10
20
30
LAG MENSILE


Autocorrelazione mensile − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.0
0.5
0.0
−0.5
−1.0
0
10
20
30
LAG MENSILE


Autocorrelazione annuale − TEMPERATURA MASSIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
−0.25
0
5
10
15
20
LAG ANNUALE


Autocorrelazione annuale − TEMPERATURA MINIMA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
20
LAG ANNUALE


Autocorrelazione annuale − TEMPERATURA MEDIA (°C)
AUTOCORRELAZIONE
1.00
0.75
0.50
0.25
0.00
0
5
10
15
20
LAG ANNUALE