Script R - Parte 1 - Thiago Menezes

1) Importação do conjunto de dados “dataset_st” no RStudio:

```
> dataset_st
# A tibble: 300 x 15
   mes_total   ano   mes biomassa  carvao eolica fotovoltaica     gas hidraulica multi_combustivel_diese…¹ nuclear
       <dbl> <dbl> <dbl>    <dbl>   <dbl>  <dbl>        <dbl>   <dbl>      <dbl>                     <dbl>   <dbl>
 1         1  2000     1   50756. 770966.      0            0 510250.  27711265.                         0  4.18e5
 2         2  2000     2   11439  725440.      0            0 580009.  26849928.                         0  4.00e5
 3         3  2000     3    3845. 685798.      0            0 513065.  28789384.                         0  4.00e5
 4         4  2000     4    3600  675602.      0            0 337403.  27810599.                         0  3.59e5
 5         5  2000     5    4403. 725875.      0            0 503225.  28813392.                         0  0
 6         6  2000     6    7537. 725002.      0            0 574332.  27376412.                         0  1.11e1
 7         7  2000     7    5148. 607498.      0            0 540093.  27679250.                         0  6.79e4
 8         8  2000     8    8189. 627498.      0            0 547118.  27812268.                         0  6.72e5
 9         9  2000     9   18808. 397360.      0            0 500285.  27482979.                         0  7.81e5
10        10  2000    10    3537. 337344.      0            0 551283.  30034719.                         0  5.45e5
# i 290 more rows
# i abbreviated name: 'multi_combustivel_diesel_oleo
# i 4 more variables: oleo_combustivel <dbl>, oleo_diesel <dbl>, outras_multi_combustivel <dbl>,
#   residuos_industriais <dbl>
# i Use `print(n = ...)` to see more rows
```

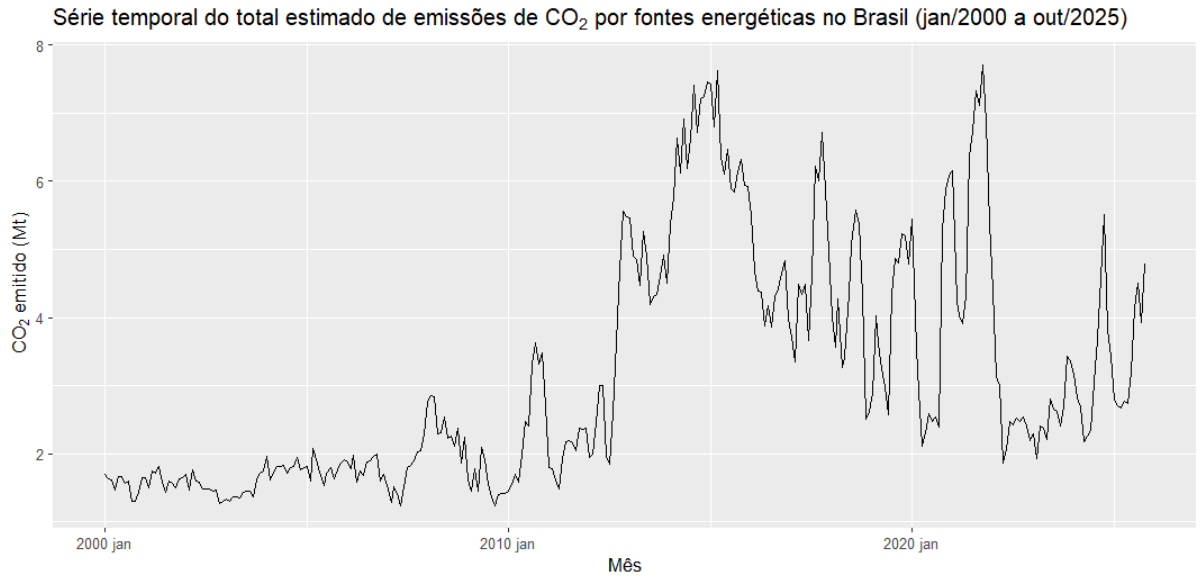
2) Conversão dos dados para em um objeto tsibble:

```
install.packages("dplyr")
install.packages("tsibble")
install.packages("fabletools")
install.packages("ggplot2")

library(dplyr)
library(tsibble)
library(fabletools)
library(ggplot2)

dataset_st <- dataset_st |>
  mutate(mes = yearmonth(mes)) |>
  as_tsibble(index = mes)

dataset_st
```

```
> dataset_st
# A tsibble: 310 x 16 [1M]
       mes co2_mt biomassa  carvao eolica fotovoltaica     gas hidraulica multi_combustivel_diesel_oleo  nuclear
     <mth>  <dbl>    <dbl>   <dbl>  <dbl>        <dbl>   <dbl>      <dbl>                         <dbl>    <dbl>
 1 2000 jan   1.70   50756. 770966.      0            0 510250.  27711265.                             0 418078.
 2 2000 fev   1.64   11439  725440.      0            0 580009.  26849928.                             0 400043.
 3 2000 mar   1.61    3845. 685798.      0            0 513065.  28789384.                             0 400187.
 4 2000 abr   1.46    3600  675602.      0            0 337403.  27810599.                             0 358886.
 5 2000 mai   1.66    4403. 725875.      0            0 503225.  28813392.                             0      0
 6 2000 jun   1.67    7537. 725002.      0            0 574332.  27376412.                             0     11.1
 7 2000 jul   1.56    5148. 607498.      0            0 540093.  27679250.                             0  67881
 8 2000 ago   1.59    8189. 627498.      0            0 547118.  27812268.                             0 672391.
 9 2000 set   1.30   18808. 397360.      0            0 500285.  27482979.                             0 781196.
10 2000 out   1.30    3537. 337344.      0            0 551283.  30034719.                             0 544916.
# i 300 more rows
# i 6 more variables: oleo_combustivel <dbl>, oleo_diesel <dbl>, grupo_befn <dbl>, grupo_cg <dbl>,
#   grupo_hidraulica <dbl>, grupo_o <dbl>
# i Use `print(n = ...)` to see more rows
```

3) Geração do gráfico da emissão de CO2:

```
dataset_st |>
  autoplot(co2_mt) +
  labs(
    y = expression("CO"[2] * " emitido (Mt)"),
    x = "Mês",
     title = expression("Série temporal do total estimado de emissões de CO"[2] * " por fontes energéticas no Brasil (jan/2000 a out/2025)"
  ))
```



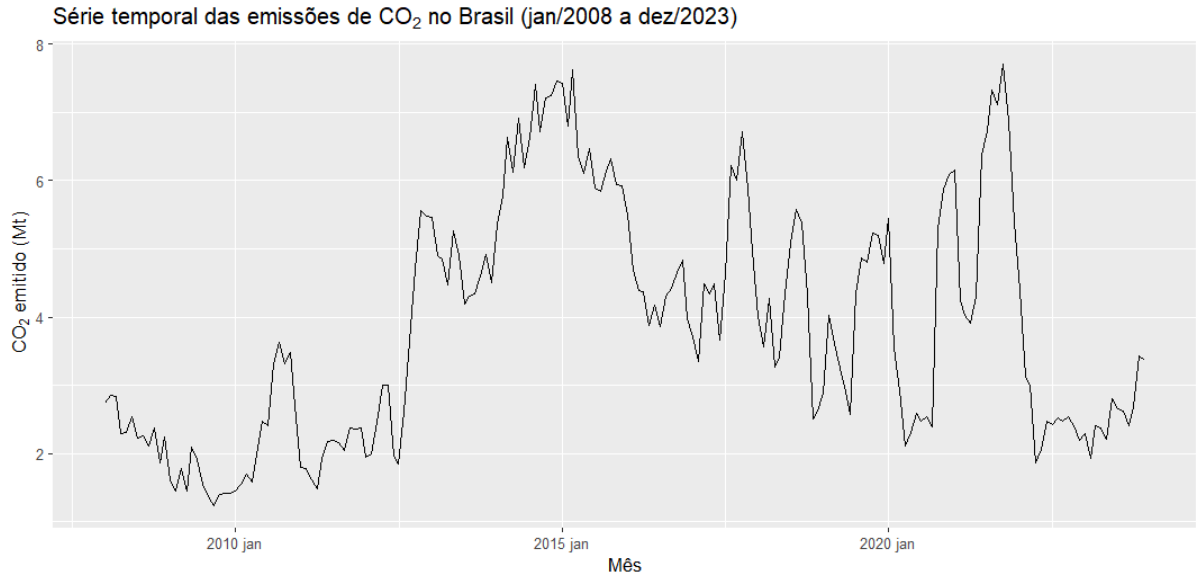
4) Divisão dos dados em treinamento e teste:

Treino: de jan/2008 a dez/2023

Teste: de jan/2024 a out/2025

```
library(tsibble)
library(dplyr)
library(ggplot2)
library(feasts)  # autoplot() para tsibbles

dataset_st_treino_2008_2023 <- dataset_st |>
  filter_index("2008 Jan" ~ "2023 Dec")

dataset_st_teste_2024_25 <- dataset_st |>
  filter_index("2024 Jan" ~ "2025 Oct")
```

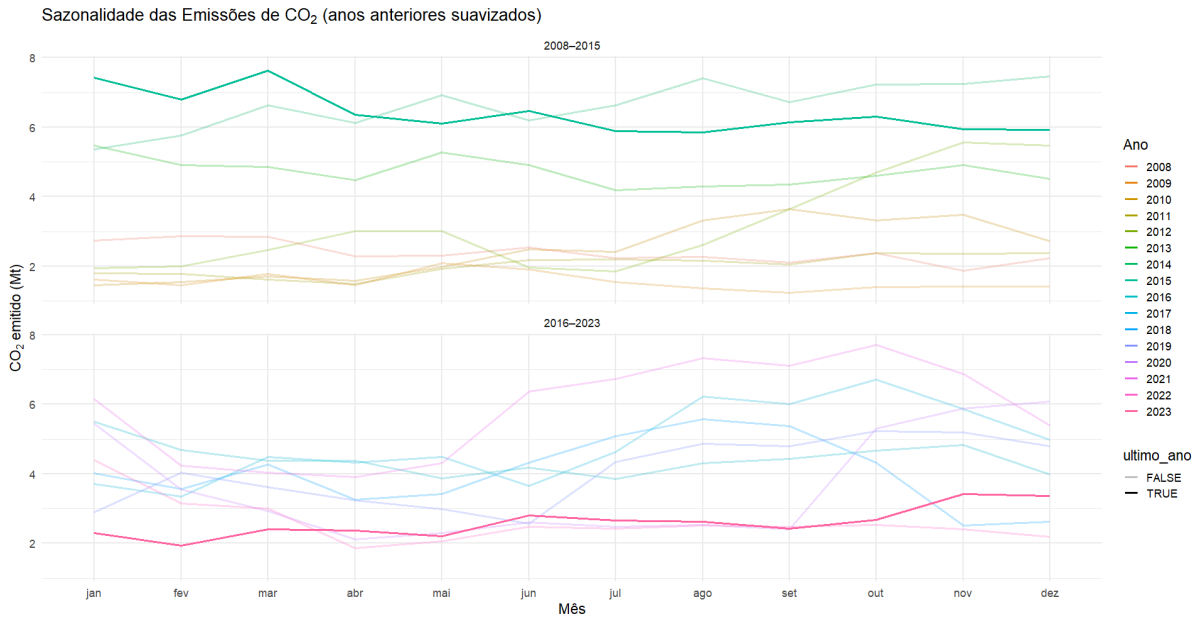
```
# Plotar apenas o conjunto de treino
dataset_st_treino_2008_2023 |>
  autoplot(co2_mt) +
  labs(
      title = expression("Série temporal das emissões de CO"[2] * " no Brasil (jan/2008 a dez/2023)"),
    y = expression("CO"[2] * " emitido (Mt)"),
    x = "Mês"
  )
```



4) Geração de gráfico para avaliar sazonalidade:

```
library(dplyr)
library(lubridate)
library(feasts)
library(ggplot2)

dataset_st_treino_2008_2023 |>
    mutate(
        ano   = year(mes),
        grupo = case_when(
            ano >= 2008 & ano <= 2015 ~ "2008–2015",
            ano >= 2016 & ano <= 2023 ~ "2016–2023",
            TRUE ~ NA_character_
        )
    ) |>
    filter(!is.na(grupo)) |>
    mutate(ano_fator = factor(ano)) |>
    group_by(grupo) |>
    mutate(
```

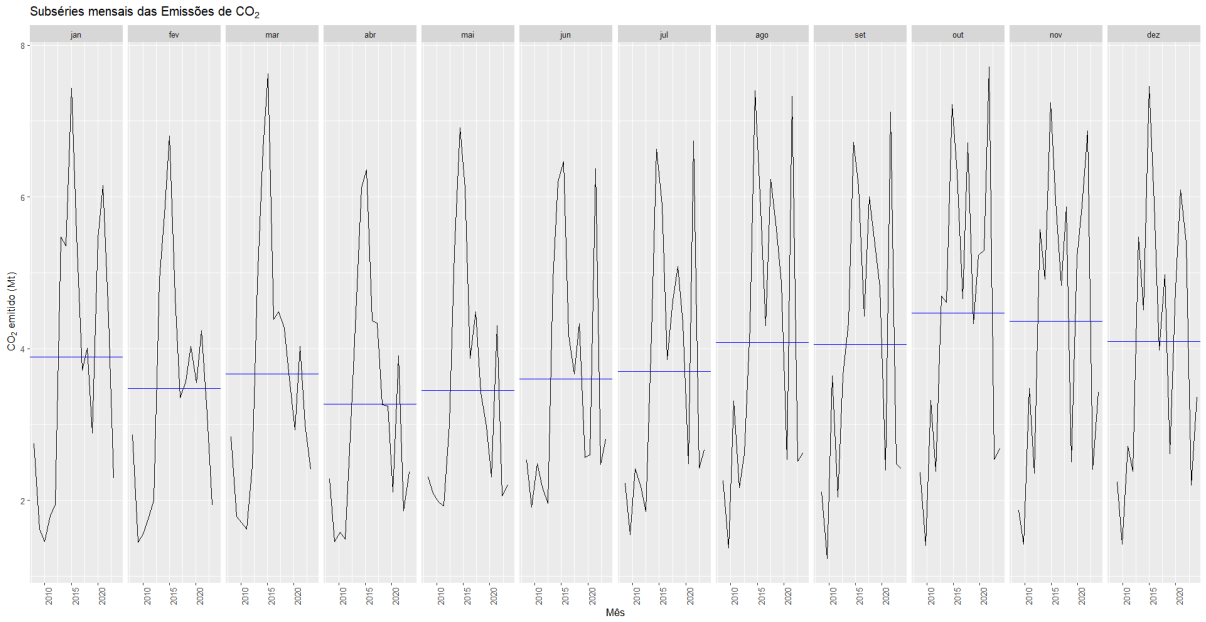
```
    ultimo_ano = (ano == max(ano))  # marca o ano mais recente em cada painel
  ) |>
  ungroup() |>
  ggplot(aes(x = month(mes, label = TRUE, abbr = TRUE), y = co2_mt,
             group = ano_fator, color = ano_fator, alpha = ultimo_ano)) +

  geom_line(linewidth = 1) +

  scale_alpha_manual(values = c(`FALSE` = 0.25, `TRUE` = 1)) +
  labs(
    y = expression("CO"[2] * " emitido (Mt)"),
    x = "Mês",
    title = expression("Sazonalidade das Emissões de CO"[2] * " (anos anteriores suavizados)"),
    color = "Ano"
  ) +
  facet_wrap(~ grupo, ncol = 1, scales = "fixed") +   # <-- ALTERAÇÃO AQUI
  theme_minimal(base_size = 15) +
  theme(legend.position = "right")
```



5) Geração de gráficos sazonais para subséries:

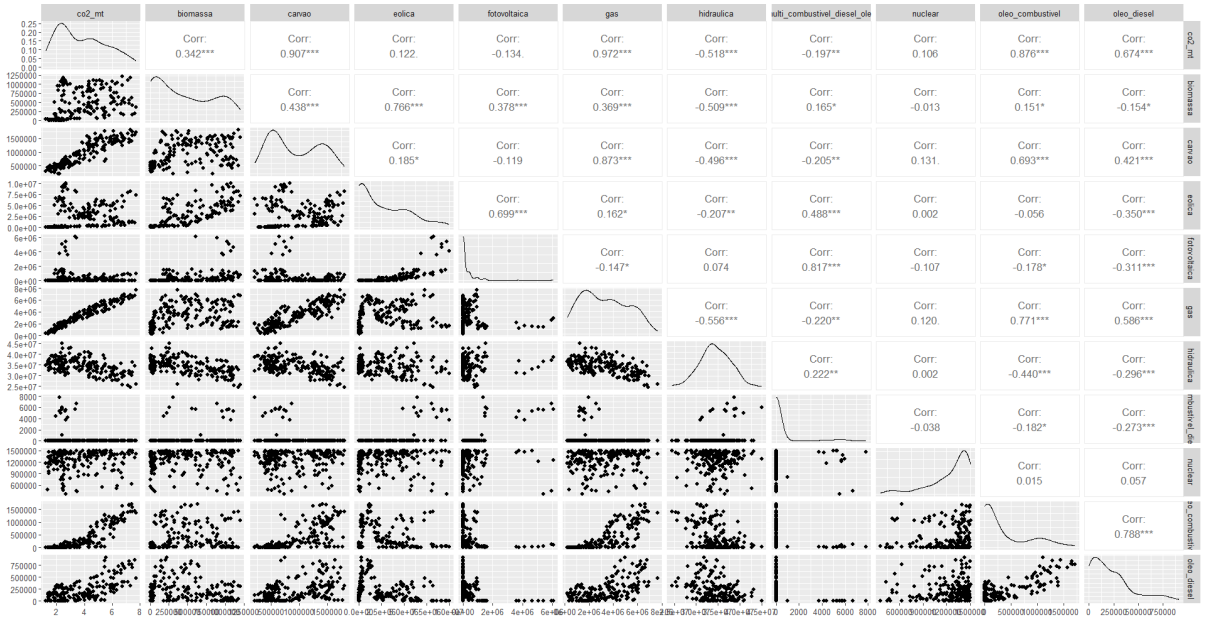
```
dataset_st_treino_2008_2023 |>
  gg_subseries(co2_mt) +
  labs(
    y = expression("CO"[2] * " emitido (Mt)"),
    x = "Mês",
    title = expression("Subséries mensais das Emissões de CO"[2] * ""
```

))



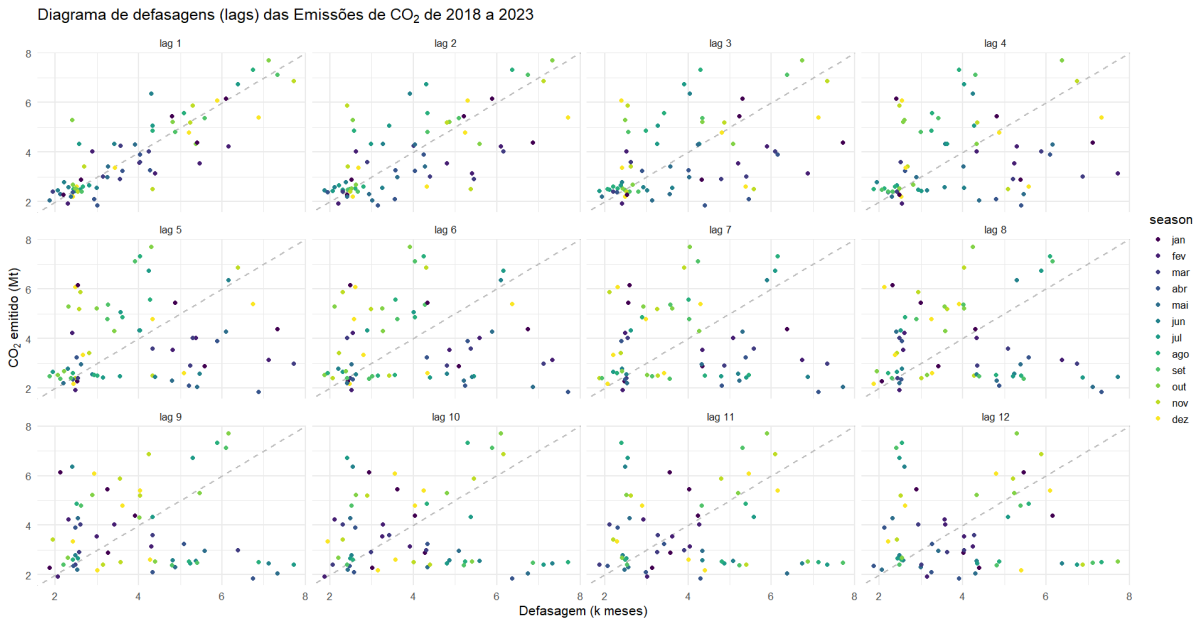
6) Gráficos de dispersão em painel:

```
library(dplyr)
library(GGally)

dataset_tbl <- tibble::as_tibble(dataset_st_treino_2008_2023)

num_cols <- dataset_tbl |> select(where(is.numeric))

num_cols |>
  tidyr::drop_na() |>
  GGally::ggpairs(progress = FALSE)
```



7) Gráficos para as defasagens (lags) de 2018 em diante:

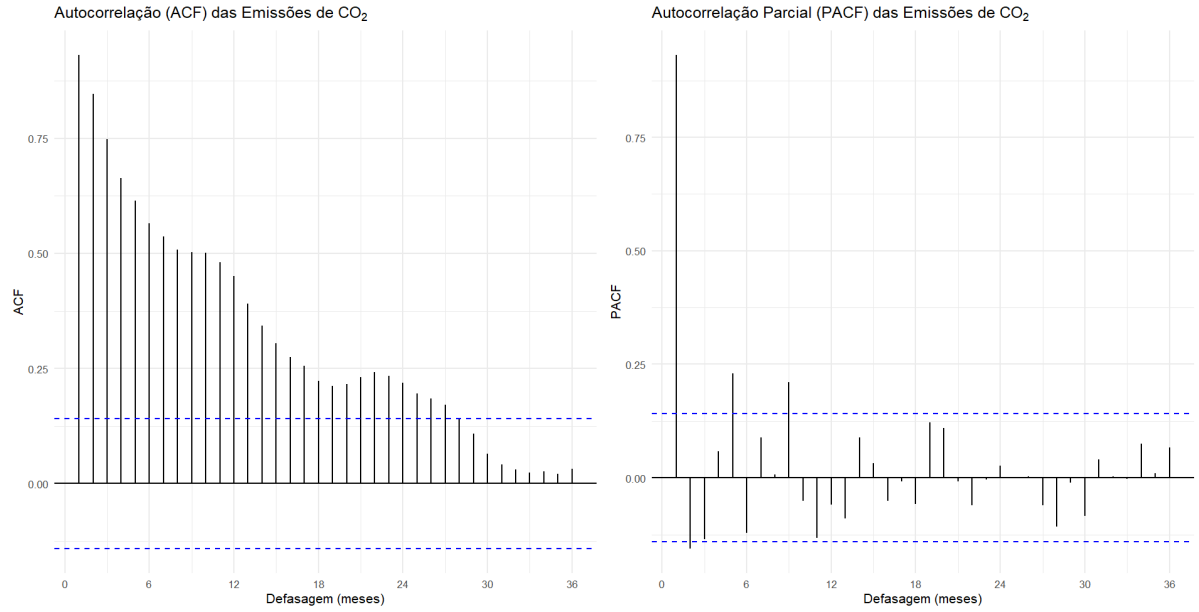
```
library(lubridate)
library(feasts)
library(ggplot2)

recent_emissions <- dataset_st_treino_2008_2023 |>
  dplyr::filter(year(mes) >= 2018)

recent_emissions |>
  gg_lag(co2_mt, lags = 1:12, geom = "point") +
  labs(
    x = "Defasagem (k meses)",
    y = expression("CO"[2] * " emitido (Mt)"),
     title = expression("Diagrama de defasagens (lags) das Emissões de CO"[2] * " de 2018 a
2023"
  )) +
  theme_minimal(base_size = 14)
```



8) Autocorrelação - Tendência e sazonalidade em gráficos ACF e PACF:

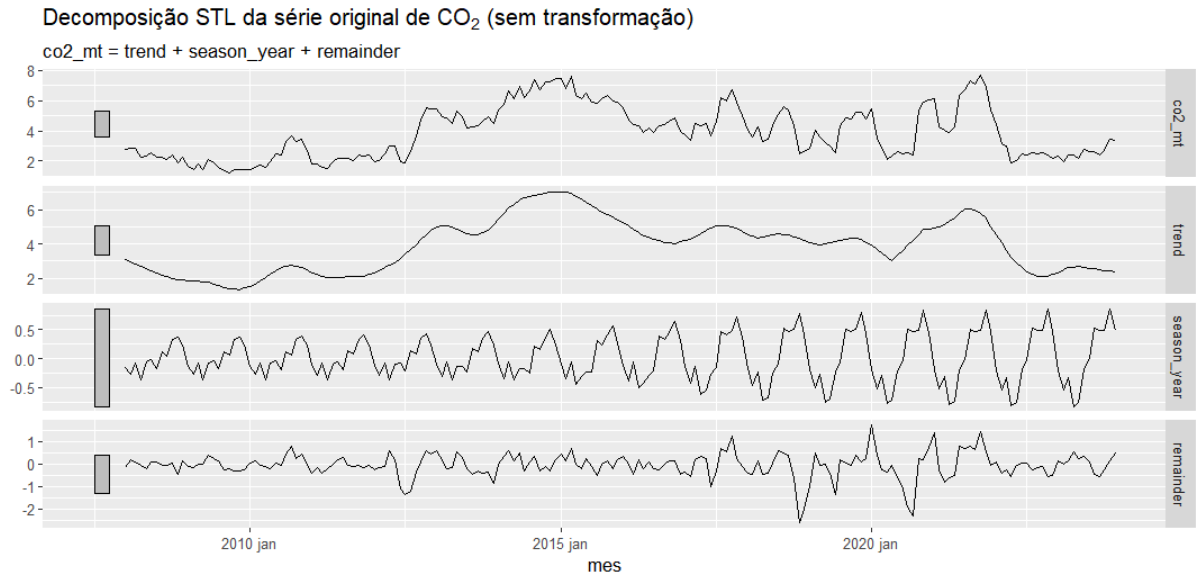
```
library(fable)
library(feasts)
library(ggplot2)
library(patchwork)

# ACF
p_acf <- dataset_st_treino_2008_2023 |>
  ACF(co2_mt, lag_max = 36) |>
  autoplot() +
  labs(
    title = expression("Autocorrelação (ACF) das Emissões de CO"[2]),
    x = "Defasagem (meses)",
    y = "ACF"
  ) +
  theme_minimal(base_size = 14)

# PACF
p_pacf <- dataset_st_treino_2008_2023 |>
  PACF(co2_mt, lag_max = 36) |>
  autoplot() +
  labs(
    title = expression("Autocorrelação Parcial (PACF) das Emissões de CO"[2]),
    x = "Defasagem (meses)",
    y = "PACF"
  ) +
  theme_minimal(base_size = 14)
```

```
# Mostrar lado a lado com patchwork
p_acf | p_pacf
```



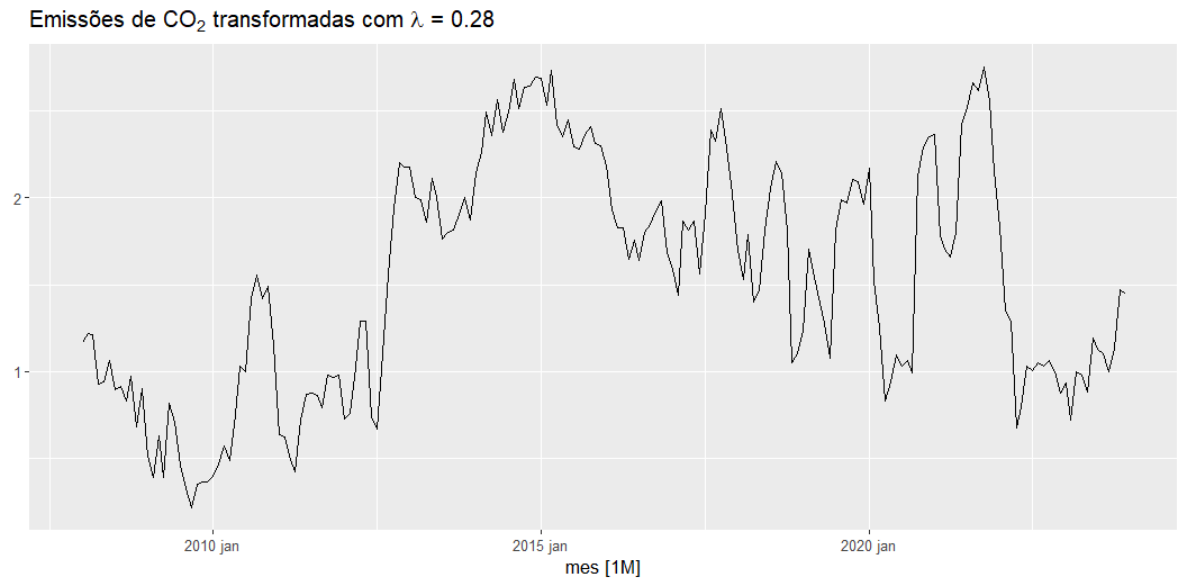
9) STL da série original:

```
library(fabletools)
library(feasts)
library(dplyr)
library(ggplot2)

# STL na SÉRIE ORIGINAL (sem Box–Cox)
dataset_st_treino_2008_2023 |>
  model(
    STL(co2_mt ~ trend(window = 12) +
                 season(window = 12),
        robust = TRUE)
  ) |>
  components() |>
  autoplot() +
  labs(
          title = expression("Decomposição STL da série original de CO"[2] * " (sem transformação)",
    y = ""
  ))
```

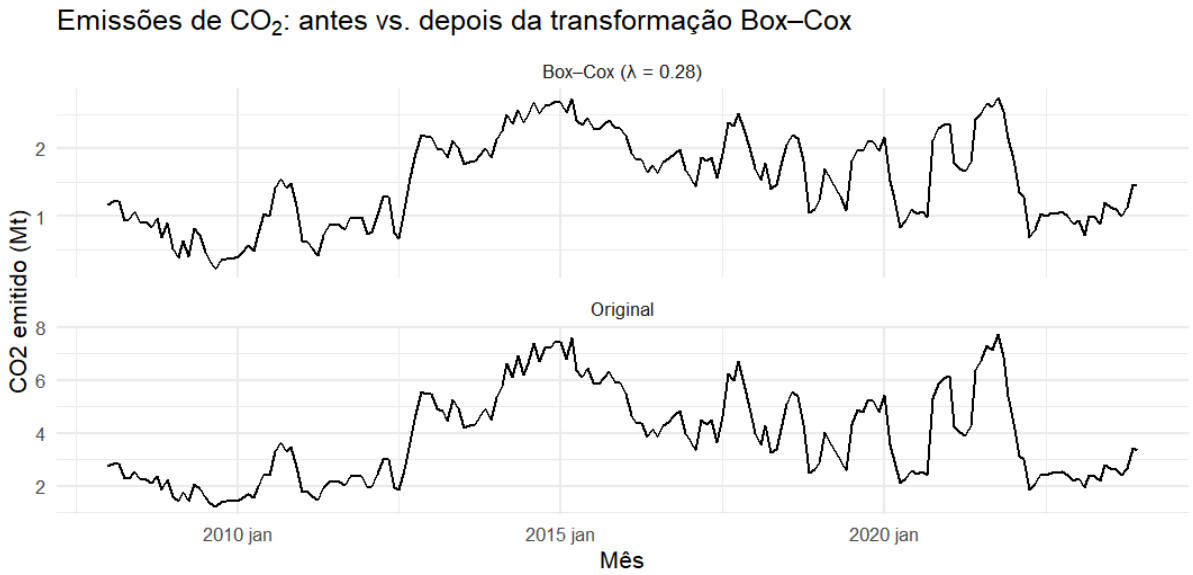


10) Transformação box-cox e ajustes:

```
library(fabletools)
library(feasts)
library(latex2exp)
library(ggplot2)

lambda <- dataset_st_treino_2008_2023 |>
  features(co2_mt, features = guerrero) |>
  pull(lambda_guerrero)
dataset_st_treino_2008_2023 |>
  autoplot(box_cox(co2_mt, lambda)) +
  labs(y = "",
       title = latex2exp::TeX(paste0(
         "Emissões de CO$_2$ transformadas com $\\lambda$ = ",
         round(lambda,2))))
```

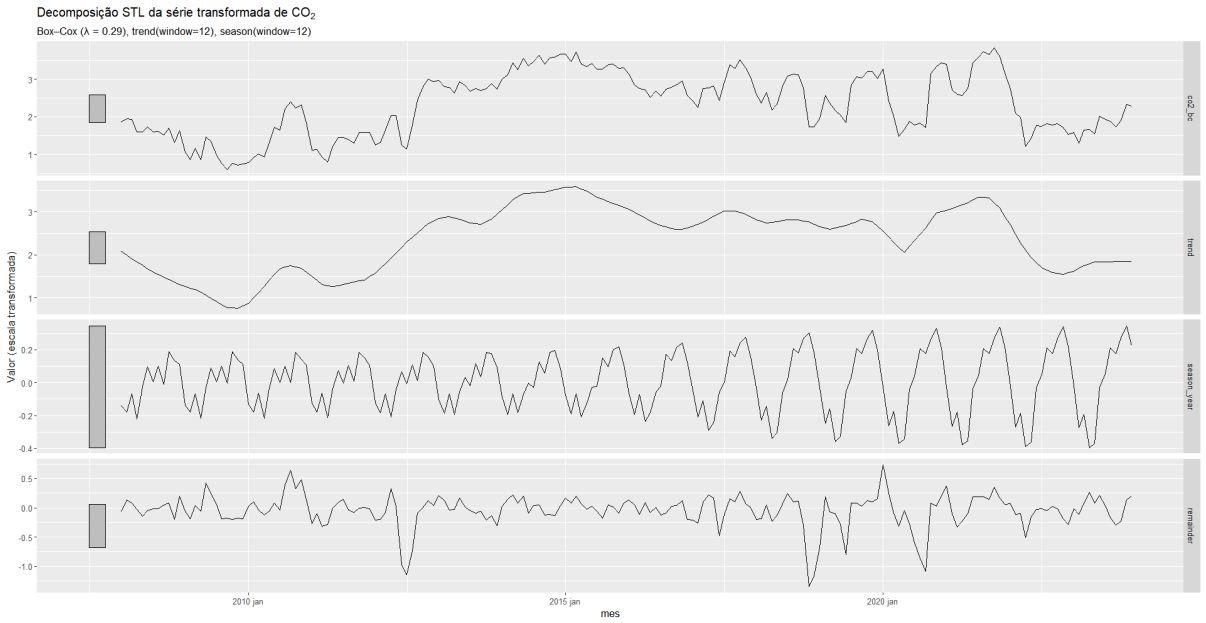



```
library(dplyr)
library(tidyr)
library(fabletools)
library(feasts)
library(ggplot2)

# 1) Estima λ (Guerrero)
lambda <- dataset_st_treino_2008_2023 |>
  features(co2_mt, features = guerrero) |>
  pull(lambda_guerrero)

# 2) Cria série transformada e organiza para facetas
plot_df <- dataset_st_treino_2008_2023 |>
  mutate(co2_bc = box_cox(co2_mt, lambda)) |>
  pivot_longer(c(co2_mt, co2_bc),
               names_to = "serie", values_to = "valor") |>
  mutate(serie = dplyr::recode(
    serie,
    co2_mt = "Original",
    co2_bc = paste0("Box–Cox (λ = ", round(lambda, 2), ")")
  ))

# 3) Gráfico comparativo (painéis verticalizados)
ggplot(plot_df, aes(x = mes, y = valor)) +
  geom_line() +
  facet_wrap(~ serie, ncol = 1, scales = "free_y") +
  labs(
    x = "Mês",
    y = "CO2 emitido (Mt)",
    title = expression("Emissões de CO"[2] * ": antes vs. depois da transformação Box–Cox"
  )) +
  theme_minimal(base_size = 14)
```



11) STL da transformada:

```
library(dplyr)
library(tsibble)
library(fable)
library(fabletools)
library(feasts)
library(ggplot2)

# 1) Criar a coluna transformada (co2_bc) no tsibble de treino
dataset_train_bc <- dataset_st_treino_2008_2023 |>
  mutate(co2_bc = box_cox(co2_mt, lambda))

# 2) Ajustar STL na série transformada (co2_bc)
stl_bc_mod <- dataset_train_bc |>
  model(stl = STL(co2_bc ~ trend(window = 12) + season(window = 12), robust = TRUE))

# 3) Extrair componentes
stl_bc_comp <- stl_bc_mod |> components()

# 4) Plot dos componentes na escala transformada
stl_bc_comp |>
  autoplot() +
  labs(
    title = expression("Decomposição STL da série transformada de CO"[2]),
          subtitle = paste0("Box–Cox (λ = ", round(lambda, 2), "), trend(window=12),
season(window=12)"),
    y = "Valor (escala transformada)"
  )
```



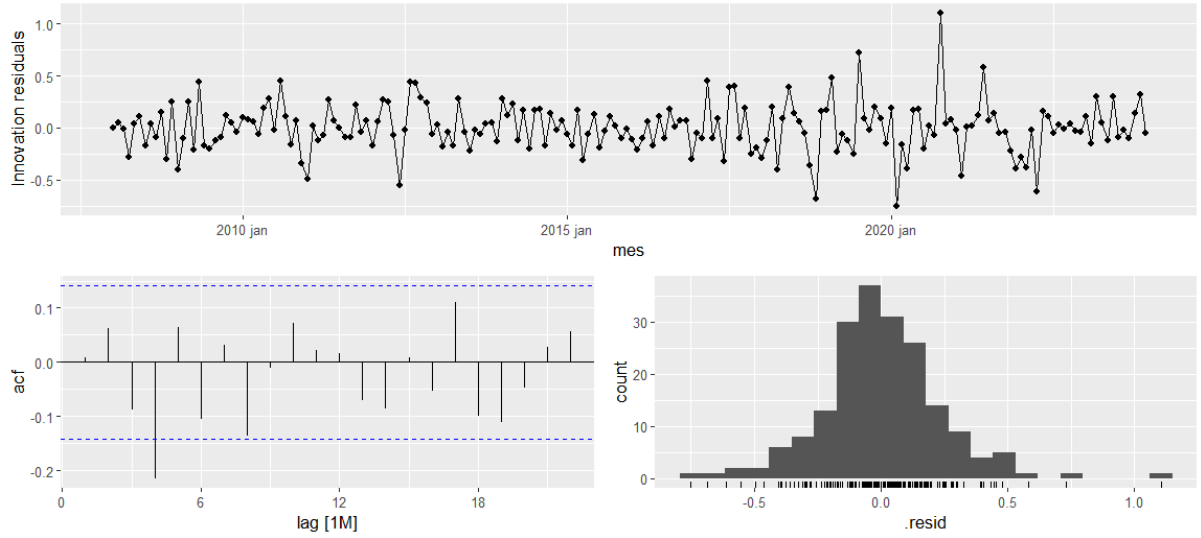
12) Ajuste do modelo SARIMA automático:

```
library(fable)
library(fabletools)
library(feasts)
library(tsibble)
library(dplyr)
library(ggplot2)
library(lubridate)

lambda_val <- lambda

dataset_original <- dataset_st_treino_2008_2023 |> as_tsibble(index = mes)

has_test <- exists("dataset_st_teste_2024_25")
if (has_test) {
  dataset_test <- dataset_st_teste_2024_25 |> as_tsibble(index = mes)
          }

# ---- AJUSTE DO MODELO (Box–Cox dentro do model) ----
fit_sarima <- dataset_original |>
  model(ARIMA(box_cox(co2_mt, lambda_val)))

# relatório do modelo (parâmetros, AIC, etc.)
report(fit_sarima)
```

```
> report(fit_sarima)
Series: co2_mt
Model: ARIMA(0,1,1)(0,0,1)[12]
Transformation: box_cox(co2_mt, lambda_val)

Coefficients:
          ma1     sma1
      0.0980  0.1689
s.e.  0.0672  0.0688

sigma^2 estimated as 0.0569:  log likelihood=3.56
AIC=-1.12   AICc=-1   BIC=8.63
```

13) Diagnóstico dos Resíduos e Ljung Box:

```
# resíduos (escala transformada — diagnóstico válido)
fit_sarima |> gg_tsresiduals()

# Ljung-Box nos resíduos (lag 24)
fit_sarima |>
  augment() |>
  features(.innov, ljung_box, lag = 24)
```

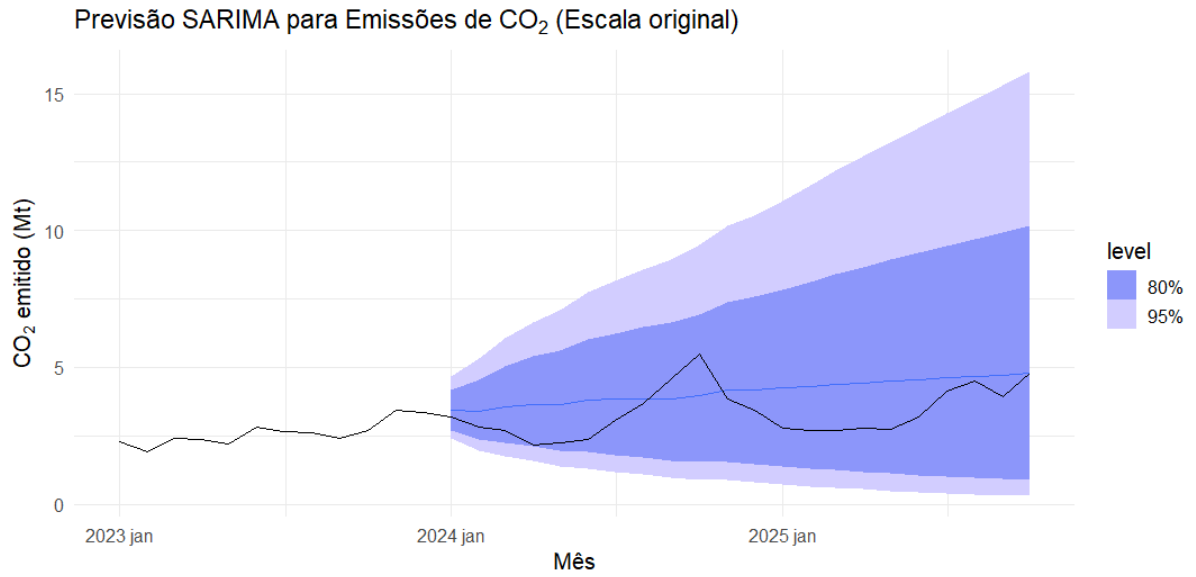



```
+     features(.innov, ljung_box, lag = 24)
# A tibble: 1 x 3
  .model lb_stat lb_pvalue
  <chr>    <dbl>     <dbl>
1 SARIMA    33.6    0.0924
```

```
# ---- PREVISÃO (fable irá retransformar automaticamente para a escala original) ----
h <- if (has_test) nrow(dataset_test) else 22
fc_sarima <- fit_sarima |> forecast(h = h)

# preview das previsões (ver colunas .mean, .lower/.upper etc.)
print(as_tibble(fc_sarima) %>% slice_head(n = 6))
```

```
> print(as_tibble(fc_sarima) %>% slice_head(n = 6))
# A tibble: 6 × 4
  .model      mes           co2_mt .mean
  <chr>     <mth>           <dist> <dbl>
1 SARIMA 2024 jan t(N(1.5, 0.057))  3.43
2 SARIMA 2024 fev  t(N(1.4, 0.13))  3.41
3 SARIMA 2024 mar  t(N(1.5, 0.19))  3.57
4 SARIMA 2024 abr  t(N(1.5, 0.26))  3.65
5 SARIMA 2024 mai  t(N(1.5, 0.33))  3.65
6 SARIMA 2024 jun   t(N(1.5, 0.4))  3.81
```

```
# ---- PLOT: previsões na escala original sobre os dados originais ----
# usa dataset_original (treino) ou dataset_full (treino + teste) para visualização
if (has_test) {
  dataset_full <- bind_rows(dataset_original, dataset_test) |> as_tsibble(index = mes)
} else {
  dataset_full <- dataset_original
}

fc_sarima |> autoplot(dataset_full |> filter_index("2023 Jan" ~ .)) +
  labs(
    title = expression("Previsão SARIMA para Emissões de CO"[2] * " (Escala original)"),
    y = expression("CO"[2] * " emitido (Mt)"),
    x = "Mês"
  ) +
  theme_minimal(base_size = 13)
```



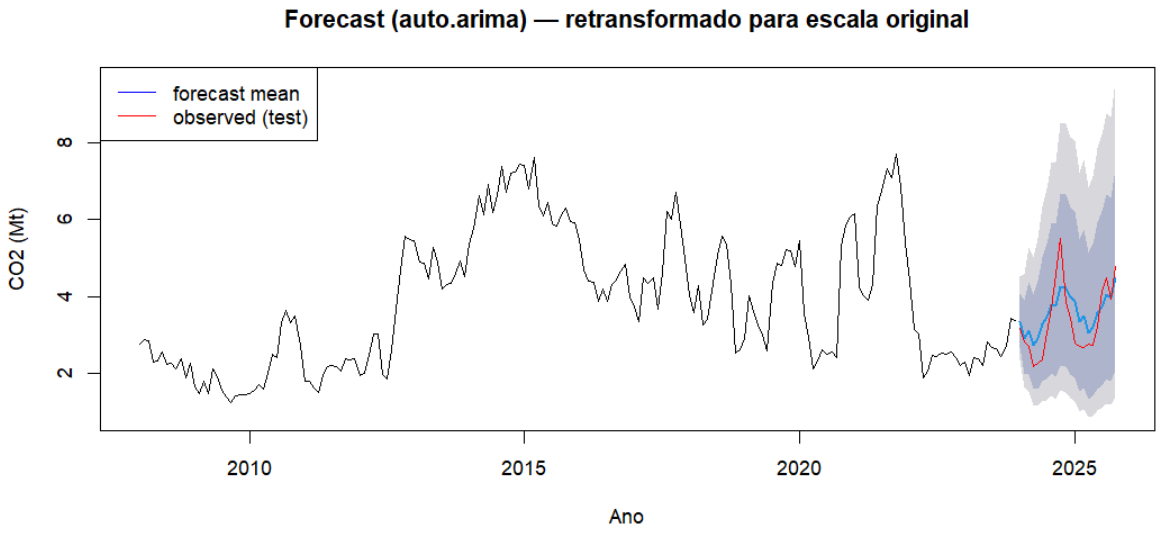
14) Ajuste do modelo ARIMA travado:

```
library(forecast)
library(lubridate)
library(dplyr)
library(ggplot2)

lambda_val <- if (exists("lambda")) lambda else 0.28

# dataset_st_treino_2008_2023  -> treino (co2_mt)
# dataset_st_teste_2024_25     -> teste (co2_mt)

# 1) transformar treino para objeto 'ts' (freq = 12)
start_year  <- year(min(dataset_st_treino_2008_2023$mes))
start_month <- month(min(dataset_st_treino_2008_2023$mes))
treino_ts  <-  ts(dataset_st_treino_2008_2023$co2_mt,  start  =  c(start_year,  start_month),
frequency = 12)

# 2) transformar teste para ts (para plot/accuracy)
if (exists("dataset_st_teste_2024_25")) {
  test_start_year  <- year(min(dataset_st_teste_2024_25$mes))
  test_start_month <- month(min(dataset_st_teste_2024_25$mes))
        test_ts  <-  ts(dataset_st_teste_2024_25$co2_mt,  start  =  c(test_start_year,
test_start_month), frequency = 12)
  h <- length(test_ts)
} else {
  test_ts <- NULL
  h <- 22
}

set.seed(123)  # reprodutibilidade (buscas stepwise podem usar aleatoriedade em alguns
backends)

modelo_auto_1 <- forecast::auto.arima(
  y = treino_ts,
  start.p = 0, start.q = 0,
  max.p = 3, max.q = 3,
  start.P = 0, max.P = 2,
  seasonal = TRUE,
  D = 1,      # força diferenciação sazonal como em seu python
  d = NA,     # deixar auto.arima escolher d (como d=None no pmdarima)
  test = "adf",     # usar ADF para escolher d, como no seu exemplo python
  stepwise = TRUE,
  trace = TRUE,
    lambda  =  lambda_val      #  aplica  Box–Cox  internamente;  forecast()  retransformará
automaticamente
)
```

```
# modelo escolhido (equivalente a model.summary() do pmdarima)
cat("\n--- auto.arima concluído ---\n")
print(summary(modelo_auto_1))

# -------- PREVISÃO (forecast) ----------
fc <- forecast(modelo_auto_1, h = h, biasadj = TRUE)  # biasadj ajuda ao retransformar

# mostra as primeiras previsões
print(head(fc))

# -------- PLOT: previsões + observados do teste ----------
# plot base rápido
plot(fc, main = "Forecast (auto.arima) — retransformado para escala original", xlab = "Ano",
ylab = "CO2 (Mt)")
if (!is.null(test_ts)) {
  lines(test_ts, col = "red", lwd = 1.2)
   legend("topleft", legend = c("forecast mean", "observed (test)"), col = c("blue", "red"), lty =
c(1,1))
}
```

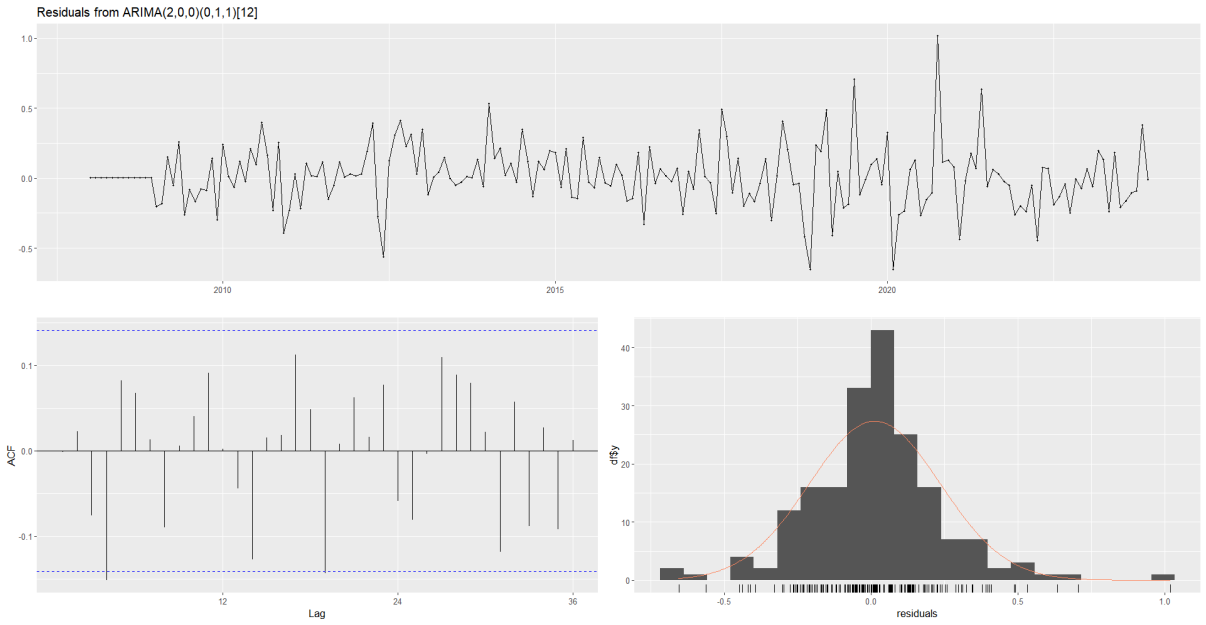
```
> print(head(fc))
$method
[1] "ARIMA(2,0,0)(0,1,1)[12]"

$model
Series: treino_ts
ARIMA(2,0,0)(0,1,1)[12]
Box Cox transformation: lambda= 0.2785684

Coefficients:
         ar1      ar2     sma1
      1.1105  -0.1818  -0.8407
s.e.  0.0728   0.0733   0.0718

sigma^2 = 0.05389:  log likelihood = 0.85
AIC=6.3   AICc=6.53   BIC=19.07

$level
[1] 80 95
```

```
> print(summary(modelo_auto_1))
Series: treino_ts
ARIMA(2,0,0)(0,1,1)[12]
Box Cox transformation: lambda= 0.2785684

Coefficients:
         ar1      ar2     sma1
      1.1105  -0.1818  -0.8407
s.e.  0.0728   0.0733   0.0718

sigma^2 = 0.05389:  log likelihood = 0.85
AIC=6.3   AICc=6.53   BIC=19.07

Training set error measures:
                     ME      RMSE       MAE         MPE     MAPE      MASE
Training set 0.05096175 0.5911895 0.409598 -0.4551357 11.22122 0.2941342
                    ACF1
Training set 0.005018579
```



15) Diagnóstico dos Resíduos e Ljung Box:

```
# 3.1) Gráficos de resíduos + Ljung–Box automático
forecast::checkresiduals(modelo_auto_1, lag = 24)

res <- residuals(modelo_auto_1)
k  <- length(coef(modelo_auto_1))  # nº de parâmetros estimados

Box.test(res,
         lag   = 24,
         type  = "Ljung-Box",
         fitdf = k)
```



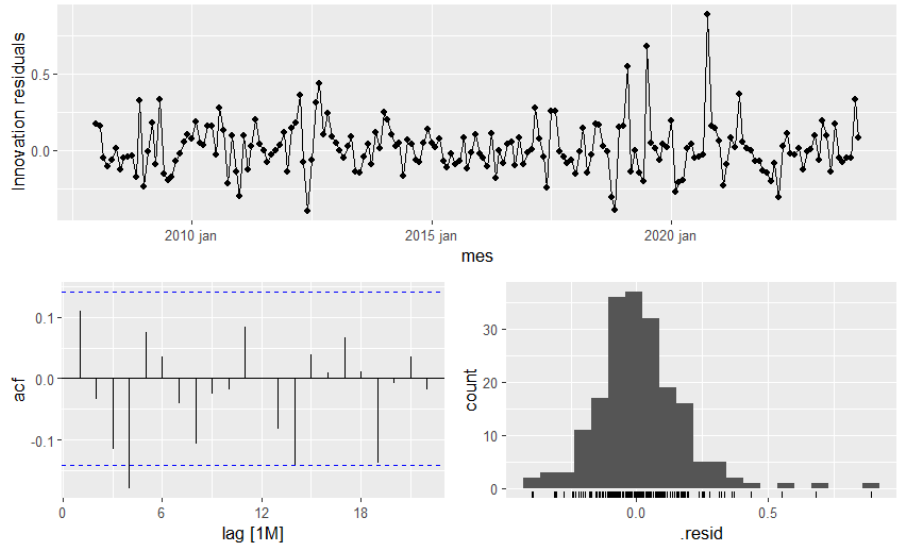
16) Ajuste do modelo ETS automático:

```
library(fable)
library(fabletools)
library(feasts)
library(tsibble)
library(dplyr)
library(ggplot2)

treino <- dataset_st_treino_2008_2023 |> as_tsibble(index = mes)

# 2) ajustar o melhor ETS automaticamente (seleção interna)
fit_ets <- treino |>
  model(ETS = ETS(co2_mt))

# 3) relatório do modelo (forma escolhida: erro/trend/season)
report(fit_ets)
```

```
> report(fit_ets)
Series: co2_mt
Model: ETS(M,N,M)
  Smoothing parameters:
    alpha = 0.9998595
    gamma = 0.0001362892

  Initial states:
     l[0]     s[0]    s[-1]    s[-2]    s[-3]    s[-4]    s[-5]     s[-6]     s[-7]     s[-8]     s[-9]
 2.356911 1.041572 1.149926 1.201478 1.028158 1.055936 0.985102 0.9798608 0.9056847 0.8375277 0.9317142
    s[-10]    s[-11]
 0.8914993 0.9915412

  sigma^2:  0.0307

     AIC     AICc      BIC
831.0673 833.7946 879.9298
```

17) Diagnóstico de resíduos e teste de Ljung-Box nos resíduos:

```
fit_ets |> gg_tsresiduals()
```

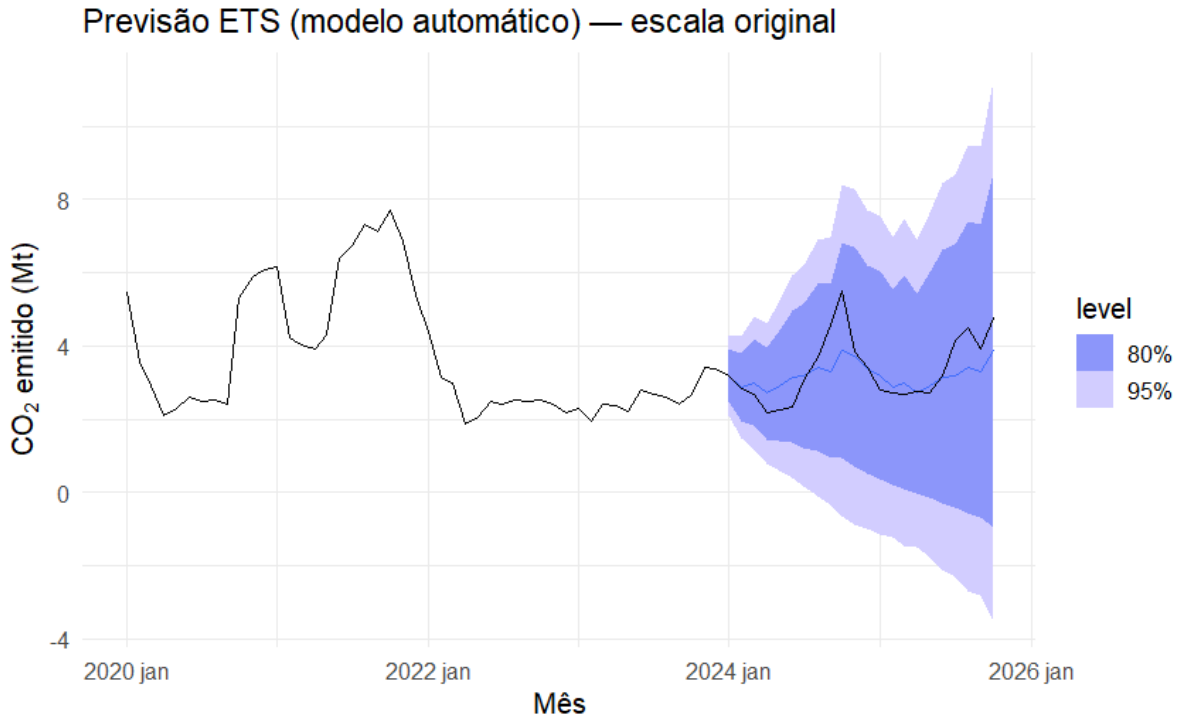



```
fit_ets |>
  augment() |>
  features(.innov, ljung_box, lag = 24)
```

```
+     augment() |>
+     features(.innov, ljung_box, lag = 24)
# A tibble: 1 x 3
  .model lb_stat lb_pvalue
  <chr>    <dbl>     <dbl>
1 ETS       30.5     0.168
```

```
# Previsão do modelo ETS

h <- if (exists("dataset_st_teste_2024_25")) nrow(dataset_st_teste_2024_25) else 22
fc_ets <- fit_ets |> forecast(h = h)

# 7) visualizar previsões sobre os dados (treino + teste se existir)
dataset_full <- if (exists("dataset_st_teste_2024_25")) {
  bind_rows(treino, dataset_st_teste_2024_25 |> as_tsibble(index = mes))
} else {
  treino
}

fc_ets |>
  autoplot(dataset_full |> filter_index("2020 Jan" ~ .)) +
  labs(
    title = "Previsão ETS (modelo automático) — escala original",
    y = expression("CO"[2] * " emitido (Mt)"),
    x = "Mês"
  ) +
```

theme_minimal(base_size = 13)



18) Ajuste do modelo NAIVE:

```
library(fable)
library(tsibble)

fit_naive <- dataset_st_treino_2008_2023 %>%
  model(NAIVE = NAIVE(box_cox(co2_mt, lambda)))

report(fit_naive)

fc_naive <- fit_naive %>% forecast(h = 22)

head(as_tibble(fc_naive), 6)
```

```
> report(fit_naive)
Series: co2_mt
Model: NAIVE
Transformation: box_cox(co2_mt, lambda)

sigma^2: 0.0591
>
> fc_naive <- fit_naive %>% forecast(h = 22)
>
> head(as_tibble(fc_naive), 6)
# A tibble: 6 x 4
  .model      mes           co2_mt .mean
  <chr>     <mth>           <dist> <dbl>
1 NAIVE  2024 jan t(N(1.4, 0.059))  3.40
2 NAIVE  2024 fev  t(N(1.4, 0.12))  3.43
3 NAIVE  2024 mar  t(N(1.4, 0.18))  3.47
4 NAIVE  2024 abr  t(N(1.4, 0.24))  3.51
5 NAIVE  2024 mai  t(N(1.4, 0.29))  3.54
6 NAIVE  2024 jun  t(N(1.4, 0.35))  3.58
```

19) Ajuste do modelo NAIVE SAZONAL:

```
library(fable)
library(tsibble)

fit_snaive <- dataset_st_treino_2008_2023 %>%
  model(SNAIVE = SNAIVE(box_cox(co2_mt, lambda)))

report(fit_snaive)

fc_snaive <- fit_snaive %>% forecast(h = 22)

head(as_tibble(fc_snaive), 6)
```

```
> report(fit_snaive)
Series: co2_mt
Model: SNAIVE
Transformation: box_cox(co2_mt, lambda)

sigma^2: 0.4499
>
> fc_snaive <- fit_snaive %>% forecast(h = 22)
>
> head(as_tibble(fc_snaive), 6)
# A tibble: 6 x 4
  .model      mes           co2_mt .mean
  <chr>     <mth>           <dist> <dbl>
1 SNAIVE 2024 jan t(N(0.93, 0.45))  2.53
2 SNAIVE 2024 fev t(N(0.72, 0.45))  2.15
3 SNAIVE 2024 mar    t(N(1, 0.45))  2.65
4 SNAIVE 2024 abr t(N(0.98, 0.45))  2.61
5 SNAIVE 2024 mai t(N(0.88, 0.45))  2.43
6 SNAIVE 2024 jun  t(N(1.2, 0.45))  3.06
```

20) Ajuste do modelo MEAN:

```
library(fable)
```

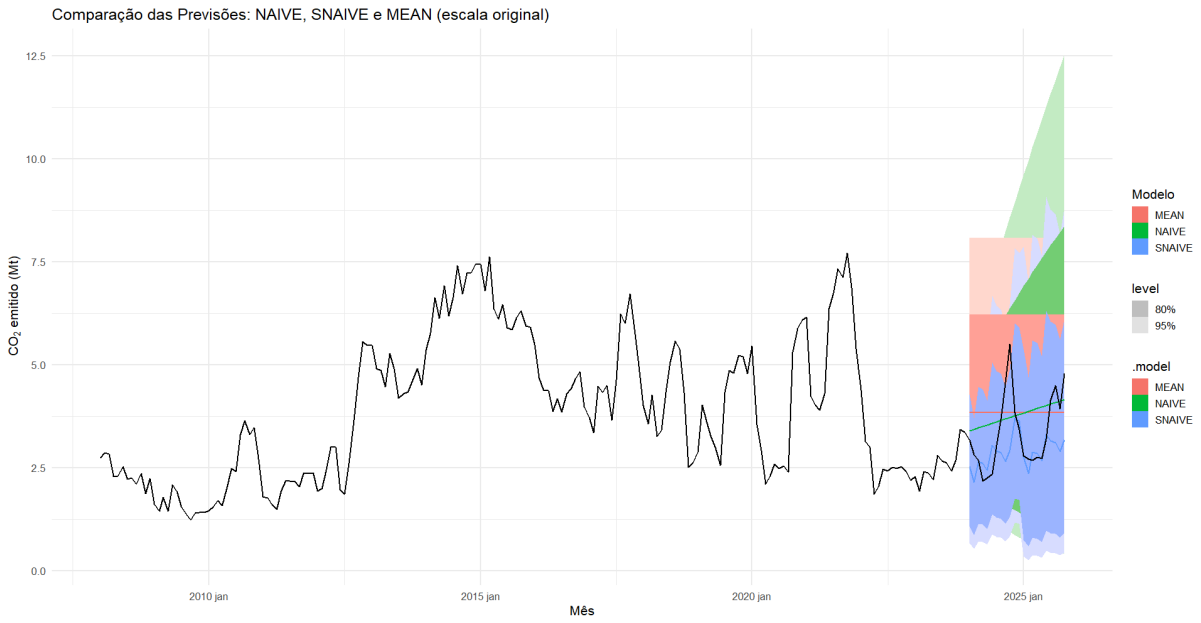
```
library(tsibble)

fit_mean <- dataset_st_treino_2008_2023 %>%
  model(MEAN = MEAN(box_cox(co2_mt, lambda)))

report(fit_mean)

fc_mean <- fit_mean %>% forecast(h = 22)

head(as_tibble(fc_mean), 6)
```

```
> report(fit_mean)
Series: co2_mt
Model: MEAN
Transformation: box_cox(co2_mt, lambda)

Mean: 1.5232
sigma^2: 0.4461
>
> fc_mean <- fit_mean %>% forecast(h = 22)
>
> head(as_tibble(fc_mean), 6)
# A tibble: 6 × 4
  .model      mes          co2_mt .mean
  <chr>     <mth>          <dist> <dbl>
1 MEAN   2024 jan t(N(1.5, 0.45))  3.84
2 MEAN   2024 fev t(N(1.5, 0.45))  3.84
3 MEAN   2024 mar t(N(1.5, 0.45))  3.84
4 MEAN   2024 abr t(N(1.5, 0.45))  3.84
5 MEAN   2024 mai t(N(1.5, 0.45))  3.84
6 MEAN   2024 jun t(N(1.5, 0.45))  3.84
```

21) Geração do gráfico gráfico dos 3 modelos (NAIVE, NAIVE SAZONAL e MEAN):

```
library(fable)
library(fabletools)
library(tsibble)
library(ggplot2)
library(dplyr)
library(lubridate)

dataset_train <- dataset_st_treino_2008_2023 %>%
  mutate(mes = yearmonth(mes)) %>%
  as_tsibble(index = mes)

dataset_test <- dataset_st_teste_2024_25 %>%
  mutate(mes = yearmonth(mes)) %>%
  as_tsibble(index = mes)

dataset_full <- bind_rows(dataset_train, dataset_test) %>%
  as_tsibble(index = mes)

# ==== 2) Ajustar os 3 modelos transformados ====
```

```r
fit_naive <- dataset_train %>%
  model(NAIVE = NAIVE(box_cox(co2_mt, lambda)))

fit_snaive <- dataset_train %>%
  model(SNAIVE = SNAIVE(box_cox(co2_mt, lambda)))

fit_mean <- dataset_train %>%
  model(MEAN = MEAN(box_cox(co2_mt, lambda)))

# ==== 3) Fazer previsões retransformadas para a escala original ====

h <- nrow(dataset_test)

fc_naive  <- fit_naive  %>% forecast(new_data = dataset_test)
fc_snaive <- fit_snaive %>% forecast(new_data = dataset_test)
fc_mean   <- fit_mean   %>% forecast(new_data = dataset_test)

# ==== 4) Unir previsões em um único objeto ====

fc_all <- bind_rows(
  fc_naive  %>% mutate(modelo = "NAIVE"),
  fc_snaive %>% mutate(modelo = "SNAIVE"),
  fc_mean   %>% mutate(modelo = "MEAN")
)

# ==== 5) Plot: valores reais + previsões ====

fc_all %>%
  autoplot(dataset_full) +
  labs(
    title = "Comparação das Previsões: NAIVE, SNAIVE e MEAN (escala original)",
    y = expression("CO"[2] * " emitido (Mt)"),
    x = "Mês",
    colour = "Modelo"
  ) +
  theme_minimal(base_size = 14)
```



22) Cálculo das Métricas:

```
library(dplyr)
library(tsibble)
library(fable)
library(fabletools)

# garantir índices coerentes (yearmonth)
dataset_train <- dataset_st_treino_2008_2023 %>% mutate(mes = yearmonth(mes)) %>% as_tsibble(index = mes)
dataset_test  <- dataset_st_teste_2024_25 %>% mutate(mes = yearmonth(mes)) %>% as_tsibble(index = mes)

# ajustar todos os modelos em um só passo (Box–Cox dentro dos modelos)
all_models <- dataset_train %>%
  model(
    ARIMA_auto = ARIMA(box_cox(co2_mt, lambda)),
    ARIMA_trav  = ARIMA(box_cox(co2_mt, lambda) ~ pdq(2,0,0) + PDQ(0,1,1)),
    ETS_model  = ETS(box_cox(co2_mt, lambda)),
    NAIVE      = NAIVE(box_cox(co2_mt, lambda)),
    SNAIVE     = SNAIVE(box_cox(co2_mt, lambda)),
    MEAN       = MEAN(box_cox(co2_mt, lambda)),
)

# previsões alinhadas com o conjunto de teste
fc_all <- all_models %>% forecast(new_data = dataset_test)

# acurácia (compara previsões com observados do dataset_test)
metrics <- accuracy(fc_all, dataset_test)
```

mostrar métricas ordenadas por RMSE
metrics %>% as_tibble() %>% arrange(RMSE) %>% print()

```
> metrics %>% as_tibble() %>% arrange(RMSE) %>% print()
# A tibble: 6 x 10
  .model     .type     ME  RMSE   MAE    MPE  MAPE  MASE RMSSE  ACF1
  <chr>      <chr>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
1 ARIMA_trav Test  -0.215 0.596 0.509 -10.3   16.4   NaN   NaN 0.528
2 NAIVE      Test  -0.426 0.918 0.781 -19.3   26.4   NaN   NaN 0.668
3 ETS_model  Test  -0.428 0.919 0.781 -19.3   26.5   NaN   NaN 0.667
4 SNAIVE     Test   0.474 0.937 0.650   9.84  17.0   NaN   NaN 0.531
5 MEAN       Test  -0.491 1.01  0.888 -22.2   30.3   NaN   NaN 0.688
6 ARIMA_auto Test  -0.764 1.12  0.970 -29.5   33.4   NaN   NaN 0.667
```

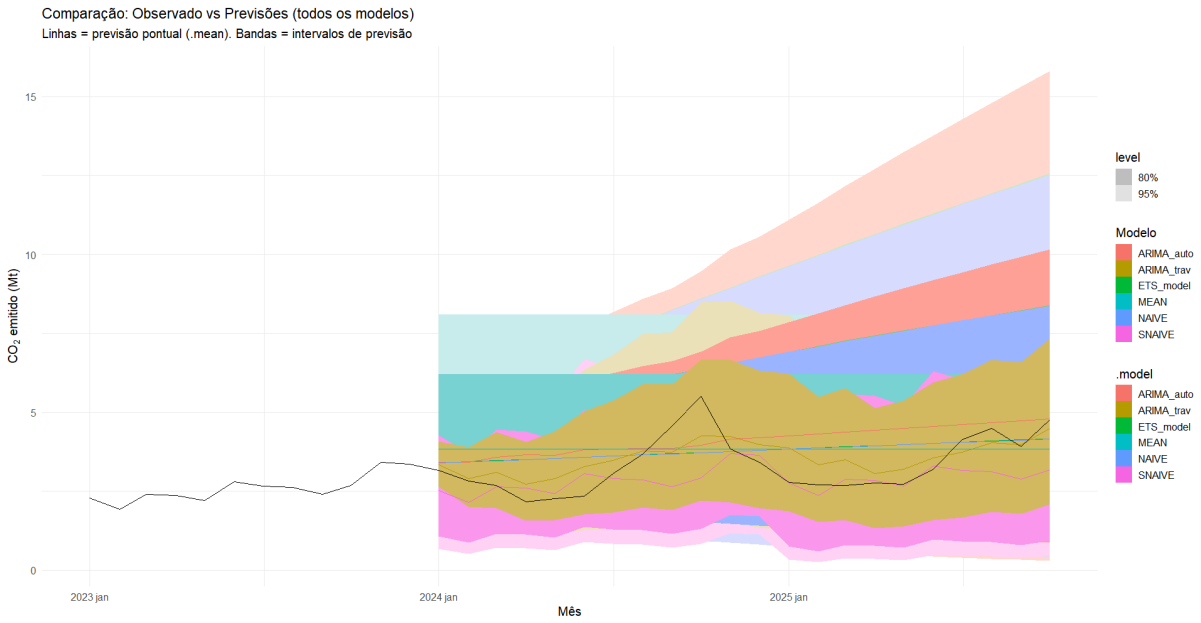
23) Gráfico de previsão de todos os modelos:

library(dplyr); library(tsibble); library(ggplot2); library(fabletools)

# 1) garantir índices coerentes
dataset_train <- dataset_train %>% mutate(mes = yearmonth(mes)) %>% as_tsibble(index = mes)
dataset_test <- dataset_test %>% mutate(mes = yearmonth(mes)) %>% as_tsibble(index = mes)

# 2) criar dataset_full (treino + teste) e escolher janela para plot
dataset_full <- bind_rows(dataset_train, dataset_test) %>% as_tsibble(index = mes)
start_plot <- "2023 Jan"   # ajustar conforme quiser

3) Plot único: observados + previsões de todos os modelos (fc_all)
autoplot pinta cada modelo (col = .model) e mostra intervalos
fc_all %>%
  autoplot(dataset_full %>% filter_index(start_plot ~ .)) +
  labs(
    title = "Comparação: Observado vs Previsões (todos os modelos)",
    subtitle = "Linhas = previsão pontual (.mean). Bandas = intervalos de previsão",
    y = expression("CO"[2] * " emitido (Mt)"),
    x = "Mês",
    colour = "Modelo"
) +
  theme_minimal(base_size = 13)


Comparação: Observado vs Previsões (todos os modelos)
Linhas = previsão pontual (.mean). Bandas = intervalos de previsão
15
10
5
0
CO_2 emitido (Mt)
2023 jan
2024 jan
2025 jan
Mês
level
80%
95%
Modelo
ARIMA_auto
ARIMA_trav
ETS_model
MEAN
NAIVE
SNAIVE
.model
ARIMA_auto
ARIMA_trav
ETS_model
MEAN
NAIVE
SNAIVE