Cálculo de Largura de Banda para GWR

Seu Nome/Projeto

2025-06-17

Table of Contents

# 1. Carregar bibliotecas necessárias  
message("Carregando bibliotecas essenciais para análise espacial e GWR...")

## Carregando bibliotecas essenciais para análise espacial e GWR...

library(sf) # Para manipulação de dados espaciais vetoriais (Simple Features)

## Linking to GEOS 3.13.1, GDAL 3.10.2, PROJ 9.5.1; sf\_use\_s2() is TRUE

library(GWmodel) # Para Geographically Weighted Regression (GWR) e funções relacionadas

## Carregando pacotes exigidos: robustbase

## Carregando pacotes exigidos: sp

## Carregando pacotes exigidos: Rcpp

## Welcome to GWmodel version 2.4-2.

library(dplyr) # Para manipulação e transformação de dados

##   
## Anexando pacote: 'dplyr'

## Os seguintes objetos são mascarados por 'package:stats':  
##   
## filter, lag

## Os seguintes objetos são mascarados por 'package:base':  
##   
## intersect, setdiff, setequal, union

library(ggplot2) # Para visualização e criação de mapas  
library(sp) # Necessário para o pacote GWmodel, que utiliza objetos Spatial\*DataFrame  
library(spdep) # Para testes de autocorrelação espacial como o I de Moran <--- ADICIONADO

## Carregando pacotes exigidos: spData

## To access larger datasets in this package, install the spDataLarge  
## package with: `install.packages('spDataLarge',  
## repos='https://nowosad.github.io/drat/', type='source')`

message("Bibliotecas carregadas com sucesso.")

## Bibliotecas carregadas com sucesso.

# --- INÍCIO: Definição dos caminhos dos arquivos ---  
# É uma boa prática centralizar a definição de caminhos para facilitar a manutenção  
# e a portabilidade do script. Considere usar 'here::here()' para caminhos relativos  
# ao projeto, o que torna o script mais robusto em diferentes ambientes.  
# Exemplo: path\_base <- here::here("data", "shp")  
# path\_roubos <- file.path(path\_base, "roubo.shp")  
# path\_drogas <- file.path(path\_base, "drogas.shp")  
# path\_area\_estudo <- file.path(path\_base, "centro\_expandido", "centro\_expandido\_dissolve.shp")  
  
path\_roubos <- "C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/roubo.shp"  
path\_drogas <- "C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/drogas.shp"  
path\_area\_estudo <- "C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/centro\_expandido\_dissolve.shp"  
# --- FIM: Definição dos caminhos dos arquivos ---  
  
# 2. Ler os arquivos shapefile  
message("Iniciando a leitura dos arquivos shapefile...")

## Iniciando a leitura dos arquivos shapefile...

tryCatch({  
 # st\_read com quiet = TRUE suprime mensagens de progresso, mantendo o output limpo.  
 pontos\_roubo <- st\_read(path\_roubos, quiet = TRUE)  
 pontos\_drogas <- st\_read(path\_drogas, quiet = TRUE)  
 area\_estudo <- st\_read(path\_area\_estudo, quiet = TRUE)  
}, error = function(e) {  
 # Em caso de erro na leitura (e.g., arquivo não encontrado, corrompido),  
 # o script é interrompido com uma mensagem clara.  
 stop("Erro crítico ao ler um ou mais arquivos shapefile. Verifique os caminhos e a integridade dos arquivos: ", e$message)  
})  
message("Arquivos shapefile carregados com sucesso.")

## Arquivos shapefile carregados com sucesso.

# 3. Definir CRS projetado alvo e transformar camadas  
# Para análises de distância e área, como GWR e criação de grades, é CRÍTICO usar  
# um CRS (Coordinate Reference System) projetado (em metros, quilômetros, etc.),  
# e não um CRS geográfico (em graus de latitude/longitude).  
crs\_projetado\_epsg <- 31983 # SIRGAS 2000 / UTM zone 23S - um CRS comum para o Brasil.  
crs\_projetado\_desejado <- st\_crs(crs\_projetado\_epsg)  
message(paste("CRS projetado desejado para a análise de distância e área: EPSG:", crs\_projetado\_epsg))

## CRS projetado desejado para a análise de distância e área: EPSG: 31983

# Função auxiliar para transformar CRS de forma segura e com feedback  
transformar\_crs\_se\_necessario <- function(sf\_object, target\_crs\_obj, nome\_camada) {  
 if (st\_crs(sf\_object) != target\_crs\_obj) {  
 message(paste("Transformando CRS da camada '", nome\_camada, "' (EPSG:", st\_crs(sf\_object)$epsg, ") para EPSG:", st\_crs(target\_crs\_obj)$epsg, "..."))  
 return(st\_transform(sf\_object, crs = target\_crs\_obj))  
 } else {  
 message(paste("CRS da camada '", nome\_camada, "' já é o desejado (EPSG:", st\_crs(target\_crs\_obj)$epsg, ")."))  
 return(sf\_object)  
 }  
}  
  
tryCatch({  
 area\_estudo <- transformar\_crs\_se\_necessario(area\_estudo, crs\_projetado\_desejado, "area\_estudo")  
 pontos\_roubo <- transformar\_crs\_se\_necessario(pontos\_roubo, crs\_projetado\_desejado, "pontos\_roubo")  
 pontos\_drogas <- transformar\_crs\_se\_necessario(pontos\_drogas, crs\_projetado\_desejado, "pontos\_drogas")  
}, error = function(e) {  
 stop("Erro durante a transformação de CRS de uma ou mais camadas: ", e$message)  
})

## Transformando CRS da camada ' area\_estudo ' (EPSG: 4326 ) para EPSG: 31983 ...

## Transformando CRS da camada ' pontos\_roubo ' (EPSG: 4326 ) para EPSG: 31983 ...

## Transformando CRS da camada ' pontos\_drogas ' (EPSG: 4326 ) para EPSG: 31983 ...

# Verificação final para garantir que todos os CRS foram harmonizados  
if (st\_crs(area\_estudo) != crs\_projetado\_desejado || st\_crs(pontos\_roubo) != crs\_projetado\_desejado || st\_crs(pontos\_drogas) != crs\_projetado\_desejado) {  
 stop("Falha crítica ao harmonizar CRS para todas as camadas. Verifique as configurações de CRS.")  
} else {  
 message(paste("Todas as camadas foram harmonizadas com sucesso para o CRS projetado (EPSG:", st\_crs(area\_estudo)$epsg, ")."))  
}

## Todas as camadas foram harmonizadas com sucesso para o CRS projetado (EPSG: 31983 ).

# Confirmação explícita de que o CRS é projetado para a criação da grade  
if (st\_is\_longlat(area\_estudo)) {  
 stop("O CRS da área de estudo ainda é geográfico (latitude/longitude). A criação da grade e a análise GWR requerem um CRS projetado (em unidades de distância, como metros).")  
} else {  
 message(paste("CRS para criação da grade é projetado (EPSG:", st\_crs(area\_estudo)$epsg, "), adequado para cálculos de distância e área."))  
}

## CRS para criação da grade é projetado (EPSG: 31983 ), adequado para cálculos de distância e área.

# --- Função Auxiliar para Impressão Robusta de Diagnósticos ---  
# Verifica se o valor é numérico e finito antes de imprimir  
print\_diagnostic\_robust <- function(value, name, description) {  
 if (!is.null(value) && is.numeric(value) && is.finite(value)) {  
 cat(paste0(name, ": ", round(value, 4), "\n")) # Aumenta precisão para AIC/R2/etc.  
 cat(paste0(" -> ", description, "\n"))  
 } else {  
 # Tenta extrair o valor mesmo que não seja numérico/finito para mostrar o que encontrou  
 val\_str <- if (is.null(value)) "NULL" else as.character(value)  
 cat(paste0(name, ": Não disponível ou não numérico/finito (Valor: ", val\_str, ").\n"))  
 cat(paste0(" -> ", description, " (Não calculado/Reportado)\n"))  
 }  
}  
  
# Nota: Certifique-se de que o pacote 'e1071' está instalado para Assimetria e Curtose dos Resíduos  
# install.packages('e1071') # Rode isto no console R se não tiver instalado  
# library(e1071) # Carregue o pacote no início do seu script

# 4. Criar uma grade hexagonal de análise sobre a área de estudo  
cell\_size\_m <- 800   
message(paste("Criando grade hexagonal de análise com 'side length' de", cell\_size\_m, "metros..."))

## Criando grade hexagonal de análise com 'side length' de 800 metros...

area\_estudo\_union <- st\_union(area\_estudo)  
grade\_base <- st\_make\_grid(area\_estudo\_union, cellsize = cell\_size\_m, what = "polygons", square = FALSE)  
grade\_sf\_obj <- st\_sf(id\_celula\_grade\_temp = 1:length(grade\_base), geometry = grade\_base)  
grade\_area\_estudo\_raw <- st\_intersection(grade\_sf\_obj, area\_estudo\_union)

## Warning: attribute variables are assumed to be spatially constant throughout  
## all geometries

grade\_area\_estudo <- grade\_area\_estudo\_raw[!st\_is\_empty(grade\_area\_estudo\_raw), ]  
grade\_area\_estudo <- grade\_area\_estudo[st\_is\_valid(grade\_area\_estudo$geometry), ]  
  
if(nrow(grade\_area\_estudo) == 0) {  
 stop("Nenhuma célula da grade intersecta a área de estudo ou todas as geometrias resultantes são inválidas/vazias. Verifique 'cell\_size\_m' e 'area\_estudo'.")  
}  
grade\_area\_estudo$id\_celula\_grade <- 1:nrow(grade\_area\_estudo)  
grade\_area\_estudo <- grade\_area\_estudo %>% dplyr::select(id\_celula\_grade, geometry)  
message(paste("Grade hexagonal de análise criada com", nrow(grade\_area\_estudo), "células válidas dentro da área de estudo."))

## Grade hexagonal de análise criada com 406 células válidas dentro da área de estudo.

# Bloco 5: Agregar os pontos à grade e aplicar transformação log1p  
message("Iniciando a agregação dos pontos de roubo e drogas às células da grade...")

## Iniciando a agregação dos pontos de roubo e drogas às células da grade...

agregar\_pontos\_a\_grade <- function(pontos\_sf, grade\_sf, nome\_variavel\_contagem) {  
 pontos\_na\_grade <- suppressMessages(st\_join(pontos\_sf, grade\_sf, join = st\_intersects))  
 contagem <- pontos\_na\_grade %>%  
 st\_drop\_geometry() %>%  
 filter(!is.na(id\_celula\_grade)) %>%   
 group\_by(id\_celula\_grade) %>%  
 summarise(!!sym(nome\_variavel\_contagem) := n(), .groups = 'drop')   
 return(contagem)  
}  
  
contagem\_roubos <- agregar\_pontos\_a\_grade(pontos\_roubo, grade\_area\_estudo, "n\_roubos")  
contagem\_drogas <- agregar\_pontos\_a\_grade(pontos\_drogas, grade\_area\_estudo, "n\_drogas")  
  
dados\_gwr\_sf <- grade\_area\_estudo %>%  
 left\_join(contagem\_roubos, by = "id\_celula\_grade") %>%  
 left\_join(contagem\_drogas, by = "id\_celula\_grade")  
  
dados\_gwr\_sf$n\_roubos[is.na(dados\_gwr\_sf$n\_roubos)] <- 0  
dados\_gwr\_sf$n\_drogas[is.na(dados\_gwr\_sf$n\_drogas)] <- 0  
message("Dados de roubos e drogas agregados às células da grade.")

## Dados de roubos e drogas agregados às células da grade.

message("Aplicando transformação log1p às variáveis de contagem (n\_roubos, n\_drogas)...")

## Aplicando transformação log1p às variáveis de contagem (n\_roubos, n\_drogas)...

dados\_gwr\_sf <- dados\_gwr\_sf %>%  
 mutate(  
 log1p\_n\_roubos = log1p(n\_roubos),  
 log1p\_n\_drogas = log1p(n\_drogas)  
 )  
message("Transformação log1p aplicada. Novas colunas criadas: 'log1p\_n\_roubos' e 'log1p\_n\_drogas'.")

## Transformação log1p aplicada. Novas colunas criadas: 'log1p\_n\_roubos' e 'log1p\_n\_drogas'.

formula\_gwr <- log1p\_n\_roubos ~ log1p\_n\_drogas  
message(paste("Fórmula GWR definida como:", deparse(formula\_gwr)))

## Fórmula GWR definida como: log1p\_n\_roubos ~ log1p\_n\_drogas

# 6. Preparar dados para GWR  
message("Preparando dados para GWR: convertendo para formato SpatialPointsDataFrame (SPDF)...")

## Preparando dados para GWR: convertendo para formato SpatialPointsDataFrame (SPDF)...

dados\_gwr\_sf\_validos <- NULL  
dados\_spdf\_gwr <- NULL  
  
tryCatch({  
 dados\_gwr\_sf\_validos <- dados\_gwr\_sf  
 if(nrow(dados\_gwr\_sf\_validos) == 0) stop("Nenhuma célula com geometria válida após a agregação. Impossível prosseguir com GWR.")  
  
 message("Calculando pontos representativos (st\_point\_on\_surface) para as células da grade...")  
 locais\_regressao\_sf\_centroids <- st\_point\_on\_surface(dados\_gwr\_sf\_validos)  
 coords\_locais\_regressao <- st\_coordinates(locais\_regressao\_sf\_centroids)  
  
 if(any(is.na(coords\_locais\_regressao))) {  
 stop("Coordenadas NA encontradas após st\_point\_on\_surface. Isso pode indicar geometrias problemáticas em 'dados\_gwr\_sf\_validos'.")  
 }  
  
 message("Extraindo atributos (incluindo colunas transformadas log1p) para o SPDF...")  
 dados\_atributos\_para\_spdf <- st\_drop\_geometry(dados\_gwr\_sf\_validos)  
  
 dados\_spdf\_gwr <- SpatialPointsDataFrame(coords = coords\_locais\_regressao,  
 data = dados\_atributos\_para\_spdf,   
 proj4string = CRS(st\_crs(dados\_gwr\_sf\_validos)$proj4string))  
  
 message(paste("Dados SPDF preparados com sucesso. Número de Pontos SPDF para GWR:", length(dados\_spdf\_gwr)))  
 message(paste("Nomes das colunas de atributos em dados\_spdf\_gwr@data:", paste(names(dados\_spdf\_gwr@data), collapse=", ")))  
   
 if(!("log1p\_n\_roubos" %in% names(dados\_spdf\_gwr@data) && "log1p\_n\_drogas" %in% names(dados\_spdf\_gwr@data))){  
 warning("AVISO: Colunas 'log1p\_n\_roubos' ou 'log1p\_n\_drogas' não encontradas em dados\_spdf\_gwr@data. Verifique o Bloco 5.")  
 } else {  
 message("Confirmação: Colunas 'log1p\_n\_roubos' e 'log1p\_n\_drogas' estão presentes em dados\_spdf\_gwr@data.")  
 }  
}, error = function(e) {  
 stop(paste("Erro ao preparar dados para GWR (conversão para SPDF):", e$message))  
})

## Calculando pontos representativos (st\_point\_on\_surface) para as células da grade...

## Warning: st\_point\_on\_surface assumes attributes are constant over geometries

## Extraindo atributos (incluindo colunas transformadas log1p) para o SPDF...

## Dados SPDF preparados com sucesso. Número de Pontos SPDF para GWR: 406

## Nomes das colunas de atributos em dados\_spdf\_gwr@data: id\_celula\_grade, n\_roubos, n\_drogas, log1p\_n\_roubos, log1p\_n\_drogas

## Confirmação: Colunas 'log1p\_n\_roubos' e 'log1p\_n\_drogas' estão presentes em dados\_spdf\_gwr@data.

# --- INÍCIO: Inspeção da Esparsidade dos Dados ---  
message("------------------------------------------------------")

## ------------------------------------------------------

message("Inspeção da Esparsidade dos Dados para GWR:")

## Inspeção da Esparsidade dos Dados para GWR:

message("------------------------------------------------------")

## ------------------------------------------------------

if (!is.null(dados\_spdf\_gwr) && nrow(dados\_spdf\_gwr@data) > 0) {  
 cat("Resumo estatístico para 'n\_roubos' (variável dependente):\n")  
 print(summary(dados\_spdf\_gwr$n\_roubos))  
 cat("\nFrequência de células com e sem ocorrências de roubos:\n")  
 print(table(Com\_Roubos = dados\_spdf\_gwr$n\_roubos > 0))  
 message(paste("\nPorcentagem de células SEM roubos:",  
 round(sum(dados\_spdf\_gwr$n\_roubos == 0) / length(dados\_spdf\_gwr$n\_roubos) \* 100, 2), "%"))  
  
 cat("\nResumo estatístico para 'n\_drogas' (variável independente):\n")  
 print(summary(dados\_spdf\_gwr$n\_drogas))  
 cat("\nFrequência de células com e sem ocorrências de drogas:\n")  
 print(table(Com\_Drogas = dados\_spdf\_gwr$n\_drogas > 0))  
 message(paste("\nPorcentagem de células SEM drogas:",  
 round(sum(dados\_spdf\_gwr$n\_drogas == 0) / length(dados\_spdf\_gwr$n\_drogas) \* 100, 2), "%"))  
  
 message(paste("\nNúmero de células com n\_roubos = 0 E n\_drogas = 0:",  
 sum(dados\_spdf\_gwr$n\_roubos == 0 & dados\_spdf\_gwr$n\_drogas == 0)))  
 message(paste("Porcentagem de células com n\_roubos = 0 E n\_drogas = 0:",  
 round(sum(dados\_spdf\_gwr$n\_roubos == 0 & dados\_spdf\_gwr$n\_drogas == 0) / length(dados\_spdf\_gwr$n\_roubos) \* 100, 2), "%"))  
  
 message(paste("\nTotal de células (pontos de regressão) na análise GWR:", length(dados\_spdf\_gwr)))  
 if (length(dados\_spdf\_gwr) < 50) {  
 warning("O número de células para a análise GWR é baixo (< 50). Isso pode afetar a estabilidade dos resultados e a busca pela largura de banda ótima. Considere aumentar o 'cell\_size\_m' ou a área de estudo.")  
 }  
} else {  
 warning("dados\_spdf\_gwr está vazio ou nulo. Não foi possível inspecionar a esparsidade dos dados.")  
}

## Resumo estatístico para 'n\_roubos' (variável dependente):  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.000 0.000 1.000 4.335 3.000 177.000   
##   
## Frequência de células com e sem ocorrências de roubos:  
## Com\_Roubos  
## FALSE TRUE   
## 159 247

##   
## Porcentagem de células SEM roubos: 39.16 %

##   
## Resumo estatístico para 'n\_drogas' (variável independente):  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.0 25.0 60.0 128.5 129.5 1552.0   
##   
## Frequência de células com e sem ocorrências de drogas:  
## Com\_Drogas  
## FALSE TRUE   
## 19 387

##   
## Porcentagem de células SEM drogas: 4.68 %

##   
## Número de células com n\_roubos = 0 E n\_drogas = 0: 16

## Porcentagem de células com n\_roubos = 0 E n\_drogas = 0: 3.94 %

##   
## Total de células (pontos de regressão) na análise GWR: 406

message("------------------------------------------------------\n")

## ------------------------------------------------------

# --- FIM: Inspeção da Esparsidade dos Dados ---  
  
# 7. Calcular a largura de banda ótima e matriz de distância  
message(paste("Fórmula GWR definida como:", deparse(formula\_gwr)))

## Fórmula GWR definida como: log1p\_n\_roubos ~ log1p\_n\_drogas

if(is.null(dados\_spdf\_gwr) || length(dados\_spdf\_gwr) < 30) { # Mínimo sugerido para GWR robusto  
 stop(paste("Número insuficiente de pontos (<30) para GWR ou dados\_spdf\_gwr é nulo. Atualmente:",   
 ifelse(is.null(dados\_spdf\_gwr), 0, length(dados\_spdf\_gwr)),  
 "\nConsidere aumentar 'cell\_size\_m' ou verificar a área de estudo e a agregação de dados."))  
}  
  
vars\_na\_formula <- all.vars(formula\_gwr)  
if(!all(vars\_na\_formula %in% names(dados\_spdf\_gwr@data))){  
 colunas\_faltantes <- setdiff(vars\_na\_formula, names(dados\_spdf\_gwr@data))  
 stop(paste("ERRO CRÍTICO: Variáveis da fórmula ('", paste(colunas\_faltantes, collapse=", "), "') não encontradas em 'dados\_spdf\_gwr@data'. Verifique os Blocos 5 e 6."))  
} else {  
 message(paste("Confirmação: Todas as variáveis da fórmula ('", paste(vars\_na\_formula, collapse=", "), "') estão presentes em 'dados\_spdf\_gwr@data'."))  
}

## Confirmação: Todas as variáveis da fórmula (' log1p\_n\_roubos, log1p\_n\_drogas ') estão presentes em 'dados\_spdf\_gwr@data'.

# --- INÍCIO DA CORREÇÃO: Calcular a matriz de distância ---  
message("Calculando a matriz de distância para a seleção da largura de banda...")

## Calculando a matriz de distância para a seleção da largura de banda...

dMat\_calibracao <- NULL   
if (is.null(dados\_spdf\_gwr) || nrow(dados\_spdf\_gwr) == 0) {  
 stop("dados\_spdf\_gwr está vazio ou nulo. Não é possível calcular a matriz de distância.")  
}  
coords\_para\_dmat <- coordinates(dados\_spdf\_gwr)  
if(any(is.na(coords\_para\_dmat))) {  
 stop("Coordenadas NA encontradas em dados\_spdf\_gwr. Impossível calcular a matriz de distância.")  
}  
  
# Verifica se o CRS é geográfico (longlat=TRUE) ou projetado (longlat=FALSE)  
# O CRS foi definido como projetado no Bloco 3.  
crs\_spdf <- st\_crs(dados\_spdf\_gwr)  
is\_longlat\_check <- FALSE # Default para projetado  
if (!is.na(crs\_spdf) && !is.null(crs\_spdf)) {  
 is\_longlat\_check <- st\_is\_longlat(crs\_spdf)  
 if (is.na(is\_longlat\_check)) { # Se st\_is\_longlat retorna NA  
 warning("Não foi possível determinar programaticamente se o CRS é longlat (st\_is\_longlat retornou NA). Assumindo projetado (longlat=FALSE) com base na configuração do script (EPSG:31983).")  
 is\_longlat\_check <- FALSE   
 }  
} else {  
 warning("CRS de dados\_spdf\_gwr é NA ou NULL. Assumindo projetado (longlat=FALSE). Verifique a preparação dos dados.")  
 is\_longlat\_check <- FALSE  
}  
message(paste("Para o cálculo da matriz de distância: longlat =", is\_longlat\_check))

## Para o cálculo da matriz de distância: longlat = FALSE

tryCatch({  
 dMat\_calibracao <- GWmodel::gw.dist(  
 dp.locat = coords\_para\_dmat,  
 # rp.locat = coords\_para\_dmat, # Não é necessário para bw.gwr/bw.ggwr se dp.locat é usado para cálculo interno da matriz de pesos  
 p = 2, # Distância Euclidiana  
 theta = 0, # Nenhuma rotação  
 longlat = is\_longlat\_check   
 )  
 message("Matriz de distância 'dMat\_calibracao' calculada com sucesso.")  
 message(paste("Dimensões da dMat\_calibracao:", paste(dim(dMat\_calibracao), collapse = "x")))  
}, error = function(e) {  
 stop(paste("Erro ao calcular a matriz de distância 'dMat\_calibracao' com gw.dist:", e$message))  
})

## Matriz de distância 'dMat\_calibracao' calculada com sucesso.

## Dimensões da dMat\_calibracao: 406x406

if (is.null(dMat\_calibracao) || !is.matrix(dMat\_calibracao) || nrow(dMat\_calibracao) != length(dados\_spdf\_gwr) || ncol(dMat\_calibracao) != length(dados\_spdf\_gwr)) {  
 stop("A matriz de distância 'dMat\_calibracao' não foi criada corretamente ou tem dimensões inválidas.")  
}  
# --- FIM DA CORREÇÃO ---  
  
message("Iniciando a busca pela largura de banda adaptativa ótima...")

## Iniciando a busca pela largura de banda adaptativa ótima...

bw\_adaptativa <- NULL  
error\_message\_bw <- ""   
kernel\_usado\_para\_bw <- "gaussian" # Default  
  
tentativas\_bw\_gwr <- list(  
 list(approach = "AIC", kernel = "gaussian"),  
 list(approach = "CV", kernel = "gaussian"),  
 list(approach = "AIC", kernel = "bisquare"),  
 list(approach = "CV", kernel = "bisquare")  
)  
  
tentativas\_bw\_ggwr <- list(  
 list(approach = "AICc", kernel = "gaussian"),  
 list(approach = "CV", kernel = "gaussian"),  
 list(approach = "AICc", kernel = "bisquare"),  
 list(approach = "CV", kernel = "bisquare")  
)  
  
funcao\_bw\_usada <- ""  
  
message("\n--- Priorizando GWmodel::bw.gwr (específico para GWR Gaussiano) ---")

##   
## --- Priorizando GWmodel::bw.gwr (específico para GWR Gaussiano) ---

for (i in 1:length(tentativas\_bw\_gwr)) {  
 params <- tentativas\_bw\_gwr[[i]]  
 message(paste0("\n--- Tentativa com bw.gwr ", i, ": approach = '", params$approach, "', kernel = '", params$kernel, "' ---"))  
 tryCatch({  
 bw\_adaptativa\_temp <- GWmodel::bw.gwr(   
 formula = formula\_gwr,  
 data = dados\_spdf\_gwr,  
 dMat = dMat\_calibracao, # AGORA dMat\_calibracao EXISTE  
 approach = params$approach,   
 kernel = params$kernel,  
 adaptive = TRUE  
 )  
 if (!is.null(bw\_adaptativa\_temp) && is.numeric(bw\_adaptativa\_temp) && bw\_adaptativa\_temp > 0 && bw\_adaptativa\_temp <= nrow(dados\_spdf\_gwr@data)) {  
 bw\_adaptativa <- bw\_adaptativa\_temp  
 kernel\_usado\_para\_bw <- params$kernel  
 funcao\_bw\_usada <- "bw.gwr"  
 message(paste("Sucesso com bw.gwr na Tentativa ", i, "! Largura de banda:", round(bw\_adaptativa, 0), "vizinhos, kernel:", kernel\_usado\_para\_bw))  
 break   
 } else {  
 msg <- paste0("bw.gwr Tentativa ", i, " (", params$approach, ", ", params$kernel, ") não retornou uma largura de banda válida (valor: ", bw\_adaptativa\_temp, ").")  
 message(msg)  
 error\_message\_bw <- paste0(error\_message\_bw, msg, "; ")   
 }  
 }, error = function(e) {  
 message(paste("ERRO com bw.gwr na Tentativa ", i, " (", params$approach, ", ", params$kernel, "):", e$message))  
 error\_message\_bw <- paste0(error\_message\_bw, "Erro bw.gwr (", params$approach, ",", params$kernel, "): ", e$message, "; ")   
 })  
 if (!is.null(bw\_adaptativa)) break   
}

##   
## --- Tentativa com bw.gwr 1: approach = 'AIC', kernel = 'gaussian' ---

## Adaptive bandwidth (number of nearest neighbours): 258 AICc value: 1027.573   
## Adaptive bandwidth (number of nearest neighbours): 167 AICc value: 1013.533   
## Adaptive bandwidth (number of nearest neighbours): 110 AICc value: 995.649   
## Adaptive bandwidth (number of nearest neighbours): 75 AICc value: 977.411   
## Adaptive bandwidth (number of nearest neighbours): 53 AICc value: 957.1655   
## Adaptive bandwidth (number of nearest neighbours): 40 AICc value: 950.0465   
## Adaptive bandwidth (number of nearest neighbours): 31 AICc value: 929.7529   
## Adaptive bandwidth (number of nearest neighbours): 26 AICc value: 929.3024   
## Adaptive bandwidth (number of nearest neighbours): 22 AICc value: 928.6423   
## Adaptive bandwidth (number of nearest neighbours): 20 AICc value: 924.5499   
## Adaptive bandwidth (number of nearest neighbours): 18 AICc value: 911.9409   
## Adaptive bandwidth (number of nearest neighbours): 18 AICc value: 911.9409

## Sucesso com bw.gwr na Tentativa 1 ! Largura de banda: 18 vizinhos, kernel: gaussian

if (is.null(bw\_adaptativa)) {  
 message("\n--- bw.gwr falhou ou não encontrou largura de banda. Tentando GWmodel::bw.ggwr (Gaussiano implícito) ---")  
 for (i in 1:length(tentativas\_bw\_ggwr)) {  
 params <- tentativas\_bw\_ggwr[[i]]  
 message(paste0("\n--- Tentativa com bw.ggwr ", i, ": approach = '", params$approach, "', kernel = '", params$kernel, "' ---"))  
 tryCatch({  
 bw\_adaptativa\_temp <- GWmodel::bw.ggwr(  
 formula = formula\_gwr,  
 data = dados\_spdf\_gwr,  
 dMat = dMat\_calibracao, # AGORA dMat\_calibracao EXISTE  
 approach = params$approach,  
 kernel = params$kernel,  
 adaptive = TRUE  
 )  
 if (!is.null(bw\_adaptativa\_temp) && is.numeric(bw\_adaptativa\_temp) && bw\_adaptativa\_temp > 0 && bw\_adaptativa\_temp <= nrow(dados\_spdf\_gwr@data)) {  
 bw\_adaptativa <- bw\_adaptativa\_temp  
 kernel\_usado\_para\_bw <- params$kernel  
 funcao\_bw\_usada <- "bw.ggwr"  
 message(paste("Sucesso com bw.ggwr na Tentativa ", i, "! Largura de banda:", round(bw\_adaptativa, 0), "vizinhos, kernel:", kernel\_usado\_para\_bw))  
 break   
 } else {  
 msg <- paste0("bw.ggwr Tentativa ", i, " (", params$approach, ", ", params$kernel, ") não retornou uma largura de banda válida (valor: ", bw\_adaptativa\_temp, ").")  
 message(msg)  
 error\_message\_bw <- paste0(error\_message\_bw, msg, "; ")   
 }  
 }, error = function(e) {  
 message(paste("ERRO com bw.ggwr na Tentativa ", i, " (", params$approach, ", ", params$kernel, "):", e$message))  
 if (grepl("the condition has length > 1", e$message, fixed = TRUE)) {  
 message("AVISO: O erro 'the condition has length > 1' persistiu mesmo sem especificar 'family' em bw.ggwr.")  
 }  
 error\_message\_bw <- paste0(error\_message\_bw, "Erro bw.ggwr (", params$approach, ",", params$kernel, "): ", e$message, "; ")   
 })  
 if (!is.null(bw\_adaptativa)) break   
 }  
}  
  
if (is.null(bw\_adaptativa)) {  
 cat("-------------------------------------------------------------------------------------\n")  
 cat("ATENÇÃO: Não foi possível calcular automaticamente a largura de banda ótima (bw\_adaptativa)\n")  
 cat("após todas as tentativas com bw.gwr e bw.ggwr (para modelo Gaussiano).\n")   
 cat("Mensagens de erro/aviso acumuladas das tentativas:\n", error\_message\_bw, "\n\n")  
 cat("Este problema frequentemente ocorre devido a:\n")  
 cat(" a) Esparsidade dos dados (mesmo após transformação log1p, muitos valores 'log1p(0)=0' podem ser problemáticos se levarem a pouca variância local).\n")   
 cat(" b) Colinearidade local: Variáveis independentes fortemente correlacionadas em algumas vizinhanças locais.\n")  
 cat(" c) Número insuficiente de observações (células) para o GWR, especialmente se a largura de banda ótima for muito pequena.\n")  
 cat(" d) Se o erro 'valor ausente onde TRUE/FALSE necessário' ou 'the condition has length > 1' persistiu, pode ser um problema nos dados ou no pacote.\n\n")   
   
 cat("Próximas Etapas Sugeridas para Solução:\n")  
 cat("1. REVISE CUIDADOSAMENTE O OUTPUT DA 'Inspeção da Esparsidade dos Dados' e as mensagens de erro detalhadas das tentativas de bw.gwr/bw.ggwr.\n")  
 cat("2. TENTAR UMA LARGURA DE BANDA ADAPTATIVA MANUALMENTE MAIOR (PARA DIAGNÓSTICO):\n")  
 cat("# bw\_adaptativa <- round(0.1 \* nrow(dados\_spdf\_gwr@data)) # Exemplo: 10% dos pontos\n")  
 cat('# kernel\_usado\_para\_bw <- "gaussian" \n')  
 cat('# funcao\_bw\_usada <- "manual"\n')  
 cat('# message(paste("Usando largura de banda adaptativa definida manualmente:", bw\_adaptativa, "vizinhos e kernel:", kernel\_usado\_para\_bw))\n\n')  
 cat("3. AUMENTAR O TAMANHO DA CÉLULA DA GRADE ('cell\_size\_m').\n")  
 cat("4. RECONSIDERAR AS VARIÁVEIS DO MODELO OU A TRANSFORMAÇÃO.\n")  
 cat("5. VERIFICAR A VERSÃO DO PACOTE GWmodel E ATUALIZAR SE POSSÍVEL.\n\n")  
 cat("-------------------------------------------------------------------------------------\n")  
 stop("Cálculo automático da largura de banda falhou. Analise as mensagens e sugestões acima para depuração.")  
} else {  
 message(paste("\nLargura de banda adaptativa ótima calculada com sucesso usando", funcao\_bw\_usada, ":", round(bw\_adaptativa, 0),   
 "vizinhos, utilizando o kernel:", kernel\_usado\_para\_bw, "."))  
 bw\_adaptativa <- round(bw\_adaptativa, 0) # Garante que seja um inteiro  
 if (bw\_adaptativa < 2) { # Uma largura de banda de 1 vizinho não faz sentido  
 warning(paste0("A largura de banda calculada (", bw\_adaptativa, ") é muito pequena. Pode indicar problemas ou levar a resultados instáveis. Considerar forçar um mínimo."))  
 # Exemplo: bw\_adaptativa <- max(bw\_adaptativa, 20)   
 }  
 message(paste("Largura de banda final (arredondada para o inteiro mais próximo):", bw\_adaptativa, "vizinhos."))  
}

##   
## Largura de banda adaptativa ótima calculada com sucesso usando bw.gwr : 18 vizinhos, utilizando o kernel: gaussian .

## Largura de banda final (arredondada para o inteiro mais próximo): 18 vizinhos.

message(paste("\nCONFIRMAÇÃO ANTES DO GWR: Usando largura de banda adaptativa (número de vizinhos):", bw\_adaptativa,  
 "e kernel:", kernel\_usado\_para\_bw, "para gwr.basic (modelo Gaussiano implícito)."))

##   
## CONFIRMAÇÃO ANTES DO GWR: Usando largura de banda adaptativa (número de vizinhos): 18 e kernel: gaussian para gwr.basic (modelo Gaussiano implícito).

if (is.null(bw\_adaptativa) || !is.numeric(bw\_adaptativa) || bw\_adaptativa <= 0) {  
 stop("ERRO CRÍTICO: 'bw\_adaptativa' é inválida (NULL, não numérica ou não positiva) antes de iniciar a Etapa 8 (gwr.basic). Verifique o cálculo da largura de banda.")  
}  
if (is.null(kernel\_usado\_para\_bw) || !kernel\_usado\_para\_bw %in% c("gaussian", "bisquare", "tricube", "boxcar", "exponential")) { # Kernels comuns para GWmodel  
 stop(paste("ERRO CRÍTICO: 'kernel\_usado\_para\_bw' ('", kernel\_usado\_para\_bw, "') é inválido para gwr.basic. Verifique os nomes dos kernels suportados."))  
}

# 8. Executar o GWR usando gwr.basic  
message("Executando a Regressão Geograficamente Ponderada (GWR) com gwr.basic (modelo Gaussiano implícito)...")

## Executando a Regressão Geograficamente Ponderada (GWR) com gwr.basic (modelo Gaussiano implícito)...

gwr\_resultado\_lista <- NULL  
  
tryCatch({  
 gwr\_resultado\_lista <- GWmodel::gwr.basic(  
 formula = formula\_gwr,  
 data = dados\_spdf\_gwr,  
 bw = bw\_adaptativa,  
 kernel = kernel\_usado\_para\_bw,  
 adaptive = TRUE,  
 dMat = dMat\_calibracao  
 )  
}, error = function(e) {  
 cat("Erro crítico ao executar gwr.basic (modelo Gaussiano implícito):\n", e$message, "\n")  
   
 if (grepl("the condition has length > 1", e$message, fixed = TRUE)) {  
 message("AVISO: O erro 'the condition has length > 1' persistiu em gwr.basic mesmo sem especificar 'family'. Isso indica um problema fundamental no pacote GWmodel que afeta até mesmo o GWR Gaussiano padrão.")  
 }  
 stop(paste("Falha ao executar gwr.basic (modelo Gaussiano implícito). Verifique a largura de banda, o kernel, os dados de entrada (dados\_spdf\_gwr), e a matriz de distância (dMat\_calibracao).\n",  
 "Causas comuns (além de bugs no pacote) incluem: singularidade devido a poucos vizinhos, dados problemáticos, ou dMat incorreta."))  
})  
  
if (is.null(gwr\_resultado\_lista) || is.null(gwr\_resultado\_lista$SDF)) {  
 stop("Falha ao executar gwr.basic (modelo Gaussiano implícito) ou o resultado não contém o componente 'SDF' (SpatialDataFrame com os resultados).")  
} else {  
 message("Análise GWR (gwr.basic, modelo Gaussiano implícito) concluída com sucesso. Processando os resultados...")  
}

## Análise GWR (gwr.basic, modelo Gaussiano implícito) concluída com sucesso. Processando os resultados...

gwr\_resultados\_sdf <- gwr\_resultado\_lista$SDF  
message(paste("Resultados GWR (SDF) obtidos. Nomes das colunas no SDF do GWR:", paste(names(gwr\_resultados\_sdf), collapse=", ")))

## Resultados GWR (SDF) obtidos. Nomes das colunas no SDF do GWR: Intercept, log1p\_n\_drogas, y, yhat, residual, CV\_Score, Stud\_residual, Intercept\_SE, log1p\_n\_drogas\_SE, Intercept\_TV, log1p\_n\_drogas\_TV, Local\_R2

# Verificar se a variável dependente no SDF corresponde à VD da fórmula  
var\_dependente\_formula <- all.vars(formula\_gwr)[1]  
if ("y" %in% names(gwr\_resultados\_sdf)) {  
 if (all(gwr\_resultados\_sdf$y == dados\_spdf\_gwr@data[[var\_dependente\_formula]], na.rm = TRUE)) {  
 message(paste0("Confirmação: A coluna 'y' no SDF do GWR corresponde aos valores de '", var\_dependente\_formula, "' dos dados de entrada."))  
 } else {  
 warning(paste0("AVISO: A coluna 'y' no SDF do GWR pode não corresponder exatamente aos valores de '", var\_dependente\_formula, "' dos dados de entrada. Verifique."))  
 }  
} else {  
 warning("AVISO: Coluna 'y' (variável dependente) não encontrada no SDF do GWR.")  
}

## Confirmação: A coluna 'y' no SDF do GWR corresponde aos valores de 'log1p\_n\_roubos' dos dados de entrada.

# Verificar se a coluna da variável independente principal da fórmula existe nos resultados  
var\_independente\_principal\_formula <- all.vars(formula\_gwr)[2]  
if (!(var\_independente\_principal\_formula %in% names(gwr\_resultados\_sdf))) {  
 warning(paste0("AVISO: A coluna do coeficiente para '", var\_independente\_principal\_formula, "' não foi encontrada diretamente com esse nome no SDF do GWR. Verifique os nomes das colunas listados acima."))  
}  
  
  
if(nrow(dados\_gwr\_sf\_validos) == nrow(gwr\_resultados\_sdf)) {  
 # Extrair os dados do SDF do GWR para um data.frame  
 gwr\_resultados\_df\_para\_join <- as.data.frame(gwr\_resultados\_sdf)  
  
 # Adicionar 'id\_celula\_grade' de 'dados\_gwr\_sf\_validos' aos resultados do GWR  
 if ("id\_celula\_grade" %in% names(dados\_gwr\_sf\_validos)) {  
 gwr\_resultados\_df\_para\_join$id\_celula\_grade <- dados\_gwr\_sf\_validos$id\_celula\_grade  
 message("Coluna 'id\_celula\_grade' adicionada aos resultados do GWR a partir de 'dados\_gwr\_sf\_validos'.")  
 } else {  
 stop("ERRO CRÍTICO: 'id\_celula\_grade' não encontrada em 'dados\_gwr\_sf\_validos'. Impossível adicionar ao join e garantir a integridade dos dados.")  
 }  
  
 # Verificar se 'id\_celula\_grade' está presente  
 if (!"id\_celula\_grade" %in% names(gwr\_resultados\_df\_para\_join)) {  
 stop("Falha inesperada ao adicionar 'id\_celula\_grade' a 'gwr\_resultados\_df\_para\_join'. Verifique a lógica de atribuição.")  
 }  
  
 # Renomear a coluna do coeficiente 'n\_drogas' para 'coef\_n\_drogas'  
 if ("n\_drogas" %in% names(gwr\_resultados\_df\_para\_join)) {  
 names(gwr\_resultados\_df\_para\_join)[names(gwr\_resultados\_df\_para\_join) == "n\_drogas"] <- "coef\_n\_drogas"  
 message("Coluna do coeficiente para 'n\_drogas' nos resultados GWR renomeada para 'coef\_n\_drogas'.")  
 } else {  
 warning("Coluna do coeficiente para 'n\_drogas' não encontrada diretamente nos resultados do GWR. Verifique os nomes das variáveis no modelo GWR e o output do GWmodel.")  
 }  
  
 # Definir as colunas de resultados do GWR que serão mantidas  
 cols\_gwr\_a\_manter <- c("id\_celula\_grade",  
 "Intercept", "Intercept\_TV",  
 "coef\_n\_drogas", "n\_drogas\_TV",  
 "Local\_R2", "y", "yhat", "residual")  
  
 # Filtrar colunas para o join  
 cols\_existentes\_no\_join\_df <- intersect(cols\_gwr\_a\_manter, names(gwr\_resultados\_df\_para\_join))  
 gwr\_resultados\_filtrados\_df <- gwr\_resultados\_df\_para\_join %>%  
 dplyr::select(dplyr::all\_of(cols\_existentes\_no\_join\_df))  
  
 # Realizar o join dos resultados do GWR com o objeto sf dos polígonos da grade  
 resultados\_gwr\_poligonos\_sf <- dplyr::left\_join(dados\_gwr\_sf\_validos,  
 gwr\_resultados\_filtrados\_df,  
 by = "id\_celula\_grade")  
  
 message("Resultados GWR combinados com os polígonos originais da grade usando 'id\_celula\_grade'.")  
  
 # Verificação pós-join  
 if (any(is.na(resultados\_gwr\_poligonos\_sf$Local\_R2))) {  
 warning("NAs encontrados em 'Local\_R2' após o join. Isso pode indicar problemas na correspondência dos IDs ou dados ausentes em algumas células da grade.")  
 }  
  
} else {  
 stop(paste("Número de linhas nos resultados GWR (", nrow(gwr\_resultados\_sdf),  
 ") não corresponde aos dados de polígonos originais (", nrow(dados\_gwr\_sf\_validos),  
 "). Não é possível combinar os resultados de forma segura. Verifique a integridade dos dados antes do GWR ou a função de modelagem utilizada."))  
}

## Coluna 'id\_celula\_grade' adicionada aos resultados do GWR a partir de 'dados\_gwr\_sf\_validos'.

## Warning: Coluna do coeficiente para 'n\_drogas' não encontrada diretamente nos  
## resultados do GWR. Verifique os nomes das variáveis no modelo GWR e o output do  
## GWmodel.

## Resultados GWR combinados com os polígonos originais da grade usando 'id\_celula\_grade'.

# --- Bloco 9: Combinar Resultados GWR aos Polígonos e Gerar Mapas Essenciais (SOMENTE RESULTADOS GWR) ---  
# Combina os resultados tabulares do GWR (coeficientes, R2, etc.) de volta ao objeto espacial dos polígonos  
# e gera os mapas essenciais para visualização e análise, FOCANDO APENAS NOS RESULTADOS DO MODELO GWR.  
cat("\n\n--- BL. 9: COMBINANDO RESULTADOS GWR E GERANDO MAPAS ESSENCIAIS (SOMENTE RESULTADOS GWR) ---")

##   
##   
## --- BL. 9: COMBINANDO RESULTADOS GWR E GERANDO MAPAS ESSENCIAIS (SOMENTE RESULTADOS GWR) ---

cat("\n----------------------------------------------------------------------------------------\n")

##   
## ----------------------------------------------------------------------------------------

# --- 9.1. Verificar objetos necessários ---  
# Verifica se os principais objetos gerados nos blocos anteriores existem, não são NULL e não estão vazios.  
message("--- 9.1. Verificando objetos necessários ---")

## --- 9.1. Verificando objetos necessários ---

if (!exists("dados\_gwr\_sf\_validos") || is.null(dados\_gwr\_sf\_validos) || nrow(dados\_gwr\_sf\_validos) == 0) {  
 stop("ERRO CRÍTICO Bloco 9: O objeto 'dados\_gwr\_sf\_validos' (polígonos da grade com dados originais) não foi encontrado, é NULL ou está vazio.")  
}  
if (!exists("gwr\_resultado\_lista") || is.null(gwr\_resultado\_lista) || is.null(gwr\_resultado\_lista$SDF) || nrow(gwr\_resultado\_lista$SDF) == 0) {  
 stop("ERRO CRÍTICO Bloco 9: O objeto 'gwr\_resultado\_lista' (resultado do GWR) não foi encontrado, é NULL, seu componente SDF é NULL ou o SDF está vazio.")  
}  
  
# Verifica se o objeto da área de estudo (contorno para mapas) existe. Emite apenas aviso se não.  
if (!exists("area\_estudo\_union") || is.null(area\_estudo\_union)) {  
 warning("AVISO Bloco 9: Objeto 'area\_estudo\_union' (contorno da área de estudo) não encontrado. A camada de contorno não será adicionada aos mapas.")  
 area\_estudo\_union\_mapa <- NULL # Define como NULL para o bloco de plotagem  
} else {  
 # Se existir, verifica se o CRS coincide com os resultados GWR antes de usá-lo nos mapas  
 if (st\_crs(area\_estudo\_union) != st\_crs(dados\_gwr\_sf\_validos)) { # Compara com dados\_gwr\_sf\_validos que já tem o CRS projetado  
 warning("AVISO Bloco 9: CRS da área de estudo difere do CRS dos resultados GWR. A área de estudo não será adicionada como contorno.")  
 area\_estudo\_union\_mapa <- NULL  
 } else {  
 area\_estudo\_union\_mapa <- area\_estudo\_union # Usa o objeto se o CRS for compatível  
 }  
}  
  
message("Objetos de dados necessários para o Bloco 9 encontrados. Prosseguindo com a combinação de resultados.")

## Objetos de dados necessários para o Bloco 9 encontrados. Prosseguindo com a combinação de resultados.

# --- 9.2. Preparar Resultados GWR para Join ---  
# O SDF do GWR contém os resultados (coeficientes, R2, resíduos) nos centróides.  
# Precisamos extrair estes dados (em formato data.frame) para juntá-los aos polígonos SF originais.  
message("\n--- 9.2. Extraindo resultados do GWR (SDF) para data.frame ---")

##   
## --- 9.2. Extraindo resultados do GWR (SDF) para data.frame ---

gwr\_resultados\_df\_para\_join <- as.data.frame(gwr\_resultado\_lista$SDF)  
  
# Adicionar a chave de join ('id\_celula\_grade') ao data.frame de resultados do GWR.  
# Esta coluna DEVE existir em 'dados\_gwr\_sf\_validos' (creada no Bloco 4/5).  
# Assumimos que a ordem das linhas no SDF corresponde à ordem em dados\_spdf\_gwr,  
# pois o SDF foi criado a partir de dados\_gwr\_sf\_validos, que por sua vez veio diretamente de dados\_gwr\_sf\_validos.  
# Embora o join por ID seja mais seguro, garantir a ordem pode ser uma camada extra de verificação se necessário.  
# Aqui, confiamos no join por ID após adicionar a coluna.  
if ("id\_celula\_grade" %in% names(dados\_gwr\_sf\_validos)) {  
 # Adiciona a coluna id\_celula\_grade ao dataframe de resultados do GWR  
 gwr\_resultados\_df\_para\_join$id\_celula\_grade <- dados\_gwr\_sf\_validos$id\_celula\_grade  
 message("Coluna 'id\_celula\_grade' adicionada aos resultados do GWR (DF) para join.")  
} else {  
 stop("ERRO CRÍTICO Bloco 9: A coluna 'id\_celula\_grade' NÃO foi encontrada no objeto SF 'dados\_gwr\_sf\_validos'. Verifique os Blocos 4 e 5 onde a grade foi criada e agregada. Impossível combinar resultados de forma segura.")  
}

## Coluna 'id\_celula\_grade' adicionada aos resultados do GWR (DF) para join.

# --- 9.3. Renomear Colunas de Resultados GWR para Clareza e Padronização (CORRIGIDO v5) ---  
# Renomeia colunas do data.frame extraído do SDF para nomes mais descritivos antes do join.  
# Isso evita conflitos de nomes e deixa claro que são resultados do GWR.  
message("\n--- 9.3. Renomeando colunas de resultados GWR ---")

##   
## --- 9.3. Renomeando colunas de resultados GWR ---

# Nome da variável independente principal da fórmula original (e.g. "log1p\_n\_drogas")  
# ESSA VARIÁVEL DEVE TER SIDO DEFINIDA NO SEU SCRIPT ANTES DESTE BLOCO (e.g., no Bloco 5 ou 7)  
# Ex: var\_indep\_na\_formula <- all.vars(formula\_gwr)[2]  
if (!exists("var\_indep\_na\_formula") || is.null(var\_indep\_na\_formula)) {  
 # Tenta definir aqui se não foi definida antes, baseando-se na formula\_gwr  
 if (exists("formula\_gwr") && !is.null(formula\_gwr) && length(all.vars(formula\_gwr)) > 1) {  
 var\_indep\_na\_formula <- all.vars(formula\_gwr)[2]  
 message(paste0("Variável 'var\_indep\_na\_formula' definida dentro do Bloco 9 como: '", var\_indep\_na\_formula, "' baseada em formula\_gwr."))  
 } else {  
 stop("ERRO CRÍTICO Bloco 9: Variável 'var\_indep\_na\_formula' não definida e não pôde ser inferida de 'formula\_gwr'.")  
 }  
} else {  
 message(paste0("Variável 'var\_indep\_na\_formula' encontrada (valor: '", var\_indep\_na\_formula, "')."))  
}

## Variável 'var\_indep\_na\_formula' definida dentro do Bloco 9 como: 'log1p\_n\_drogas' baseada em formula\_gwr.

# Define os nomes originais das colunas de resultados no SDF do GWR (verificar output GWmodel se diferente)  
# Estes são os nomes que esperamos encontrar NO DATA.FRAME gwr\_resultados\_df\_para\_join antes de renomear.  
nome\_coef\_original\_sdf <- var\_indep\_na\_formula # O nome do coeficiente é o nome da variável no modelo  
nome\_tval\_original\_sdf <- paste0(var\_indep\_na\_formula, "\_TV") # Nome padrão para o t-valor  
nome\_intercept\_original\_sdf <- "Intercept" # Nome padrão do intercepto  
nome\_intercept\_tv\_original\_sdf <- "Intercept\_TV" # Nome padrão para o t-valor do intercepto  
nome\_local\_r2\_original\_sdf <- "Local\_R2" # Nome padrão para o R2 Local  
nome\_yhat\_original\_sdf <- "yhat" # Nome padrão para o valor previsto  
nome\_residual\_original\_sdf <- "residual" # Nome padrão para o resíduo  
nome\_y\_observado\_original\_sdf <- "y" # Nome padrão para a VD observada na escala transformada  
  
  
# Define os NOVOS nomes padronizados desejados no data.frame para o join  
# \*\*\* DEFINIÇÃO EXPLÍCITA DE TODAS AS VARIÁVEIS ANTES DE CRIAR O VETOR DE MAPEAMENTO \*\*\*  
# Garante que estas variáveis existem no ambiente.  
novo\_nome\_coef <- paste0("GWR\_coef\_", var\_indep\_na\_formula) # Ex: "GWR\_coef\_log1p\_n\_drogas"  
novo\_nome\_tval <- paste0("GWR\_tval\_", var\_indep\_na\_formula) # Ex: "GWR\_tval\_log1p\_n\_drogas"  
novo\_nome\_intercept <- "GWR\_coef\_Intercept" # Ex: "GWR\_coef\_Intercept"  
novo\_nome\_intercept\_tv <- "GWR\_tval\_Intercept" # Ex: "GWR\_tval\_Intercept"  
novo\_nome\_r2\_local <- "GWR\_Local\_R2" # Ex: "GWR\_Local\_R2"  
novo\_nome\_yhat <- "GWR\_yhat" # Ex: "GWR\_yhat"  
novo\_nome\_residual <- "GWR\_residual" # Ex: "GWR\_residual"  
novo\_nome\_y\_observado <- "GWR\_y\_observed\_transf" # Nome para a coluna 'y' do SDF, que é a VD observada transformada  
  
  
# Cria o mapeamento correto para dplyr::rename: c(novo\_nome = "nome\_original")  
# Usa setNames() para criar o vetor nomeado onde os nomes vêm das variáveis.  
nomes\_originais\_esperados <- c(  
 nome\_coef\_original\_sdf,  
 nome\_tval\_original\_sdf,  
 nome\_intercept\_original\_sdf,  
 nome\_intercept\_tv\_original\_sdf,  
 nome\_local\_r2\_original\_sdf,  
 nome\_yhat\_original\_sdf,  
 nome\_residual\_original\_sdf,  
 nome\_y\_observado\_original\_sdf  
)  
  
novos\_nomes\_desejados <- c(  
 novo\_nome\_coef,  
 novo\_nome\_tval,  
 novo\_nome\_intercept,  
 novo\_nome\_intercept\_tv,  
 novo\_nome\_r2\_local,  
 novo\_nome\_yhat,  
 novo\_nome\_residual,  
 novo\_nome\_y\_observado  
)  
  
# Cria o vetor de mapeamento onde nomes = novos\_nomes, valores = nomes\_originais\_esperados  
names\_mapping\_for\_rename <- setNames(nomes\_originais\_esperados, novos\_nomes\_desejados)  
  
  
# Filtra o mapeamento: mantém apenas as entradas onde o NOME ORIGINAL (o valor no vetor names\_mapping\_for\_rename)  
# realmente existe nos nomes das colunas do data.frame gwr\_resultados\_df\_para\_join.  
# Isso previne o erro "Can't rename columns that don't exist".  
names\_to\_check\_in\_df <- as.character(names\_mapping\_for\_rename) # Extrai os nomes originais esperados  
existing\_original\_names <- intersect(names\_to\_check\_in\_df, names(gwr\_resultados\_df\_para\_join)) # Verifica quais existem no DF  
  
# Cria o mapeamento filtrado: apenas entradas cujo nome original existe no DF  
# names\_mapping\_for\_rename é um vetor nomeado (nomes=novos nomes, valores=nomes originais)  
# Precisamos filtrar baseado nos \*valores\* que existem em existing\_original\_names  
names\_mapping\_for\_rename\_existing <- names\_mapping\_for\_rename[names\_mapping\_for\_rename %in% existing\_original\_names]  
  
  
# Aplica a renomeação usando o mapeamento filtrado.  
# A função rename espera c(novo\_nome = "nome\_original"). Nosso vetor names\_mapping\_for\_rename\_existing  
# tem nomes = novo\_nome e valores = nome\_original. O !!! expande corretamente.  
if (length(names\_mapping\_for\_rename\_existing) > 0) {  
 # Renomeia o data.frame. O operador !!! expande o vetor nomeado para argumentos new\_name = old\_name.  
 gwr\_resultados\_df\_para\_join <- dplyr::rename(gwr\_resultados\_df\_para\_join, !!!names\_mapping\_for\_rename\_existing)  
  
 # Mensagem de confirmação das colunas renomeadas (mostra "nome\_original -> novo\_nome")  
 # Para mostrar "nome\_original -> novo\_nome": os VALUES do vetor são os nomes originais, os NAMES são os novos nomes.  
 rename\_message\_text <- paste(as.character(names\_mapping\_for\_rename\_existing), "->", names(names\_mapping\_for\_rename\_existing), collapse = ", ")  
 message(paste0("Colunas de resultados GWR renomeadas no data.frame: ", rename\_message\_text))  
} else {  
 message("Nenhuma coluna de resultado GWR padrão encontrada (com os nomes originais esperados) que exista no dataframe de resultados do GWR para renomear.")  
}

## Colunas de resultados GWR renomeadas no data.frame: log1p\_n\_drogas -> GWR\_coef\_log1p\_n\_drogas, log1p\_n\_drogas\_TV -> GWR\_tval\_log1p\_n\_drogas, Intercept -> GWR\_coef\_Intercept, Intercept\_TV -> GWR\_tval\_Intercept, Local\_R2 -> GWR\_Local\_R2, yhat -> GWR\_yhat, residual -> GWR\_residual, y -> GWR\_y\_observed\_transf

# Verifica AGORA (após a tentativa de renomear) se as colunas com os NOVOS nomes esperados existem.  
# Isso confirma quais colunas estarão disponíveis para o join e plotagem.  
expected\_cols\_renamed <- c(novo\_nome\_coef, novo\_nome\_tval, novo\_nome\_r2\_local, novo\_nome\_residual,  
 novo\_nome\_yhat, novo\_nome\_intercept, novo\_nome\_intercept\_tv, novo\_nome\_y\_observado) # Inclui todas as variáveis 'novo\_nome\_...'  
missing\_expected\_cols\_after\_rename <- setdiff(expected\_cols\_renamed, names(gwr\_resultados\_df\_para\_join))  
  
if (length(missing\_expected\_cols\_after\_rename) > 0) {  
 warning(paste0("AVISO Bloco 9: As seguintes colunas de resultados GWR esperadas (APÓS renomeação) NÃO foram encontradas no DF de resultados para join: ", paste(missing\_expected\_cols\_after\_rename, collapse = ", "), ".\n Isso significa que elas NÃO EXISTIAM com os nomes originais esperados no SDF do GWR, ou a renomeação falhou por outro motivo. Alguns mapas ou partes da análise final podem não ser gerados corretamente."))  
} else {  
 message("Todas as colunas de resultados GWR esperadas encontradas (com os novos nomes) e prontas para join.")  
}

## Todas as colunas de resultados GWR esperadas encontradas (com os novos nomes) e prontas para join.

message("Colunas de resultados GWR preparadas para join com nomes padronizados.")

## Colunas de resultados GWR preparadas para join com nomes padronizados.

# --- 9.4. Realizar o Join Final ao Objeto SF ---  
# Junta o data.frame de resultados GWR (agora com nomes padronizados) ao objeto SF dos polígonos originais.  
# O objeto SF base 'dados\_gwr\_sf\_validos' já contém as geometrias e as variáveis originais/transformadas.  
# Seleciona apenas as colunas de resultados GWR renomeadas (e o id\_celula\_grade) do data.frame para o join,  
# para evitar duplicar colunas originais já presentes em 'dados\_gwr\_sf\_validos'.  
message("\n--- 9.4. Realizando o join dos resultados GWR ao objeto SF ---")

##   
## --- 9.4. Realizando o join dos resultados GWR ao objeto SF ---

# Colunas do DF de resultados GWR a serem mantidas no join (id + colunas renomeadas que existem após renomear)  
# Pega todos os nomes no DF de resultados que NÃO são id\_celula\_grade.  
cols\_gwr\_data\_only\_to\_keep <- setdiff(names(gwr\_resultados\_df\_para\_join), "id\_celula\_grade")  
  
# Realiza o left\_join  
resultados\_gwr\_poligonos\_sf <- dplyr::left\_join(  
 dados\_gwr\_sf\_validos, # Objeto SF base com polígonos originais e dados originais  
 gwr\_resultados\_df\_para\_join %>% dplyr::select(dplyr::all\_of(c("id\_celula\_grade", cols\_gwr\_data\_only\_to\_keep))), # DF de resultados GWR filtrado por colunas  
 by = "id\_celula\_grade" # Chave de join  
)  
  
message("Resultados GWR combinados com sucesso aos polígonos originais da grade no objeto SF 'resultados\_gwr\_poligonos\_sf'.")

## Resultados GWR combinados com sucesso aos polígonos originais da grade no objeto SF 'resultados\_gwr\_poligonos\_sf'.

# Verificação rápida pós-join para NAs em uma coluna chave (usando o novo nome esperado)  
# Verifica se a coluna existe antes de checar NAs ou emitir aviso.  
if (novo\_nome\_r2\_local %in% names(resultados\_gwr\_poligonos\_sf) && any(is.na(resultados\_gwr\_poligonos\_sf[[novo\_nome\_r2\_local]]))) {  
 warning(paste0("AVISO Bloco 9: Foram encontrados NAs na coluna '", novo\_nome\_r2\_local, "' após o join. Isso pode indicar problemas na correspondência de 'id\_celula\_grade' ou dados faltantes no resultado do GWR original."))  
} else if (!(novo\_nome\_r2\_local %in% names(resultados\_gwr\_poligonos\_sf))) {  
 # AVISO: A coluna R2 Local renomeada nem chegou no objeto final após o join.  
 warning(paste0("AVISO Bloco 9: A coluna de resultados GWR esperada '", novo\_nome\_r2\_local, "' não foi encontrada no objeto SF final após o join."))  
}  
  
  
# --- 9.5. Preparar a Área de Estudo para Plotagem (se existir) ---  
# Se area\_estudo\_union foi carregado e tem CRS compatível, ele será usado como contorno nos mapas.  
message("\n--- 9.5. Preparando contorno da área de estudo para mapas ---")

##   
## --- 9.5. Preparando contorno da área de estudo para mapas ---

# area\_estudo\_union\_mapa já foi definido e verificado no Bloco 9.1  
  
  
# --- 9.6. Gerar e Salvar Mapas Essenciais (SOMENTE RESULTADOS GWR) ---  
message("\n--- 9.6. Iniciando a geração dos mapas essenciais de resultados GWR ---")

##   
## --- 9.6. Iniciando a geração dos mapas essenciais de resultados GWR ---

# Define o diretório para salvar os mapas. Sugestão: usar o mesmo diretório dos shapefiles de entrada.  
# Você pode alterar 'path\_area\_estudo' para outra variável que contenha o caminho base desejado.  
output\_map\_dir <- dirname(path\_area\_estudo) # Ex: pega o diretório do arquivo de área de estudo  
if (!dir.exists(output\_map\_dir)) {  
 warning(paste0("AVISO Bloco 9: Diretório de saída para mapas NÃO EXISTE: '", output\_map\_dir, "'. Os mapas não serão salvos."))  
 output\_map\_dir\_valid <- FALSE # Flag para não tentar salvar  
} else {  
 message(paste0("Mapas serão salvos no diretório: '", output\_map\_dir, "'\n"))  
 output\_map\_dir\_valid <- TRUE # Flag para permitir salvar  
}

## Mapas serão salvos no diretório: 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido'

# ==========================================================================  
# ADIÇÃO PARA DEBUG: Imprimir variáveis antes da lista map\_configs  
# Confirma se as variáveis 'novo\_nome\_...' estão definidas neste ponto  
cat("\n--- Debug: Verificando variáveis 'novo\_nome\_...' antes da lista map\_configs ---\n")

##   
## --- Debug: Verificando variáveis 'novo\_nome\_...' antes da lista map\_configs ---

print(paste0("Valor de novo\_nome\_coef: ", if(exists("novo\_nome\_coef")) novo\_nome\_coef else "NÃO DEFINIDO"))

## [1] "Valor de novo\_nome\_coef: GWR\_coef\_log1p\_n\_drogas"

print(paste0("Valor de novo\_nome\_tval: ", if(exists("novo\_nome\_tval")) novo\_nome\_tval else "NÃO DEFINIDO"))

## [1] "Valor de novo\_nome\_tval: GWR\_tval\_log1p\_n\_drogas"

print(paste0("Valor de novo\_nome\_intercept: ", if(exists("novo\_nome\_intercept")) novo\_nome\_intercept else "NÃO DEFINIDO"))

## [1] "Valor de novo\_nome\_intercept: GWR\_coef\_Intercept"

print(paste0("Valor de novo\_nome\_tval\_intercept: ", if(exists("novo\_nome\_tval\_intercept")) novo\_nome\_tval\_intercept else "NÃO DEFINIDO"))

## [1] "Valor de novo\_nome\_tval\_intercept: NÃO DEFINIDO"

print(paste0("Valor de novo\_nome\_r2\_local: ", if(exists("novo\_nome\_r2\_local")) novo\_nome\_r2\_local else "NÃO DEFINIDO"))

## [1] "Valor de novo\_nome\_r2\_local: GWR\_Local\_R2"

print(paste0("Valor de novo\_nome\_yhat: ", if(exists("novo\_nome\_yhat")) novo\_nome\_yhat else "NÃO DEFINIDO"))

## [1] "Valor de novo\_nome\_yhat: GWR\_yhat"

print(paste0("Valor de novo\_nome\_residual: ", if(exists("novo\_nome\_residual")) novo\_nome\_residual else "NÃO DEFINIDO"))

## [1] "Valor de novo\_nome\_residual: GWR\_residual"

print(paste0("Valor de novo\_nome\_y\_observado: ", if(exists("novo\_nome\_y\_observed")) novo\_nome\_y\_observed else "NÃO DEFINIDO"))

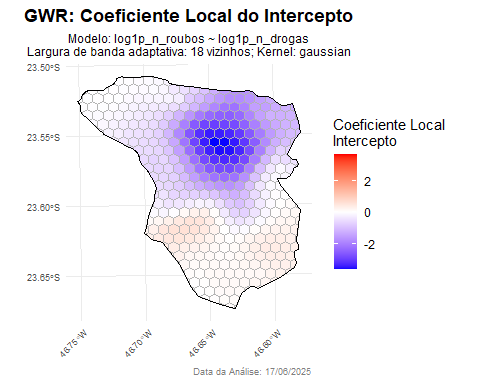
## [1] "Valor de novo\_nome\_y\_observado: NÃO DEFINIDO"

cat("------------------------------------------------------------------------------\n\n")

## ------------------------------------------------------------------------------

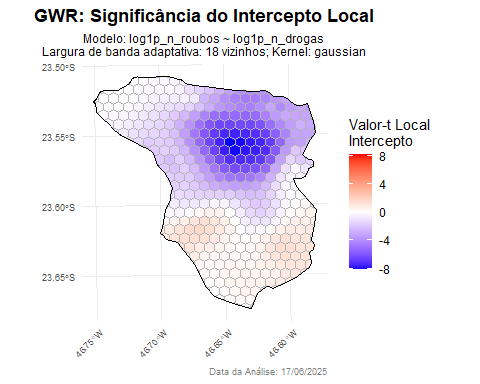
# ==========================================================================  
  
  
# Nomes das colunas a serem mapeadas e seus títulos/descrições para o loop de plotagem  
# ESTA LISTA AGORA CONTÉM APENAS OS MAPAS DE RESULTADOS GWR CONFORME SOLICITADO  
map\_configs <- list(  
 # --- 1. Mapas para Analisar a Relação Drogas-Crimes (Coeficientes e Significância): ---  
 # c. Intercept  
 # Descrição: Este é o intercepto local estimado para cada célula da grade no GWR.  
 # Motivo para Mapear: Indica o valor esperado de log1p\_n\_roubos quando todas as variáveis independentes (como log1p\_n\_drogas) são zero, variando localmente. Útil para entender o nível básico de roubo em cada área, controlando pelo impacto de drogas.  
 list(col = novo\_nome\_intercept, name = paste0("Coeficiente Local\nIntercepto"), title = paste0("GWR: Coeficiente Local do Intercepto"), type = "diverging", file\_suffix = "gwr\_coef\_intercept",  
 desc = "Mostra o valor base estimado para os roubos (na escala log1p) em cada localidade quando a variável de drogas é zero."),  
  
 # Significância do Intercepto Local (Adicionado por completude dos resultados GWR, embora não listado na última mensagem do usuário)  
 # É útil para saber onde o intercepto é estatisticamente diferente de zero.  
 list(col = novo\_nome\_intercept\_tv, name = paste0("Valor-t Local\nIntercepto"), title = paste0("GWR: Significância do Intercepto Local"), type = "diverging", file\_suffix = "gwr\_tval\_intercept",  
 desc = "Indica a significância estatística do Intercepto local. Valores |t| > ~1.96 sugerem significância a p < 0.05."),  
  
 # a. coef\_log1p\_n\_drogas  
 # Descrição: Este é o coeficiente local estimado (b) para a relação entre log1p\_n\_drogas (indicador de drogas transformado usando log1p) e log1p\_n\_roubos (indicador de roubos transformado).  
 # Motivo para Mapear: Mostra como a força da relação entre drogas e crimes varia no espaço. Permite identificar regiões onde o impacto do consumo de drogas sobre os roubos é maior, menor, ou até inexistente. Áreas com coeficientes mais altos indicam uma associação mais forte entre drogas e roubos.  
 list(col = novo\_nome\_coef, name = paste0("Coeficiente Local\n(", var\_indep\_na\_formula, ")"), title = paste0("GWR: Coeficientes Locais para ", var\_indep\_na\_formula), type = "sequential", file\_suffix = paste0("gwr\_coef\_", var\_indep\_na\_formula),  
 desc = paste0("Mede a força e direção da associação local entre '", var\_indep\_na\_formula, "' e 'log1p\_n\_roubos'. Variação mostra não estacionariedade.")),  
  
 # b. log1p\_n\_drogas\_TV  
 # Descrição: Este é o valor-t local associado ao coeficiente coef\_log1p\_n\_drogas.  
 # Motivo para Mapear: Permite identificar onde o coeficiente local é estatisticamente significativo. Regiões onde |log1p\_n\_drogas\_TV| > ~1.96 (para p < 0.05) indicam que o impacto de log1p\_n\_drogas sobre log1p\_n\_roubos é significativo. Mostra se o efeito local detectado pelo GWR é confiável ou pode ser atribuído ao acaso.  
 # Saída esperada: Um mapa binário ou categorizado (a escala divergente ajuda a ver positivo/negativo e magnitude para significância).  
 list(col = novo\_nome\_tval, name = paste0("Valor-t Local\n(", var\_indep\_na\_formula, ")"), title = paste0("GWR: Significância do Coeficiente de ", var\_indep\_na\_formula), type = "diverging", file\_suffix = paste0("gwr\_tval\_", var\_indep\_na\_formula),  
 desc = paste0("Indica onde o coeficiente local de '", var\_indep\_na\_formula, "' é estatisticamente significativo. Valores |t| > ~1.96 (p<0.05).")),  
  
 # --- 2. Mapas de Ajuste e Diagnóstico do Modelo: ---  
 # a. Local\_R2  
 # Descrição: Este é o R² local, que informa quanto da variação local na variável dependente (log1p\_n\_roubos) o modelo GWR conseguiu explicar.  
 # Motivo para Mapear: Quão bem o modelo performa em cada área? Áreas com Local\_R2 mais altos refletem regiões onde os roubos são mais bem explicados por drogas e pelas relações capturadas pelo modelo. Áreas com valores baixos sugerem influência de outros fatores que não foram incluídos no modelo.  
 list(col = novo\_nome\_r2\_local, name = "R2 Local\n(Ajuste do Modelo)", title = "GWR: R2 Local (Poder Explicativo do Modelo)", type = "sequential\_0\_1", file\_suffix = "gwr\_local\_r2", # R2 deve ir de 0 a 1  
 desc = "Mostra a proporção da variação local da VD explicada pelo modelo. Áreas com R2 alto = bom ajuste local."),  
  
 # c. yhat (Colocado antes de residual porque é o valor que gera o residual)  
 # Descrição: Este é o valor previsto do modelo para cada hexágono (log1p\_n\_roubos previsto com base em log1p\_n\_drogas e o intercepto).  
 # Motivo para Mapear: Permite visualizar os valores previstos de crime diretamente. Útil para comparar com os valores observados e verificar a apropriação geral do modelo.  
 list(col = novo\_nome\_yhat, name = paste0("VD Prevista\n(Escala Transf.)"), title = paste0("GWR: VD Prevista (", all.vars(formula\_gwr)[1], ")"), type = "sequential", file\_suffix = "gwr\_vd\_predicted",  
 desc = paste0("Valores de '", all.vars(formula\_gwr)[1], "' previstos pelo modelo GWR local.")),  
  
 # Variável Dependente Observada Transformada (Para comparação visual com yhat e residual)  
 # É crucial mapear a VD observada na mesma escala para entender yhat e residual.  
 # Descrição: A variável dependente observada na escala transformada (log1p\_n\_roubos).  
 # Motivo para Mapear: Comparação visual direta com os valores previstos (yhat) e para entender os resíduos.  
 list(col = novo\_nome\_y\_observado, name = paste0("VD Observada\n(Escala Transf.)"), title = paste0("GWR: VD Observada (", all.vars(formula\_gwr)[1], ")"), type = "sequential", file\_suffix = "gwr\_vd\_observed",  
 desc = paste0("Valores de '", all.vars(formula\_gwr)[1], "' observados (input para o GWR).")),  
  
 # b. residual  
 # Descrição: Este é o resíduo local, calculado como a diferença entre o valor observado (log1p\_n\_roubos) e o valor previsto (yhat) pelo modelo GWR (Fórmula: residual = log1p\_n\_roubos - yhat).  
 # Motivo para Mapear: A análise dos resíduos pode revelar padrões espaciais não capturados pelo modelo. Agrupamentos espaciais nos resíduos sugerem que fatores importantes foram omitidos ou que o modelo não conseguiu capturar processos locais específicos. Caso os resíduos apresentem autocorrelação significativa (testável com o Índice de Moran), pode ser necessário ajustar o modelo ou incluir novas variáveis.  
 list(col = novo\_nome\_residual, name = paste0("Resíduo Local"), title = paste0("GWR: Resíduos do Modelo (", all.vars(formula\_gwr)[1], ")"), type = "diverging", file\_suffix = "gwr\_residual", # Resíduos são divergentes  
 desc = "Diferença entre valor observado e previsto (VD transformada). Mostra onde o modelo sub/superestima.")  
)  
  
# Loop para gerar e salvar cada mapa definido em map\_configs  
for (map\_cfg in map\_configs) {  
 col\_name <- map\_cfg$col # Nome da coluna no objeto SF final  
 map\_title <- map\_cfg$title # Título principal do mapa  
 legend\_name <- map\_cfg$name # Nome para a legenda da cor  
 map\_type <- map\_cfg$type # Tipo de escala (sequential, diverging, sequential\_0\_1)  
 file\_suffix <- map\_cfg$file\_suffix # Sufixo para o nome do arquivo PNG  
 # Descrição é útil nos comentários do código, mas não usada na plotagem direta aqui.  
  
 # Verifica novamente se a coluna existe no objeto SF COMBINADO antes de tentar plotar  
 # Usa [[col\_name]] para acessar a coluna pelo nome da string de forma segura  
 if (!(col\_name %in% names(resultados\_gwr\_poligonos\_sf))) {  
 warning(paste0("AVISO Bloco 9: Coluna '", col\_name, "' não encontrada no objeto 'resultados\_gwr\_poligonos\_sf' para mapeamento. Pulando este mapa."))  
 next # Pula para a próxima configuração de mapa no loop  
 }  
  
 message(paste0(" -> Gerando mapa para: '", legend\_name, "' (coluna: '", col\_name, "')..."))  
  
 tryCatch({  
 # --- Configura a escala de cor baseada no tipo ---  
 if (map\_type == "sequential") {  
 # Escala sequencial padrão (ex: Viridis)  
 color\_scale <- scale\_fill\_viridis\_c(name = legend\_name, option = "viridis", na.value = "grey80")  
 } else if (map\_type == "diverging") {  
 # Escala divergente (centralizada em 0)  
 # Calcula o limite máximo absoluto para centralizar a escala de cor  
 # Acessa a coluna de dados de forma segura com [[col\_name]]  
 max\_abs\_val <- max(abs(resultados\_gwr\_poligonos\_sf[[col\_name]]), na.rm = TRUE)  
 # Define limites para a escala, garantindo que não sejam infinitos ou muito pequenos  
 plot\_limits <- c(-max\_abs\_val, max\_abs\_val)  
 if (!is.finite(max\_abs\_val) || max\_abs\_val == 0) {  
 plot\_limits <- c(-1, 1) # Limite padrão seguro se valores são zero/infinito  
 } else if (max\_abs\_val < 0.1) {  
 plot\_limits <- c(-0.1, 0.1) # Ajusta limite se valores forem muito pequenos  
 }  
  
 color\_scale <- scale\_fill\_gradient2(  
 name = legend\_name,  
 low = "blue", mid = "white", high = "red", # Azul (negativo) - Branco (zero) - Vermelho (positivo)  
 midpoint = 0, # Centraliza em zero  
 limits = plot\_limits, # Define os limites da escala  
 space = "Lab",  
 na.value = "grey80", # Cor para valores NA  
 oob = scales::squish # Comprime valores fora do limite para o limite mais próximo  
 )  
 } else if (map\_type == "sequential\_0\_1") {  
 # Escala sequencial específica para valores entre 0 e 1 (como R2)  
 # Trunca valores < 0 para 0 para visualização no mapa (R2 não deve ser negativo)  
 plot\_data\_r2 <- resultados\_gwr\_poligonos\_sf  
 # Acessa a coluna de dados de forma segura com [[col\_name]]  
 plot\_data\_r2[[col\_name]] <- ifelse(plot\_data\_r2[[col\_name]] < 0, 0, plot\_data\_r2[[col\_name]])  
 # Define limite superior do R2 (máx entre 1 e o valor máximo observado >=0)  
 lim\_max\_r2 <- max(1, max(plot\_data\_r2[[col\_name]], na.rm = TRUE))  
 color\_scale <- scale\_fill\_viridis\_c(  
 name = legend\_name,  
 option = "plasma", # Outra opção de paleta sequencial  
 limits = c(0, lim\_max\_r2), # Limita a escala de 0 até o valor máximo (ou 1)  
 labels = scales::percent\_format(accuracy = 1), # Formata legenda como porcentagem  
 na.value = "grey80", # Cor para valores NA  
 oob = scales::squish # Comprime valores fora do limite  
 )  
 # Usa os dados com R2 truncado para plotar, mas o objeto original permanece inalterado  
 data\_to\_plot <- plot\_data\_r2  
 } else {  
 # Tipo de mapa desconhecido, usa escala sequencial padrão  
 warning(paste0("Tipo de mapa desconhecido: '", map\_type, "'. Usando escala sequencial padrão."))  
 color\_scale <- scale\_fill\_viridis\_c(name = legend\_name, na.value = "grey80")  
 }  
  
 # Define os dados a serem plotados (usa plot\_data\_r2 se for tipo sequential\_0\_1, senão usa o objeto original)  
 data\_to\_plot <- if (map\_type == "sequential\_0\_1") plot\_data\_r2 else resultados\_gwr\_poligonos\_sf  
  
 # --- Cria o objeto ggplot para o mapa atual ---  
 p <- ggplot() +  
 # Camada dos polígonos com os resultados GWR ou dados originais  
 # Usa .data[[col\_name]] dentro do aes para acessar a coluna pela string col\_name  
 geom\_sf(data = data\_to\_plot, aes(fill = .data[[col\_name]]), color = "grey70", linewidth = 0.1) +  
 # Camada de contorno da área de estudo (se o objeto existir e for válido)  
 {if(!is.null(area\_estudo\_union\_mapa)) geom\_sf(data = area\_estudo\_union\_mapa, fill = NA, color = "black", linewidth = 0.7)} +  
 # Aplica a escala de cor configurada acima  
 color\_scale +  
 # Define títulos e legendas para o mapa  
 labs(  
 title = map\_title,  
 subtitle = paste0("Modelo: ", deparse(formula\_gwr), # Mostra a fórmula usada no GWR  
 "\nLargura de banda adaptativa: ", round(gwr\_resultado\_lista$GW.arguments$bw, 0), " vizinhos; Kernel: ", gwr\_resultado\_lista$GW.arguments$kernel), # Informa os parâmetros do GWR  
 caption = paste("Data da Análise:", format(Sys.Date(), "%d/%m/%Y")) # Data atual da análise  
 ) +  
 # Aplica um tema minimalista para mapas  
 theme\_minimal(base\_size = 11) +  
 theme(  
 plot.title = element\_text(hjust = 0.5, face = "bold"), # Título centralizado e em negrito  
 plot.subtitle = element\_text(hjust = 0.5, size = 9, lineheight = 0.9), # Subtítulo centralizado e menor  
 plot.caption = element\_text(hjust = 1, size = 7, color = "grey50"), # Legenda no canto inferior direito, menor e cinza  
 legend.position = "right", # Posição da legenda  
 axis.text.x = element\_text(angle = 45, hjust = 1, size = 7), # Rótulos do eixo X rotacionados  
 axis.text.y = element\_text(size = 7) # Rótulos do eixo Y  
 ) +  
 # Remove rótulos dos eixos (coordenadas) para um visual de mapa limpo  
 labs(x = NULL, y = NULL)  
  
 # Imprime o mapa (útil para visualização imediata no RStudio antes de salvar)  
 print(p)  
  
 # --- Salva o mapa em arquivo PNG ---  
 if (output\_map\_dir\_valid) { # Verifica se o diretório de saída é válido antes de tentar salvar  
 file\_name <- file.path(output\_map\_dir, paste0(file\_suffix, ".png")) # Usa o sufixo definido para o nome do arquivo  
 # Salva o plot p no caminho e nome especificados  
 ggsave(file\_name, plot = p, width = 8, height = 7, units = "in", dpi = 300) # Define tamanho e resolução  
 message(paste0(" Mapa salvo com sucesso em '", file\_name, "'.\n"))  
 } else {  
 message(" Diretório de saída não definido ou não encontrado/válido. O mapa não foi salvo em arquivo.\n")  
 }  
  
 }, error = function(e\_plot) {  
 # Captura e reporta erros específicos durante a plotagem ou salvamento de um mapa  
 cat(paste0("ERRO ao gerar ou salvar o mapa para '", legend\_name, "' (coluna '", col\_name, "'):\n"))  
 cat(paste(" Mensagem de erro:", e\_plot$message, "\n\n"))  
 })  
}

## -> Gerando mapa para: 'Coeficiente Local  
## Intercepto' (coluna: 'GWR\_coef\_Intercept')...



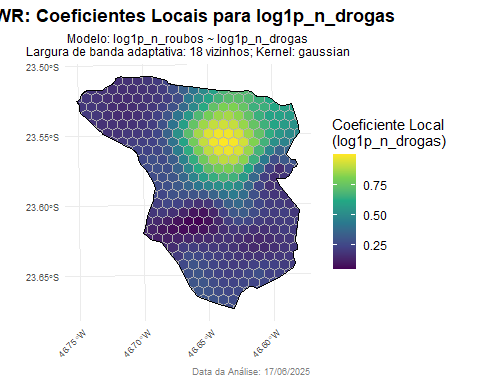
## Mapa salvo com sucesso em 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/gwr\_coef\_intercept.png'.

## -> Gerando mapa para: 'Valor-t Local  
## Intercepto' (coluna: 'GWR\_tval\_Intercept')...



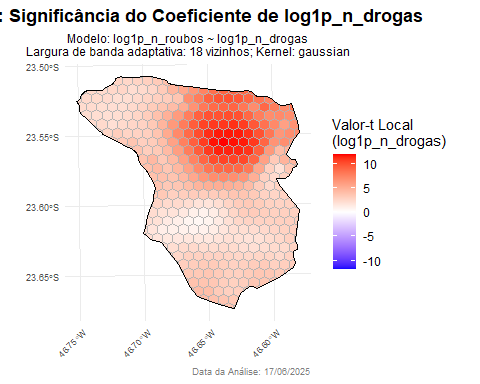
## Mapa salvo com sucesso em 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/gwr\_tval\_intercept.png'.

## -> Gerando mapa para: 'Coeficiente Local  
## (log1p\_n\_drogas)' (coluna: 'GWR\_coef\_log1p\_n\_drogas')...



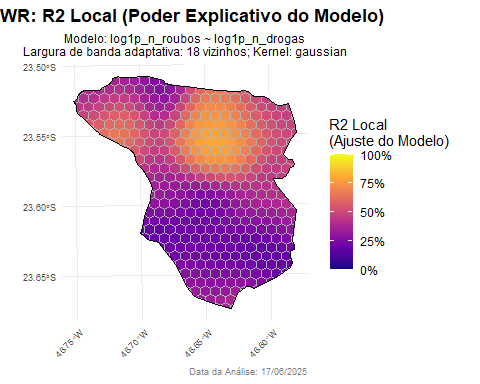
## Mapa salvo com sucesso em 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/gwr\_coef\_log1p\_n\_drogas.png'.

## -> Gerando mapa para: 'Valor-t Local  
## (log1p\_n\_drogas)' (coluna: 'GWR\_tval\_log1p\_n\_drogas')...



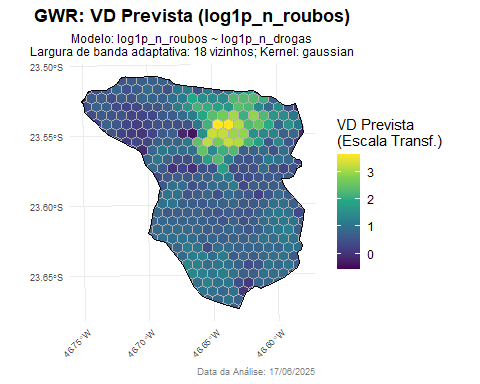
## Mapa salvo com sucesso em 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/gwr\_tval\_log1p\_n\_drogas.png'.

## -> Gerando mapa para: 'R2 Local  
## (Ajuste do Modelo)' (coluna: 'GWR\_Local\_R2')...



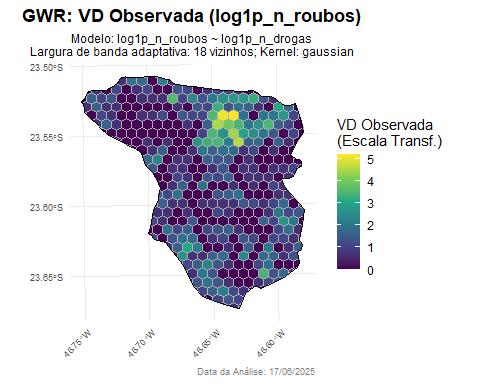
## Mapa salvo com sucesso em 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/gwr\_local\_r2.png'.

## -> Gerando mapa para: 'VD Prevista  
## (Escala Transf.)' (coluna: 'GWR\_yhat')...



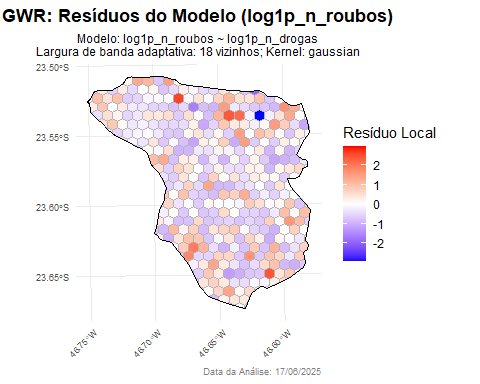
## Mapa salvo com sucesso em 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/gwr\_vd\_predicted.png'.

## -> Gerando mapa para: 'VD Observada  
## (Escala Transf.)' (coluna: 'GWR\_y\_observed\_transf')...



## Mapa salvo com sucesso em 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/gwr\_vd\_observed.png'.

## -> Gerando mapa para: 'Resíduo Local' (coluna: 'GWR\_residual')...



## Mapa salvo com sucesso em 'C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/shp/centro\_expandido/gwr\_residual.png'.

message("\n--- Geração de mapas de resultados GWR concluída. Todos os mapas definidos foram processados. ---")

##   
## --- Geração de mapas de resultados GWR concluída. Todos os mapas definidos foram processados. ---

# --- Fim do Bloco 9 ---  
cat("\n--- Fim do BL. 9 ---\n")

##   
## --- Fim do BL. 9 ---

cat("--------------------\n")

## --------------------

#--- Bloco 9: Combinar Resultados GWR aos Polígonos e Visualizar ---  
  
message("\n--- Iniciando Bloco 9: Combinando Resultados GWR aos Polígonos e Preparando Visualizações ---")

##   
## --- Iniciando Bloco 9: Combinando Resultados GWR aos Polígonos e Preparando Visualizações ---

# Verificar se os objetos necessários existem  
if (!exists("dados\_gwr\_sf\_validos") || is.null(dados\_gwr\_sf\_validos)) {  
 stop("ERRO CRÍTICO Bloco 9: 'dados\_gwr\_sf\_validos' não encontrado ou é NULL.")  
}  
if (!exists("gwr\_resultados\_sdf") || is.null(gwr\_resultados\_sdf)) {  
 stop("ERRO CRÍTICO Bloco 9: 'gwr\_resultados\_sdf' (output do GWR) não encontrado ou é NULL.")  
}  
  
# Combinar os resultados do GWR (que estão em formato SPDF, nos centróides)  
# de volta aos polígonos originais (dados\_gwr\_sf\_validos).  
# É crucial garantir que a correspondência entre os resultados e os polígonos da grade seja correta.  
# Como dados\_spdf\_gwr (entrada para gwr.basic) foi criado diretamente a partir de dados\_gwr\_sf\_validos  
# (na mesma ordem e com os mesmos pontos representativos), a ordem das linhas DEVE ser a mesma.  
# No entanto, um join explícito por 'id\_celula\_grade' é mais robusto.  
  
resultados\_gwr\_poligonos\_sf <- NULL # Inicializar  
  
if(nrow(dados\_gwr\_sf\_validos) == nrow(gwr\_resultados\_sdf)) {  
 message("Número de linhas corresponde entre polígonos originais e resultados GWR. Prosseguindo com a combinação...")  
   
 # Extrair os dados (atributos) do SDF do GWR para um data.frame.  
 gwr\_resultados\_df\_para\_join <- as.data.frame(gwr\_resultados\_sdf)  
  
 # Adicionar 'id\_celula\_grade' de 'dados\_gwr\_sf\_validos' aos resultados do GWR para um join robusto.  
 # Assumimos que a ordem das linhas em dados\_gwr\_sf\_validos corresponde à ordem em gwr\_resultados\_sdf  
 # porque gwr\_resultados\_sdf foi gerado a partir de dados\_spdf\_gwr, que por sua vez  
 # foi gerado a partir de dados\_gwr\_sf\_validos.  
 if ("id\_celula\_grade" %in% names(dados\_gwr\_sf\_validos)) {  
 gwr\_resultados\_df\_para\_join$id\_celula\_grade <- dados\_gwr\_sf\_validos$id\_celula\_grade  
 message("Coluna 'id\_celula\_grade' adicionada aos resultados do GWR (gwr\_resultados\_df\_para\_join) para join.")  
 } else {  
 stop("ERRO CRÍTICO Bloco 9: 'id\_celula\_grade' não encontrada em 'dados\_gwr\_sf\_validos'. Impossível realizar join seguro.")  
 }  
  
 # --- INÍCIO DA MODIFICAÇÃO: Renomear coluna do coeficiente e valor-t da VI ---  
 # A VI na fórmula é all.vars(formula\_gwr)[2], que deve ser "log1p\_n\_drogas"  
 var\_indep\_na\_formula <- all.vars(formula\_gwr)[2] # Ex: "log1p\_n\_drogas"  
   
 nome\_coef\_original\_no\_sdf <- var\_indep\_na\_formula  
 novo\_nome\_coef\_para\_join <- paste0("coef\_", var\_indep\_na\_formula) # Ex: "coef\_log1p\_n\_drogas"  
   
 nome\_tval\_original\_no\_sdf <- paste0(var\_indep\_na\_formula, "\_TV") # Ex: "log1p\_n\_drogas\_TV"  
 # Não vamos renomear a coluna do valor-t, mas vamos referenciá-la pelo nome correto.  
  
 if (nome\_coef\_original\_no\_sdf %in% names(gwr\_resultados\_df\_para\_join)) {  
 names(gwr\_resultados\_df\_para\_join)[names(gwr\_resultados\_df\_para\_join) == nome\_coef\_original\_no\_sdf] <- novo\_nome\_coef\_para\_join  
 message(paste0("Coluna do coeficiente para '", nome\_coef\_original\_no\_sdf, "' nos resultados GWR renomeada para '", novo\_nome\_coef\_para\_join, "'."))  
 } else {  
 warning(paste0("Coluna do coeficiente '", nome\_coef\_original\_no\_sdf, "' não encontrada diretamente nos resultados do GWR (gwr\_resultados\_df\_para\_join). Verifique os nomes das colunas no output do GWmodel. O mapa de coeficientes pode falhar."))  
 }  
 # --- FIM DA MODIFICAÇÃO ---  
  
 # Definir as colunas de resultados do GWR que serão mantidas após o join.  
 # 'y' no SDF do GWR corresponde à variável dependente da fórmula (log1p\_n\_roubos).  
 cols\_gwr\_a\_manter <- c("id\_celula\_grade",  
 "Intercept", "Intercept\_TV",  
 novo\_nome\_coef\_para\_join, nome\_tval\_original\_no\_sdf, # Usando os nomes corretos/renomeados  
 "Local\_R2", "y", "yhat", "residual")  
  
 # Filtrar colunas para o join, selecionando apenas as relevantes.  
 # Usar `all\_of` garante que apenas as colunas existentes e listadas sejam selecionadas,  
 # gerando um erro se alguma coluna essencial estiver faltando.  
 cols\_existentes\_no\_join\_df <- intersect(cols\_gwr\_a\_manter, names(gwr\_resultados\_df\_para\_join))  
   
 # Checar se colunas essenciais para plotagem estão presentes  
 if (!(novo\_nome\_coef\_para\_join %in% cols\_existentes\_no\_join\_df)) {  
 warning(paste0("AVISO Bloco 9: Coluna do coeficiente '", novo\_nome\_coef\_para\_join, "' não está presente para o join. O mapa de coeficientes falhará."))  
 }  
 if (!(nome\_tval\_original\_no\_sdf %in% cols\_existentes\_no\_join\_df)) {  
 warning(paste0("AVISO Bloco 9: Coluna do valor-t '", nome\_tval\_original\_no\_sdf, "' não está presente para o join. O mapa de valores-t falhará."))  
 }  
 if (!("Local\_R2" %in% cols\_existentes\_no\_join\_df)) {  
 warning(paste0("AVISO Bloco 9: Coluna 'Local\_R2' não está presente para o join. O mapa de R² Local falhará."))  
 }  
  
 gwr\_resultados\_filtrados\_df <- gwr\_resultados\_df\_para\_join %>%  
 dplyr::select(dplyr::all\_of(cols\_existentes\_no\_join\_df))  
  
 # Realizar o join dos resultados do GWR com o objeto sf dos polígonos da grade.  
 # dados\_gwr\_sf\_validos já contém as colunas originais e as transformadas log1p.  
 # Vamos selecionar explicitamente as colunas de dados\_gwr\_sf\_validos para evitar duplicatas desnecessárias,  
 # mantendo as colunas originais de contagem (n\_roubos, n\_drogas) e as transformadas (log1p\_n\_roubos, log1p\_n\_drogas)  
 # para referência e análise.  
   
 resultados\_gwr\_poligonos\_sf <- dplyr::left\_join(  
 dados\_gwr\_sf\_validos, # Contém id\_celula\_grade, geometry, n\_roubos, n\_drogas, log1p\_n\_roubos, log1p\_n\_drogas  
 gwr\_resultados\_filtrados\_df, # Contém id\_celula\_grade e resultados GWR selecionados  
 by = "id\_celula\_grade"  
 )  
   
 message("Resultados GWR combinados com os polígonos originais da grade usando 'id\_celula\_grade'.")  
  
 # Verificação pós-join: NAs em colunas críticas podem indicar problemas de correspondência.  
 if (any(is.na(resultados\_gwr\_poligonos\_sf$Local\_R2))) {  
 warning("NAs encontrados em 'Local\_R2' após o join. Isso pode indicar problemas na correspondência dos IDs ou dados ausentes em algumas células da grade.")  
 }  
  
} else {  
 stop(paste("Número de linhas nos resultados GWR (", nrow(gwr\_resultados\_sdf),  
 ") não corresponde aos dados de polígonos originais (", nrow(dados\_gwr\_sf\_validos),  
 "). Não é possível combinar os resultados de forma segura. Verifique a integridade dos dados antes do GWR ou a função de modelagem utilizada."))  
}

## Número de linhas corresponde entre polígonos originais e resultados GWR. Prosseguindo com a combinação...

## Coluna 'id\_celula\_grade' adicionada aos resultados do GWR (gwr\_resultados\_df\_para\_join) para join.

## Coluna do coeficiente para 'log1p\_n\_drogas' nos resultados GWR renomeada para 'coef\_log1p\_n\_drogas'.

## Resultados GWR combinados com os polígonos originais da grade usando 'id\_celula\_grade'.

# --- Visualizar os resultados ---  
message("\n--- Preparando visualizações dos resultados do GWR (modelo com variáveis log1p) ---")

##   
## --- Preparando visualizações dos resultados do GWR (modelo com variáveis log1p) ---

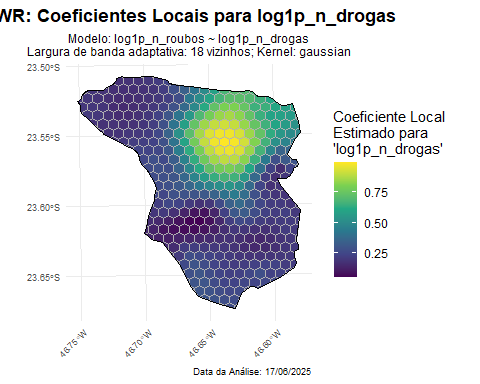
if (!is.null(resultados\_gwr\_poligonos\_sf) && nrow(resultados\_gwr\_poligonos\_sf) > 0) {  
 message("Iniciando a criação dos mapas de resultados GWR...")  
 cat("Primeiras linhas dos resultados GWR combinados com os polígonos (atributos):\n")  
 print(head(st\_drop\_geometry(resultados\_gwr\_poligonos\_sf)))  
 col\_names\_results <- names(resultados\_gwr\_poligonos\_sf)  
 message(paste("\nNomes das colunas nos resultados finais (polígonos):\n", paste(col\_names\_results, collapse=", ")))  
  
 # --- INÍCIO DA MODIFICAÇÃO: Atualizar nomes de colunas e display para variáveis transformadas ---  
 # var\_indep\_na\_formula foi definido acima como all.vars(formula\_gwr)[2] (ex: "log1p\_n\_drogas")  
 coef\_col\_name\_plot <- novo\_nome\_coef\_para\_join # Ex: "coef\_log1p\_n\_drogas"  
 t\_val\_col\_name\_plot <- nome\_tval\_original\_no\_sdf # Ex: "log1p\_n\_drogas\_TV"  
   
 var\_independente\_nome\_amigavel <- paste0(var\_indep\_na\_formula) # Ex: "log1p\_n\_drogas"  
 formula\_display <- deparse(formula\_gwr) # Ex: "log1p\_n\_roubos ~ log1p\_n\_drogas"  
 # --- FIM DA MODIFICAÇÃO ---  
  
 # -- Mapa 1: Coeficientes Locais da variável transformada --  
 if (coef\_col\_name\_plot %in% col\_names\_results) {  
 message(paste("Criando mapa para os coeficientes locais da variável '", var\_independente\_nome\_amigavel, "' (coluna '", coef\_col\_name\_plot, "')...", sep=""))  
 tryCatch({  
 mapa\_coef\_drogas <- ggplot() +  
 geom\_sf(data = resultados\_gwr\_poligonos\_sf, aes(fill = .data[[coef\_col\_name\_plot]]), color = "grey70", linewidth = 0.1) +  
 geom\_sf(data = area\_estudo\_union, fill = NA, color = "black", linewidth = 0.7) +  
 scale\_fill\_viridis\_c(name = paste0("Coeficiente Local\nEstimado para\n'", var\_independente\_nome\_amigavel, "'"), option = "viridis") +  
 labs(  
 title = paste("GWR: Coeficientes Locais para", var\_independente\_nome\_amigavel),  
 subtitle = paste0("Modelo: ", formula\_display,  
 "\nLargura de banda adaptativa: ", bw\_adaptativa, " vizinhos; Kernel: ", kernel\_usado\_para\_bw),  
 caption = paste("Data da Análise:", format(Sys.Date(), "%d/%m/%Y"))  
 ) +  
 theme\_minimal(base\_size = 11) +   
 theme(  
 plot.title = element\_text(hjust = 0.5, face = "bold"),   
 plot.subtitle = element\_text(hjust = 0.5, size = 9),   
 plot.caption = element\_text(hjust = 1, size = 7, lineheight = 1.1),   
 legend.position = "right",   
 axis.text.x = element\_text(angle = 45, hjust = 1, size=7),   
 axis.text.y = element\_text(size=7)   
 )  
 print(mapa\_coef\_drogas)  
 ggsave(paste0("mapa\_coef\_",var\_indep\_na\_formula, "\_gwr.png"), plot = mapa\_coef\_drogas, width = 8, height = 7, dpi = 300, bg = "white")  
 message(paste0("Mapa dos coeficientes locais de '", var\_independente\_nome\_amigavel, "' salvo como mapa\_coef\_",var\_indep\_na\_formula, "\_gwr.png."))  
 }, error = function(e\_plot) {  
 cat(paste("Erro ao gerar ou salvar o mapa dos coeficientes locais de '", var\_independente\_nome\_amigavel, "':\n", e\_plot$message, "\n"))  
 })  
 } else {  
 warning(paste0("Coluna do coeficiente '", coef\_col\_name\_plot, "' não encontrada nos resultados. O mapa de coeficientes não será gerado."))  
 }  
  
 # -- Mapa 2: R² Local --  
 r2\_col\_name <- "Local\_R2"  
 if (r2\_col\_name %in% col\_names\_results) {  
 message(paste("Criando mapa para o R² Local (coluna '", r2\_col\_name, "')...", sep=""))  
 tryCatch({  
 resultados\_gwr\_poligonos\_sf$Local\_R2\_plot <- ifelse(resultados\_gwr\_poligonos\_sf[[r2\_col\_name]] < 0, 0, resultados\_gwr\_poligonos\_sf[[r2\_col\_name]])  
 lim\_max\_r2 <- max(1, max(resultados\_gwr\_poligonos\_sf$Local\_R2\_plot, na.rm = TRUE))  
  
 mapa\_r2\_local <- ggplot() +  
 geom\_sf(data = resultados\_gwr\_poligonos\_sf, aes(fill = Local\_R2\_plot), color = "grey70", linewidth = 0.1) +  
 geom\_sf(data = area\_estudo\_union, fill = NA, color = "black", linewidth = 0.7) +  
 scale\_fill\_viridis\_c(name = "R² Local\n(Ajuste do Modelo)", option = "plasma", limits = c(0, lim\_max\_r2),  
 labels = scales::percent\_format(accuracy = 1)) +   
 labs(  
 title = "GWR: R² Local (Poder Explicativo do Modelo)",  
 subtitle = paste0("Modelo: ", formula\_display,  
 "\nLargura de banda adaptativa: ", bw\_adaptativa, " vizinhos; Kernel: ", kernel\_usado\_para\_bw),  
 caption = paste("Valores de R² Local < 0 foram truncados para 0 na visualização.\nData da Análise:", format(Sys.Date(), "%d/%m/%Y"))  
 ) +  
 theme\_minimal(base\_size = 11) +  
 theme(  
 plot.title = element\_text(hjust = 0.5, face = "bold"),  
 plot.subtitle = element\_text(hjust = 0.5, size = 9),  
 plot.caption = element\_text(hjust = 1, size = 7, lineheight = 0.9),  
 legend.position = "right",  
 axis.text.x = element\_text(angle = 45, hjust = 1, size=7),  
 axis.text.y = element\_text(size=7)  
 )  
 print(mapa\_r2\_local)  
 ggsave(paste0("mapa\_r2\_local\_gwr\_",var\_indep\_na\_formula,".png"), plot = mapa\_r2\_local, width = 8, height = 7, dpi = 300, bg = "white")  
 message(paste0("Mapa do R² local salvo como mapa\_r2\_local\_gwr\_",var\_indep\_na\_formula,".png."))  
 }, error = function(e\_plot) {  
 cat(paste("Erro ao gerar ou salvar o mapa do R² local:\n", e\_plot$message, "\n"))  
 })  
 } else {  
 warning(paste0("Coluna '", r2\_col\_name, "' não encontrada nos resultados. O mapa de R² Local não será gerado."))  
 }  
  
 # -- Mapa 3: Valores-t Locais do Coeficiente da variável transformada --  
 if (t\_val\_col\_name\_plot %in% col\_names\_results) {  
 message(paste("Criando mapa para os valores-t locais do coeficiente de '", var\_independente\_nome\_amigavel, "' (coluna '", t\_val\_col\_name\_plot, "')...", sep=""))  
 tryCatch({  
 lim\_max\_abs\_t <- max(abs(resultados\_gwr\_poligonos\_sf[[t\_val\_col\_name\_plot]]), na.rm = TRUE)  
 if (!is.finite(lim\_max\_abs\_t) || is.na(lim\_max\_abs\_t) || lim\_max\_abs\_t == 0) {  
 lim\_max\_abs\_t <- 2   
 }  
  
 mapa\_t\_valor\_drogas <- ggplot() +  
 geom\_sf(data = resultados\_gwr\_poligonos\_sf, aes(fill = .data[[t\_val\_col\_name\_plot]]), color = "grey70", linewidth = 0.1) +  
 geom\_sf(data = area\_estudo\_union, fill = NA, color = "black", linewidth = 0.7) +  
 scale\_fill\_gradient2(  
 name = paste0("Valor-t Local\n(", var\_independente\_nome\_amigavel, ")"),  
 low = "blue", mid = "white", high = "red",   
 midpoint = 0,   
 limits = c(-lim\_max\_abs\_t, lim\_max\_abs\_t),   
 oob = scales::squish   
 ) +  
 labs(  
 title = paste("GWR: Significância do Coeficiente de", var\_independente\_nome\_amigavel),  
 subtitle = paste0("Valores-t locais. |t| > ~1.96 sugere significância a p < 0.05 (bilateral).",  
 "\nLargura de banda: ", bw\_adaptativa, " vizinhos; Kernel: ", kernel\_usado\_para\_bw,  
 "\nModelo: ", formula\_display),  
 caption = paste("Data da Análise:", format(Sys.Date(), "%d/%m/%Y"))  
 ) +  
 theme\_minimal(base\_size = 11) +  
 theme(  
 plot.title = element\_text(hjust = 0.5, face = "bold"),  
 plot.subtitle = element\_text(hjust = 0.5, size = 9, lineheight = 0.9),  
 plot.caption = element\_text(hjust = 1, size = 7),  
 legend.position = "right",  
 axis.text.x = element\_text(angle = 45, hjust = 1, size=7),  
 axis.text.y = element\_text(size=7)  
 )  
 print(mapa\_t\_valor\_drogas)  
 ggsave(paste0("mapa\_t\_valor\_",var\_indep\_na\_formula, "\_gwr.png"), plot = mapa\_t\_valor\_drogas, width = 8, height = 7, dpi = 300, bg = "white")  
 message(paste0("Mapa dos valores-t locais de '", var\_independente\_nome\_amigavel, "' salvo como mapa\_t\_valor\_",var\_indep\_na\_formula, "\_gwr.png."))  
 }, error = function(e\_plot) {  
 cat(paste("Erro ao gerar ou salvar o mapa dos valores-t de '", var\_independente\_nome\_amigavel, "':\n", e\_plot$message, "\n"))  
 })  
 } else {  
 warning(paste0("Coluna do valor-t '", t\_val\_col\_name\_plot, "' não encontrada nos resultados. O mapa de valores-t não será gerado."))  
 }  
   
 # Mapa de significância opcional (precisaria ser ajustado para usar t\_val\_col\_name\_plot e formula\_display)  
 # ...  
  
} else {  
 message("Nenhum resultado GWR válido (resultados\_gwr\_poligonos\_sf) para plotar. Verifique as etapas anteriores da análise.")  
}

## Iniciando a criação dos mapas de resultados GWR...

## Primeiras linhas dos resultados GWR combinados com os polígonos (atributos):  
## id\_celula\_grade n\_roubos n\_drogas log1p\_n\_roubos log1p\_n\_drogas Intercept  
## 1 1 0 0 0.0000000 0.000000 -0.04735255  
## 2 2 0 0 0.0000000 0.000000 -0.02552189  
## 3 3 1 10 0.6931472 2.397895 -0.03699986  
## 4 4 0 5 0.0000000 1.791759 -0.01846921  
## 5 5 0 9 0.0000000 2.302585 -0.06323534  
## 6 6 4 26 1.6094379 3.295837 -0.02155795  
## Intercept\_TV coef\_log1p\_n\_drogas log1p\_n\_drogas\_TV Local\_R2 y  
## 1 -0.15952030 0.1944990 2.268392 0.4621949 0.0000000  
## 2 -0.08860688 0.1937619 2.356343 0.4119160 0.0000000  
## 3 -0.12338008 0.1972008 2.258865 0.4369848 0.6931472  
## 4 -0.06663292 0.1902470 2.434022 0.3945564 0.0000000  
## 5 -0.19811937 0.2045261 2.132905 0.4846298 0.0000000  
## 6 -0.07466145 0.1936369 2.329266 0.4061033 1.6094379  
## yhat residual  
## 1 -0.04735255 0.04735255  
## 2 -0.02552189 0.02552189  
## 3 0.43586710 0.25728008  
## 4 0.32240757 -0.32240757  
## 5 0.40770352 -0.40770352  
## 6 0.61663758 0.99280033

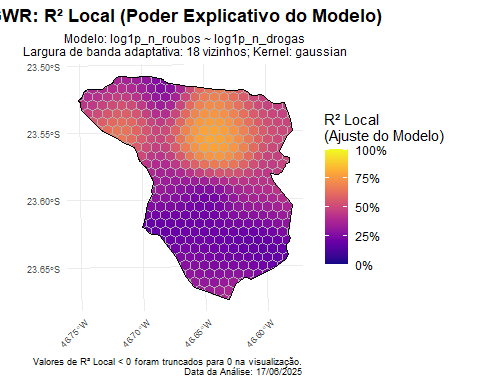
##   
## Nomes das colunas nos resultados finais (polígonos):  
## id\_celula\_grade, n\_roubos, n\_drogas, log1p\_n\_roubos, log1p\_n\_drogas, Intercept, Intercept\_TV, coef\_log1p\_n\_drogas, log1p\_n\_drogas\_TV, Local\_R2, y, yhat, residual, geometry

## Criando mapa para os coeficientes locais da variável 'log1p\_n\_drogas' (coluna 'coef\_log1p\_n\_drogas')...



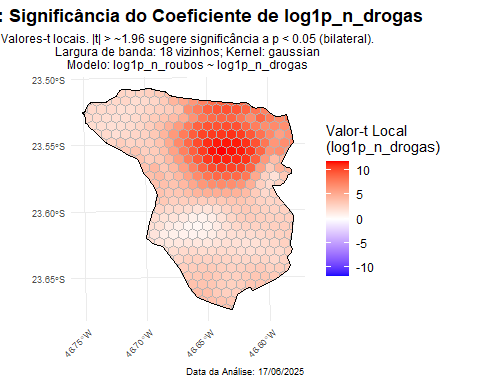
## Mapa dos coeficientes locais de 'log1p\_n\_drogas' salvo como mapa\_coef\_log1p\_n\_drogas\_gwr.png.

## Criando mapa para o R² Local (coluna 'Local\_R2')...



## Mapa do R² local salvo como mapa\_r2\_local\_gwr\_log1p\_n\_drogas.png.

## Criando mapa para os valores-t locais do coeficiente de 'log1p\_n\_drogas' (coluna 'log1p\_n\_drogas\_TV')...



## Mapa dos valores-t locais de 'log1p\_n\_drogas' salvo como mapa\_t\_valor\_log1p\_n\_drogas\_gwr.png.

message("\n--- Análise GWR (modelo com variáveis log1p) com grade hexagonal e visualizações concluídas ---")

##   
## --- Análise GWR (modelo com variáveis log1p) com grade hexagonal e visualizações concluídas ---

# ```  
  
# (O script continua com os Blocos 10 e 11)

# Bloco Final: Inspeção dos Nomes das Colunas Finais  
if (!is.null(resultados\_gwr\_poligonos\_sf) && nrow(resultados\_gwr\_poligonos\_sf) > 0) {  
 cat("\n\n--- Nomes Finais das Colunas no Objeto 'resultados\_gwr\_poligonos\_sf' ---\n")  
 print(names(resultados\_gwr\_poligonos\_sf))  
 cat("\n")  
} else {  
 cat("\n'resultados\_gwr\_poligonos\_sf' é NULL ou vazio. Verifique as etapas anteriores.\n")  
}

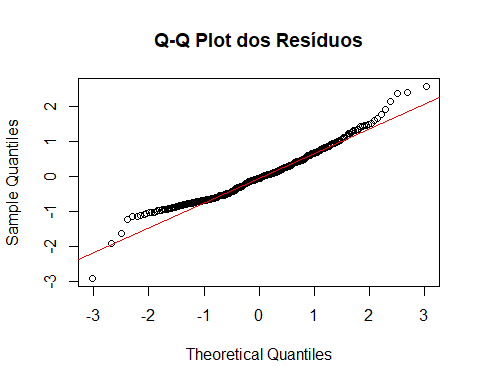
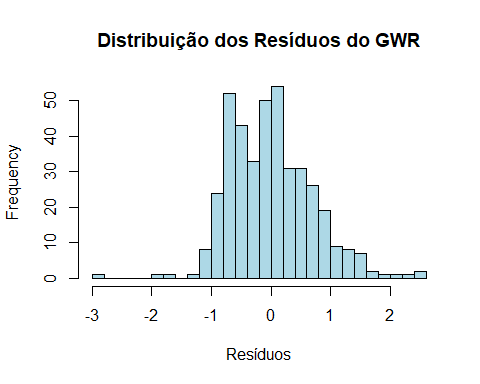
##   
##   
## --- Nomes Finais das Colunas no Objeto 'resultados\_gwr\_poligonos\_sf' ---  
## [1] "id\_celula\_grade" "n\_roubos" "n\_drogas"   
## [4] "log1p\_n\_roubos" "log1p\_n\_drogas" "Intercept"   
## [7] "Intercept\_TV" "coef\_log1p\_n\_drogas" "log1p\_n\_drogas\_TV"   
## [10] "Local\_R2" "y" "yhat"   
## [13] "residual" "geometry" "Local\_R2\_plot"

# Bloco 10: Resumo dos Resultados GWR e Análise de Resíduos  
cat("\n--- Bloco 10: Resumo dos Resultados GWR e Análise de Resíduos ---\n")

##   
## --- Bloco 10: Resumo dos Resultados GWR e Análise de Resíduos ---

if (!is.null(resultados\_gwr\_poligonos\_sf) && nrow(resultados\_gwr\_poligonos\_sf) > 0) {  
   
 # SEÇÃO 1: Resumo do R² Local  
 if ("Local\_R2" %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat("\n--- SEÇÃO 1: Resumo do R² Local ---\n")  
 r2\_summary <- summary(resultados\_gwr\_poligonos\_sf$Local\_R2)  
 print(r2\_summary)  
 cat("\n")  
 } else {  
 cat("Coluna 'Local\_R2' não encontrada. Verifique os resultados do GWR.\n")  
 }  
   
 # SEÇÃO 2: Resumo dos Coeficientes  
 coef\_col\_name <- "coef\_n\_drogas"  
 if (coef\_col\_name %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat(paste0("\n--- SEÇÃO 2: Resumo dos Coeficientes '", coef\_col\_name, "' ---\n"))  
 coef\_summary <- summary(resultados\_gwr\_poligonos\_sf[[coef\_col\_name]])  
 print(coef\_summary)  
 cat("\n")  
 } else {  
 cat(paste0("Coluna de coeficiente '", coef\_col\_name, "' não encontrada.\n"))  
 }  
   
 # SEÇÃO 3: Resumo dos Valores-t  
 t\_val\_col\_name <- "n\_drogas\_TV"  
 if (t\_val\_col\_name %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat(paste0("\n--- SEÇÃO 3: Resumo dos Valores-t '", t\_val\_col\_name, "' ---\n"))  
 t\_val\_summary <- summary(resultados\_gwr\_poligonos\_sf[[t\_val\_col\_name]])  
 print(t\_val\_summary)  
 cat("\n")  
 } else {  
 cat(paste0("Coluna de valor-t '", t\_val\_col\_name, "' não encontrada.\n"))  
 }  
   
 # SEÇÃO 4: Análise de Resíduos  
 if ("residual" %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat("\n--- SEÇÃO 4: Análise de Resíduos ---\n")  
 residual\_summary <- summary(resultados\_gwr\_poligonos\_sf$residual)  
 print(residual\_summary)  
 cat("\n")  
   
 # Histograma dos resíduos  
 hist(resultados\_gwr\_poligonos\_sf$residual,   
 main = "Distribuição dos Resíduos do GWR",  
 xlab = "Resíduos",   
 col = "lightblue",   
 breaks = 30)  
   
 # Q-Q plot dos resíduos  
 qqnorm(resultados\_gwr\_poligonos\_sf$residual,   
 main = "Q-Q Plot dos Resíduos")  
 qqline(resultados\_gwr\_poligonos\_sf$residual,   
 col = "red")  
 } else {  
 cat("Coluna 'residual' não encontrada. Análise de resíduos não pode ser realizada.\n")  
 }  
   
 # SEÇÃO 5: Teste de Autocorrelação Espacial dos Resíduos (I de Moran)  
 if ("residual" %in% names(resultados\_gwr\_poligonos\_sf) && nrow(resultados\_gwr\_poligonos\_sf) > 0) {  
 cat("\n--- SEÇÃO 5: Teste de Autocorrelação Espacial dos Resíduos ---\n")  
   
 tryCatch({  
 # Criar matriz de pesos espaciais  
 coords\_residuos <- st\_coordinates(st\_centroid(resultados\_gwr\_poligonos\_sf))  
 nb\_residuos <- knn2nb(knearneigh(coords\_residuos, k = 8))  
 listw\_residuos <- nb2listw(nb\_residuos, style = "W")  
   
 # Teste I de Moran  
 moran\_test <- moran.test(resultados\_gwr\_poligonos\_sf$residual, listw\_residuos)  
 cat("\nTeste I de Moran para os resíduos:\n")  
 print(moran\_test)  
   
 if (moran\_test$p.value < 0.05) {  
 cat("\nAVISO: Autocorrelação espacial significativa detectada nos resíduos (p < 0.05).\n")  
 cat("Isso pode indicar que o modelo GWR não capturou completamente a estrutura espacial.\n")  
 } else {  
 cat("\nNenhuma autocorrelação espacial significativa nos resíduos (p >= 0.05).\n")  
 cat("O modelo GWR parece ter capturado adequadamente a estrutura espacial.\n")  
 }  
 }, error = function(e) {  
 cat("\nErro ao calcular o I de Moran para os resíduos.\n")  
 })  
 } else {  
 cat("Coluna 'residual' não encontrada em 'resultados\_gwr\_poligonos\_sf' ou o objeto está vazio.\n")  
 cat("A análise de autocorrelação espacial dos resíduos não pode ser realizada.\n")  
 }  
  
}

##   
## --- SEÇÃO 1: Resumo do R² Local ---  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.1788 0.2551 0.3980 0.4171 0.5398 0.8013   
##   
## Coluna de coeficiente 'coef\_n\_drogas' não encontrada.  
## Coluna de valor-t 'n\_drogas\_TV' não encontrada.  
##   
## --- SEÇÃO 4: Análise de Resíduos ---  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -2.924125 -0.548303 -0.048512 -0.004367 0.411694 2.588215



##   
## --- SEÇÃO 5: Teste de Autocorrelação Espacial dos Resíduos ---

## Warning: st\_centroid assumes attributes are constant over geometries

##   
## Teste I de Moran para os resíduos:  
##   
## Moran I test under randomisation  
##   
## data: resultados\_gwr\_poligonos\_sf$residual   
## weights: listw\_residuos   
##   
## Moran I statistic standard deviate = 3.2095, p-value = 0.0006649  
## alternative hypothesis: greater  
## sample estimates:  
## Moran I statistic Expectation Variance   
## 0.074480965 -0.002469136 0.000574839   
##   
##   
## AVISO: Autocorrelação espacial significativa detectada nos resíduos (p < 0.05).  
## Isso pode indicar que o modelo GWR não capturou completamente a estrutura espacial.

cat("\n--- Fim do Bloco 10: Resumo GWR (com variáveis transformadas log1p e Análise de Resíduos) ---\n")

##   
## --- Fim do Bloco 10: Resumo GWR (com variáveis transformadas log1p e Análise de Resíduos) ---

# --- Bloco 10 Final: Resumo Completo dos Resultados e Diagnósticos do Modelo GWR ---  
# Consolida e refina a análise final, incluindo diagnósticos e interpretações aprofundadas.  
cat("\n\n--- BL. 10: RESUMO COMPLETO DOS RESULTADOS E DIAGNÓSTICOS DO MODELO GWR ---")

##   
##   
## --- BL. 10: RESUMO COMPLETO DOS RESULTADOS E DIAGNÓSTICOS DO MODELO GWR ---

cat("\n--------------------------------------------------------------------\n")

##   
## --------------------------------------------------------------------

# Verifica se os objetos necessários existem e não estão vazios antes de prosseguir  
if (!exists("gwr\_resultado\_lista") || is.null(gwr\_resultado\_lista) ||  
 !exists("resultados\_gwr\_poligonos\_sf") || is.null(resultados\_gwr\_poligonos\_sf) ||  
 nrow(resultados\_gwr\_poligonos\_sf) == 0) {  
  
 cat("\nERRO: Os resultados do GWR ('gwr\_resultado\_lista' ou 'resultados\_gwr\_poligonos\_sf') não foram encontrados ou estão vazios.\n")  
 cat("Impossível gerar o relatório de análise final. Verifique se os blocos anteriores (carregamento, agregação, GWR) foram executados com sucesso.\n")  
  
} else { # Se os objetos existem e não estão vazios, procede com a análise  
  
 cat("\n## 1. Informações Gerais do Modelo GWR ##")  
 cat("\n------------------------------------\n")  
 # Usa a fórmula real do script (definida anteriormente, e.g., formula\_gwr <- log1p\_n\_roubos ~ log1p\_n\_drogas)  
 cat(paste0("Fórmula do modelo: ", deparse(formula\_gwr), "\n"))  
 cat(paste0("Tipo de kernel: ", gwr\_resultado\_lista$GW.arguments$kernel, "\n"))  
  
 # Tenta extrair o método de seleção da largura de banda de forma segura  
 bw\_approach <- tryCatch({ gwr\_resultado\_lista$GW.arguments$approach }, error = function(e) "Não especificado/Disponível")  
 cat(paste0("Método de seleção de largura de banda: ", bw\_approach, "\n"))  
  
 cat(paste0("Largura de banda ótima (Adaptativa): ", round(gwr\_resultado\_lista$GW.arguments$bw, 2), " vizinhos\n"))  
 cat(paste0("Número de observações (pontos de regressão): ", nrow(gwr\_resultado\_lista$SDF), "\n"))  
  
 # Tenta extrair o CRS final projetado de forma segura  
 final\_crs\_epsg <- tryCatch({ st\_crs(resultados\_gwr\_poligonos\_sf)$epsg }, error = function(e) NA\_integer\_)  
 final\_crs\_epsg\_str <- if (is.na(final\_crs\_epsg)) "Não disponível" else paste0("EPSG:", final\_crs\_epsg)  
 cat(paste0("CRS Final Projetado Utilizado: ", final\_crs\_epsg\_str, "\n"))  
  
 cat("\n## 2. Diagnósticos Globais do Modelo GWR ##")  
 cat("\n---------------------------------------\n")  
  
 # Extrai os diagnósticos do objeto GWR  
 diagnostics <- gwr\_resultado\_lista$GW.diagnostic  
  
 # Imprime diagnósticos essenciais usando a função robusta  
 print\_diagnostic\_robust(diagnostics$AIC, "AIC", "Métrica de qualidade de ajuste, penaliza complexidade. Menor é melhor.")  
 print\_diagnostic\_robust(diagnostics$AICc, "AICc", "AIC corrigido para amostras pequenas. Mais confiável em GWR.")  
 print\_diagnostic\_robust(diagnostics$BIC, "BIC", "Similar ao AIC/AICc, penaliza mais a complexidade. Favorece modelos mais simples.")  
  
 # ENP (Effective Number of Parameters) - Tenta extrair de diferentes locais possíveis ou usa NULL se não encontrar  
 enp\_val <- tryCatch({ diagnostics$ENP }, error = function(e) NULL) # Tenta do slot principal  
 if (is.null(enp\_val) && !is.null(gwr\_resultado\_lista$GW.arguments$ENP)) { # Tenta se estiver nos argumentos  
 enp\_val <- gwr\_resultado\_lista$GW.arguments$ENP  
 }  
 print\_diagnostic\_robust(enp\_val, "ENP (Effective Number of Parameters)", "Mede a complexidade 'efetiva' do modelo GWR.")  
  
 # EDF (Effective Degrees of Freedom) - Geralmente disponível em diagnostics  
 print\_diagnostic\_robust(diagnostics$edf, "EDF (Effective Degrees of Freedom)", "Graus de liberdade efetivos. Relacionado ao ENP.")  
  
 # Sigma (Residual standard error) - Tenta extrair de diferentes locais ou calcula do resíduo se necessário  
 sigma\_val <- tryCatch({ diagnostics$sigma }, error = function(e) NULL) # Tenta do slot principal  
 if (is.null(sigma\_val) || !is.finite(sigma\_val)) { # Se não encontrou ou é inválido, tenta calcular do desvio padrão dos resíduos  
 sigma\_val\_alt <- tryCatch({ sd(resultados\_gwr\_poligonos\_sf$residual, na.rm = TRUE) }, error = function(e) NULL)  
 if (!is.null(sigma\_val\_alt) && is.finite(sigma\_val\_alt)) {  
 sigma\_val <- sigma\_val\_alt  
 print\_diagnostic\_robust(sigma\_val, "Sigma (Est. Desvio Padrão Resíduos - Calculado do SDF)", "Dispersão típica dos resíduos. Menor é melhor.")  
 } else {  
 print\_diagnostic\_robust(NULL, "Sigma (Est. Desvio Padrão Resíduos)", "Dispersão típica dos resíduos. Menor é melhor. (Não disponível/calculável)")  
 }  
 } else {  
 print\_diagnostic\_robust(sigma\_val, "Sigma (Est. Desvio Padrão Resíduos)", "Dispersão típica dos resíduos. Menor é melhor.")  
 }  
  
 # RSS (Residual Sum of Squares) - Geralmente disponível em diagnostics  
 print\_diagnostic\_robust(diagnostics$RSS, "RSS (Residual Sum of Squares)", "Soma dos resíduos quadrados (na escala transformada). Menor é melhor.")  
  
 # R2 Global (Calculado manualmente na escala log1p, como feito no script original)  
 # Verifica se as colunas 'y' (observado) e 'yhat' (previsto) existem no SDF do GWR  
 if ("y" %in% names(gwr\_resultado\_lista$SDF) && "yhat" %in% names(gwr\_resultado\_lista$SDF)) {  
 y\_obs <- gwr\_resultado\_lista$SDF$y  
 y\_hat <- gwr\_resultado\_lista$SDF$yhat  
 # Remove NAs para calcular TSS e RSS  
 valid\_indices <- !is.na(y\_obs) & !is.na(y\_hat)  
 y\_obs\_valid <- y\_obs[valid\_indices]  
 y\_hat\_valid <- y\_hat[valid\_indices]  
  
 if (length(y\_obs\_valid) > 1) { # Certifica-se de que há dados suficientes após remover NAs  
 tss <- sum((y\_obs\_valid - mean(y\_obs\_valid))^2)  
 rss\_calc <- sum((y\_obs\_valid - y\_hat\_valid)^2)  
  
 # Evita divisão por zero se TSS for zero (ocorre se todos os y\_obs forem iguais)  
 if (tss > 1e-9) { # Usa uma pequena tolerância para zero  
 r2\_global\_calc <- 1 - (rss\_calc / tss)  
 cat("\nR2 Global (Calculado - escala transformada):\n")  
 print\_diagnostic\_robust(r2\_global\_calc, " R2 Global", "Proporção da variância da VD (transformada) explicada pelo modelo.")  
  
 # Cálculo R2 Ajustado (requer EDF ou ENP válidos)  
 r2\_ajustado\_calc <- NA\_real\_ # Inicializa com NA  
 n\_obs\_valid <- length(y\_obs\_valid)  
 k\_eff <- if (!is.null(diagnostics$edf) && is.finite(diagnostics$edf)) {  
 diagnostics$edf  
 } else if (!is.null(enp\_val) && is.finite(enp\_val)) {  
 enp\_val  
 } else {  
 NULL # Nenhum valor efetivo de parâmetros encontrado  
 }  
  
 if (!is.null(k\_eff) && n\_obs\_valid > k\_eff + 1) { # Certifica-se de que tem observações > k\_eff + 1  
 r2\_ajustado\_calc <- 1 - ((1 - r2\_global\_calc) \* (n\_obs\_valid - 1) / (n\_obs\_valid - k\_eff - 1))  
 print\_diagnostic\_robust(r2\_ajustado\_calc, " R2 Ajustado", "R2 global penalizado pela complexidade (EDF/ENP).")  
 } else {  
 cat(" R2 Ajustado: Não calculado (EDF/ENP não disponível ou insuficiente observações válidas).\n")  
 }  
 } else {  
 cat(" R2 Global: Não calculável (Todos os valores observados de 'y' são constantes).\n")  
 cat(" R2 Ajustado: Não calculável.\n")  
 }  
 } else {  
 cat("R2 Global (Calculado): Dados insuficientes ou com muitos NAs após filtragem para cálculo.\n")  
 cat("R2 Ajustado: Não calculável.\n")  
 }  
 } else {  
 cat("R2 Global (Calculado): Colunas 'y' ou 'yhat' não encontradas no SDF do GWR.\n")  
 cat("R2 Ajustado: Não calculável.\n")  
 }  
  
 cat("\n## 3. Resumo dos Coeficientes Locais e Significância (na escala log1p) ##")  
 cat("\n---------------------------------------------------------------------\n")  
  
 # Nomes das colunas de interesse (Ajustados para log1p - devem coincidir com os nomes criados no Bloco 9)  
 var\_indep\_formula <- all.vars(formula\_gwr)[2] # Ex: "log1p\_n\_drogas"  
 coef\_col\_name <- paste0("coef\_", var\_indep\_formula) # Ex: "coef\_log1p\_n\_drogas"  
 t\_val\_col\_name <- paste0(var\_indep\_formula, "\_TV") # Ex: "log1p\_n\_drogas\_TV"  
 intercept\_col\_name <- "Intercept" # Nome padrão para o intercepto  
 intercept\_tv\_col\_name <- "Intercept\_TV" # Nome padrão para o t-valor do intercepto  
  
 # Resumo do Intercepto Local  
 if (intercept\_col\_name %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat(paste0("Resumo Estatístico para '", intercept\_col\_name, " (Intercepto Local)':\n"))  
 print(summary(resultados\_gwr\_poligonos\_sf[[intercept\_col\_name]]))  
 cat(" -> Representa o valor esperado da variável dependente (log1p\_n\_roubos) quando todas as variáveis independentes são zero, variando em cada localidade.\n")  
 } else {  
 cat(paste0("Coluna de coeficiente '", intercept\_col\_name, "' não encontrada nos resultados finais (esperado nome: ", intercept\_col\_name, ").\n"))  
 }  
  
 # Resumo do Valor-t do Intercepto  
 if (intercept\_tv\_col\_name %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat(paste0("\nResumo Estatístico para '", intercept\_tv\_col\_name, " (Valor-t Intercepto Local)':\n"))  
 print(summary(resultados\_gwr\_poligonos\_sf[[intercept\_tv\_col\_name]]))  
 cat(" -> Indica a significância estatística do Intercepto local em cada ponto. |t| > ~1.96 sugere significância a p < 0.05 (bicaudal).\n")  
 } else {  
 cat(paste0("\nColuna de valor-t '", intercept\_tv\_col\_name, "' não encontrada nos resultados finais (esperado nome: ", intercept\_tv\_col\_name, ").\n"))  
 }  
  
 # Resumo do Coeficiente Local da Variável Independente Principal (log1p\_n\_drogas)  
 if (coef\_col\_name %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat(paste0("\nResumo Estatístico para '", coef\_col\_name, " (Coeficiente Local de ", var\_indep\_formula, ")':\n"))  
 print(summary(resultados\_gwr\_poligonos\_sf[[coef\_col\_name]]))  
 cat(paste0(" -> Mede a força e a direção da relação entre '", var\_indep\_formula, "' e 'log1p\_n\_roubos' em cada localidade.\n"))  
 cat(" \*\*A VARIAÇÃO NESTES VALORES (Min. a Max.) É A EVIDÊNCIA CHAVE DA NÃO ESTACIONARIEDADE ESPACIAL.\*\*\n") # COMENTÁRIO EXPANDIDO AQUI  
 cat(" Isso justifica o uso do GWR: o efeito da variável preditora não é o mesmo em toda a área de estudo.\n")  
  
 # Interpretação detalhada dos Coeficientes  
 mean\_coef <- mean(resultados\_gwr\_poligonos\_sf[[coef\_col\_name]], na.rm = TRUE)  
 sd\_coef <- sd(resultados\_gwr\_poligonos\_sf[[coef\_col\_name]], na.rm = TRUE)  
 cat(paste0("\n - Média dos coeficientes locais: ", round(mean\_coef, 4), "\n"))  
 cat(paste0(" - Desvio Padrão dos coeficientes locais: ", round(sd\_coef, 4), "\n"))  
 total\_valid\_coef <- sum(!is.na(resultados\_gwr\_poligonos\_sf[[coef\_col\_name]]))  
 if (total\_valid\_coef > 0) {  
 prop\_positivo <- sum(resultados\_gwr\_poligonos\_sf[[coef\_col\_name]] > 0, na.rm = TRUE) / total\_valid\_coef \* 100  
 prop\_negativo <- sum(resultados\_gwr\_poligonos\_sf[[coef\_col\_name]] < 0, na.rm = TRUE) / total\_valid\_coef \* 100  
 cat(paste0(" - Proporção de coeficientes positivos: ", round(prop\_positivo, 1), " %\n"))  
 cat(paste0(" - Proporção de coeficientes negativos: ", round(prop\_negativo, 1), " %\n"))  
 } else {  
 cat(" Nenhum coeficiente válido para análise de proporção.\n")  
 }  
  
 } else {  
 cat(paste0("Coluna de coeficiente '", coef\_col\_name, "' não encontrada nos resultados finais (esperado nome: ", coef\_col\_name, ").\n"))  
 }  
  
 # Resumo dos Valores-t Locais da Variável Independente Principal (log1p\_n\_drogas\_TV)  
 if (t\_val\_col\_name %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat(paste0("\nResumo Estatístico para '", t\_val\_col\_name, " (Valor-t Local de ", var\_indep\_formula, ")':\n"))  
 print(summary(resultados\_gwr\_poligonos\_sf[[t\_val\_col\_name]]))  
 cat(paste0(" -> Indica a significância estatística do coeficiente local de '", var\_indep\_formula, "' em cada localidade.\n"))  
 cat(" Valores |t| maiores sugerem maior confiança de que o coeficiente local é diferente de zero.\n")  
 cat(" |t| > ~1.96 sugere significância estatística a p < 0.05 (bicaudal, aproximação).\n")  
  
 # Análise de Significância Detalhada baseada em limites de valor-t  
 total\_valid\_t <- sum(!is.na(resultados\_gwr\_poligonos\_sf[[t\_val\_col\_name]]))  
 if (total\_valid\_t > 0) {  
 sig\_5\_count <- sum(abs(resultados\_gwr\_poligonos\_sf[[t\_val\_col\_name]]) > 1.96, na.rm = TRUE)  
 sig\_1\_count <- sum(abs(resultados\_gwr\_poligonos\_sf[[t\_val\_col\_name]]) > 2.58, na.rm = TRUE) # Limite aproximado para p < 0.01  
 cat(paste0("\n - Número de localidades significantes a 5% (|t| > 1.96): ", sig\_5\_count, " (", round(sig\_5\_count/total\_valid\_t \* 100, 1), " %)\n"))  
 cat(paste0(" - Número de localidades significantes a 1% (|t| > 2.58): ", sig\_1\_count, " (", round(sig\_1\_count/total\_valid\_t \* 100, 1), " %)\n"))  
 cat(paste0(" - Número de localidades NÃO significantes a 5%: ", total\_valid\_t - sig\_5\_count, " (", round((total\_valid\_t - sig\_5\_count)/total\_valid\_t \* 100, 1), " %)\n"))  
 } else {  
 cat(" Nenhum valor-t válido para análise de significância.\n")  
 }  
  
 } else {  
 cat(paste0("\nColuna de valor-t '", t\_val\_col\_name, "' não encontrada nos resultados finais (esperado nome: ", t\_val\_col\_name, ").\n"))  
 }  
  
 # Resumo do R2 Local  
 r2\_col\_name <- "Local\_R2"  
 if (r2\_col\_name %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat(paste0("\n## 4. Resumo do R2 Local (Poder Explicativo Variável) ##\n"))  
 cat("------------------------------------------------------\n")  
 cat(paste0("Resumo Estatístico para '", r2\_col\_name, "':\n"))  
 print(summary(resultados\_gwr\_poligonos\_sf[[r2\_col\_name]]))  
 cat(" -> Indica a proporção da variação local na variável dependente (log1p\_n\_roubos) que é explicada pelo modelo GWR em cada localidade.\n")  
 cat(" Uma alta variação (Min. a Max.) no R2 Local confirma que o poder explicativo do modelo difere no espaço.\n")  
  
 # Interpretação detalhada do R2 Local  
 mean\_r2 <- mean(resultados\_gwr\_poligonos\_sf[[r2\_col\_name]], na.rm = TRUE)  
 sd\_r2 <- sd(resultados\_gwr\_poligonos\_sf[[r2\_col\_name]], na.rm = TRUE)  
 cat(paste0("\n - Média do R2 Local: ", round(mean\_r2, 3), "\n"))  
 cat(paste0(" - Desvio Padrão do R2 Local: ", round(sd\_r2, 3), "\n"))  
 # Coeficiente de Variação do R2 Local (se a média > uma pequena tolerância para evitar divisão por zero)  
 if (mean\_r2 > 1e-9) {  
 cv\_r2 <- (sd\_r2 / mean\_r2) \* 100  
 cat(paste0(" - Coeficiente de Variação do R2 Local: ", round(cv\_r2, 1), " % (Um CV alto indica grande variabilidade na capacidade explicativa do modelo entre as localidades)\n"))  
 } else {  
 cat(" - Coeficiente de Variação do R2 Local: Não aplicável (Média do R2 é zero ou negativa).\n")  
 }  
  
 # Distribuição do R2 Local por classes predefinidas  
 r2\_classes <- cut(resultados\_gwr\_poligonos\_sf[[r2\_col\_name]],  
 breaks = c(-Inf, 0.1, 0.3, 0.5, 0.7, 1), # Classes: <0.1, 0.1-0.3, 0.3-0.5, 0.5-0.7, >0.7  
 labels = c("<0.1 (Muito Baixo)", "0.1-0.3 (Baixo)", "0.3-0.5 (Moderado)",  
 "0.5-0.7 (Alto)", ">0.7 (Muito Alto)"),  
 include.lowest = TRUE, right = TRUE) # include.lowest=TRUE inclui o 0, right=TRUE inclui o limite superior  
 cat("\n Distribuição do R2 Local por classes:\n")  
 print(table(r2\_classes, useNA = "ifany")) # useNA = "ifany" para incluir contagem de NAs, se houver  
 cat("\n")  
  
 } else {  
 cat(paste0("Coluna '", r2\_col\_name, "' não encontrada nos resultados finais (esperado nome: ", r2\_col\_name, ").\n"))  
 }  
  
 cat("\n## 5. Análise de Resíduos ##")  
 cat("\n---------------------------\n")  
  
 residual\_col\_name <- "residual"  
 if (residual\_col\_name %in% names(resultados\_gwr\_poligonos\_sf)) {  
 cat(paste0("Resumo Estatístico para '", residual\_col\_name, "':\n"))  
 print(summary(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]]))  
 cat(" -> As diferenças entre os valores observados (log1p\_n\_roubos) e os valores previstos (yhat) pelo modelo GWR em cada localidade.\n")  
 cat(" Resíduos indicam a porção da variável dependente que não foi explicada pelo modelo.\n")  
  
 # Estatísticas Adicionais dos Resíduos (Média, DP, Assimetria, Curtose)  
 mean\_res <- mean(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]], na.rm = TRUE)  
 sd\_res <- sd(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]], na.rm = TRUE)  
 cat(paste0("\n - Média dos resíduos: ", round(mean\_res, 6), " (Idealmente próximo de zero)\n"))  
 cat(paste0(" - Desvio Padrão dos resíduos: ", round(sd\_res, 4), "\n"))  
  
 # Verifica se o pacote e1071 está carregado para Assimetria e Curtose  
 if (requireNamespace("e1071", quietly = TRUE)) {  
 skew\_res <- e1071::skewness(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]], na.rm = TRUE)  
 kurt\_res <- e1071::kurtosis(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]], na.rm = TRUE)  
 cat(paste0(" - Assimetria (Skewness): ", round(skew\_res, 4), " (Mede a simetria da distribuição. 0 indica simetria)\n"))  
 cat(paste0(" - Curtose (Kurtosis): ", round(kurt\_res, 4), " (Mede o 'achatamento'/'caudas' da distribuição. 0 para Normal (Fisher), 3 para Normal (Pearson))\n"))  
 } else {  
 cat(" Instale o pacote 'e1071' para calcular Assimetria e Curtose: install.packages('e1071')\n")  
 }  
  
 # Teste de Normalidade dos Resíduos (Shapiro-Wilk)  
 # Testa apenas se houver dados residuais válidos  
 valid\_residuals <- resultados\_gwr\_poligonos\_sf[[residual\_col\_name]][!is.na(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]])]  
 if (length(valid\_residuals) > 3) { # Shapiro-Wilk requer pelo menos 4 dados não NA  
 shapiro\_test <- tryCatch({  
 # Shapiro-Wilk limitado a 5000 observações para performance. Amostra se necessário.  
 res\_sample <- if (length(valid\_residuals) > 5000) {  
 sample(valid\_residuals, 5000, replace = FALSE)  
 } else {  
 valid\_residuals  
 }  
 shapiro.test(res\_sample)  
 }, error = function(e) {  
 cat("\n Erro ao executar o Teste de Shapiro-Wilk: ", e$message, "\n")  
 NULL # Retorna NULL em caso de erro  
 })  
  
 if (!is.null(shapiro\_test)) {  
 cat("\n Teste de Normalidade dos Resíduos (Shapiro-Wilk):\n")  
 cat(paste0(" - Estatística W: ", round(shapiro\_test$statistic, 4), "\n"))  
 cat(paste0(" - p-valor: ", format(shapiro\_test$p.value, scientific = TRUE), "\n"))  
 if (shapiro\_test$p.value < 0.05) {  
 cat(" - Conclusão: Resíduos NÃO seguem distribuição normal (p < 0.05).\n")  
 cat(" -> \*\*Implicação:\*\* A não normalidade dos resíduos é uma violação de um pressuposto do modelo GWR Gaussiano.\n") # COMENTÁRIO EXPANDIDO AQUI  
 cat(" -> Isso pode afetar a validade dos testes de significância baseados no pressuposto de normalidade (valores-t).\n")  
 cat(" -> \*\*Sugestão:\*\* Considere usar Generalized Geographically Weighted Regression (GGWR) com uma família de distribuição apropriada para seus dados de contagem (ex: Binomial Negativa, 'nbinomial'), que não pressupõe normalidade dos erros.\n") # SUGESTÃO GGWR  
 } else {  
 cat(" - Conclusão: Não há evidência contra a normalidade dos resíduos (p >= 0.05).\n")  
 }  
 } else {  
 cat("\n Teste de Shapiro-Wilk não pôde ser realizado.\n")  
 }  
 } else {  
 cat("\n Dados residuais insuficientes ou inválidos para realizar o Teste de Normalidade (Shapiro-Wilk requer > 3 dados válidos).\n")  
 }  
  
 # Plotagem de Resíduos (Histograma e Q-Q Plot) - GERA GRÁFICOS, NÃO TEXTO. Comentei a chamada direta no texto de output.  
 # Para ver os gráficos, remova o '#' das linhas abaixo no seu script R.  
 # cat("\n -> Verifique o Histograma e o Q-Q Plot dos Resíduos (gerados separadamente) para visualização da distribuição.\n")  
 # hist(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]], main = "Distribuição dos Resíduos do GWR", xlab = "Resíduos", col = "lightblue", breaks = 30)  
 # qqnorm(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]], main = "Q-Q Plot dos Resíduos")  
 # qqline(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]], col = "red")  
  
 } else {  
 cat(paste0("Coluna '", residual\_col\_name, "' não encontrada nos resultados finais. Análise de resíduos não pode ser realizada.\n"))  
 }  
  
 cat("\n## 6. Autocorrelação Espacial dos Resíduos (Teste I de Moran) ##")  
 cat("\n-----------------------------------------------------------\n")  
  
 # Verifica se a coluna de resíduos existe e se há observações suficientes para o Moran  
 if (residual\_col\_name %in% names(resultados\_gwr\_poligonos\_sf) && nrow(resultados\_gwr\_poligonos\_sf) >= 30) {  
 tryCatch({  
 # --- Criação da Matriz de Vizinhança e Lista de Pesos ---  
 # Utiliza vizinhança por adjacência (rainha), mais apropriada para polígonos.  
 # É CRUCIAL que resultados\_gwr\_poligonos\_sf seja um objeto espacial (sf ou sp).  
 if (!inherits(resultados\_gwr\_poligonos\_sf, "sf") && !inherits(resultados\_gwr\_poligonos\_sf, "Spatial")) {  
 stop("O objeto de resultados ('resultados\_gwr\_poligonos\_sf') não é um objeto espacial válido (sf ou sp).")  
 }  
  
 # Converte para objeto 'sp' se for 'sf', pois 'poly2nb' ainda pode preferir 'sp'  
 resultados\_sp <- as(resultados\_gwr\_poligonos\_sf, "Spatial")  
  
 # Cria a matriz de vizinhança por adjacência (queen = TRUE considera cantos)  
 nb\_residuos <- poly2nb(resultados\_sp, queen = TRUE)  
  
 # Verifica se há polígonos isolados (sem vizinhos) e lida com eles (zero.policy=TRUE)  
 # Uma contagem de vizinhos 0 indica um polígono isolado.  
 isolated <- which(card(nb\_residuos) == 0)  
 if (length(isolated) > 0) {  
 cat(paste0(" AVISO: Foram encontrados ", length(isolated), " polígono(s) isolado(s) (sem vizinhos) na matriz de adjacência.\n"))  
 cat(" Estes polígonos serão excluídos do cálculo da lista de pesos (zero.policy=TRUE).\n")  
 }  
  
 # Converte para lista de pesos espaciais (style="W" - row-standardized é padrão e comum)  
 listw\_residuos <- nb2listw(nb\_residuos, style = "W", zero.policy = TRUE) # zero.policy=TRUE é necessário para lidar com vizinhos isolados  
  
 # --- Executa o Teste I de Moran ---  
 # 'alternative = "greater"' testa autocorrelação espacial positiva (mais comum em resíduos de regressão mal especificada)  
 moran\_test <- moran.test(resultados\_gwr\_poligonos\_sf[[residual\_col\_name]], listw\_residuos, alternative = "greater", zero.policy = TRUE)  
  
 cat("\n Teste I de Moran para os resíduos (baseado em vizinhança por adjacência):\n")  
 cat(paste0(" - Estatística I de Moran: ", round(moran\_test$estimate[1], 4), "\n"))  
 cat(paste0(" - Expectativa sob H0 (aleatoriedade espacial): ", round(moran\_test$estimate[2], 4), "\n"))  
 cat(paste0(" - Variância: ", round(moran\_test$estimate[3], 6), "\n"))  
 cat(paste0(" - Desvio padrão: ", round(sqrt(moran\_test$estimate[3]), 4), "\n"))  
 cat(paste0(" - p-valor (alternativa: maior que a expectativa): ", format(moran\_test$p.value, scientific = TRUE), "\n"))  
  
 # --- Interpretação do Resultado do Teste de Moran ---  
 if (moran\_test$p.value < 0.05) {  
 cat("\n --> \*\*AVISO CRÍTICO:\*\* Foi detectada AUTOCORRELAÇÃO ESPACIAL POSITIVA e estatisticamente SIGNIFICATIVA nos resíduos (p-valor < 0.05).\n") # COMENTÁRIO EXPANDIDO AQUI  
 cat(" Isso é um indicador forte de que o modelo GWR atual NÃO capturou completamente a estrutura espacial subjacente nos dados de roubos.\n")  
 cat(" Resíduos em localidades próximas tendem a ser semelhantes (altos resíduos perto de altos resíduos, baixos perto de baixos).\n")  
 cat(" \*\*Possíveis causas e soluções:\*\*\n")  
 cat(" - \*\*Variáveis Omitidas:\*\* Fatores espaciais importantes que influenciam os roubos e/ou a relação drogas-roubos não foram incluídos no modelo.\n")  
 cat(" - \*\*Forma Funcional:\*\* A relação entre as variáveis pode ser mais complexa do que a modelada (e.g., não linear, interações espaciais não capturadas).\n")  
 cat(" - \*\*Necessidade de Modelagem de Erro Espacial:\*\* Pode ser apropriado usar modelos que incluam um termo de erro espacial explícito (ex: modelos no framework INLA, ou GWR-ESF - Geographically Weighted Regression with Eigenvector Spatial Filtering).\n") # SUGESTÕES ADICIONAIS  
 } else {  
 cat("\n --> Nenhuma autocorrelação espacial significativa detectada nos resíduos (p-valor >= 0.05).\n")  
 cat(" Isso sugere que o modelo GWR parece ter capturado a maior parte da estrutura espacial nos dados.\n")  
 }  
  
 }, error = function(e) {  
 # Reporta erros específicos que podem ocorrer no cálculo do Moran  
 cat("\nERRO INESPERADO ao calcular ou reportar o Teste I de Moran para os resíduos:\n")  
 cat(paste(" Mensagem de erro:", e$message, "\n"))  
 cat(" Verifique se o objeto espacial final 'resultados\_gwr\_poligonos\_sf' é válido, se o pacote 'spdep' está carregado e se a matriz de vizinhança pode ser criada.\n")  
 })  
 } else {  
 # Informa por que o Moran não foi calculado  
 cat("\nNão foi possível calcular o Teste I de Moran para os resíduos.\n")  
 if (!(residual\_col\_name %in% names(resultados\_gwr\_poligonos\_sf))) {  
 cat(paste0(" - A coluna '", residual\_col\_name, "' (resíduos) não foi encontrada nos resultados finais.\n"))  
 }  
 if (nrow(resultados\_gwr\_poligonos\_sf) < 30) {  
 cat(paste0(" - Número insuficiente de observações (", nrow(resultados\_gwr\_poligonos\_sf), ") para calcular o Teste I de Moran (mínimo recomendado: 30).\n"))  
 }  
 }  
  
  
   
  
} # Fim do if(exists)

##   
## ## 1. Informações Gerais do Modelo GWR ##  
## ------------------------------------  
## Fórmula do modelo: log1p\_n\_roubos ~ log1p\_n\_drogas  
## Tipo de kernel: gaussian  
## Método de seleção de largura de banda:   
## Largura de banda ótima (Adaptativa): 18 vizinhos  
## Número de observações (pontos de regressão): 406  
## CRS Final Projetado Utilizado: EPSG:31983  
##   
## ## 2. Diagnósticos Globais do Modelo GWR ##  
## ---------------------------------------  
## AIC: 881.229  
## -> Métrica de qualidade de ajuste, penaliza complexidade. Menor é melhor.  
## AICc: 911.9409  
## -> AIC corrigido para amostras pequenas. Mais confiável em GWR.  
## BIC: 600.4164  
## -> Similar ao AIC/AICc, penaliza mais a complexidade. Favorece modelos mais simples.  
## ENP (Effective Number of Parameters): Não disponível ou não numérico/finito (Valor: NULL).  
## -> Mede a complexidade 'efetiva' do modelo GWR. (Não calculado/Reportado)  
## EDF (Effective Degrees of Freedom): 370.3623  
## -> Graus de liberdade efetivos. Relacionado ao ENP.  
## Sigma (Est. Desvio Padrão Resíduos - Calculado do SDF): 0.6954  
## -> Dispersão típica dos resíduos. Menor é melhor.  
## RSS (Residual Sum of Squares): 195.8601  
## -> Soma dos resíduos quadrados (na escala transformada). Menor é melhor.  
##   
## R2 Global (Calculado - escala transformada):  
## R2 Global: 0.5059  
## -> Proporção da variância da VD (transformada) explicada pelo modelo.  
## R2 Ajustado: -4.7775  
## -> R2 global penalizado pela complexidade (EDF/ENP).  
##   
## ## 3. Resumo dos Coeficientes Locais e Significância (na escala log1p) ##  
## ---------------------------------------------------------------------  
## Resumo Estatístico para 'Intercept (Intercepto Local)':  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -3.64426 -1.21255 -0.26722 -0.67367 0.08167 0.57886   
## -> Representa o valor esperado da variável dependente (log1p\_n\_roubos) quando todas as variáveis independentes são zero, variando em cada localidade.  
##   
## Resumo Estatístico para 'Intercept\_TV (Valor-t Intercepto Local)':  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -8.1480 -3.2234 -0.8166 -1.6113 0.2879 1.6091   
## -> Indica a significância estatística do Intercepto local em cada ponto. |t| > ~1.96 sugere significância a p < 0.05 (bicaudal).  
##   
## Resumo Estatístico para 'coef\_log1p\_n\_drogas (Coeficiente Local de log1p\_n\_drogas)':  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.05208 0.19378 0.25260 0.35453 0.48359 0.99487   
## -> Mede a força e a direção da relação entre 'log1p\_n\_drogas' e 'log1p\_n\_roubos' em cada localidade.  
## \*\*A VARIAÇÃO NESTES VALORES (Min. a Max.) É A EVIDÊNCIA CHAVE DA NÃO ESTACIONARIEDADE ESPACIAL.\*\*  
## Isso justifica o uso do GWR: o efeito da variável preditora não é o mesmo em toda a área de estudo.  
##   
## - Média dos coeficientes locais: 0.3545  
## - Desvio Padrão dos coeficientes locais: 0.2314  
## - Proporção de coeficientes positivos: 100 %  
## - Proporção de coeficientes negativos: 0 %  
##   
## Resumo Estatístico para 'log1p\_n\_drogas\_TV (Valor-t Local de log1p\_n\_drogas)':  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.4926 2.4993 3.2024 4.4190 6.1797 11.8244   
## -> Indica a significância estatística do coeficiente local de 'log1p\_n\_drogas' em cada localidade.  
## Valores |t| maiores sugerem maior confiança de que o coeficiente local é diferente de zero.  
## |t| > ~1.96 sugere significância estatística a p < 0.05 (bicaudal, aproximação).  
##   
## - Número de localidades significantes a 5% (|t| > 1.96): 368 (90.6 %)  
## - Número de localidades significantes a 1% (|t| > 2.58): 295 (72.7 %)  
## - Número de localidades NÃO significantes a 5%: 38 (9.4 %)  
##   
## ## 4. Resumo do R2 Local (Poder Explicativo Variável) ##  
## ------------------------------------------------------  
## Resumo Estatístico para 'Local\_R2':  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.1788 0.2551 0.3980 0.4171 0.5398 0.8013   
## -> Indica a proporção da variação local na variável dependente (log1p\_n\_roubos) que é explicada pelo modelo GWR em cada localidade.  
## Uma alta variação (Min. a Max.) no R2 Local confirma que o poder explicativo do modelo difere no espaço.  
##   
## - Média do R2 Local: 0.417  
## - Desvio Padrão do R2 Local: 0.173  
## - Coeficiente de Variação do R2 Local: 41.6 % (Um CV alto indica grande variabilidade na capacidade explicativa do modelo entre as localidades)  
##   
## Distribuição do R2 Local por classes:  
## r2\_classes  
## <0.1 (Muito Baixo) 0.1-0.3 (Baixo) 0.3-0.5 (Moderado) 0.5-0.7 (Alto)   
## 0 138 141 97   
## >0.7 (Muito Alto)   
## 30   
##   
##   
## ## 5. Análise de Resíduos ##  
## ---------------------------  
## Resumo Estatístico para 'residual':  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -2.924125 -0.548303 -0.048512 -0.004367 0.411694 2.588215   
## -> As diferenças entre os valores observados (log1p\_n\_roubos) e os valores previstos (yhat) pelo modelo GWR em cada localidade.  
## Resíduos indicam a porção da variável dependente que não foi explicada pelo modelo.  
##   
## - Média dos resíduos: -0.004367 (Idealmente próximo de zero)  
## - Desvio Padrão dos resíduos: 0.6954  
## - Assimetria (Skewness): 0.4607 (Mede a simetria da distribuição. 0 indica simetria)  
## - Curtose (Kurtosis): 1.1451 (Mede o 'achatamento'/'caudas' da distribuição. 0 para Normal (Fisher), 3 para Normal (Pearson))  
##   
## Teste de Normalidade dos Resíduos (Shapiro-Wilk):  
## - Estatística W: 0.9717  
## - p-valor: 4.326061e-07  
## - Conclusão: Resíduos NÃO seguem distribuição normal (p < 0.05).  
## -> \*\*Implicação:\*\* A não normalidade dos resíduos é uma violação de um pressuposto do modelo GWR Gaussiano.  
## -> Isso pode afetar a validade dos testes de significância baseados no pressuposto de normalidade (valores-t).  
## -> \*\*Sugestão:\*\* Considere usar Generalized Geographically Weighted Regression (GGWR) com uma família de distribuição apropriada para seus dados de contagem (ex: Binomial Negativa, 'nbinomial'), que não pressupõe normalidade dos erros.  
##   
## ## 6. Autocorrelação Espacial dos Resíduos (Teste I de Moran) ##  
## -----------------------------------------------------------  
##   
## Teste I de Moran para os resíduos (baseado em vizinhança por adjacência):  
## - Estatística I de Moran: 0.102  
## - Expectativa sob H0 (aleatoriedade espacial): -0.0025  
## - Variância: 0.000935  
## - Desvio padrão: 0.0306  
## - p-valor (alternativa: maior que a expectativa): 3.17255e-04  
##   
## --> \*\*AVISO CRÍTICO:\*\* Foi detectada AUTOCORRELAÇÃO ESPACIAL POSITIVA e estatisticamente SIGNIFICATIVA nos resíduos (p-valor < 0.05).  
## Isso é um indicador forte de que o modelo GWR atual NÃO capturou completamente a estrutura espacial subjacente nos dados de roubos.  
## Resíduos em localidades próximas tendem a ser semelhantes (altos resíduos perto de altos resíduos, baixos perto de baixos).  
## \*\*Possíveis causas e soluções:\*\*  
## - \*\*Variáveis Omitidas:\*\* Fatores espaciais importantes que influenciam os roubos e/ou a relação drogas-roubos não foram incluídos no modelo.  
## - \*\*Forma Funcional:\*\* A relação entre as variáveis pode ser mais complexa do que a modelada (e.g., não linear, interações espaciais não capturadas).  
## - \*\*Necessidade de Modelagem de Erro Espacial:\*\* Pode ser apropriado usar modelos que incluam um termo de erro espacial explícito (ex: modelos no framework INLA, ou GWR-ESF - Geographically Weighted Regression with Eigenvector Spatial Filtering).

# --- Bloco 11: Salvar Resultados Finais em GeoPackage ---  
cat("\n\n--- Bloco 11: Salvar Resultados Finais em GeoPackage ---\n")

##   
##   
## --- Bloco 11: Salvar Resultados Finais em GeoPackage ---

# Verificar se a biblioteca 'sf' está carregada, pois é necessária para st\_write.  
if (!requireNamespace("sf", quietly = TRUE)) {  
 cat("AVISO: O pacote 'sf' não está instalado. Não é possível salvar em GeoPackage.\n")  
 cat("Por favor, instale o pacote com: install.packages('sf')\n")  
} else if (!exists("resultados\_gwr\_poligonos\_sf") || is.null(resultados\_gwr\_poligonos\_sf) || nrow(resultados\_gwr\_poligonos\_sf) == 0) {  
 cat("AVISO: O objeto 'resultados\_gwr\_poligonos\_sf' não foi encontrado ou está vazio.\n")  
 cat("Nenhum resultado para salvar em GeoPackage. Verifique as etapas anteriores do script.\n")  
} else {  
 # Definir o caminho completo e o nome do arquivo GeoPackage de saída  
 caminho\_diretorio\_saida <- "C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/h3/resultados\_gwr/"  
 nome\_base\_arquivo <- "resultados\_analise\_gwr\_final.gpkg" # Você pode alterar o nome do arquivo se desejar  
 nome\_arquivo\_gpkg\_completo <- file.path(caminho\_diretorio\_saida, nome\_base\_arquivo)  
  
 nome\_camada\_gpkg <- "gwr\_coeficientes\_locais" # Nome da camada dentro do GeoPackage  
  
 cat(paste("Tentando salvar os resultados GWR no arquivo GeoPackage em:", nome\_arquivo\_gpkg\_completo, "\n"))  
 cat(paste("Nome da camada a ser criada:", nome\_camada\_gpkg, "\n"))  
  
 # Opcional: Verificar se o diretório de saída existe.  
 # Se não existir, você pode querer criá-lo com dir.create(caminho\_diretorio\_saida, recursive = TRUE, showWarnings = FALSE)  
 # ou simplesmente alertar o usuário. Por ora, vamos assumir que ele existe.  
 if (!dir.exists(caminho\_diretorio\_saida)) {  
 cat(paste("ALERTA: O diretório de saída especificado NÃO EXISTE:", caminho\_diretorio\_saida, "\n"))  
 cat("Por favor, crie o diretório manualmente ou ajuste o caminho no script.\n")  
 # Você pode optar por parar o script aqui se o diretório for crucial.  
 # stop("Diretório de saída não encontrado.")  
 }  
  
 tryCatch({  
 # Salvar o objeto sf (SpatialPolygonsDataFrame com resultados GWR) em um arquivo GeoPackage.  
 sf::st\_write(obj = resultados\_gwr\_poligonos\_sf,  
 dsn = nome\_arquivo\_gpkg\_completo, # Usar o caminho completo  
 layer = nome\_camada\_gpkg,  
 driver = "GPKG", # Especifica o driver para GeoPackage  
 delete\_layer = TRUE, # Sobrescreve a camada se já existir  
 # Use delete\_dsn = TRUE para sobrescrever o arquivo inteiro  
 quiet = FALSE) # Mostra mensagens do processo  
  
 cat(paste("\nResultados GWR salvos com sucesso em:", nome\_arquivo\_gpkg\_completo, "como camada:", nome\_camada\_gpkg, "\n"))  
 cat("Você pode abrir este arquivo em um software GIS (QGIS, ArcGIS, etc.) para visualizar os mapas de coeficientes, valores-t e R² local.\n")  
  
 }, error = function(e) {  
 cat("\nERRO AO SALVAR EM GEOPACKAGE:\n")  
 cat(paste("Mensagem de erro:", e$message, "\n"))  
 cat(paste("Verifique se o caminho do diretório está correto, se o diretório existe e se você tem permissão de escrita nele:\n", caminho\_diretorio\_saida, "\n"))  
 cat("Certifique-se também de que o pacote 'sf' está funcionando corretamente.\n")  
 })  
}

## Tentando salvar os resultados GWR no arquivo GeoPackage em: C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/h3/resultados\_gwr//resultados\_analise\_gwr\_final.gpkg   
## Nome da camada a ser criada: gwr\_coeficientes\_locais   
## Deleting layer `gwr\_coeficientes\_locais' using driver `GPKG'  
## Writing layer `gwr\_coeficientes\_locais' to data source   
## `C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/h3/resultados\_gwr//resultados\_analise\_gwr\_final.gpkg' using driver `GPKG'  
## Writing 406 features with 14 fields and geometry type Polygon.  
##   
## Resultados GWR salvos com sucesso em: C:/Users/Rodrigo - H2R/OneDrive - Conhecimento e Ação em Marketing Ltda/Documentos/mba/mba\_arrumado/nova abordagem/h3/resultados\_gwr//resultados\_analise\_gwr\_final.gpkg como camada: gwr\_coeficientes\_locais   
## Você pode abrir este arquivo em um software GIS (QGIS, ArcGIS, etc.) para visualizar os mapas de coeficientes, valores-t e R² local.

cat("\n--- Fim do Bloco 11: Salvar Resultados ---\n")

##   
## --- Fim do Bloco 11: Salvar Resultados ---

# Dentro do Bloco 10, na seção "## 2. Diagnósticos Globais do Modelo GWR ##"  
  
if (!is.null(gwr\_resultado\_lista$GW.diagnostic)) {  
 diagnostics <- gwr\_resultado\_lista$GW.diagnostic  
  
 # Função auxiliar para verificar e imprimir diagnósticos  
 print\_diagnostic <- function(value, name, description) {  
 if (!is.null(value) && is.numeric(value) && is.finite(value)) { # is.finite() verifica NA, NaN, Inf  
 cat(paste0(name, ": ", round(value, 2), "\n"))  
 cat(paste0(" -> ", description, "\n"))  
 } else {  
 cat(paste0(name, ": Não disponível ou não numérico (Valor: ", as.character(value), ").\n"))  
 }  
 }  
  
 # AICc  
 print\_diagnostic(diagnostics$AICc, "AICc (Akaike Information Criterion Corrigido)", "O AICc é uma métrica de qualidade de ajuste que penaliza modelos com mais parâmetros...")  
  
 # ENP  
 print\_diagnostic(diagnostics$ENP, "ENP (Effective Number of Parameters)", "O ENP reflete a complexidade do modelo GWR, indicando o número efetivo de parâmetros...")  
  
 # EDF  
 print\_diagnostic(diagnostics$edf, "EDF (Effective Degrees of Freedom)", "Similar ao ENP, o EDF é usado em cálculos estatísticos e representa os graus de liberdade efetivos do modelo...")  
  
 # Sigma  
 print\_diagnostic(diagnostics$sigma, "Sigma (Estimativa do Desvio Padrão dos Resíduos)", "O sigma representa o desvio padrão dos resíduos, uma medida da dispersão dos erros do modelo...")  
  
 # RSS  
 print\_diagnostic(diagnostics$RSS, "RSS (Residual Sum of Squares)", "O RSS mede a soma das diferenças quadráticas entre os valores observados e previstos, indicando a variação não explicada pelo modelo...")  
  
 # Cálculo do R² Global (manter como está, pois já tem uma verificação if)  
 if ("y" %in% names(gwr\_resultado\_lista$SDF) && "yhat" %in% names(gwr\_resultado\_lista$SDF)) {  
 # ... (código do R² global) ...  
 } else {  
 cat("R² Global (Calculado): Não foi possível calcular pois 'y' ou 'yhat' não estão presentes no SDF do GWR.\n")  
 }  
} else {  
 cat("Diagnósticos globais (gwr\_resultado\_lista$GW.diagnostic) não disponíveis ou a estrutura está vazia.\n")  
}

## AICc (Akaike Information Criterion Corrigido): 911.94  
## -> O AICc é uma métrica de qualidade de ajuste que penaliza modelos com mais parâmetros...  
## ENP (Effective Number of Parameters): Não disponível ou não numérico (Valor: ).  
## EDF (Effective Degrees of Freedom): 370.36  
## -> Similar ao ENP, o EDF é usado em cálculos estatísticos e representa os graus de liberdade efetivos do modelo...  
## Sigma (Estimativa do Desvio Padrão dos Resíduos): Não disponível ou não numérico (Valor: ).  
## RSS (Residual Sum of Squares): 195.86  
## -> O RSS mede a soma das diferenças quadráticas entre os valores observados e previstos, indicando a variação não explicada pelo modelo...

## NULL