

```

# Groundwater U-238 Contamination Mapping and Contouring
# Auto-install and load required packages
required_packages <- c("gstat", "sp", "ggplot2", "viridis", "gridExtra", "fields")

for (pkg in required_packages) {
  if (!require(pkg, character.only = TRUE, quietly = TRUE)) {
    install.packages(pkg, dependencies = TRUE)
    library(pkg, character.only = TRUE)
  }
}

## Spam version 2.10-0 (2023-10-23) is loaded.
## Type 'help( Spam)' or 'demo( spam)' for a short introduction
## and overview of this package.
## Help for individual functions is also obtained by adding the
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.
##
## Attaching package: 'spam'
## The following objects are masked from 'package:base':
##
##      backsolve, forwardsolve
##
## Try help(fields) to get started.

# Set seed for reproducibility
set.seed(123)

# 1. Generate simulated sampling points (monitoring wells)
n_samples <- 30
x <- runif(n_samples, 0, 100)
y <- runif(n_samples, 0, 100)

# Simulate U-238 concentration (g/L) with contamination plumes
hotspot1_x <- 30
hotspot1_y <- 70
hotspot2_x <- 60
hotspot2_y <- 40

u238 <- 50 * exp(-((x - hotspot1_x)^2 + (y - hotspot1_y)^2) / 500) +
  30 * exp(-((x - hotspot2_x)^2 + (y - hotspot2_y)^2) / 300) +
  rnorm(n_samples, 0, 2)

u238 <- pmax(u238, 0.1) # No negative values

# Create data frame
contamination_data <- data.frame(x = x, y = y, u238 = u238)

```

```

cat("Sample Data Summary:\n")

## Sample Data Summary:

print(summary(contamination_data))

##           x           y           u238
## Min.      : 4.206   Min.      : 2.461   Min.      : 0.1000
## 1st Qu.:34.818   1st Qu.:20.900   1st Qu.: 0.8132
## Median :58.339   Median :39.409   Median : 4.1629
## Mean    :57.240   Mean     :43.864   Mean     : 9.5974
## 3rd Qu.:85.934   3rd Qu.:73.766   3rd Qu.:16.8782
## Max.    :99.427   Max.      :96.302   Max.      :41.6827

cat("\n")

```

```

# 2. Create 50x50 grid for interpolation
grid_x <- seq(0, 100, length.out = 50)
grid_y <- seq(0, 100, length.out = 50)
grid <- expand.grid(x = grid_x, y = grid_y)

# 3. Interpolation Method 1: Inverse Distance Weighting (IDW)
coordinates(contamination_data) <- ~x+y
coordinates(grid) <- ~x+y
gridded(grid) <- TRUE

cat("Performing IDW interpolation...\n")

## Performing IDW interpolation...

idw_result <- idw(u238 ~ 1, contamination_data, grid, idp = 2)

## [inverse distance weighted interpolation]

idw_df <- as.data.frame(idw_result)
names(idw_df)[3] <- "u238_idw"

# 4. Interpolation Method 2: Kriging
cat("Fitting variogram and performing kriging...\n")

## Fitting variogram and performing kriging...

v <- variogram(u238 ~ 1, contamination_data)
v_fit <- fit.variogram(v, vgm(psill = 100, model = "Sph", range = 30, nugget = 1))

## Warning in fit.variogram(v, vgm(psill = 100, model = "Sph", range
= 30, : No convergence after 200 iterations: try different initial
values?

```

```

## Warning in fit.variogram(object, model, fit.sills = fit.sills, fit.ranges
= fit.ranges, : No convergence after 200 iterations: try different
initial values?

krige_result <- krige(u238 ~ 1, contamination_data, grid, model = v_fit)

## [using ordinary kriging]

krige_df <- as.data.frame(krige_result)
names(krige_df)[3] <- "u238_krige"

# 5. Interpolation Method 3: Thin Plate Spline
cat("Performing thin plate spline interpolation...\n")

## Performing thin plate spline interpolation...

tps_fit <- Tps(x = contamination_data@coords,
               Y = contamination_data$u238,
               lambda = 0)

grid_coords <- expand.grid(x = grid_x, y = grid_y)
tps_predictions <- predict(tps_fit, grid_coords)

tps_df <- grid_coords
tps_df$u238_tps <- tps_predictions

# Convert sampling points back to regular data frame
contamination_points <- as.data.frame(contamination_data)

# 6. Define contour levels and regulatory threshold
contour_levels <- c(1, 5, 10, 15, 20, 25, 30, 40, 50)
mcl_u238 <- 30 # EPA Maximum Contaminant Level (g/L)

cat("\nGenerating maps...\n\n")

##
## Generating maps...

# Plot 1: IDW Interpolation with Contours
p1 <- ggplot() +
  geom_tile(data = idw_df, aes(x = x, y = y, fill = u238_idw)) +
  geom_contour(data = idw_df, aes(x = x, y = y, z = u238_idw),
               breaks = contour_levels, color = "white", linewidth = 0.5, alpha = 0.7) +
  geom_contour(data = idw_df, aes(x = x, y = y, z = u238_idw),
               breaks = mcl_u238, color = "red", linewidth = 1.2) +
  geom_point(data = contamination_points, aes(x = x, y = y),
             color = "black", size = 2.5, shape = 21, fill = "yellow", stroke = 1) +

```

```

scale_fill_viridis(name = "U-238\n(g/L)", option = "plasma", limits = c(0, 60)) +
labs(title = "IDW Interpolation (Inverse Distance Weighting)",
     subtitle = "Red contour = MCL threshold (30 g/L) | Yellow points = sampling wells",
     x = "X Coordinate (m)", y = "Y Coordinate (m)") +
theme_minimal(base_size = 11) +
theme(plot.title = element_text(face = "bold")) +
coord_fixed()

# Plot 2: Kriging with Contours
p2 <- ggplot() +
  geom_tile(data = krige_df, aes(x = x, y = y, fill = u238_krige)) +
  geom_contour(data = krige_df, aes(x = x, y = y, z = u238_krige),
              breaks = contour_levels, color = "white", linewidth = 0.5, alpha = 0.7) +
  geom_contour(data = krige_df, aes(x = x, y = y, z = u238_krige),
              breaks = mcl_u238, color = "red", linewidth = 1.2) +
  geom_point(data = contamination_points, aes(x = x, y = y),
            color = "black", size = 2.5, shape = 21, fill = "yellow", stroke = 1) +
  scale_fill_viridis(name = "U-238\n(g/L)", option = "plasma", limits = c(0, 60)) +
  labs(title = "Kriging Interpolation (Geostatistical)",
       subtitle = "Red contour = MCL threshold (30 g/L) | Yellow points = sampling wells",
       x = "X Coordinate (m)", y = "Y Coordinate (m)") +
  theme_minimal(base_size = 11) +
  theme(plot.title = element_text(face = "bold")) +
  coord_fixed()

# Plot 3: Thin Plate Spline with Contours
p3 <- ggplot() +
  geom_tile(data = tps_df, aes(x = x, y = y, fill = u238_tps)) +
  geom_contour(data = tps_df, aes(x = x, y = y, z = u238_tps),
              breaks = contour_levels, color = "white", linewidth = 0.5, alpha = 0.7) +
  geom_contour(data = tps_df, aes(x = x, y = y, z = u238_tps),
              breaks = mcl_u238, color = "red", linewidth = 1.2) +
  geom_point(data = contamination_points, aes(x = x, y = y),
            color = "black", size = 2.5, shape = 21, fill = "yellow", stroke = 1) +
  scale_fill_viridis(name = "U-238\n(g/L)", option = "plasma", limits = c(0, 60)) +
  labs(title = "Thin Plate Spline Interpolation",
       subtitle = "Red contour = MCL threshold (30 g/L) | Yellow points = sampling wells",
       x = "X Coordinate (m)", y = "Y Coordinate (m)") +
  theme_minimal(base_size = 11) +
  theme(plot.title = element_text(face = "bold")) +
  coord_fixed()

# Plot 4: Comparison - Contours Only (All Methods Overlaid)
p4 <- ggplot() +
  geom_contour(data = idw_df, aes(x = x, y = y, z = u238_idw, color = "IDW"),

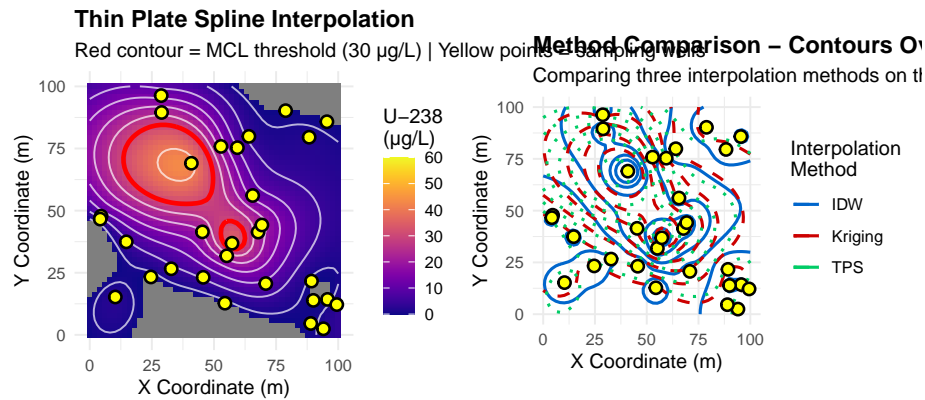
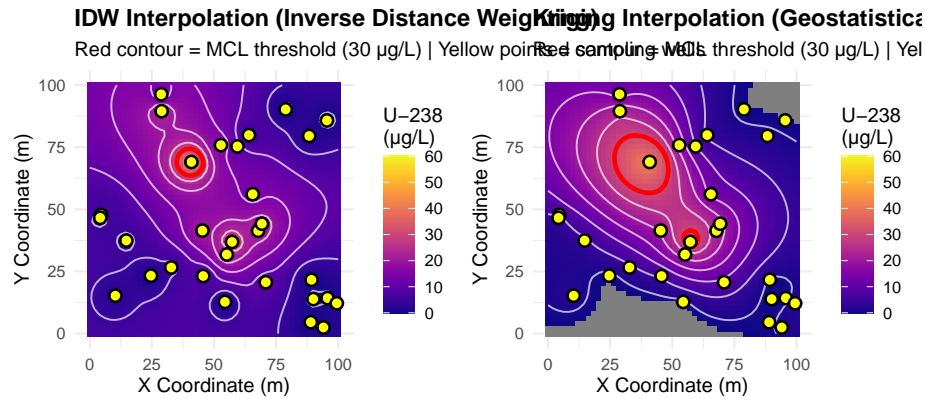
```

```

        breaks = contour_levels, linewidth = 0.8) +
geom_contour(data = krige_df, aes(x = x, y = y, z = u238_krige, color = "Kriging"),
             breaks = contour_levels, linewidth = 0.8, linetype = "dashed") +
geom_contour(data = tps_df, aes(x = x, y = y, z = u238_tps, color = "TPS"),
             breaks = contour_levels, linewidth = 0.8, linetype = "dotted") +
geom_point(data = contamination_points, aes(x = x, y = y),
           color = "black", size = 2.5, shape = 21, fill = "yellow", stroke = 1) +
scale_color_manual(name = "Interpolation\\nMethod",
                  values = c("IDW" = "#0066CC", "Kriging" = "#CC0000", "TPS" = "#00CC66"))
labs(title = "Method Comparison - Contours Overlay",
     subtitle = "Comparing three interpolation methods on the same domain",
     x = "X Coordinate (m)", y = "Y Coordinate (m)") +
theme_minimal(base_size = 11) +
theme(plot.title = element_text(face = "bold"),
      legend.position = "right") +
coord_fixed()

# Display all plots in a 2x2 grid
grid.arrange(p1, p2, p3, p4, ncol = 2)

```



```
# 7. Statistical comparison
cat("=== INTERPOLATION METHOD COMPARISON ===\n\n")

## === INTERPOLATION METHOD COMPARISON ===

cat("IDW (Inverse Distance Weighting):\n")

## IDW (Inverse Distance Weighting):

cat(sprintf(" Minimum: %.2f g/L\n", min(idw_df$u238_idw, na.rm = TRUE)))

## Minimum: 0.12 g/L

cat(sprintf(" Maximum: %.2f g/L\n", max(idw_df$u238_idw, na.rm = TRUE)))

## Maximum: 41.61 g/L

cat(sprintf(" Mean: %.2f g/L\n", mean(idw_df$u238_idw, na.rm = TRUE)))
```

```

## Mean: 10.64 g/L
cat(sprintf(" Std Dev: %.2f g/L\n\n", sd(idw_df$u238_idw, na.rm = TRUE)))
## Std Dev: 6.51 g/L
cat("Kriging (Geostatistical):\n")
## Kriging (Geostatistical):
cat(sprintf(" Minimum: %.2f g/L\n", min(krige_df$u238_krige, na.rm = TRUE)))
## Minimum: -2.54 g/L
cat(sprintf(" Maximum: %.2f g/L\n", max(krige_df$u238_krige, na.rm = TRUE)))
## Maximum: 41.25 g/L
cat(sprintf(" Mean: %.2f g/L\n", mean(krige_df$u238_krige, na.rm = TRUE)))
## Mean: 10.42 g/L
cat(sprintf(" Std Dev: %.2f g/L\n\n", sd(krige_df$u238_krige, na.rm = TRUE)))
## Std Dev: 9.92 g/L
cat("Thin Plate Spline:\n")
## Thin Plate Spline:
cat(sprintf(" Minimum: %.2f g/L\n", min(tps_df$u238_tps, na.rm = TRUE)))
## Minimum: -4.67 g/L
cat(sprintf(" Maximum: %.2f g/L\n", max(tps_df$u238_tps, na.rm = TRUE)))
## Maximum: 43.09 g/L
cat(sprintf(" Mean: %.2f g/L\n", mean(tps_df$u238_tps, na.rm = TRUE)))
## Mean: 10.82 g/L
cat(sprintf(" Std Dev: %.2f g/L\n\n", sd(tps_df$u238_tps, na.rm = TRUE)))
## Std Dev: 11.98 g/L

# Calculate area exceeding MCL
idw_exceed <- sum(idw_df$u238_idw > mcl_u238, na.rm = TRUE) / nrow(idw_df) * 100
krige_exceed <- sum(krige_df$u238_krige > mcl_u238, na.rm = TRUE) / nrow(krige_df) * 100
tps_exceed <- sum(tps_df$u238_tps > mcl_u238, na.rm = TRUE) / nrow(tps_df) * 100

cat("Area Exceeding MCL (30 g/L):\n")

```

```

## Area Exceeding MCL (30 g/L):

cat(sprintf("  IDW:      %.1f%% of study area\n", idw_exceed))

##  IDW:      1.0% of study area

cat(sprintf("  Kriging: %.1f%% of study area\n", krige_exceed))

##  Kriging: 4.0% of study area

cat(sprintf("  TPS:      %.1f%% of study area\n\n", tps_exceed))

##  TPS:      9.7% of study area

cat("Script completed successfully!\n")

## Script completed successfully!

cat("Note: To save plots, uncomment the ggsave() line at the end of the script.\n")

## Note: To save plots, uncomment the ggsave() line at the end of the script.

```