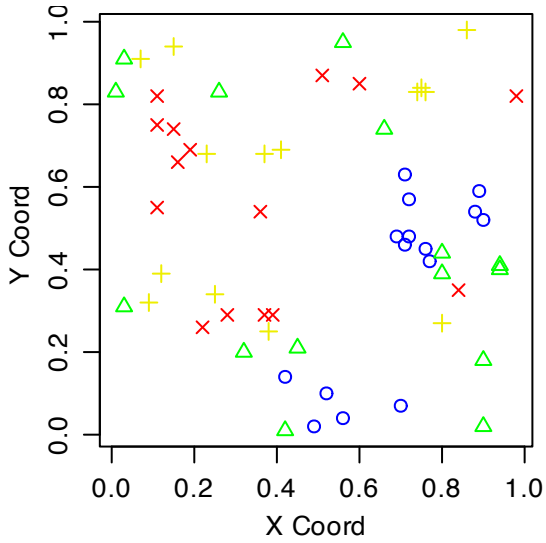
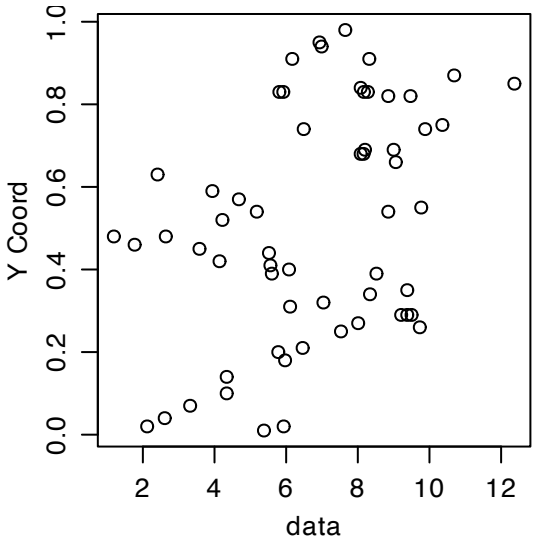
STA 5377, Homework 3

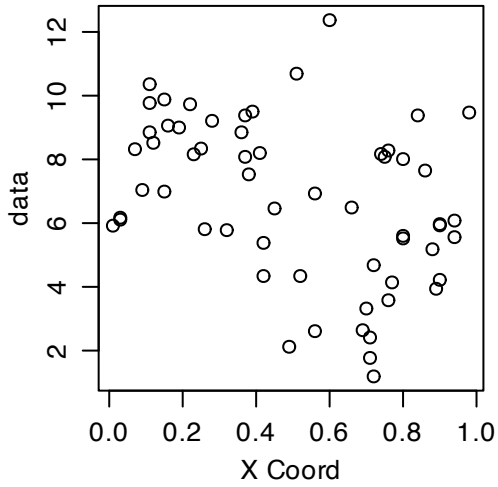
Carson Slater *Baylor University*

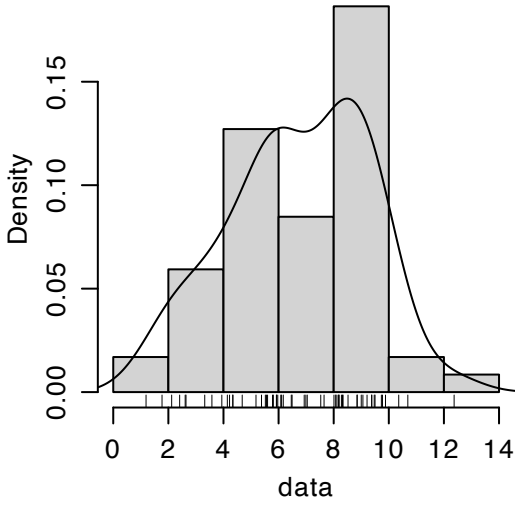
Consider the `wells` data (`wells.txt`). `x` and `y` are spatial coordinates, and `z` is the variable of interest. Our goal in this problem is to use ordinary kriging and universal kriging for the wells data, generating a grid of prediction values and comparing OK and UK using cross-validation.

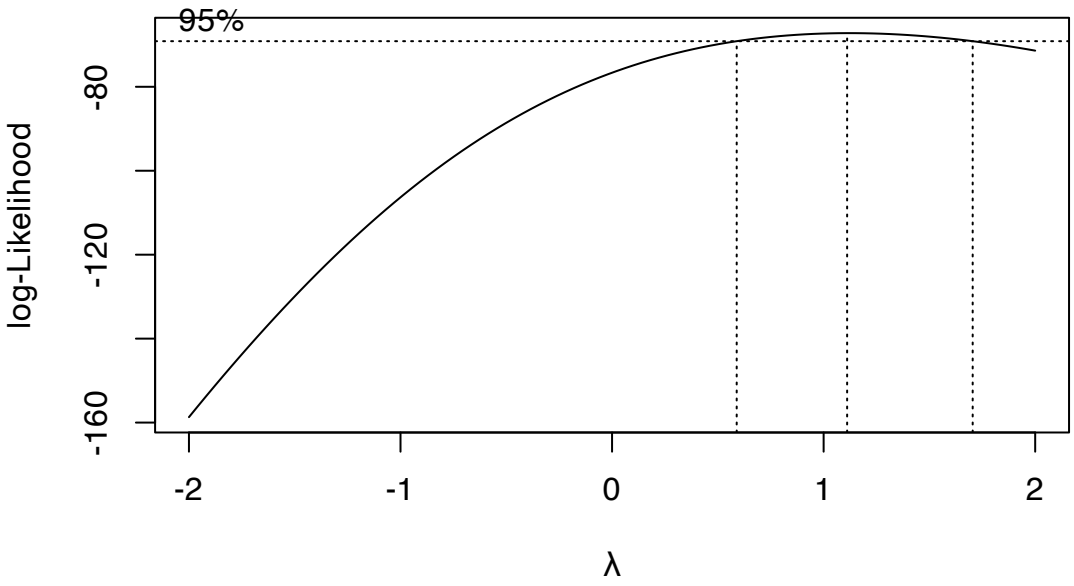
(1) Perform exploratory spatial data analysis.









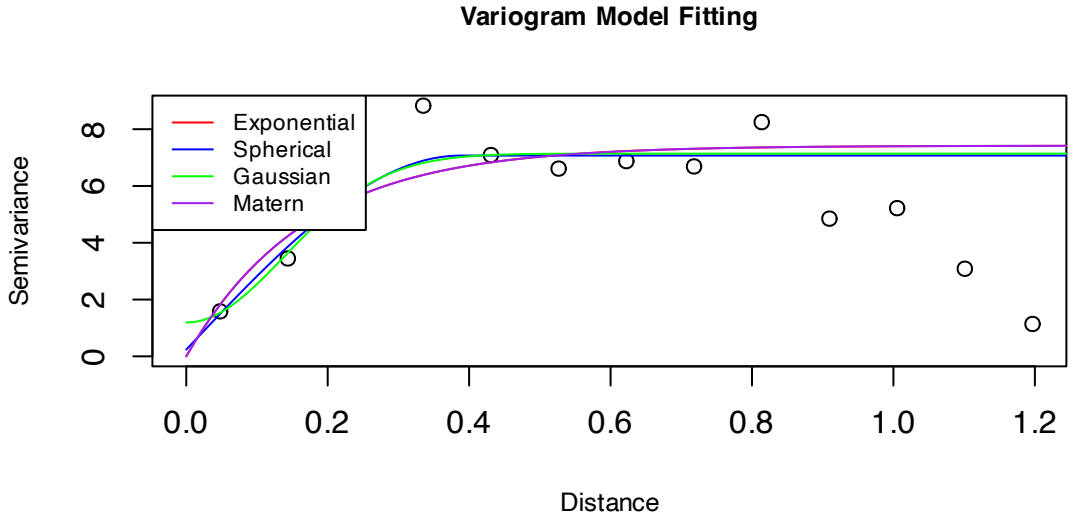


This data seems to have clusters of the ranges in the top left and bottom right of the data plot (see red "x" points and blue "o" points). It seems that there is no need of a transformation for normality.

(2) Can you find any trend based on the spatial coordinates?

It can be argued that these data are negatively correlated with the x coordinate and then also positively correlated with the y coordinate.

(3) Perform variogram analysis. Find the best variogram fit.



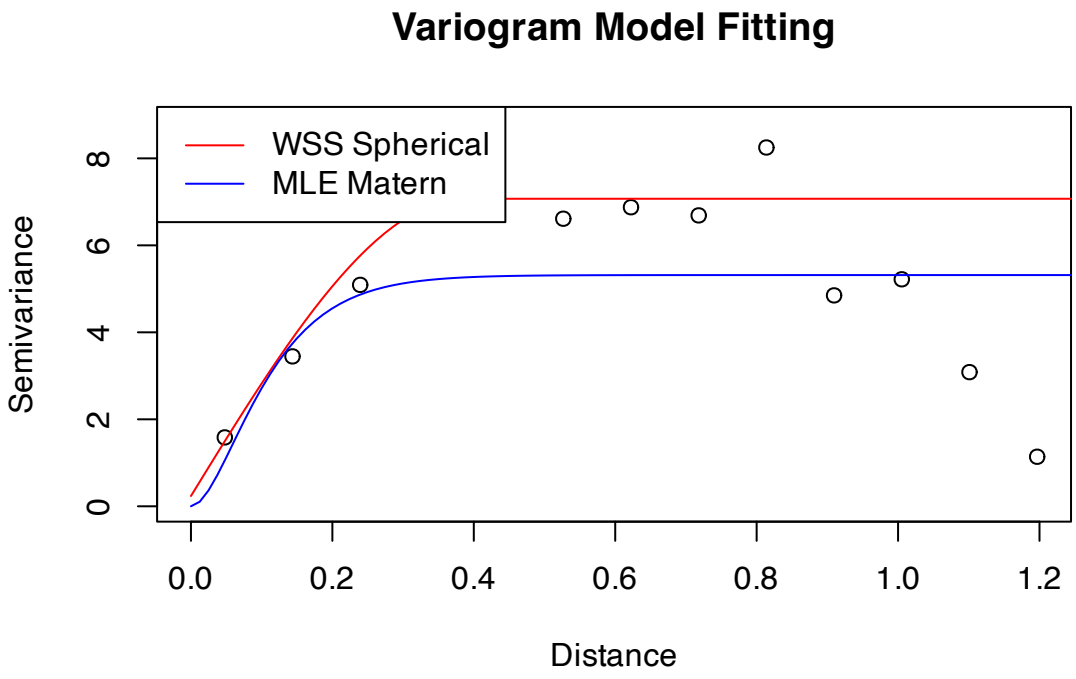
	Exponential	Spherical	Gaussian	Matern
WSS	56.68289	40.01399	40.87849	56.68289

Here it appears the Spherical covariance structure yielded the lowest weighted sum of squares, giving evidence that it is the most viable model. Now we examine MLE variograms and compare.

Table 2: Performance of each covariance structure using the MLE for estimation.

Model	LogLikelihood	AIC
Exponential	-111.2289	230.4578
Spherical	-110.2738	228.5476
Gaussian	-110.7989	229.5978
Matern	-108.5703	225.1406

From the table, the best model was the Matern model, with the highest log-likelihood and the lowest AIC.



It appears the MLE semivariogram was actually better than the WLS Spherical semivariogram. This is debatable, as the MLE could be underestimating the partial sill, but the empirical semivariogram also tapers down substantially.

(4) Construct image plots of the spatial predictions using OK and UK on the following grid.

```
locations <- expand.grid(0:20/20, 0:20/20)
```

Comment on your results.

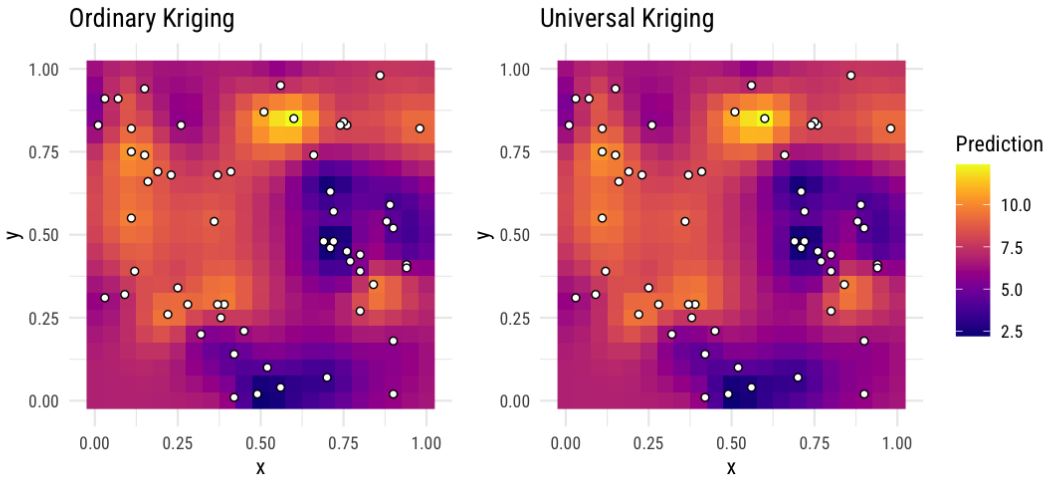


Figure 1: Kriging image plots with both ordinary and universal kriging.

For UK, we construct the following trend:

```
trend_formula <- data ~ coords[,1] + coords[,2]
```

As stated in (1) there appears to be regions with higher data values in the top left of the image plots, per the kriging predictions. This is consistent with the data being negatively correlated with the x coordinate and positively wiht the y. It is difficult to definitively say though.

(5) Construct image plots of the prediction standard errors of OK and UK. Comment again on your results.

See Figure 2.

As expected, the image plot of the standard errors demonstrates high confidence in the values around observations, and higher standard errors in areas where there are less observations.

(6) Perform leave-one-out cross-validation with OK and UK and compare the prediction performance of the methods.

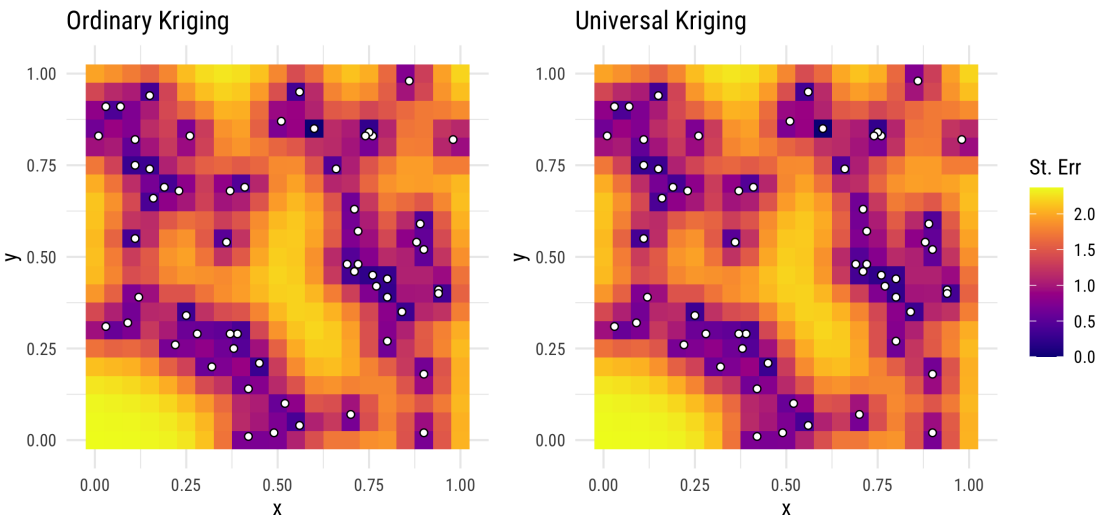


Figure 2: An image plot of standard errors for universal and ordinary kriging.

Table 3: Cross validation metrics for ordinary and universal kriging.

	MSE
OK	2.399810
UK	2.377271

Here, we have that univeral kriging is the better model, assuming we specified the true covariance structure.

(7) Construct image plots of the prediction standard errors of Bayesian OK and UK. Comment again on your results.

See Figures 3 and 4.

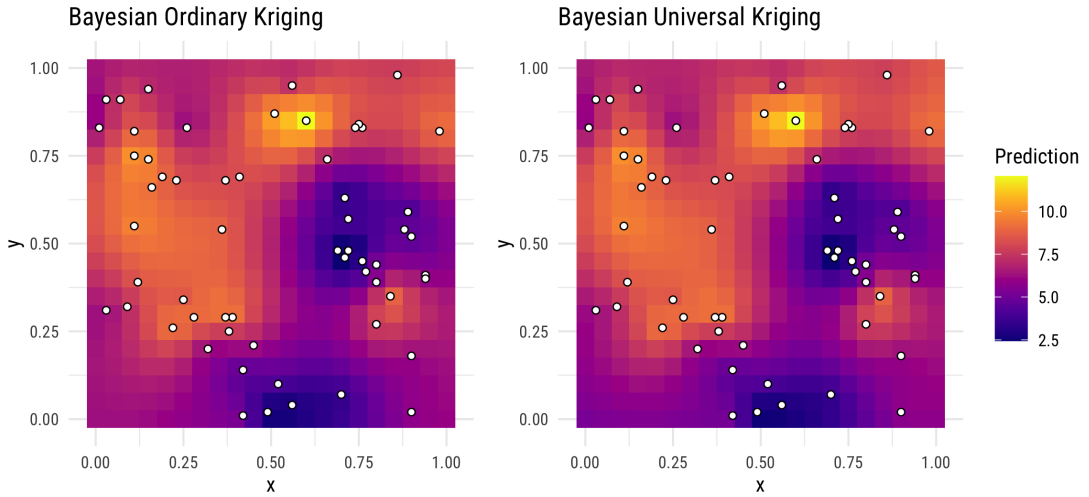


Figure 3: Predictions from Bayesian kriging on the wells dataset.

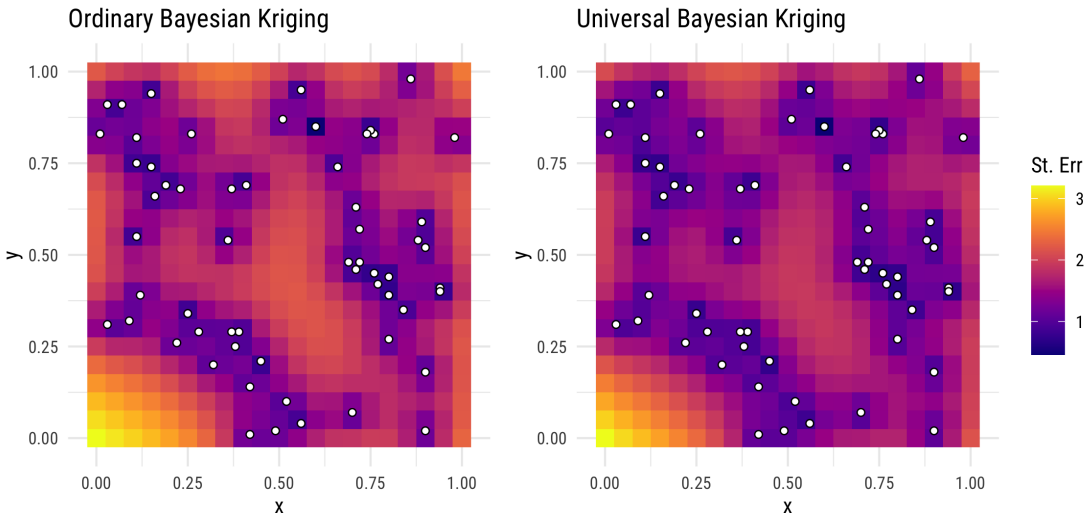


Figure 4: Standard error estimates for Bayesian kriging on the wells dataset.

For the most part, the most noticeable difference between the frequentist kriging and the Bayesian kriging is the difference in the standard errors over the predictions. For the Bayesian kriging, the standard errors are noticably different, intuitively from the integration of prior beliefs into the model.

(8) Perform leave-one-out cross-validation with Bayesian OK and UK and compare the prediction performance of the methods.

	MSE
OK	2.231493
UK	2.218462

Here, we also have that univeral kriging is the better model, the MSE's are better for both though after fitting with the Bayesian

Code Appendix

Below is all of the code used for this assignment.

```
# Setup

knitr::opts_chunk$set(dev = "cairo_pdf",
                      fig.width = 6,
                      fig.height = 4,
                      fig.align = 'center',
                      echo = FALSE,
```

```
                      message = FALSE,
                      warning = TRUE,
                      error = FALSE)
library("tidyverse"); library("patchwork"); library("glue")
library("scales", warn.conflicts = FALSE); library("extrafont")
library("tinytex"); library("rjags"); library("coda")
library("bayesplot"); library("patchwork"); library("geoR")
library("gridExtra"); library("tidyr"); library("latex2exp")
library("gstat"); library("furrr"); library("knitr")
theme_set(theme_minimal(base_family = "Roboto Condensed"))

conflicted::conflicts_prefer(
  readr::col_factor(),
  purrr::discard(),
  rstan::extract(),
  dplyr::lag(),
  rstan::traceplot(),
  viridis::viridis_pal(),
  readr::parse_date
)
# Read the data
wells <- read.table("wells.txt", header = TRUE) |>
  as.geodata(coords.col = 2:3, data.col = 4)

wells_tb <- read.table("wells.txt", header = TRUE) |>
  as_tibble()
# 1
plot.geodata(wells)
MASS::boxcox(wells$data ~ 1)
# 3
vario <- variog(wells, messages = FALSE)
# Fit variogram models
v_exp <- variofit(vario, cov.model = "exponential",
                  weights = "cressie", messages = FALSE)
v_sph <- variofit(vario, cov.model = "spherical",
                  weights = "cressie", messages = FALSE)
v_gau <- variofit(vario, cov.model = "gaussian",
                  weights = "cressie", messages = FALSE)
v_mat <- variofit(vario, cov.model = "matern",
                  weights = "cressie", kappa = 0.5, messages = FALSE)  # Fixi

plot(vario, main = "Variogram Model Fitting",
     xlab = "Distance", ylab = "Semivariance",
     cex.main = 0.8, cex.lab = 0.8)
lines(v_exp, col = "red")
lines(v_sph, col = "blue")
lines(v_gau, col = "green")
```

```r
lines(v_mat, col = "purple")
legend("topleft", legend = c("Exponential", "Spherical",
                             "Gaussian", "Matern"),
       col = c("red", "blue", "green", "purple"), lty = 1, cex = 0.7)

tibble(
  "WSS" = list(v_exp, v_sph, v_gau, v_mat) |> map_vec(\(x) pluck(x, "value"))
) |> t() |>
  `colnames<-`(c("Exponential", "Spherical", "Gaussian", "Matern")) |>
  knitr::kable()
initial_sill <- var(wells$data)

initial_range <- max(vario$u) / 3  # A common heuristic

ini_cov_pars <- c(initial_sill, initial_range)

# Fit models using MLE
mle_exp <- likfit(wells, cov.model = "exponential",
                  ini.cov.pars = ini_cov_pars,
                  messages = FALSE)
mle_sph <- likfit(wells, cov.model = "spherical",
                  ini.cov.pars = ini_cov_pars,
                  messages = FALSE)
mle_gau <- likfit(wells, cov.model = "gaussian",
                  ini.cov.pars = ini_cov_pars,
                  messages = FALSE)
mle_mat <- likfit(wells, cov.model = "matern",
                  ini.cov.pars = ini_cov_pars,
                  kappa = 1.5, messages = FALSE)  # kappa required for Matérn

# Extract Log-Likelihood and AIC values
model_comparison <- data.frame(
  Model = c("Exponential", "Spherical",
            "Gaussian", "Matern"),
  LogLikelihood = c(mle_exp$loglik, mle_sph$loglik,
                    mle_gau$loglik, mle_mat$loglik),
  AIC = c(AIC(mle_exp), AIC(mle_sph), AIC(mle_gau), AIC(mle_mat))
)

model_comparison |> knitr::kable()
plot(vario, main = "Variogram Model Fitting",
     xlab = "Distance", ylab = "Semivariance")
lines(v_sph, col = "red")
lines(mle_mat, col = "blue")
legend("topleft", legend = c("WSS Spherical", "MLE Matern"),
       col = c("red", "blue"), lty = 1)
```

```r
locations <- expand.grid(0:20/20, 0:20/20)
load("HW3_Spatial.RData")
# 4 - To remove messaging I don't run this code but load results from
#     previously running it.
wells_krige_cont <- krige.control(
    cov.model = "matern",
    obj.model = mle_mat,
    nugget = 0
  )
wells_ok <- krige.conv(wells, loc = locations, krige = wells_krige_cont)

locations <- locations |> as.data.frame()

locations$predict_ok <- wells_ok$predict

p1 <- locations |>
  ggplot(aes(x = Var1, y = Var2, fill = predict_ok)) +
  geom_raster() +
  geom_point(data = wells_tb, aes(x = x, y = y), shape = 21,
             color = "black", fill = "white", inherit.aes = FALSE) +
  scale_fill_viridis(option = "C") +
  labs(title = "Ordinary Kriging",
       x = "x",
       y = "y",
       fill = "Prediction") +
  coord_equal() +
  theme(legend.position = "none")
trend_formula <- data ~ coords[,1] + coords[,2]

# Define kriging control with the trend
wells_uk_krige_cont <- krige.control(
  trend.d = trend_formula,  # Define universal kriging trend
  cov.model = "matern",
  obj.model = mle_mat,
  nugget = 0
)

# Convert locations to matrix for trend specification
coords <- as.matrix(locations)

# Perform Universal Kriging
wells_uk <- krige.conv(wells, loc = coords, krige = wells_uk_krige_cont)

# Store predictions
locations$predict_uk <- wells_uk$predict

p2 <- locations |>
```

```
  ggplot(aes(x = Var1, y = Var2, fill = predict_uk)) +
  geom_raster() +
  geom_point(data = wells_tb, aes(x = x, y = y), shape = 21,
             color = "black", fill = "white", inherit.aes = FALSE) +
  scale_fill_viridis(option = "C") +
  coord_equal() +
  labs(title = "Universal Kriging",
       x = "x",
       y = "y",
       fill = "Prediction")
p1 + p2
trend_formula <- data ~ coords[,1] + coords[,2]
locations$ok_se <- wells_ok$krige.var |> sqrt()
locations$uk_se <- wells_uk$krige.var |> sqrt()

p3 <- locations |>
  ggplot(aes(x = Var1, y = Var2, fill = ok_se)) +
  geom_raster() +
  geom_point(data = wells_tb, aes(x = x, y = y), shape = 21,
             color = "black", fill = "white", inherit.aes = FALSE) +
  scale_fill_viridis(option = "C") +
  labs(title = "Ordinary Kriging",
       x = "x",
       y = "y",
       fill = "Prediction") +
  coord_equal() +
  theme(legend.position = "none")

p4 <- locations |>
  ggplot(aes(x = Var1, y = Var2, fill = uk_se)) +
  geom_raster() +
  geom_point(data = wells_tb, aes(x = x, y = y), shape = 21,
             color = "black", fill = "white", inherit.aes = FALSE) +
  scale_fill_viridis(option = "C") +
  coord_equal() +
  labs(title = "Universal Kriging",
       x = "x",
       y = "y",
       fill = "St. Err")

p3 + p4
library("gstat")
library("sp")

# Convert wells data to a spatial data frame
coordinates(wells_tab) <- ~ x + y
```

```r
ok_cv <- krige.cv(z ~ 1, wells_tab,
                  model = vgm(psill = 5.3164,
                              model = "Mat",
                              range = 0.0584,
                              nugget = 0))

uk_cv <- krige.cv(z ~ x + y, wells_tab,
                  model = vgm(psill = 5.3164,
                              model = "Mat",
                              range = 0.0584,
                              nugget = 0))

# Compute RMSE for performance comparison
mse <- \(residuals) mean(residuals2)

ok_cv <- read_rds("ok_cv.rds")
uk_cv <- read_rds("uk_cv.rds")

data.frame(
  "MSE" = c(mse(ok_cv$residual), mse(uk_cv$residual))
) |>
  `row.names<-`(c("OK", "UK")) |>
  knitr::kable()
bayes_mod_ok <- model.control(
  trend.d = "cte",
  cov.model = "matern",
  kappa = 0.5
)

bayes_prior <- prior.control(
  beta.prior = "flat",
  sigmasq.prior = "reciprocal",
  tausq.rel.prior = "uniform",
  tausq.rel.discrete = seq(from = 0, to = 1, by = 0.01)
)

wells_bayes_ok <- krige.bayes(
  wells,
  locations = locations[,1:2],
  model = bayes_mod_ok,
  prior = bayes_prior
)

bayes_mod_uk <- model.control(
  trend.d = trend_formula,
  trend.l = trend_formula,
  cov.model = "matern",
```

```
  kappa = 0.5
)

bayes_prior <- prior.control(
  beta.prior = "flat",
  sigmasq.prior = "reciprocal",
  tausq.rel.prior = "uniform",
  tausq.rel.discrete = seq(from = 0, to = 1, by = 0.01)
)

wells_bayes_uk <- krige.bayes(
  wells,
  locations = locations[,1:2],
  model = bayes_mod_uk,
  prior = bayes_prior
)
wells_bayes_ok <- read_rds("wells_bayes_ok.rds")
wells_bayes_uk <- read_rds("wells_bayes_uk.rds")

locations$predict_ok_bayes <- wells_bayes_ok$predictive$mean
locations$predict_uk_bayes <- wells_bayes_uk$predictive$mean

p5 <- locations |>
  ggplot(aes(x = Var1, y = Var2, fill = predict_ok_bayes)) +
  geom_raster() +
  geom_point(data = wells_tb, aes(x = x, y = y), shape = 21,
             color = "black", fill = "white", inherit.aes = FALSE) +
  scale_fill_viridis(option = "C") +
  labs(title = "Bayesian Ordinary Kriging",
       x = "x",
       y = "y",
       fill = "Prediction") +
  coord_equal() +
  theme(legend.position = "none")

p6 <- locations |>
  ggplot(aes(x = Var1, y = Var2, fill = predict_uk_bayes)) +
  geom_raster() +
  geom_point(data = wells_tb, aes(x = x, y = y), shape = 21,
             color = "black", fill = "white", inherit.aes = FALSE) +
  scale_fill_viridis(option = "C") +
  labs(title = "Bayesian Universal Kriging",
       x = "x",
       y = "y",
       fill = "Prediction") +
  coord_equal()
```

```
p5 + p6
locations$ok_se_bayes <- wells_bayes_ok$predictive$variance |> sqrt()
locations$uk_se_bayes <- wells_bayes_uk$predictive$variance |> sqrt()

p7 <- locations |>
  ggplot(aes(x = Var1, y = Var2, fill = ok_se_bayes)) +
  geom_raster() +
  geom_point(data = wells_tb, aes(x = x, y = y), shape = 21,
             color = "black", fill = "white", inherit.aes = FALSE) +
  scale_fill_viridis(option = "C") +
  labs(title = "Ordinary Bayesian Kriging",
       x = "x",
       y = "y",
       fill = "Prediction") +
  coord_equal() +
  theme(legend.position = "none")

p8 <- locations |>
  ggplot(aes(x = Var1, y = Var2, fill = uk_se_bayes)) +
  geom_raster() +
  geom_point(data = wells_tb, aes(x = x, y = y), shape = 21,
             color = "black", fill = "white", inherit.aes = FALSE) +
  scale_fill_viridis(option = "C") +
  coord_equal() +
  labs(title = "Universal Bayesian Kriging",
       x = "x",
       y = "y",
       fill = "St. Err")

p7 + p8
wells_bayes_uk$posterior$phi$summary[[1]]
wells_bayes_uk$posterior$sigmasq$summary[[1]]
wells_bayes_uk$posterior$tausq.rel$summary[[1]]

ok_cv_bayes <- krige.cv(z ~ 1, wells_tab,
                  model = vgm(psill = wells_bayes_ok$posterior$sigmasq$summar
                              model = "Mat",
                              range = wells_bayes_ok$posterior$sigmasq$summary
                              nugget = wells_bayes_ok$posterior$tausq.rel$summ

uk_cv_bayes <- krige.cv(z ~ x + y, wells_tab,
                  model = vgm(psill = wells_bayes_uk$posterior$sigmasq$summar
                              model = "Mat",
                              range = wells_bayes_uk$posterior$phi$summary[[1]
                              nugget = wells_bayes_uk$posterior$tausq.rel$summ
ok_cv_bayes <- read_rds("ok_cv_bayes.rds")
uk_cv_bayes <- read_rds("uk_cv_bayes.rds")
```

```r
# Compute RMSE for performance comparison
mse <- \(residuals) mean(residuals2)

mse_ok_bayes <- mse(ok_cv_bayes$residual)
mse_uk_bayes <- mse(uk_cv_bayes$residual)

data.frame(
  "MSE" = c(mse_ok_bayes, mse_uk_bayes)
) |>
  `row.names<-`(c("OK", "UK")) |>
  knitr::kable()
```