Data analysis of soil erosion data

Xiaodan Lyu, Emily Berg and Heike Hofmann

2020-04-03

```
rm(list = ls(all = T))
library(dplyr)
library(ggplot2)
## load model package
if(!require(saezero)) devtools::install_github("XiaodanLyu/saezero")
library(saezero)
## load pseudo data due to confidentiality
data(erosion)
glimpse(erosion)
```

```
## Observations: 646
## Variables: 10
## $ ctylab <chr> "Aurora", "Aurora", "Aurora", "Aurora", "Aurora", "Aurora", ...
## $ cty    <chr> "003", "003", "003", "003", "003", "003", "003", "003", "005...
## $ mukey  <dbl> 354836, 354837, 354841, 354844, 354847, 354851, 354866, 3548...
## $ crop   <chr> "others", "soybean", "others", "others", "others", "others",...
## $ logR   <dbl> 4.553877, 4.553877, 4.553877, 4.553877, 4.553877, 4.553877, ...
## $ logK   <dbl> -1.4271164, -1.4271164, -1.4271164, -0.9942523, -1.1394343, ...
## $ logS   <dbl> -0.3549698, -2.1465136, -1.0483327, -2.1465136, -1.0483327, ...
## $ crop2  <dbl> 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, ...
## $ crop3  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, ...
## $ RUSLE2 <dbl> 0.091, 0.009, 0.007, 0.034, 0.087, 0.000, 0.006, 0.001, 0.00...
```

Table 4

```
## data pre-processing
erosion_2p <- as.2pdata(f_pos = RUSLE2~logR+logK+logS,
                        f_zero = ~logR+logS+crop2+crop3,
                        f_area = ~cty, data = erosion)
## Maximum Likelihood Estimates
fit <- mleLBH(erosion_2p)
## fixed effect coefficients and standard errors
est <- c(beta = fit$fixed$p1[-1], alpha = fit$fixed$p0[-1])
se <- c(beta = sqrt(diag(fit$vcov$p1)[-1]),
        alpha = sqrt(diag(fit$vcov$p0)[-1]))
## positive part
paste0(round(est[1:3], 2), " (", round(se[1:3], 2), ")")
```

```
## [1] "1.82 (0.34)" "0.37 (0.17)" "0.52 (0.07)"
```

```r
## binary part
paste0(round(est[4:7], 2), " (", round(se[4:7], 2), ")")
```

```
## [1] "4.76 (0.62)" "0.17 (0.14)" "0.59 (0.31)" "1.4 (0.37)"
```

```r
## variance components
c(sig2lu = fit$refvar1, sig2b = fit$refvar0,
  sig2le = fit$errorvar, rho = fit$refcor) %>%
  print(digit = 2)
```

```
## sig2lu  sig2b sig2le    rho
##   0.22   0.44   1.20   0.77
```

Figure 2

```r
predictions <- ebLBH(Xaux, f_q = ~cnt, erosion_2p, fit, fullpop = TRUE)
predictions <- predictions %>% mutate(cv = sqrt(mse)/eb)
predictions %>% glimpse()
```

```
## Observations: 64
## Variables: 4
## $ area <fct> 003, 005, 007, 009, 011, 013, 015, 017, 021, 023, 025, 027, 02...
## $ eb   <dbl> 0.017535227, 0.022068328, 0.002422145, 0.020669755, 0.02734260...
## $ mse  <dbl> 3.541407e-05, 2.785866e-05, 2.528140e-06, 4.285971e-05, 5.3237...
## $ cv   <dbl> 0.3393724, 0.2391722, 0.6564480, 0.3167301, 0.2668512, 0.21409...
```

```r
## parametric bootstrap estimates of M2
set.seed(2020)
b <- 0
eb_boot.store <- mmse_boot.store <- c()
repeat{
  b <- b + 1
  ys <- simLBH(fit, erosion, f_pos = ~logR+logK+logS,
             f_zero = ~logR+logS+crop2+crop3, f_area = ~cty)
  erosion_boot <- erosion %>% mutate(RUSLE2 = ys)
  erosion_boot_2p <- as.2pdata(f_pos = RUSLE2~logR+logK+logS,
                               f_zero = ~logR+logS+crop2+crop3,
                               f_area = ~cty, data = erosion_boot)
  fit_boot <- mleLBH(erosion_boot_2p)
  eb_boot <- ebLBH(Xaux, f_q = ~cnt, erosion_boot_2p, fit_boot, fullpop = TRUE)$eb
  mmse_boot <- ebLBH(Xaux, f_q = ~cnt, erosion_boot_2p, fit, fullpop = TRUE)$eb
  eb_boot.store <- rbind(eb_boot.store, eb_boot)
  mmse_boot.store <- rbind(mmse_boot.store, mmse_boot)
  if(b>=100) break
}
m2_boot <- colMeans((eb_boot.store - mmse_boot.store)^2)
```

```r
## pooled standard error of direct estimator
direct <- erosion %>% group_by(cty) %>%
  summarise(nis = n(),
            pool = sqrt(var(erosion$RUSLE2)/nis))
se_all <- direct %>% mutate(
  ID = 1:n(),
  group = cut(nis, c(0, 5, 10, 20, 30)),
  semiboot = sqrt(predictions$mse+m2_boot),
  onestep = sqrt(predictions$mse))
se_long <- se_all %>%
  tidyr::gather(method, SE, -cty, -nis, -ID, -group) %>%
  mutate(method = forcats::fct_recode(
    method, "Direct" = "pool",
    "EB (Semi-Boot)" = "semiboot",
    "EB (One-step)" = "onestep") %>%
      forcats::fct_relevel("Direct", "EB (One-step)"))
ggplot(se_long, aes(x = ID, y = SE)) +
  geom_line(aes(group = method, color = method, linetype = method)) +
  scale_linetype_manual(values = c(1,1,0)) +
  geom_point(aes(shape = method, color = method),
             size = rel(2), stroke = rel(1)) +
  facet_grid(.~group, scales = "free_x") +
  scale_color_manual(values = c("darkgrey", "black", "black")) +
  scale_shape_manual(values = c(8, 1, 4)) +
  labs(x = "County", y = "Standard Error",
       color = "", shape = "", linetype = "") +
  # theme_bw(base_size = 18) +
  theme(legend.position = "bottom",
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        legend.background = element_rect(colour = "black", size = 0.5))
```

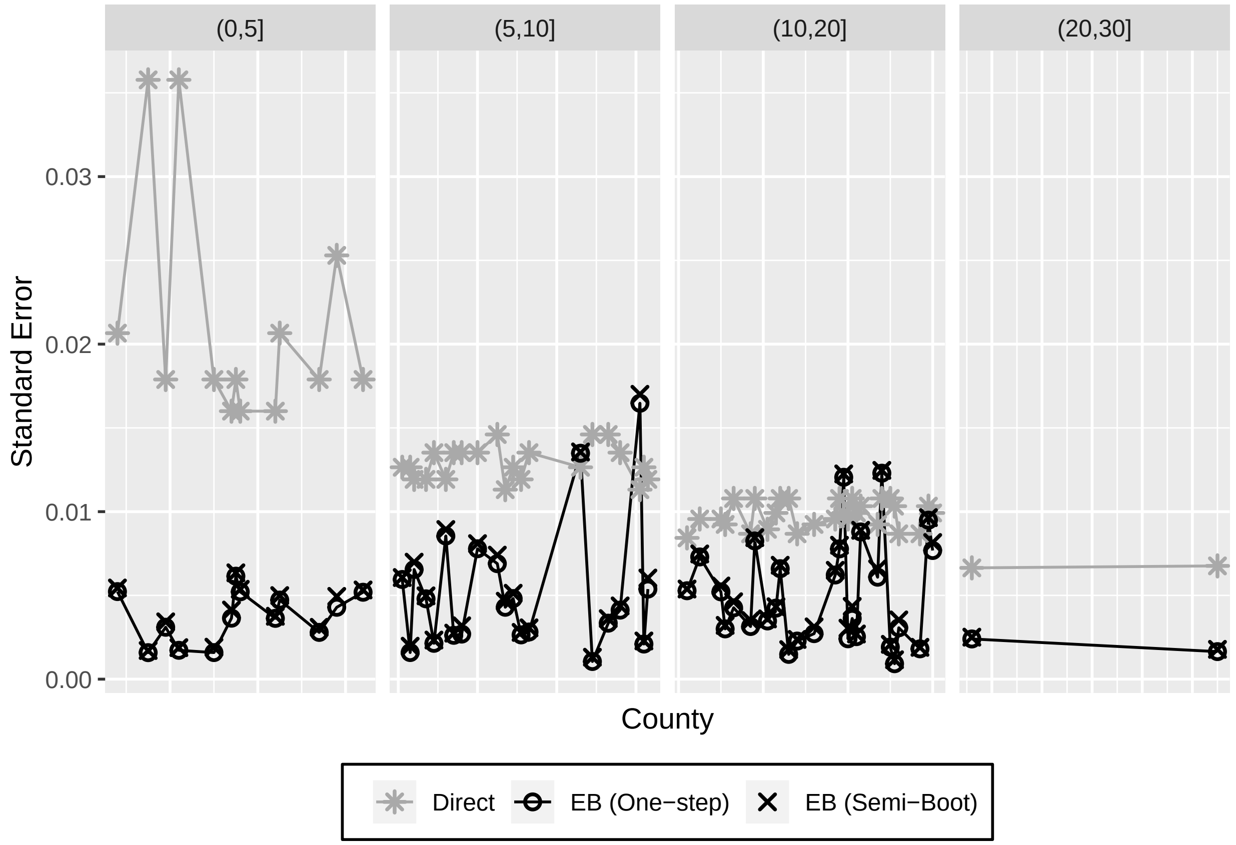


Figure 3

```
if(!require(ggmapr)) devtools::install_github("heike/ggmapr")
## EB predictions and one-step MSE estimates
map.eb <- map_sd %>% left_join(predictions, by = c("COUNTY" = "area"))
```

```
## Warning: Column `COUNTY`/`area` joining character vector and factor, coercing
## into character vector
```

```
b <- (0.2-1)/diff(range(map.eb$cv, na.rm = T))
map.eb.df <- map.eb %>% mutate(s = b*(cv-min(cv, na.rm = T))+1)
scale_cty <- function(cty) {
  map.eb.df %>% filter(COUNTY == cty) %>% ggmapr::scale(scale = unique(.$s))
}
map.eb.scale <- do.call("rbind", lapply(unique(map.eb.df$COUNTY), scale_cty))
map.eb.scale %>%
  ggplot(aes(x = long, y = lat, group = factor(group))) +
  geom_polygon(aes(fill = eb), colour = NA) +
  geom_path(data = map_sd) +
  scale_fill_distiller("", palette = "OrRd", direction = 1) +
  scale_alpha(trans = "reciprocal", range = c(0.3, 1)) +
  ggthemes::theme_map(base_size = 18) +
  theme(legend.position = "right") +
  coord_equal()
```

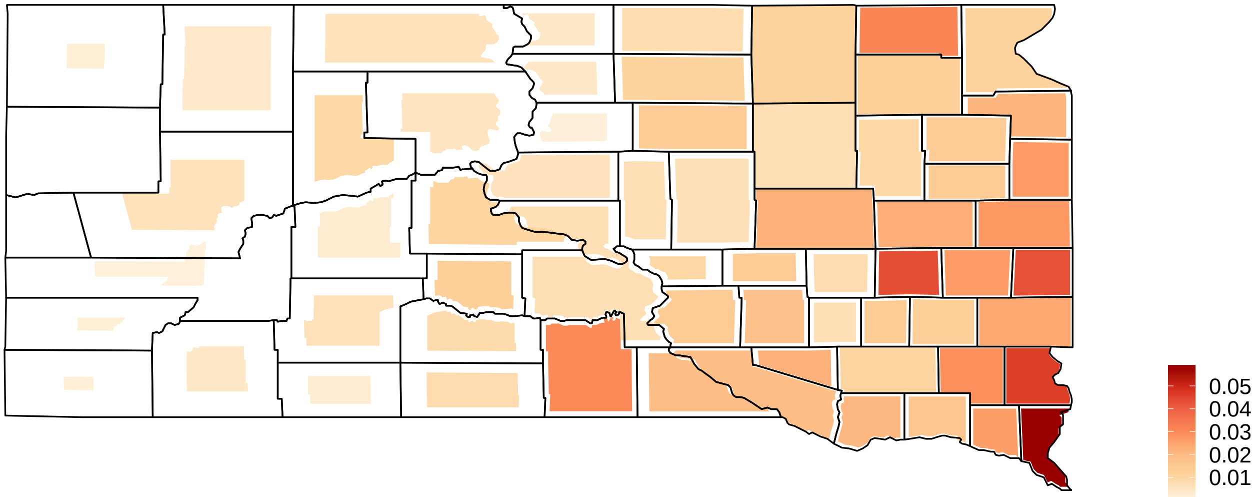
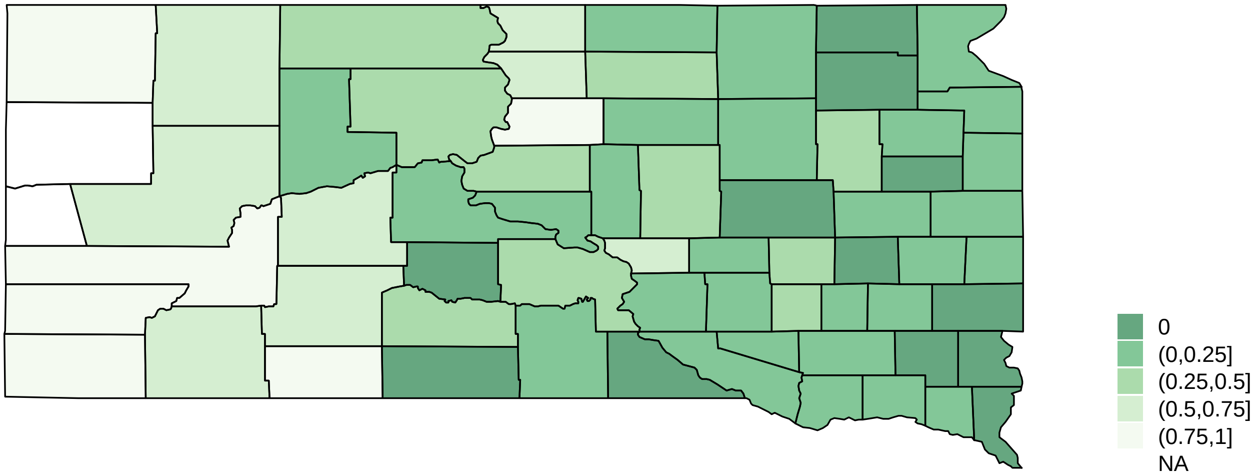
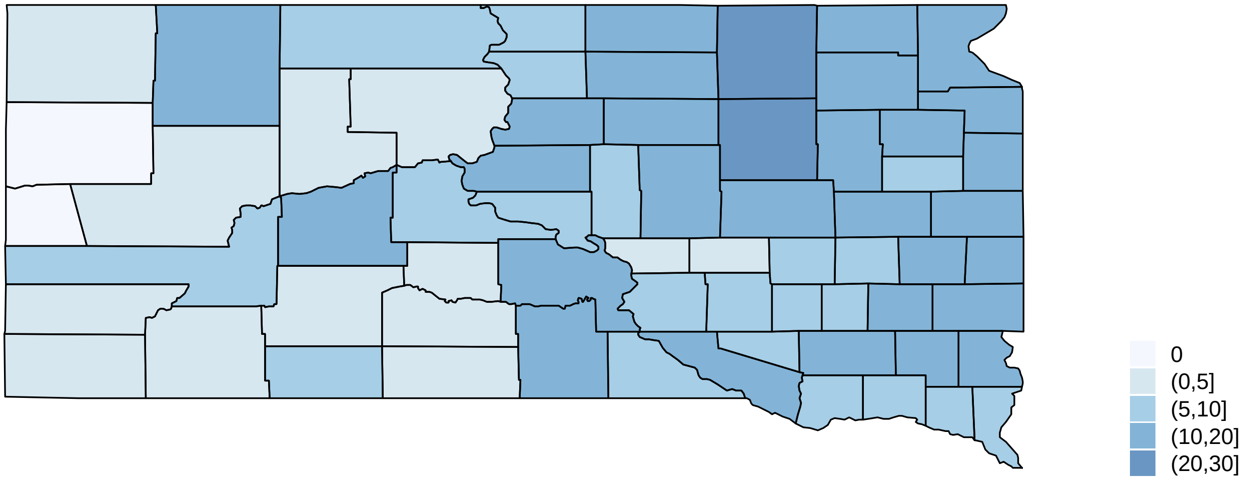


Figure S3

```
cty <- erosion %>% group_by(ctylab = tolower(ctylab)) %>%
  summarise(nis = n(), qi = mean(RUSLE2==0)) %>%
  mutate(sizegroup = cut(nis, c(0, 5, 10, 20, 30)),
         qigroup = cut(qi, seq(0, 1, 0.25)))
levels(cty$qigroup) <- c(levels(cty$qigroup), "0")
levels(cty$sizegroup) <- c(levels(cty$sizegroup), "0")
cty <- cty %>% mutate(qigroup = replace(qigroup, is.na(qigroup), "0"),
                      qigroup = relevel(qigroup, "0"))
cty2 <- left_join(map_sd, cty, by = c("subregion" = "ctylab"))
cty2 <- cty2 %>% mutate(sizegroup = replace(sizegroup, is.na(sizegroup), "0"),
                        sizegroup = relevel(sizegroup, "0"))
p1 <- cty2 %>% ggplot() +
  geom_polygon(aes(x = long, y = lat, group = factor(group), fill = qigroup), alpha = 0.6) +
  scale_fill_brewer(name = "", palette = "Greens", direction = -1) +
  geom_path(aes(x = long, y = lat, group = factor(group))) +
  coord_equal() + ggthemes::theme_map(base_size = 18) +
  theme(legend.position = "right")
p2 <- cty2 %>% ggplot() +
  geom_polygon(aes(x = long, y = lat, group = factor(group),
                   fill = sizegroup), alpha = 0.6) +
  scale_fill_brewer(name = "", palette = "Blues") +
  geom_path(aes(x = long, y = lat, group = factor(group))) +
  coord_equal() + ggthemes::theme_map(base_size = 18) +
  theme(legend.position = "right")
p1 / p2
```


0
(0,0.25]
(0.25,0.5]
(0.5,0.75]
(0.75,1]
NA


0
(0,5]
(5,10]
(10,20]
(20,30]