TCC Final

Tiago Paulino

2025-11-02

Instalando as bibliotecas

Carregando os Pacotes

```
library(readxl)
library(dplyr)
library(tidyr)
library(stringr)
library(psych)
library(ggplot2)
library(plotly)
library(corrplot)
library(car)
library(reshape2)
library(purrr)
```

Importando o Dataframe de preços de Topload, Fogão e Refrigerador Topfreezer

```
df <- read_excel("C:/Users/USER/OneDrive/Documentos/TCC/Resultados Premiliminares - R/dataframe.xlsx")
```

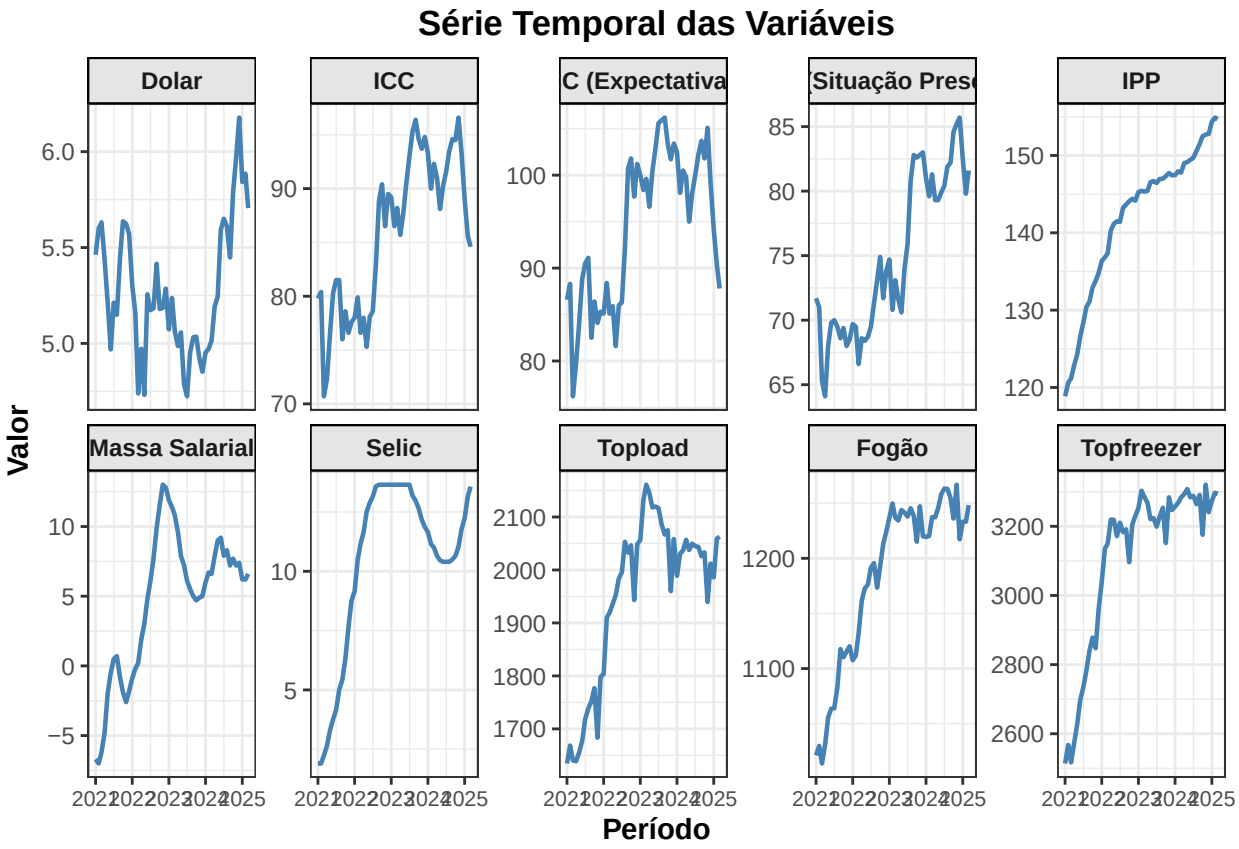
Prévia do Dataframe

```
head(df)
```

```
## # A tibble: 6 x 11
##   Período             `Massa Salarial`   IPP Dolar Selic   ICC
##   <dttm>                         <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 2021-01-01 00:00:00             -6.7  119.  5.46  1.9   79.8
## 2 2021-02-01 00:00:00             -7    121.  5.60  1.9   80.4
## 3 2021-03-01 00:00:00             -6.2  121.  5.63  2.23  70.7
## 4 2021-04-01 00:00:00             -4.8  123.  5.44  2.65  72.3
## 5 2021-05-01 00:00:00             -2    124.  5.22  3.29  76.5
## 6 2021-06-01 00:00:00             -0.5  127.  4.97  3.76  80.3
## # i 5 more variables: `ICC (Situação Presente)` <dbl>,
## #   `ICC (Expectativas)` <dbl>, Topload <dbl>, Topfreezer <dbl>, Fogão <dbl>
```

```r
df_long <- pivot_longer(
  df,
  cols = -Período,
  names_to = "Variavel",
  values_to = "Valor"
)

# Definindo a ordem das variáveis para garantir 5 em cada linha
# (A ordem que as variáveis aparecem no dataframe long será a ordem de plotagem)
ordem_variaveis <- c("Dolar", "ICC", "ICC (Expectativas)",
                     "ICC (Situação Presente)", "IPP", "Massa Salarial",
                     "Selic", "Topload", "Fogão", "Topfreezer")

df_long$Variavel <- factor(df_long$Variavel, levels = ordem_variaveis)

# Criação do gráfico
ggplot(df_long, aes(x = Período, y = Valor)) +
  geom_line(color = "steelblue", size = 0.8) +

  facet_wrap(~Variavel, scales = "free_y", nrow = 2) +

  labs(title = "Série Temporal das Variáveis",
       x = "Período", y = "Valor") +

  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title.x = element_text(face = "bold"),
    axis.title.y = element_text(face = "bold"),

strip.background = element_rect(fill = "gray90", color = "black"),
    strip.text = element_text(face = "bold"),

    axis.text.x = element_text(size = 8)
  ) + NULL
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



Padronizando as variáveis com método Z-Score

```
df_z <- df
cols_para_padronizar <- setdiff(names(df), c("Período", "Topload",
                                             "Fogão", "Topfreezer"))
df_z[ , cols_para_padronizar] <- scale(df[ , cols_para_padronizar])
head(df_z)
```

```
## # A tibble: 6 x 11
##   Período             `Massa Salarial`   IPP  Dolar Selic    ICC
##   <dttm>                         <dbl> <dbl>  <dbl> <dbl>  <dbl>
## 1 2021-01-01 00:00:00           -2.18  -2.40  0.530 -2.29 -0.845
## 2 2021-02-01 00:00:00           -2.24  -2.22  0.926 -2.29 -0.762
## 3 2021-03-01 00:00:00           -2.09  -2.16  1.02  -2.20 -2.12
## 4 2021-04-01 00:00:00           -1.82  -1.99  0.454 -2.09 -1.89
## 5 2021-05-01 00:00:00           -1.28  -1.84 -0.184 -1.92 -1.31
## 6 2021-06-01 00:00:00           -0.989 -1.60 -0.908 -1.79 -0.775
## # i 5 more variables: `ICC (Situação Presente)` <dbl>,
## #   `ICC (Expectativas)` <dbl>, Topload <dbl>, Topfreezer <dbl>, Fogão <dbl>
```

```
summary(df_z)
```

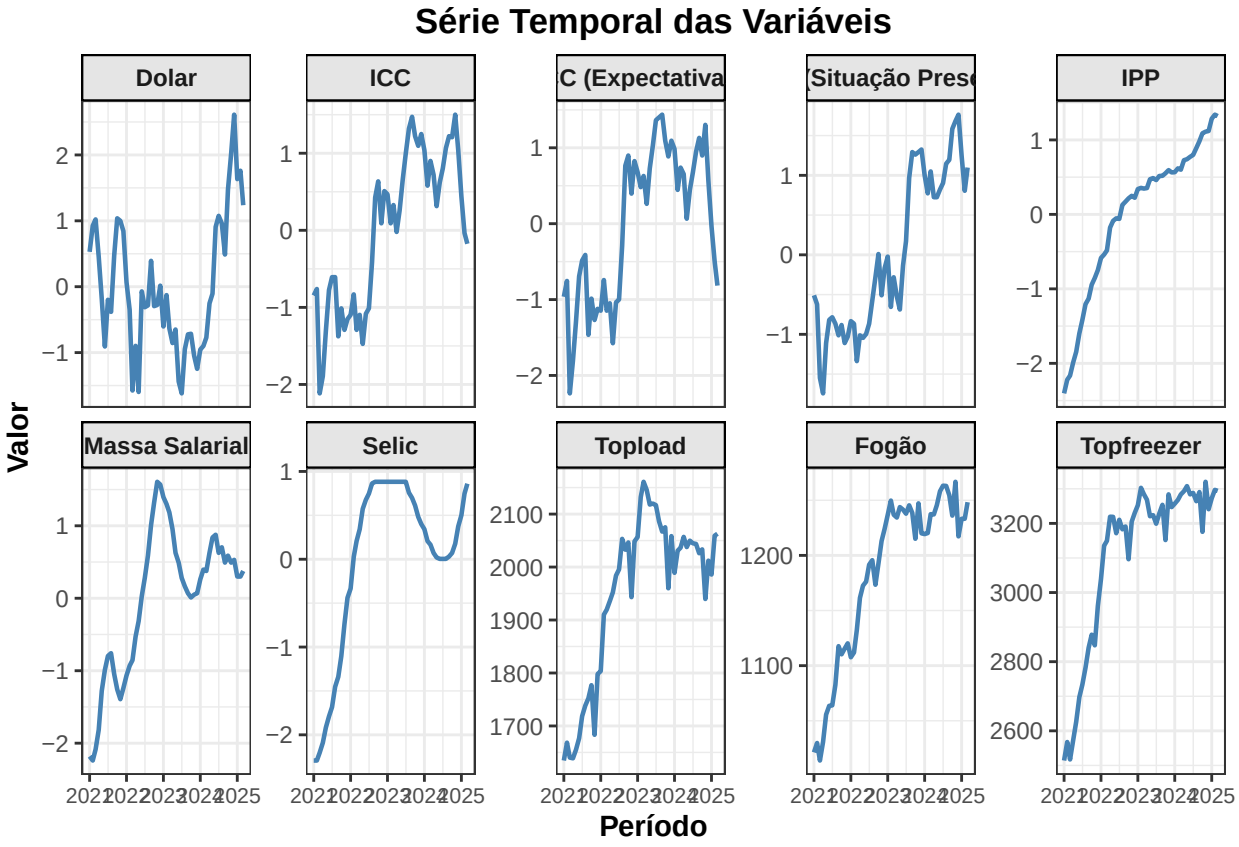
```
##     Período                      Massa Salarial         IPP
```

```
##  Min.   :2021-01-01 00:00:00.00   Min.   :-2.2396   Min.   :-2.4038
##  1st Qu.:2022-01-16 12:00:00.00   1st Qu.:-0.8258   1st Qu.:-0.5623
##  Median :2023-02-01 00:00:00.00   Median : 0.2802   Median : 0.3465
##  Mean   :2023-01-30 21:38:49.41   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.:2024-02-15 12:00:00.00   3rd Qu.: 0.6265   3rd Qu.: 0.6091
##  Max.   :2025-03-01 00:00:00.00   Max.   : 1.6074   Max.   : 1.3393
##      Dolar              Selic               ICC          ICC (Situação Presente)
##  Min.   :-1.6192   Min.   :-2.2929   Min.   :-2.1161   Min.   :-1.7430
##  1st Qu.:-0.7474   1st Qu.:-0.1648   1st Qu.:-0.9291   1st Qu.:-0.8664
##  Median :-0.1984   Median : 0.3419   Median : 0.2579   Median :-0.2820
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.8757   3rd Qu.: 0.8094   3rd Qu.: 0.8515   3rd Qu.: 0.9842
##  Max.   : 2.6114   Max.   : 0.8823   Max.   : 1.5008   Max.   : 1.7634
##  ICC (Expectativas)    Topload       Topfreezer       Fogão
##  Min.   :-2.2366    Min.   :1635   Min.   :2514   Min.   :1014
##  1st Qu.:-0.9753    1st Qu.:1857   1st Qu.:3068   1st Qu.:1119
##  Median : 0.3962    Median :2026   Median :3219   Median :1219
##  Mean   : 0.0000    Mean   :1951   Mean   :3111   Mean   :1183
##  3rd Qu.: 0.8554    3rd Qu.:2057   3rd Qu.:3271   3rd Qu.:1238
##  Max.   : 1.4370    Max.   :2161   Max.   :3320   Max.   :1267
```

Visualizando a série temporal das variáveis padronizadas

```
df_long_z <- pivot_longer(
  df_z,
  cols = -Período,
  names_to = "Variavel",
  values_to = "Valor"
)

ordem_variaveis <- c("Dolar", "ICC", "ICC (Expectativas)",
                     "ICC (Situação Presente)",
                     "IPP", "Massa Salarial", "Selic", "Topload",
                     "Fogão", "Topfreezer")

df_long_z$Variavel <- factor(df_long_z$Variavel, levels = ordem_variaveis)

ggplot(df_long_z, aes(x = Período, y = Valor)) +
  geom_line(color = "steelblue", size = 0.8) +

  facet_wrap(~Variavel, scales = "free_y", nrow = 2) +

  labs(title = "Série Temporal das Variáveis",
       x = "Período", y = "Valor") +

  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title.x = element_text(face = "bold"),
    axis.title.y = element_text(face = "bold"),

strip.background = element_rect(fill = "gray90", color = "black"),
```

```
    strip.text = element_text(face = "bold"),

    axis.text.x = element_text(size = 8)
  ) + NULL
```



Criando uma variável dummy "Sazonalidade", para os mêses de Janeiro e Novembro; período cíclico onde é esperado um vale no preço médio da categoria.

```
df_z$Sazonalidade <- ifelse(format(df$Período, "%m") %in% c("01", "11"), 1, 0)
head(df_z)
```

```
## # A tibble: 6 x 12
##   Período             `Massa Salarial`   IPP  Dolar Selic    ICC
##   <dttm>                         <dbl> <dbl>  <dbl> <dbl>  <dbl>
## 1 2021-01-01 00:00:00           -2.18  -2.40  0.530 -2.29 -0.845
## 2 2021-02-01 00:00:00           -2.24  -2.22  0.926 -2.29 -0.762
## 3 2021-03-01 00:00:00           -2.09  -2.16  1.02  -2.20 -2.12
## 4 2021-04-01 00:00:00           -1.82  -1.99  0.454 -2.09 -1.89
## 5 2021-05-01 00:00:00           -1.28  -1.84 -0.184 -1.92 -1.31
## 6 2021-06-01 00:00:00           -0.989 -1.60 -0.908 -1.79 -0.775
## # i 6 more variables: `ICC (Situação Presente)` <dbl>,
## #   `ICC (Expectativas)` <dbl>, Topload <dbl>, Topfreezer <dbl>, Fogão <dbl>,
## #   Sazonalidade <dbl>
```

Correlacionando as variáveis para selecionar aquelas com principal poder explicativo e identificar possíveis multicolinearidades

```
# --- 1. Definição de Variáveis ---
variaveis_dependentes <- c("Topload", "Fogão", "Topfreezer")
variaveis_independentes <- c("Selic", "IPP", "ICC",
                             "ICC (Expectativas)", "ICC (Situação Presente)",
                             "Dolar", "Sazonalidade", "Massa Salarial")

# --- 2. Calculo da Matriz de Correlação Completa ---
df_cor <- df_z %>%
  select(all_of(variaveis_dependentes), all_of(variaveis_independentes))

# Calcular a matriz de correlação
matriz_cor <- cor(df_cor, use = "pairwise.complete.obs")

# --- 3. Extrai e Reformata as Correlações Relevantes ---

# (Variáveis Dependentes vs Variáveis Independentes)
cor_df <- matriz_cor[variaveis_independentes, variaveis_dependentes, drop = FALSE]

cor_long <- cor_df %>%
  as.data.frame() %>%
  tibble::rownames_to_column(var = "Variavel_Independente") %>%
  pivot_longer(
    cols = all_of(variaveis_dependentes),
    names_to = "Variavel_Dependente",
    values_to = "Correlacao"
  )

cor_long$Variavel_Dependente <- factor(
  cor_long$Variavel_Dependente,
  levels = variaveis_dependentes
)

cor_long$Variavel_Independente <- factor(
  cor_long$Variavel_Independente,
  levels = rev(variaveis_independentes)
)


# --- 4. Criação da Visualização com ggplot2 ---

ggplot(cor_long, aes(x = 1, y = Variavel_Independente, fill = Correlacao)) +

  # Mapa de Calor (Tile)
  geom_tile(color = "white") +

  # Valores da Correlação como Texto
  geom_text(aes(label = sprintf("%.2f", Correlacao)), color = "black", size = 4) +

  scale_fill_gradient2(
    low = "#BC5449",     # Vermelho/Marrom para negativo
```

```r
  mid = "white",       # Branco para perto de zero
  high = "#0047AB",     # Azul Escuro para positivo
  midpoint = 0,        # Ponto médio em zero
  limit = c(-1, 1),    # Limites fixos de -1 a 1
  space = "Lab",
  name = "Correlação"
) +

# Configurações do Layout para Gráficos Paralelos
facet_wrap(~ Variavel_Dependente, ncol = 3) +

# Remover eixos desnecessários e ajustar o tema
labs(
  x = "",
  y = ""
) +
scale_x_continuous(breaks = NULL) + # Remove os ticks e labels do eixo X
theme_minimal() +
theme(
  # Ajustes de texto
  axis.text.x = element_blank(),
  axis.text.y = element_text(angle = 0, hjust = 1, size = 10),
  # Ajustes do painel e grade
  panel.grid.major = element_blank(),
  panel.border = element_blank(),
  # Títulos do Facet
  strip.text = element_text(face = "bold", size = 12)
)
```

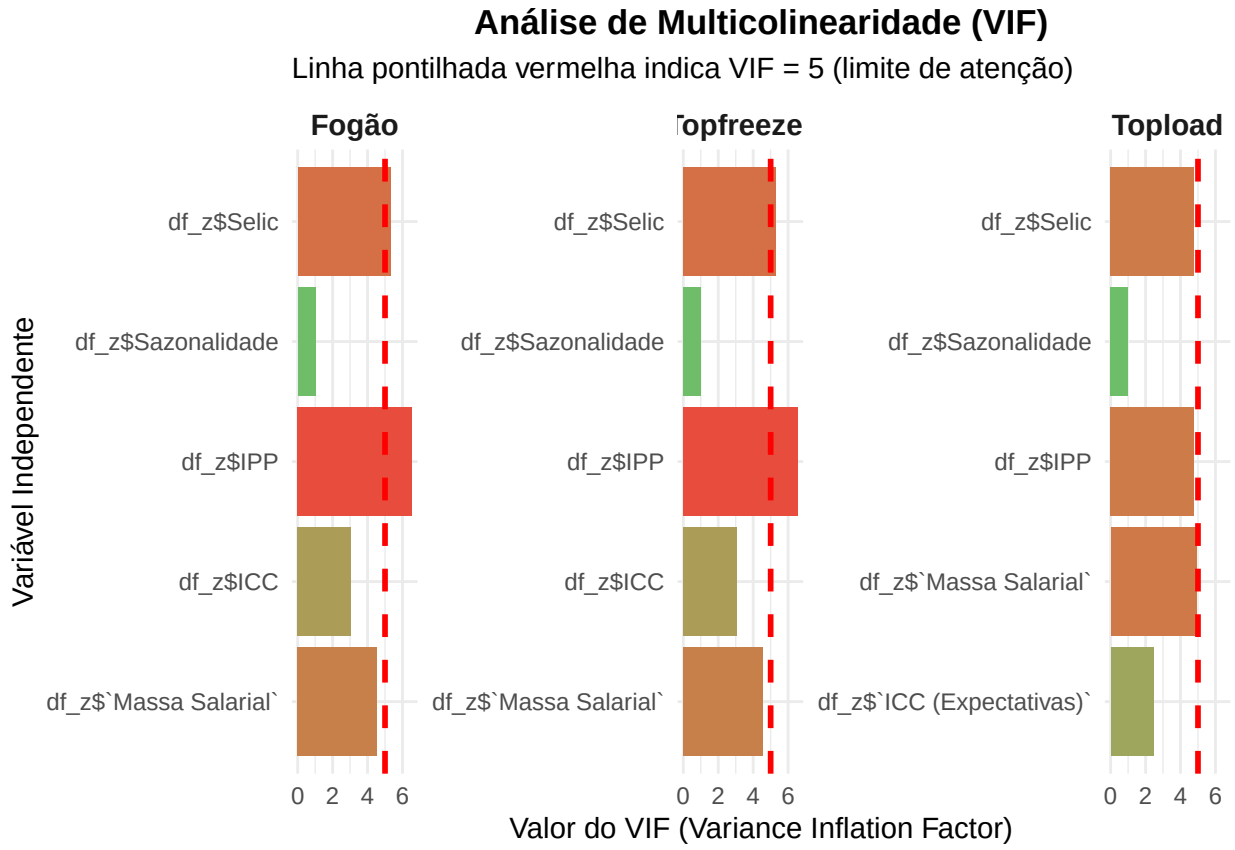
Verificar multicolinearidade ou alta correlação entre variáveis independentes

Aplicação do VIF (Variance Inflation Factor)

```
#Criação de um modelo preliminar com as variáveis mais correlacionadas

modelo_vif_tl <- lm(df_z$Topload ~ df_z$Selic + df_z$IPP +
                      df_z$`Massa Salarial` + df_z$`ICC (Expectativas)` +
                      df_z$Sazonalidade)

modelo_vif_fg <- lm(df_z$Fogão ~ df_z$Selic + df_z$IPP +
                      df_z$`Massa Salarial` + df_z$ICC +
                      df_z$Sazonalidade)

modelo_vif_tf <- lm(df_z$Topfreezer ~ df_z$Selic + df_z$IPP +
                      df_z$`Massa Salarial` + df_z$ICC +
                      df_z$Sazonalidade)

# --- 2. Calcular VIFs e Consolidar os Dados ---

# Função auxiliar para calcular VIF e formatar em um dataframe
calcular_vif_df <- function(modelo, nome_dependente) {
  vif_valores <- car::vif(modelo)
```

```
  # Cria um dataframe com os resultados
  df_resultado <- data.frame(
    Variavel_Dependente = nome_dependente,
    Variavel_Independente = names(vif_valores),
    VIF = as.numeric(vif_valores)
  )
  return(df_resultado)
}

# Calcular e combinar os VIFs de todos os modelos
df_vif_final <- bind_rows(
  calcular_vif_df(modelo_vif_tl, "Topload"),
  calcular_vif_df(modelo_vif_fg, "Fogão"),
  calcular_vif_df(modelo_vif_tf, "Topfreezer")
)

# --- 3. Criar a Visualização com ggplot2 ---

# Definir o limite de VIF (o valor 5 é um limite comum de atenção)
VIF_LIMITE <- 5

ggplot(df_vif_final, aes(x = VIF, y = Variavel_Independente, fill = VIF)) +

  # Gráfico de Barras Horizontal
  geom_bar(stat = "identity") +

  # Linha de Referência para o limite de multicolinearidade (VIF > 5)
  geom_vline(
    xintercept = VIF_LIMITE,
    linetype = "dashed",
    color = "red",
    size = 1
  ) +

  # Texto no gráfico para indicar a linha de limite
  annotate("text",
           x = VIF_LIMITE + 0.5, y = Inf,
           label = "",
           color = "red",
           hjust = 0, vjust = 1.5,
           size = 3.5, fontface = "bold") +

  # Escala de Cor (para colorir as barras com base no valor VIF)
  scale_fill_gradient(low = "#2ECC71", high = "#E74C3C",
                      limits = c(0, max(c(df_vif_final$VIF, VIF_LIMITE)))) +

  # Dividir em painéis (facetas) por Variável Dependente
  facet_wrap(~ Variavel_Dependente, ncol = 3, scales = "free_y") +

  # Títulos e Rótulos
  labs(
    title = "Análise de Multicolinearidade (VIF)",
    subtitle = "Linha pontilhada vermelha indica VIF = 5 (limite de atenção)",
```

```
    x = "Valor do VIF (Variance Inflation Factor)",
    y = "Variável Independente",
    fill = "VIF"
  ) +

  # Tema para Estilização
  theme_minimal() +
  theme(
    legend.position = "none", # A legenda de cor é redundante com a barra
    plot.title = element_text(face = "bold", hjust = 0.5),
    strip.text = element_text(face = "bold", size = 11) # Títulos dos Facets
  )
```



$1 < \text{VIF} < 5$ indica problema de multicolinearidade modearada.

Portanto, existe correlação entre as variáveis, mas não de forma grave.

Próximo passo é verificar a distribuição dos resíduos.

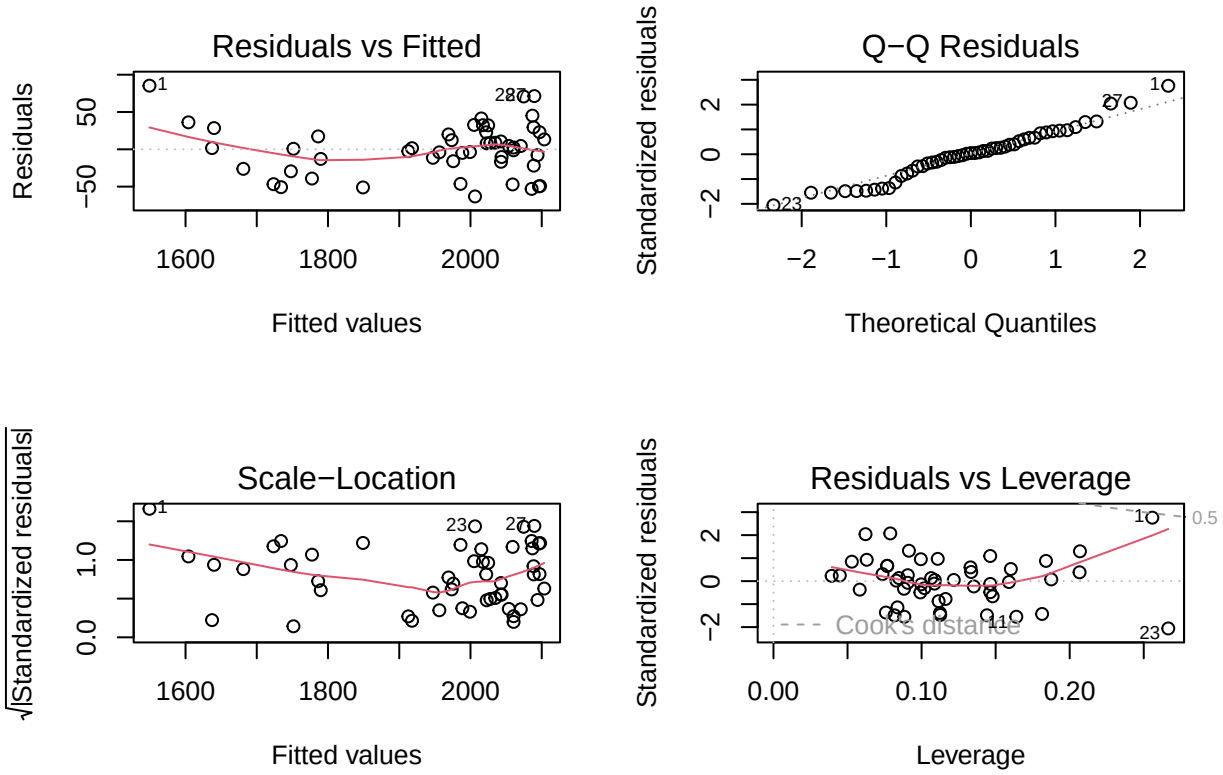
- Comportamento aleatório
- Heterocedasticidade (resíduos com variância desigual)
- Distribuição aproximadamente normal

Topload

```
modelo_tl <- lm(df_z$Topload ~ df_z$Selic + df_z$IPP + df_z$`Massa Salarial`
                + df_z$`ICC (Expectativas)` + df_z$Sazonalidade)

# Resumo
summary(modelo_tl)
```

```
##
## Call:
## lm(formula = df_z$Topload ~ df_z$Selic + df_z$IPP + df_z$`Massa Salarial` +
##     df_z$`ICC (Expectativas)` + df_z$Sazonalidade)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.174 -19.317   1.549  21.376  85.310
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1965.115      5.535 355.026  < 2e-16 ***
## df_z$Selic                   85.799     11.073   7.749 8.02e-10 ***
## df_z$IPP                     31.960     11.086   2.883 0.006024 **
## df_z$`Massa Salarial`        14.783     11.234   1.316 0.194870
## df_z$`ICC (Expectativas)`    32.162      7.948   4.047 0.000202 ***
## df_z$Sazonalidade           -78.945     13.241  -5.962 3.55e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.83 on 45 degrees of freedom
## Multiple R-squared:  0.9532, Adjusted R-squared:  0.948
## F-statistic: 183.4 on 5 and 45 DF,  p-value: < 2.2e-16
```

```
# Diagnóstico gráfico
par(mfrow = c(2, 2))
plot(modelo_tl)
```



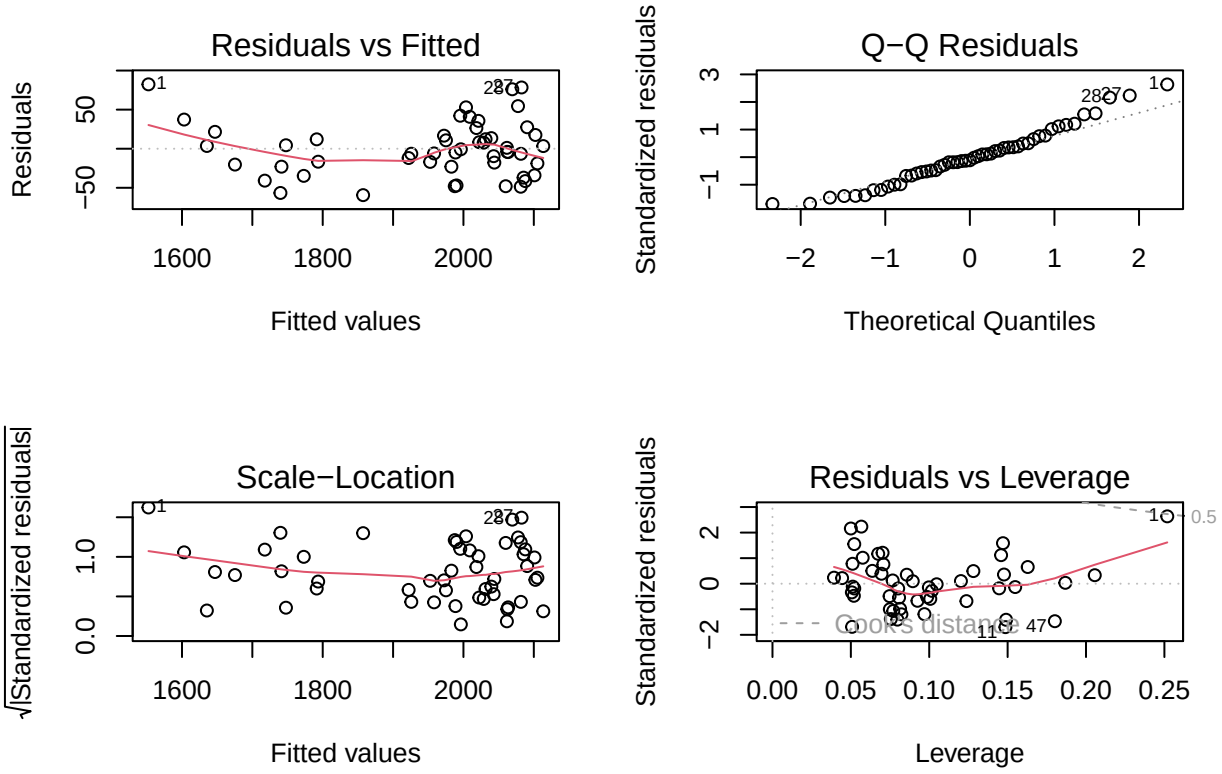
Massa Salarial não contribui para o modelo e pode ser removida.

```
modelo_tl_2 <- lm(df_z$Topload ~ df_z$Selic + df_z$IPP +
                    df_z$`ICC (Expectativas)` + df_z$Sazonalidade)

# Resumo
summary(modelo_tl_2)
```

```
##
## Call:
## lm(formula = df_z$Topload ~ df_z$Selic + df_z$IPP + df_z$`ICC (Expectativas)` +
##     df_z$Sazonalidade)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -59.39 -21.65  -4.02  17.12  82.33
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1965.395      5.575 352.546  < 2e-16 ***
## df_z$Selic                   93.097      9.659   9.638 1.30e-12 ***
## df_z$IPP                     34.882     10.948   3.186  0.00259 **
## df_z$`ICC (Expectativas)`    36.540      7.276   5.022 8.16e-06 ***
```

```
## df_z$Sazonalidade          -80.530     13.290  -6.059 2.36e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36.12 on 46 degrees of freedom
## Multiple R-squared:  0.9514, Adjusted R-squared:  0.9472
## F-statistic: 225.3 on 4 and 46 DF,  p-value: < 2.2e-16
```

```
# Diagnóstico gráfico
par(mfrow = c(2, 2))
plot(modelo_tl_2)
```



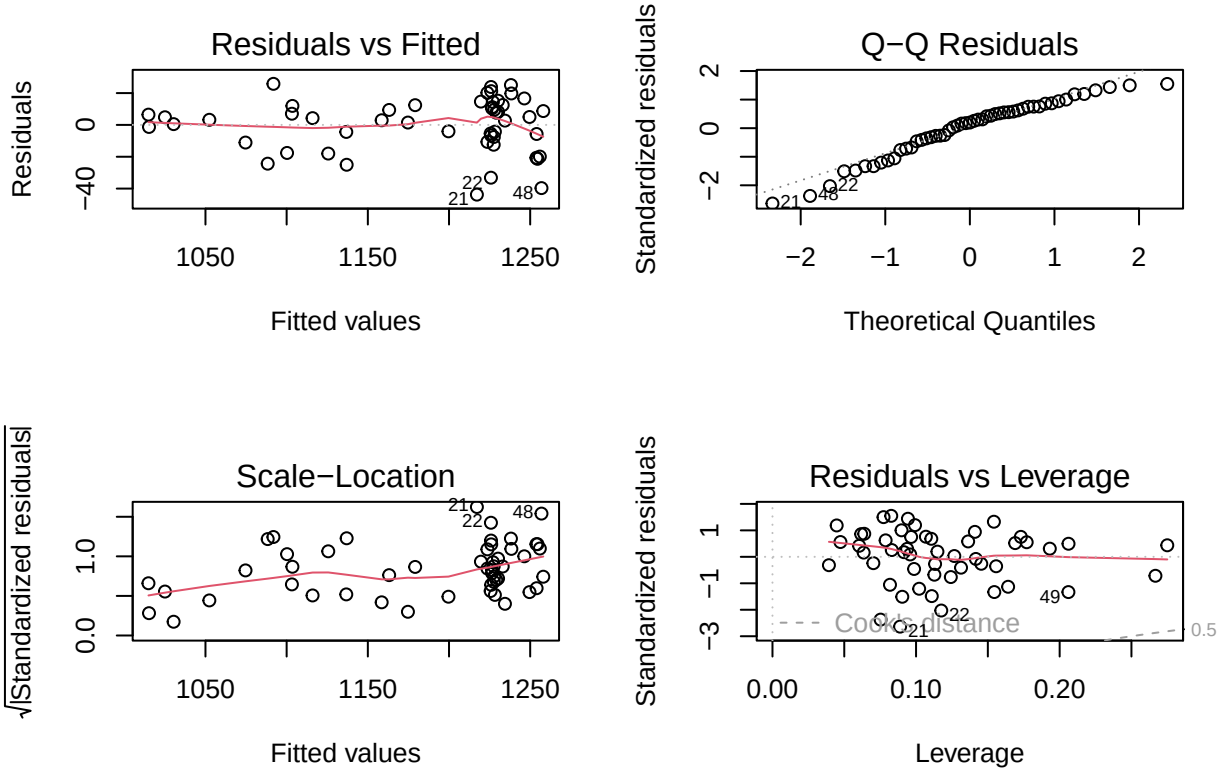
Fogão

```
modelo_fg <- lm(df_z$Fogão ~ df_z$Selic + df_z$IPP +
                  df_z$`Massa Salarial` + df_z$ICC + df_z$Sazonalidade)

# Resumo
summary(modelo_fg)
```

```
##
## Call:
## lm(formula = df_z$Fogão ~ df_z$Selic + df_z$IPP + df_z$`Massa Salarial` +
```

```
##     df_z$ICC + df_z$Sazonalidade)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -43.801  -9.134   3.215  11.555  25.805
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1183.782      2.690 440.066  < 2e-16 ***
## df_z$Selic                7.515      5.670   1.325   0.1918
## df_z$IPP                 45.779      6.289   7.279 3.93e-09 ***
## df_z$`Massa Salarial`    13.972      5.242   2.666   0.0106 *
## df_z$ICC                 11.232      4.293   2.616   0.0121 *
## df_z$Sazonalidade        -1.629      6.454  -0.252   0.8018
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.4 on 45 degrees of freedom
## Multiple R-squared:  0.952,  Adjusted R-squared:  0.9467
## F-statistic: 178.5 on 5 and 45 DF,  p-value: < 2.2e-16
```

```r
# Diagnóstico gráfico
par(mfrow = c(2, 2))
plot(modelo_fg)
```

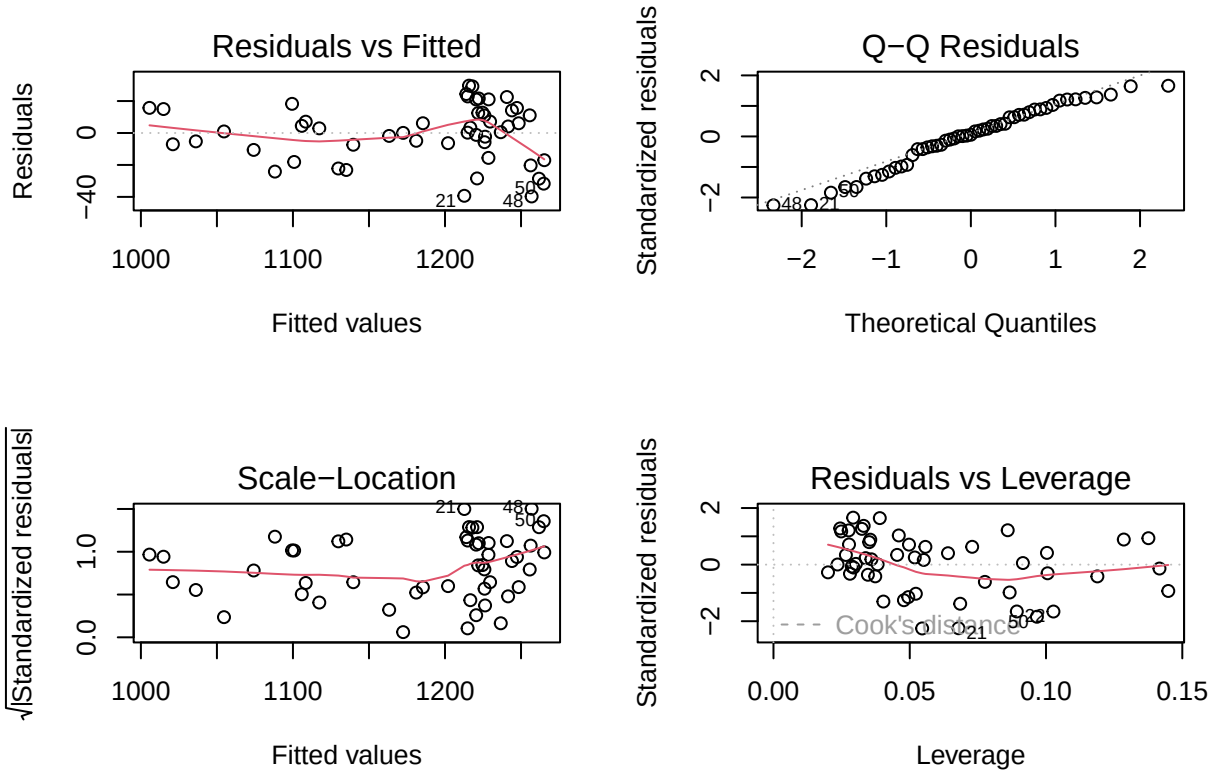


Selic, ICC e Sazonalidade não contribuem para o modelo e podem ser removidos.

```r
modelo_fg_2 <- lm(df_z$Fogão ~ df_z$IPP + df_z$`Massa Salarial`)

# Resumo
summary(modelo_fg_2)
```

```
##
## Call:
## lm(formula = df_z$Fogão ~ df_z$IPP + df_z$`Massa Salarial`)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.724  -8.987   0.958  13.431  29.618
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           1183.494      2.536 466.639  < 2e-16 ***
## df_z$IPP                56.477      4.551  12.410  < 2e-16 ***
## df_z$`Massa Salarial`   19.293      4.551   4.239 0.000101 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.11 on 48 degrees of freedom
## Multiple R-squared:  0.9445, Adjusted R-squared:  0.9422
## F-statistic: 408.7 on 2 and 48 DF,  p-value: < 2.2e-16
```

```r
# Diagnóstico gráfico
par(mfrow = c(2, 2))
plot(modelo_fg_2)
```



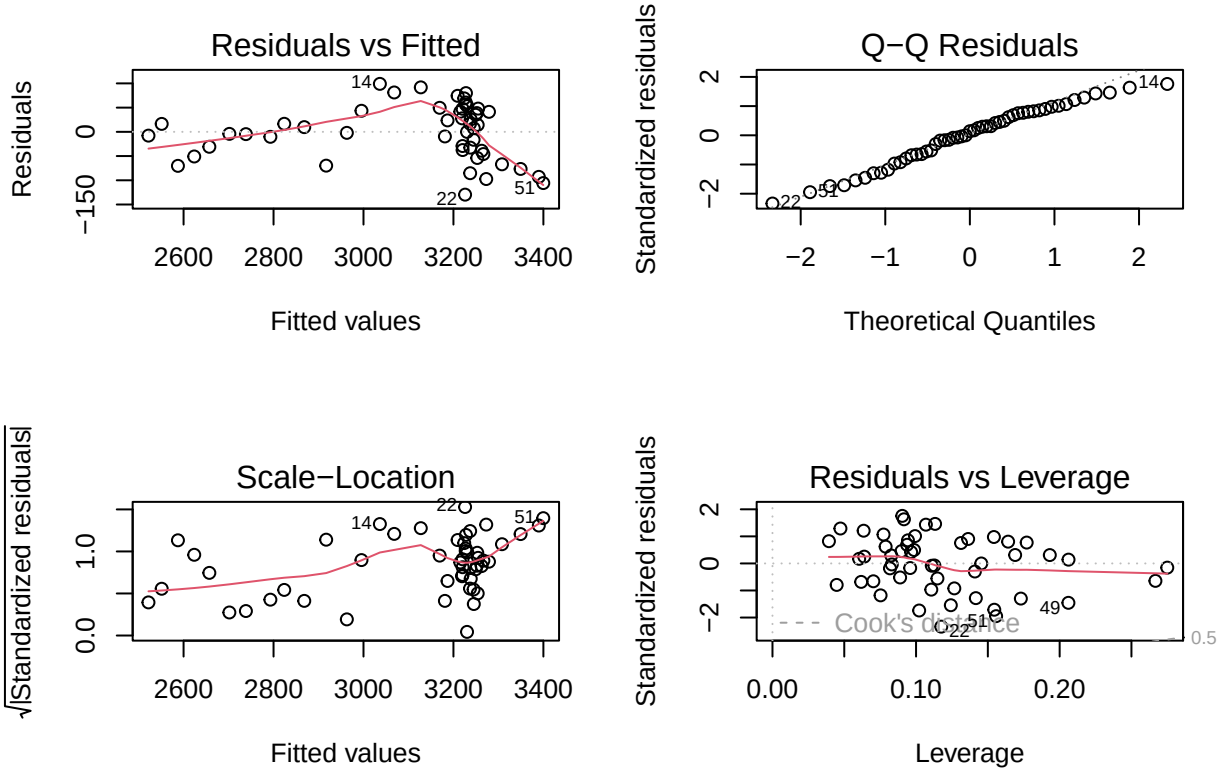
Topfreezer

```
modelo_tf <- lm(df_z$Topfreezer ~ df_z$Selic + df_z$IPP +
                  df_z$`Massa Salarial` + df_z$ICC + df_z$Sazonalidade)

# Resumo
summary(modelo_tf)
```

```
##
## Call:
## lm(formula = df_z$Topfreezer ~ df_z$Selic + df_z$IPP + df_z$`Massa Salarial` +
##     df_z$ICC + df_z$Sazonalidade)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -129.65  -38.31    7.38   42.54   99.01
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           3111.109      9.131 340.712  < 2e-16 ***
## df_z$Selic              93.611     19.248   4.863 1.45e-05 ***
## df_z$IPP               154.253     21.347   7.226 4.72e-09 ***
## df_z$`Massa Salarial`    5.736     17.792   0.322    0.749
```

```
## df_z$ICC               -13.445     14.572  -0.923    0.361
## df_z$Sazonalidade       -2.607     21.910  -0.119    0.906
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.08 on 45 degrees of freedom
## Multiple R-squared:  0.9458, Adjusted R-squared:  0.9398
## F-statistic: 157.2 on 5 and 45 DF,  p-value: < 2.2e-16
```

```r
# Diagnóstico gráfico
par(mfrow = c(2, 2))
plot(modelo_tf)
```



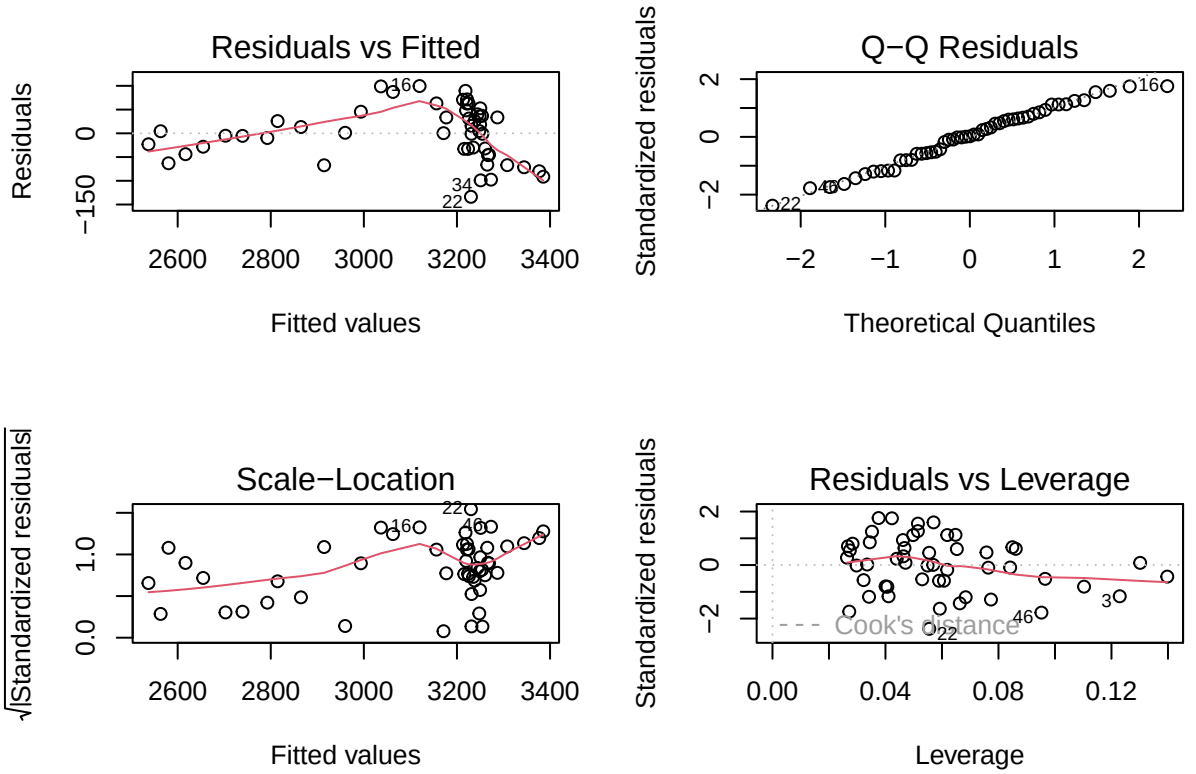
Massa Salarial, ICC e Sazonalidade não contribuem para o modelo e podem ser removidos

```r
modelo_tf_2 <- lm(df_z$Topfreezer ~ df_z$Selic + df_z$IPP)

# Resumo
summary(modelo_tf_2)
```

```
##
## Call:
```

```
## lm(formula = df_z$Topfreezer ~ df_z$Selic + df_z$IPP)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -133.786  -38.420    1.119   38.417   99.554
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3110.65       8.09 384.521  < 2e-16 ***
## df_z$Selic    100.56      15.44   6.513 4.11e-08 ***
## df_z$IPP      142.63      15.44   9.237 3.18e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 57.77 on 48 degrees of freedom
## Multiple R-squared:  0.9448, Adjusted R-squared:  0.9425
## F-statistic: 410.5 on 2 and 48 DF,  p-value: < 2.2e-16
```

```r
# Diagnóstico gráfico
par(mfrow = c(2, 2))
plot(modelo_tf_2)
```



Por fim, para avaliar a performance do modelo, irei dividir a base entre "Treino" e "Teste".

Cálculo de métricas de performance:

- Erro Quadrático Médio (RMSE);
- Erro Médio
- R^2

```r
# --- 1. Preparação dos Dados (Split Treino/Teste) ---

set.seed(123)
n <- nrow(df_z)
indices_treino <- sample(1:n, size = round(2/3 * n))
treino <- df_z[indices_treino, ]
teste <- df_z[-indices_treino, ]

# --- 2. Definição dos Modelos a serem Avaliados ---

modelos_a_avaliar <- list(
  Topload = Topload ~ Selic + IPP + `ICC (Expectativas)` + Sazonalidade,
  Fogao = Fogão ~ IPP + `Massa Salarial`,
  Topfreezer = Topfreezer ~ Selic + IPP
)

# --- 3. Função de Validação e Cálculo de Métricas (Core da Otimização) ---

validar_modelo <- function(formula, nome_modelo, dados_treino, dados_teste) {
  # 1. Treinar o modelo
  modelo_fit <- lm(formula, data = dados_treino)

  # 2. Prever nos dados de teste
  preds <- predict(modelo_fit, newdata = dados_teste)

  # Variável Real (Y)
  y_real <- dados_teste[[gsub(" .*", "", as.character(formula)[2])]]

  # 3. Calcular Métricas
  RMSE <- sqrt(mean((y_real - preds)^2, na.rm = TRUE))
  MAE <- mean(abs(y_real - preds), na.rm = TRUE)
  SS_res <- sum((y_real - preds)^2, na.rm = TRUE)
  SS_tot <- sum((y_real - mean(y_real, na.rm = TRUE))^2, na.rm = TRUE)
  R2 <- 1 - (SS_res / SS_tot)

  # 4. Retornar um dataframe com as métricas e os dados para o gráfico
  return(list(
    # Tabela de Métricas
    metrics = data.frame(
      Modelo = nome_modelo,
      RMSE = RMSE,
      MAE = MAE,
      R2 = R2
    ),
```

```
    # Dados para o Gráfico Predito vs Real
    plot_data = data.frame(
      Modelo = nome_modelo,
      Real = y_real,
      Predito = preds
    )
  ))
}

# --- 4. Executar a Validação Cruzada para todos os Modelos (Iteração) ---

# Usar 'map' para aplicar a função a cada modelo na lista
resultados_lista <- map(
  names(modelos_a_avaliar),
  ~ validar_modelo(
    formula = modelos_a_avaliar[[.x]],
    nome_modelo = .x,
    dados_treino = treino,
    dados_teste = teste
  )
)

# --- 5. Consolidar os Outputs ---

# Reunir as tabelas de métricas em um dataframe único
tabela_metricas <- map_dfr(resultados_lista, "metrics")

# Reunir os dados de plotagem para um gráfico facetado
df_plot_final <- map_dfr(resultados_lista, "plot_data")


# ------------------------------------------------------------------------
# -------------------- OTIMIZAÇÃO DA VISUALIZAÇÃO ----------------------
# ------------------------------------------------------------------------

## 1. TABELA DE MÉTRICAS CONSOLIDADAS

cat("============================================================\n")
```

```
## ============================================================
```

```
cat("          METRICAS DE PERFORMANCE EM DADOS DE TESTE       \n")
```

```
##           METRICAS DE PERFORMANCE EM DADOS DE TESTE
```

```
cat("============================================================\n")
```

```
## ============================================================
```

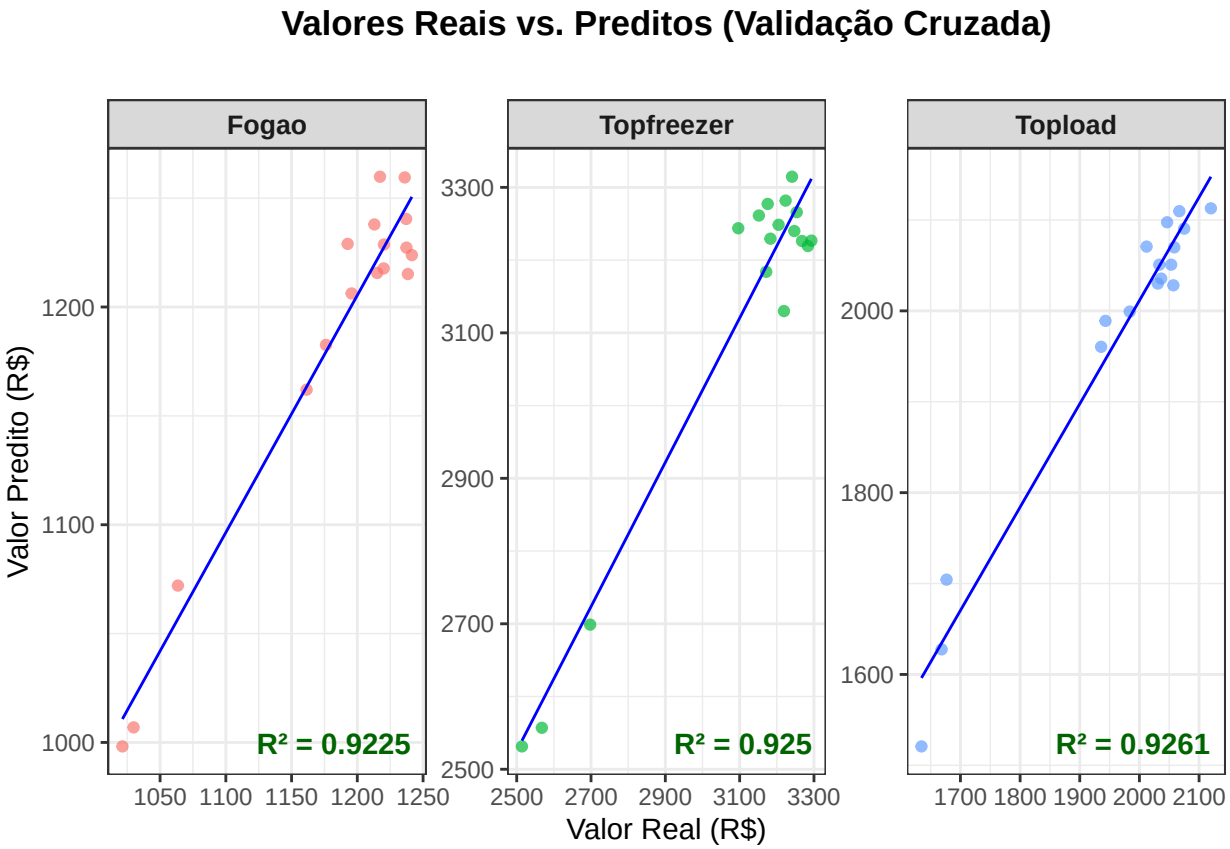
```
print(tabela_metricas %>%
        mutate(across(c(RMSE, MAE, R2), ~ round(.x, 4)))
)
```

```
##       Modelo    RMSE     MAE     R2
## 1    Topload 40.4305 29.7105 0.9261
## 2      Fogao 19.7652 15.6001 0.9225
## 3 Topfreezer 66.8534 53.1321 0.9250
```

```
cat("\n\n")
```

```
## 2. GRÁFICO DE PREDITO VS. REAL UNIFICADO

# Criar o gráfico base usando facet_wrap para os 3 modelos
grafico_final_cv <- ggplot(df_plot_final, aes(x = Real, y = Predito)) +

  # Pontos
  geom_point(aes(color = Modelo), alpha = 0.7) +

  # Linha de tendência (opcional, mostra o ajuste do modelo)
  geom_smooth(method = "lm", se = FALSE, color = "blue", linewidth = 0.5) +

  # Dividir o gráfico em 3 painéis (facets)
  facet_wrap(~ Modelo, scales = "free", ncol = 3) +

  # Adicionar o R² como anotação em cada painel
  geom_text(
    data = tabela_metricas,
    aes(
      label = paste0("R² = ", round(R2, 4)),
      x = Inf, # Posição no canto superior direito do eixo X
      y = -Inf # Posição no canto superior direito do eixo Y (inverso para topo)
    ),
    hjust = 1.1, vjust = -1, # Ajustar posição para dentro da área de plotagem
    size = 4, fontface = "bold", color = "darkgreen"
  ) +

  # Títulos e Tema
  labs(
    title = "Valores Reais vs. Preditos (Validação Cruzada)",
    subtitle = "",
    x = "Valor Real (R$)",
    y = "Valor Predito (R$)"
  ) +
  theme_bw() +
  theme(
    plot.title = element_text(face = "bold", hjust = 0.5),
    strip.text = element_text(face = "bold", size = 10),
    legend.position = "none"
  )

# Exibir o gráfico final
print(grafico_final_cv)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```


Valores Reais vs. Preditos (Validação Cruzada)
Fogao
Topfreezer
Topload
R² = 0.9225
R² = 0.925
R² = 0.9261
Valor Predito (R$)
Valor Real (R$)
1000
1100
1200
2500
2700
2900
3100
3300
1600
1800
2000
1050 1100 1150 1200 1250
2500 2700 2900 3100 3300
1700 1800 1900 2000 2100