Synthetic data experiment

Michele Nguyen

13/8/2020

Introduction

In this report, we create synthetic, unbalanced damage datasets to illustrate several advantages of ordinal regression over a nominal approach of fitting separate probit regressions per damage state. These include:

1. Higher sensitivity to the given data across the damage states;
2. Parsimony leading to lower risk of overfitting;
3. No crossing fragility curves for the different damage states.

Data generation

To aid our illustration, we create four synthetic datasets containing buildings with the peak ground acceleration (PGA) values in $\{0.1, 0.4, 0.7\}$ and damage grades ranging from 0-5 (as in the Nepal 2015 damage data). Each dataset has 100 buildings per PGA value.

For Dataset 1, the damage distribution within PGA bins varies according to an ordinal model. This is treated as a base dataset. For comparison and to show the lower sensitivity of the nominal approch to the overall structure of the damage data, Dataset 2 is created by switching the number of buildings per damage state between the groups that have states below grade 2 and above grade 2. The stacked barcharts in Figure 1(a) show the damage state proportions in Dataset 1 and 2 at the three PGA values.

To illustrate overfitting with the nominal approach, we add noise to Dataset 1 by randomly choosing $x\%$ of the buildings and randomly changing its damage state according to a discrete uniform distribution (we use $x = 25\%$). This gives us Dataset 3 which also acts as training data. To evaluate the ordinal and probit model fits, we also generate 100 test sets in the same way as Dataset 1 but with different random seeds.

To illustrate the crossing fragility curves from a nominal approach, we add noise to the fragility curves from the ordinal model used to generate Dataset 1. In particular, we vary the slope parameter using $\beta \sim N(\hat{\beta}, c*\hat{\beta})$ where $\hat{\beta}$ is the slope estimate in the ordinal model and $c \in \mathbb{R}^+$ (we use $c = 1.2$). This gives us the crossing fragility curves in Figure 1(b). Dataset 4 is generated according to these fragility curves; however, to have non-negative damage state probabilities, we apply the method in Porter et al. of reverting to the higher fragility curve in regions of crossing.

A summary of the different synthetic datasets is given in Table 1.

```
data_table <- data.frame("Dataset" = 1:4,
                         "Generation" = c("Ordinal model", "Buildings switched about Grade 2",
                                          "Added noise to x% of damage states",
                                          "Added variation to cross curves"),
                         "Purpose" = c("Base data", "Show insensitivity to data",
                                       "Show overfitting", "Show crossing fragility curves"))

kable(data_table, caption = "Summary of the synthetic datasets.")
```

Table 1: Summary of the synthetic datasets.

Dataset	Generation	Purpose
1	Ordinal model	Base data
2	Buildings switched about Grade 2	Show insensitivity to data
3	Added noise to x% of damage states	Show overfitting
4	Added variation to cross curves	Show crossing fragility curves

```
## Dataset 1: Training/Base Data ##

# Read in Nepal buildings category data:

ordinal_data <- read.csv(file = "D:/Documents/Ordinal_Fragility_Curves/Data/ordinal_data.csv",
                         stringsAsFactors = FALSE)

# Only use timber superstructure:

timber_data <- ordinal_data[ordinal_data$superstructure == "timber", ]

# Fit ordinal regression:
timber_data$damage_grade <- ordered(timber_data$damage_grade,
                                    levels = c("Grade 0", "Grade 1",
                                               "Grade 2", "Grade 3",
                                               "Grade 4", "Grade 5"))
timber_ordinal <- polr(damage_grade ~ logPGA, data = timber_data,
                               method = "probit", Hess = TRUE)

# Generate Dataset 1 via predicted probabilities:

no.buildings <- 100
# (Maximum) number of buildings per PGA bin; can delete rows later if want diff. no. per PGA bin.
pga_range <- data.frame("PGA" = seq(0.1, 0.9, by = 0.3))
pga_range$logPGA <- log(pga_range$PGA)
ds1_prob <- predict(timber_ordinal, newdata = pga_range, type = "prob")

set.seed(1)
ds1_count <- apply(ds1_prob, MARGIN = 1, FUN = function(x){sample(c("Grade 0", "Grade 1",
                                                                    "Grade 2", "Grade 3",
                                                                    "Grade 4", "Grade 5"),
                                                                  size = no.buildings,
                                                                  prob = x, replace = TRUE)})

dataset_1 <- data.frame("PGA" = rep(pga_range$PGA, each = no.buildings),
                        "logPGA" = rep(pga_range$logPGA, each = no.buildings),
                        "damage_grade" = c(ds1_count[, 1], ds1_count[, 2], ds1_count[, 3]))
table(dataset_1$damage_grade)
```

```
##
## Grade 1 Grade 2 Grade 3 Grade 4 Grade 5
##     147      65      48      23      17
```

```r
dataset_1$Data <- "Dataset 1"
```

```r
## Dataset 2: For illustration of insensitivity to data ##

# Generate Dataset 2 by focusing on the estimation of Grade 3
# and changing no. of buildings in states above/below:

dataset_2 <- dataset_1

# Switch Grade 1 with Grade 2; Grade 4 -> Grade 3; Grade 5 -> Grade 4;
dataset_2$damage_grade_2 <- NA
dataset_2$damage_grade_2[dataset_2$damage_grade == "Grade 1"] <- "Grade 2"
dataset_2$damage_grade_2[dataset_2$damage_grade == "Grade 2"] <- "Grade 1"
dataset_2$damage_grade_2[dataset_2$damage_grade == "Grade 4"] <- "Grade 3"
dataset_2$damage_grade_2[dataset_2$damage_grade == "Grade 5"] <- "Grade 4"
dataset_2$damage_grade_2[dataset_2$damage_grade == "Grade 3"] <- "Grade 5"

dataset_2$Data <- "Dataset 2"
dataset_2$damage_grade <- dataset_2$damage_grade_2
dataset_2 <- dataset_2[, colnames(dataset_1)]
table(dataset_2$damage_grade)
```

```
##
## Grade 1 Grade 2 Grade 3 Grade 4 Grade 5
##      65     147      23      17      48
```

```r
## Dataset 3 (Training set): For illustrating overfitting ##

# Generate Dataset 3 as Dataset 1 but x% chosen randomly from a
# uniform distribution across the damage states:

temp_dataset_3 <- dataset_1
x <- 0.25
n <- nrow(temp_dataset_3)
temp_dataset_3$damage_grade <- as.character(temp_dataset_3$damage_grade)

# Choose x% of rows to change damage states for:
set.seed(2)
change_id <- sample(n, x*n, replace = FALSE)
set.seed(3)
# Exclude Grade 0 for now.
change_val <- sample(c("Grade 1", "Grade 2", "Grade 3", "Grade 4", "Grade 5"),
                     size = x*n, replace = TRUE)

temp_dataset_3[change_id, "damage_grade"] <- change_val
dataset_3 <- temp_dataset_3
```

```r
## Dataset 4: For illustrating crossing curves and overfitting ##

PGA_list <- c(0.00000001, seq(0.0025, 1, by = 0.00025))
c <- 1.2
```

```r
ds4_lp_pred <- expand.grid("logPGA" = log(PGA_list),
                           "damage_grade" = c("Grade 0", "Grade 1", "Grade 2",
                                              "Grade 3","Grade 4"))
# Exclude Grade 5 because fragility curve for probability of exceedance.
ds4_zeta <- timber_ordinal$zeta
set.seed(123)
ds4_coeff <- rnorm(length(ds4_zeta), mean = timber_ordinal$coefficients,
                   sd = c*timber_ordinal$coefficients)

ds4_lp_pred$b0_id <- as.numeric(ds4_lp_pred$damage_grade)
ds4_lp_pred$b0 <- ds4_zeta[ds4_lp_pred$b0_id]

ds4_lp_pred$b1 <- ds4_coeff[ds4_lp_pred$b0_id]

ds4_lp_mean <- ds4_lp_pred$b0 - ds4_lp_pred$b1*ds4_lp_pred$logPGA
ds4_mean <- pnorm(ds4_lp_mean, lower.tail = TRUE)

# Currently, 1- exceedance probability.
ds4_mean <- 1 - ds4_mean

ds4_frag <- data.frame("PGA" = rep(PGA_list, 5),
                       "damage_grade" = rep(c("Grade 0", "Grade 1", "Grade 2",
                                              "Grade 3","Grade 4"), each = length(PGA_list)),
                       "Mean" = ds4_mean)
```

```r
# Convert exceedance probabilities to damage state probabilities:

excp_to_prob <- function(x){
  temp <- rep(NA, length(x) +1)
  temp[1] <- 1 - x$Mean[x$damage_grade == "Grade 0"]
  temp[2] <- x$Mean[x$damage_grade == "Grade 0"] - x$Mean[x$damage_grade == "Grade 1"]
  temp[3] <- x$Mean[x$damage_grade == "Grade 1"] - x$Mean[x$damage_grade == "Grade 2"]
  temp[4] <- x$Mean[x$damage_grade == "Grade 2"] - x$Mean[x$damage_grade == "Grade 3"]
  temp[5] <- x$Mean[x$damage_grade == "Grade 3"] - x$Mean[x$damage_grade == "Grade 4"]
  temp[6] <- x$Mean[x$damage_grade == "Grade 4"]
  return(temp)
}

ds4_prob <- matrix(NA, ncol = 6, nrow = nrow(pga_range))

for (i in 1:nrow(pga_range)){

  ds4_excp <- ds4_frag[ds4_frag$PGA == pga_range$PGA[i], ]

  # Use Porter method I for generating data (use higher exceedance probabilities at crossing):
  for(j in nrow(ds4_excp):2){
    if(ds4_excp$Mean[j]>ds4_excp$Mean[j-1]){
      ds4_excp$Mean[j-1] <- ds4_excp$Mean[j]
    }
  }

  ds4_prob[i, ] <- excp_to_prob(ds4_excp)
```

```
}

# Sample 100 buildings per PGA value:

set.seed(2)
ds4_count <- apply(ds4_prob, MARGIN = 1, FUN = function(x){sample(x = c("Grade 0", "Grade 1",
                                                                        "Grade 2", "Grade 3",
                                                                        "Grade 4", "Grade 5"),
                                                                  size = no.buildings,
                                                                  prob = x, replace = TRUE)})

dataset_4 <- data.frame("PGA" = rep(pga_range$PGA, each = no.buildings),
                        "logPGA" = rep(pga_range$logPGA, each = no.buildings),
                        "damage_grade" = c(ds4_count[, 1], ds4_count[, 2], ds4_count[, 3]))
table(dataset_4$damage_grade)
```

```
##
## Grade 1 Grade 2 Grade 3 Grade 4 Grade 5
##     108     126       8      31      27
```

```
dataset_4$Data <- "Dataset 4"
```

```
dataset_12 <- rbind(dataset_1, dataset_2)

widedata_12 <- unique(dataset_12[, c("PGA", "logPGA", "damage_grade", "Data")])
widedata_12$Count <- NA

for (i in 1:nrow(widedata_12)){
  widedata_12$Count[i] <- sum(dataset_12$PGA == widedata_12$PGA[i] &
                                dataset_12$damage_grade == widedata_12$damage_grade[i] &
                                dataset_12$Data == widedata_12$Data[i])
}

# New facet labels for PGA:
PGA.labs <- c("PGA = 0.1", "PGA = 0.4", "PGA = 0.7")
names(PGA.labs) <- c(0.1, 0.4, 0.7)

widedata_12$Data[widedata_12$Data == "Dataset 1"] <- "DS 1"
widedata_12$Data[widedata_12$Data == "Dataset 2"] <- "DS 2"

plot1 <- ggplot(data = widedata_12) +
  geom_bar(aes(fill = damage_grade, y = Count, x = Data), width = 0.25, position = "fill",
           stat = "identity")  + facet_wrap(~PGA, labeller = labeller(PGA = PGA.labs)) +
  labs(fill = "", x = "", y = "Proportion of buildings") + theme_classic() +
  scale_fill_viridis_d() + ggtitle("(a)") +
  theme(plot.title = element_text(hjust = 0.5), legend.position = "bottom",
        panel.spacing = grid::unit(4, "lines"),
        axis.title.x = element_text(margin=margin(10,0,0,0)),
        axis.title.y = element_text(margin=margin(0,10,0,0)))
```

```
plot2 <- ggplot(data = ds4_frag[ds4_frag$damage_grade != "Grade 0", ]) +
  geom_line(aes(x = PGA, y = Mean, color = damage_grade)) + ylim(0, 1) + ggtitle("(b)") +
```

```
  labs(y = "Probability of exceedance", x = "PGA", color = "") +
  theme_bw() + theme(plot.title = element_text(hjust = 0.5), legend.position = "bottom",
                     axis.title.x = element_text(margin=margin(10,0,0,0)),
                     axis.title.y = element_text(margin=margin(0,10,0,0))) +
  guides(color = guide_legend(order = 1))

grid.arrange(plot1, plot2, ncol = 2)
```

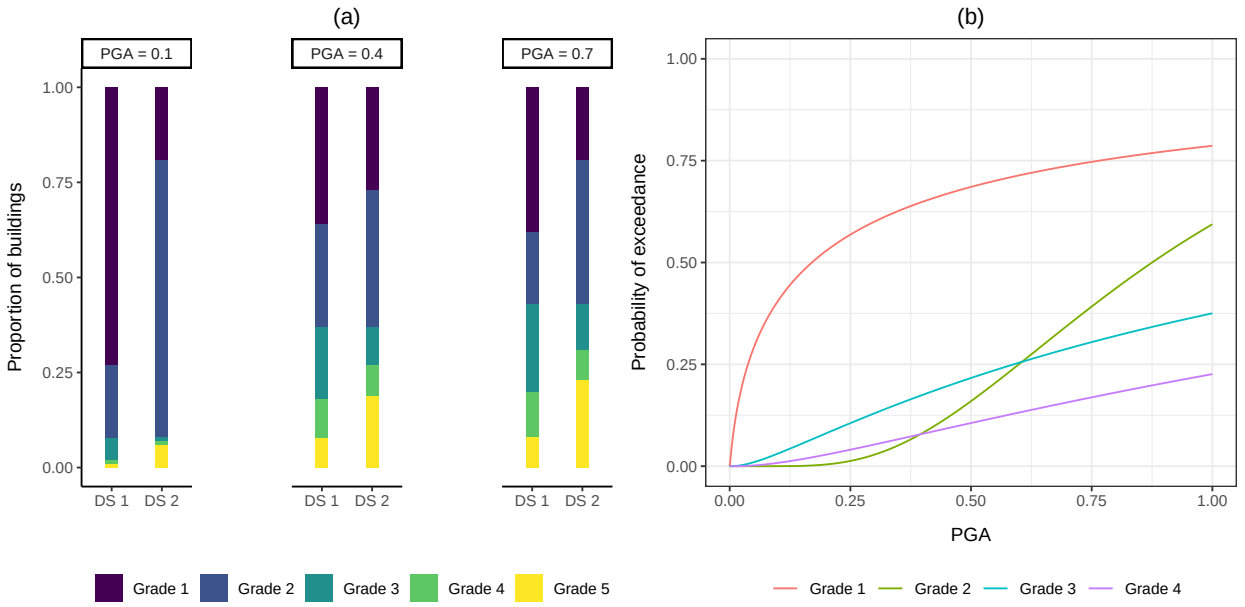

Figure 1: (a) Distribution of buildings according to PGA and damage grades in Dataset 1 and 2; (b) Crossing fragility curves used to generate Dataset 4.

Advantage 1: Higher sensitivity to the data

In this section, we use the ordinal and probit regressions to fit fragility curves for the exceedance of Grade 3 for Dataset 1 and 2. Figure 2 shows that exactly the same curve has been derived using the probit regressions for the two different datasets. This is because when we treat damage states as nominal and only consider them one state at a time, we do not make use of the information on the distribution of damage in other damage states. In comparison, by treating damage states as ordinal, we make full use of the data across all damage states. Thus, the fragility curves obtained from ordinal regressions in Figure 2 are different for Dataset 1 and 2.

```
ds1_ex3 <- dataset_1
ds1_ex3$Damage <- 1
ds1_ex3$Damage[ds1_ex3$damage_grade %in% c("Grade 0", "Grade 1", "Grade 2")] <- 0
ds1_probit3 <- glm(Damage ~ logPGA, family = binomial(link = "probit"), data = ds1_ex3)
summary(ds1_probit3)
```

```
##
## Call:
## glm(formula = Damage ~ logPGA, family = binomial(link = "probit"),
```

```
##     data = ds1_ex3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1141  -0.8878  -0.4357   1.2420   2.1917
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.1335     0.1348   0.990    0.322
## logPGA        0.6388     0.1108   5.764  8.2e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 363.07  on 299  degrees of freedom
## Residual deviance: 325.65  on 298  degrees of freedom
## AIC: 329.65
##
## Number of Fisher Scoring iterations: 4
```

```
ds2_ex3 <- dataset_2
ds2_ex3$Damage <- 1
ds2_ex3$Damage[ds2_ex3$damage_grade %in% c("Grade 0", "Grade 1", "Grade 2")] <- 0
ds2_probit3 <- glm(Damage ~ logPGA, family = binomial(link = "probit"), data = ds2_ex3)
summary(ds2_probit3)
```

```
##
## Call:
## glm(formula = Damage ~ logPGA, family = binomial(link = "probit"),
##     data = ds2_ex3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1141  -0.8878  -0.4357   1.2420   2.1917
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.1335     0.1348   0.990    0.322
## logPGA        0.6388     0.1108   5.764  8.2e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 363.07  on 299  degrees of freedom
## Residual deviance: 325.65  on 298  degrees of freedom
## AIC: 329.65
##
## Number of Fisher Scoring iterations: 4
```

```
# Fit ordinal regressions:
```

```r
dataset_1$damage_grade <- ordered(dataset_1$damage_grade,
                                  levels = c("Grade 0", "Grade 1",
                                             "Grade 2", "Grade 3",
                                             "Grade 4", "Grade 5"))
ds1_ordinal <- polr(damage_grade ~ logPGA, data = dataset_1,
                                method = "probit", Hess = TRUE)

summary(ds1_ordinal)
```

```
## Call:
## polr(formula = damage_grade ~ logPGA, data = dataset_1, Hess = TRUE,
##     method = "probit")
##
## Coefficients:
##         Value Std. Error t value
## logPGA 0.5481    0.08599   6.374
##
## Intercepts:
##                 Value   Std. Error t value
## Grade 0|Grade 1 -5.4635  7.0158    -0.7787
## Grade 1|Grade 2 -0.6770  0.1243    -5.4474
## Grade 2|Grade 3 -0.0506  0.1202    -0.4210
## Grade 3|Grade 4  0.5713  0.1254     4.5577
## Grade 4|Grade 5  1.0745  0.1442     7.4523
##
## Residual Deviance: 758.1153
## AIC: 770.1153
```

```r
dataset_2$damage_grade <- ordered(dataset_2$damage_grade,
                                  levels = c("Grade 0", "Grade 1",
                                             "Grade 2", "Grade 3",
                                             "Grade 4", "Grade 5"))
ds2_ordinal <- polr(damage_grade ~ logPGA, data = dataset_2,
                                method = "probit", Hess = TRUE)

summary(ds2_ordinal)
```

```
## Call:
## polr(formula = damage_grade ~ logPGA, data = dataset_2, Hess = TRUE,
##     method = "probit")
##
## Coefficients:
##         Value Std. Error t value
## logPGA 0.2575    0.07803   3.301
##
## Intercepts:
##                 Value   Std. Error t value
## Grade 0|Grade 1 -4.6103  4.0270    -1.1448
## Grade 1|Grade 2 -1.1114  0.1290    -8.6144
## Grade 2|Grade 3  0.2474  0.1178     2.0999
## Grade 3|Grade 4  0.5007  0.1182     4.2365
## Grade 4|Grade 5  0.7199  0.1208     5.9580
```

```
##
## Residual Deviance: 789.233
## AIC: 801.233
```

```
# PGA_list <- c(0.00000001, seq(0.0025, 1, by = 0.0025))


ds1_lp_pred <- expand.grid("logPGA" = log(PGA_list),
                           "damage_grade" = c("Grade 0", "Grade 1", "Grade 2",
                                              "Grade 3","Grade 4"))
# Exclude Grade 5 because fragility curve for probability of exceedance.
ds1_lp_pred$b0_id <- as.numeric(ds1_lp_pred$damage_grade)
ds1_lp_pred$b0 <- ds1_ordinal$zeta[ds1_lp_pred$b0_id]
ds1_ordinal_lp_mean <- ds1_lp_pred$b0 - ds1_ordinal$coefficients*ds1_lp_pred$logPGA
ds1_ordinal_mean <- pnorm(ds1_ordinal_lp_mean, lower.tail = TRUE)
# Currently, 1- exceedance probability.
ds1_ordinal_mean <- 1 - ds1_ordinal_mean

ds1_ordinal_frag <- data.frame("PGA" = rep(PGA_list, nlevels(dataset_1$damage_grade)-1),
                           "damage_grade" = rep(c("Grade 0", "Grade 1", "Grade 2",
                                                  "Grade 3","Grade 4"), each = length(PGA_list)),
                           "Mean" = ds1_ordinal_mean)

plot1b <- ggplot(data = ds1_ordinal_frag[ds1_ordinal_frag$damage_grade != "Grade 0", ]) +
  geom_line(aes(x = PGA, y = Mean, color = damage_grade)) + ylim(0, 1) +
  ggtitle("(b) Ordinal regression") +
  labs(y = "Probability of exceedance", x = "PGA", color = "Damage grade") +
  theme_bw() + theme(plot.title = element_text(hjust = 0.5)) +
  guides(color = guide_legend(order = 1))
```

```
ds2_lp_pred <- expand.grid("logPGA" = log(PGA_list),
                           "damage_grade" = c("Grade 0", "Grade 1", "Grade 2",
                                              "Grade 3","Grade 4"))
# Exclude Grade 5 because fragility curve for probability of exceedance.
ds2_lp_pred$b0_id <- as.numeric(ds2_lp_pred $damage_grade)
ds2_lp_pred$b0 <- ds2_ordinal$zeta[ds2_lp_pred$b0_id]
ds2_ordinal_lp_mean <- ds2_lp_pred$b0 - ds2_ordinal$coefficients*ds2_lp_pred$logPGA
ds2_ordinal_mean <- pnorm(ds2_ordinal_lp_mean, lower.tail = TRUE)
# Currently, 1- exceedance probability.
ds2_ordinal_mean <- 1 - ds2_ordinal_mean

ds2_ordinal_frag <- data.frame("PGA" = rep(PGA_list, nlevels(dataset_2$damage_grade)-1),
                               "damage_grade" = rep(c("Grade 0", "Grade 1", "Grade 2",
                                                      "Grade 3","Grade 4"), each = length(PGA_list)),
                               "Mean" = ds2_ordinal_mean)

plot2b <- ggplot(data = ds2_ordinal_frag[ds2_ordinal_frag$damage_grade != "Grade 0", ]) +
  geom_line(aes(x = PGA, y = Mean, color = damage_grade)) + ylim(0, 1) +
  ggtitle("(b) Ordinal regression") +
  labs(y = "Probability of exceedance", x = "PGA", color = "Damage grade") +
  theme_bw() + theme(plot.title = element_text(hjust = 0.5)) +
  guides(color = guide_legend(order = 1))
```

```
plot_pga <- data.frame("PGA" = PGA_list)
plot_pga$logPGA <- log(plot_pga$PGA)

ds1_lp_mean <- predict.glm(ds1_probit3, newdata = plot_pga, type = "link")
ds1_frag <- pnorm(ds1_lp_mean, lower.tail = TRUE)

ds2_lp_mean <- predict.glm(ds2_probit3, newdata = plot_pga, type = "link")
ds2_frag <- pnorm(ds2_lp_mean, lower.tail = TRUE)

frag_probit <- data.frame("PGA" = rep(plot_pga$PGA, 2), "Mean" = c(ds1_frag, ds2_frag),
                          "Model_Data" = rep(c("Probit: DS 1", "Probit: DS 2"),
                                             each = nrow(plot_pga)))

temp_res <- ds1_ordinal_frag[ds1_ordinal_frag$damage_grade == "Grade 2", ]
temp_df <- data.frame("PGA" = temp_res$PGA, "Mean" = temp_res$Mean)
temp_df$Model_Data <- "Ordinal: DS 1"

frag_probit <- rbind(frag_probit, temp_df)

temp_res <- ds2_ordinal_frag[ds2_ordinal_frag$damage_grade == "Grade 2", ]
temp_df <- data.frame("PGA" = temp_res$PGA, "Mean" = temp_res$Mean)
temp_df$Model_Data <- "Ordinal: DS 2"

frag_probit <- rbind(frag_probit, temp_df)

plot3 <- ggplot(data = frag_probit) +
  geom_line(aes(x = PGA, y = Mean, color = Model_Data, lty = Model_Data), lwd = 1) +
  ylim(0, 1) + labs(y = "Probability of exceedance", x = "PGA", color = "") + theme_bw() +
  guides(lty = FALSE) + scale_color_viridis_d(direction = -1) +
  theme(legend.position = "bottom", plot.title = element_text(hjust = 0.5,
                                                              margin = margin(0, 0, 25, 0)),
        axis.title.x = element_text(margin=margin(10,0,0,0)),
        axis.title.y = element_text(margin=margin(0,10,0,0))) +
  ggtitle("(b)")

grid.arrange(plot1, plot3, nrow = 1, ncol = 2)
```

Advantage 2: Lower risk of overfitting

In this section, we illustrate the issue of overfitting with the nominal approach due to the additional parameters estimated via the separate probit regressions.

We compute the Kullback-Leibler (KL) divergence between the damage state proportions estimated by the models and that from test data at each of the three PGA values. The average KL divergence across the PGA values can be used as a goodness of fit measure. Table 2 shows the mean KL divergence values at the three PGA values from 100 test sets (generated from the original ordinal model with a different random seed from Dataset 1) as well as the averages for the probit and ordinal models. Since the KL divergence value is lower at all the PGA values, this means that the ordinal model provides a better fit to the test data since the estimated damage state proportions from the model is more similar to the empirical proportions.

Figure 3 shows the behaviour of the KL values as the proportion of data replaced by uniformly distributed damage states (from Grade 1-5 since there are no Grade 0 data in the original dataset), x, varies. In general, we see that the ordinal model gives lower KL values than the separate probit models. However, the extent

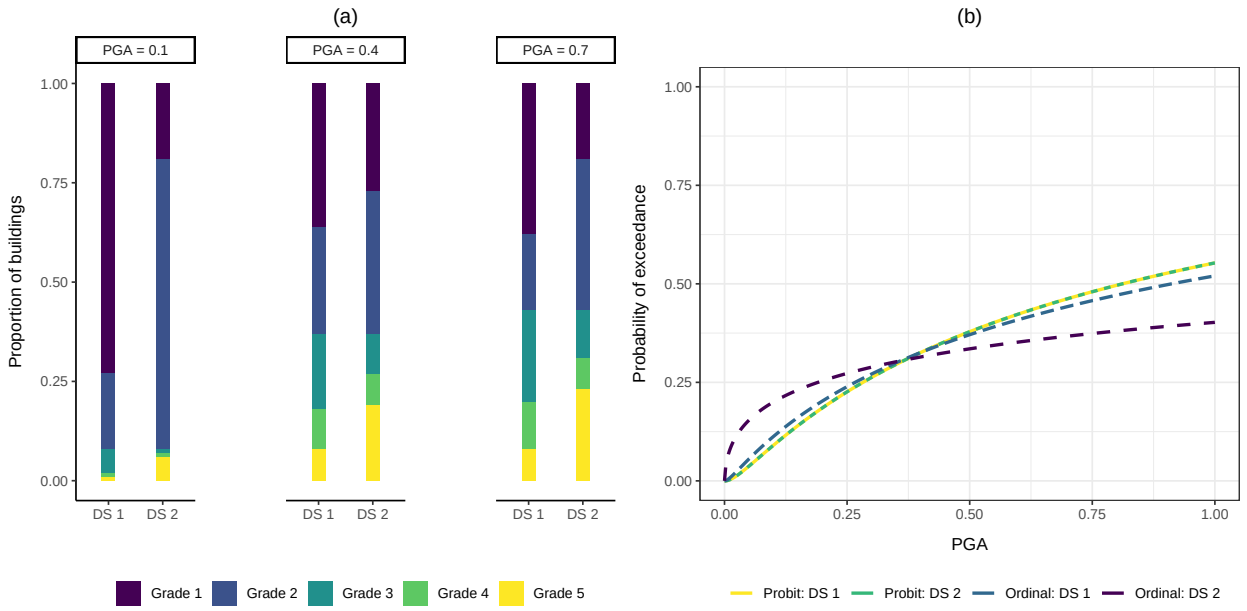

Figure 2: (a) Distribution of buildings according to PGA and damage grades in Dataset 1 and 2; (b) Fragility curves derived via ordinal and probit regressions for the exceedance of Grade 3 using Dataset 1 and 2.

of this difference as well as the increase in KL as x increases differs according to the PGA value. There also seems to be a limit of the advantage of the ordinal over the probit model which is exemplified by the switch in order of KL around $x = 0.44$ for $PGA = 0.1$. This makes sense since when a large proportion of the training data is noise, neither model can reflect the original model and hence test data as well.

```
# Fit fragility curves to Dataset 3 using separate probit regressions:

ds3_ex0 <- dataset_3
ds3_ex0$Damage <- 1
ds3_ex0$Damage[ds3_ex0$damage_grade %in% c("Grade 0")] <- 0
ds3_probit0 <- glm(Damage ~ logPGA, family = binomial(link = "probit"), data = ds3_ex0)
```

```
## Warning: glm.fit: algorithm did not converge
```

```
# summary(ds3_probit0)

ds3_ex1 <- dataset_3
ds3_ex1$Damage <- 1
ds3_ex1$Damage[ds3_ex1$damage_grade %in% c("Grade 0", "Grade 1")] <- 0
ds3_probit1 <- glm(Damage ~ logPGA, family = binomial(link = "probit"), data = ds3_ex1)
# summary(ds3_probit1)

ds3_ex2 <- dataset_3
ds3_ex2$Damage <- 1
ds3_ex2$Damage[ds3_ex2$damage_grade %in% c("Grade 0", "Grade 1", "Grade 2")] <- 0
ds3_probit2 <- glm(Damage ~ logPGA, family = binomial(link = "probit"), data = ds3_ex2)
# summary(ds3_probit2)

ds3_ex3 <- dataset_3
```

```r
ds3_ex3$Damage <- 1
ds3_ex3$Damage[ds3_ex3$damage_grade %in% c("Grade 0", "Grade 1", "Grade 2", "Grade 3")] <- 0
ds3_probit3 <- glm(Damage ~ logPGA, family = binomial(link = "probit"), data = ds3_ex3)
# summary(ds3_probit3)

ds3_ex4 <- dataset_3
ds3_ex4$Damage <- 1
ds3_ex4$Damage[ds3_ex4$damage_grade %in% c("Grade 0", "Grade 1", "Grade 2", "Grade 3", "Grade 4")] <- 0
ds3_probit4 <- glm(Damage ~ logPGA, family = binomial(link = "probit"), data = ds3_ex4)
# summary(ds3_probit4)

ex0_lp_mean <- predict.glm(ds3_probit0, newdata = plot_pga, type = "link")
frag_ex0 <- pnorm(ex0_lp_mean, lower.tail = TRUE)

ex1_lp_mean <- predict.glm(ds3_probit1, newdata = plot_pga, type = "link")
frag_ex1 <- pnorm(ex1_lp_mean, lower.tail = TRUE)

ex2_lp_mean <- predict.glm(ds3_probit2, newdata = plot_pga, type = "link")
frag_ex2 <- pnorm(ex2_lp_mean, lower.tail = TRUE)

ex3_lp_mean <- predict.glm(ds3_probit3, newdata = plot_pga, type = "link")
frag_ex3 <- pnorm(ex3_lp_mean, lower.tail = TRUE)

ex4_lp_mean <- predict.glm(ds3_probit4, newdata = plot_pga, type = "link")
frag_ex4 <- pnorm(ex4_lp_mean, lower.tail = TRUE)

frag_probit3 <- data.frame("PGA" = rep(plot_pga$PGA, 5),
                           "Mean" = c(frag_ex0, frag_ex1, frag_ex2, frag_ex3, frag_ex4),
                          "damage_grade" = rep(c("Grade 0", "Grade 1", "Grade 2", "Grade 3",
                                                 "Grade 4"), each = nrow(plot_pga)))
```

```r
# Fit ordinal regression to Dataset 3.


dataset_3$damage_grade <- ordered(dataset_3$damage_grade,
                                  levels = c("Grade 0", "Grade 1",
                                             "Grade 2", "Grade 3",
                                             "Grade 4", "Grade 5"))
ds3_ordinal <- polr(damage_grade ~ logPGA, data = dataset_3,
                                method = "probit", Hess = TRUE)

summary(ds3_ordinal)
```

```
## Call:
## polr(formula = damage_grade ~ logPGA, data = dataset_3, Hess = TRUE,
##     method = "probit")
##
## Coefficients:
##         Value Std. Error t value
## logPGA 0.3738    0.08022   4.659
##
## Intercepts:
##                 Value   Std. Error t value
```

```
## Grade 0|Grade 1 -5.0889  5.4576    -0.9324
## Grade 1|Grade 2 -0.6849  0.1222    -5.6060
## Grade 2|Grade 3 -0.1127  0.1176    -0.9584
## Grade 3|Grade 4  0.3618  0.1179     3.0684
## Grade 4|Grade 5  0.8593  0.1280     6.7150
##
## Residual Deviance: 861.2789
## AIC: 873.2789
```

```
ds3_lp_pred <- expand.grid("logPGA" = log(PGA_list),
                           "damage_grade" = c("Grade 0", "Grade 1", "Grade 2",
                                              "Grade 3","Grade 4"))
# Exclude Grade 5 because fragility curve for probability of exceedance.
ds3_lp_pred$b0_id <- as.numeric(ds3_lp_pred$damage_grade)
ds3_lp_pred$b0 <- ds3_ordinal$zeta[ds3_lp_pred$b0_id]
ds3_ordinal_lp_mean <- ds3_lp_pred$b0 - ds3_ordinal$coefficients*ds3_lp_pred$logPGA
ds3_ordinal_mean <- pnorm(ds3_ordinal_lp_mean, lower.tail = TRUE)
# Currently, 1- exceedance probability.
ds3_ordinal_mean <- 1 - ds3_ordinal_mean

ds3_ordinal_frag <- data.frame("PGA" = rep(PGA_list, nlevels(dataset_1$damage_grade)-1),
                               "damage_grade" = rep(c("Grade 0", "Grade 1", "Grade 2",
                                                      "Grade 3","Grade 4"), each = length(PGA_list)),
                               "Mean" = ds3_ordinal_mean)
```

```
#K-L divergence measure:
KL_probit <- rep(NA, nrow(pga_range))
KL_ordinal <- rep(NA, nrow(pga_range))

no.buildings.test <- no.buildings
test_KL_probit <- matrix(NA, nrow = 100, ncol = nrow(pga_range))
test_KL_ordinal <- matrix(NA, nrow = 100, ncol = nrow(pga_range))

for (j in 1:100){

  # Generate a test set:
  set.seed(j+10)
  test_count <- apply(ds1_prob, MARGIN = 1, FUN = function(x){sample(c("Grade 0", "Grade 1",
                                                                       "Grade 2", "Grade 3",
                                                                       "Grade 4", "Grade 5"),
                                                                     size = no.buildings.test,
                                                                     prob = x, replace = TRUE)})

  test_set <- data.frame("PGA" = rep(pga_range$PGA, each = no.buildings.test),
                         "logPGA" = rep(pga_range$logPGA, each = no.buildings.test),
                         "damage_grade" = c(test_count[, 1], test_count[, 2], test_count[, 3]))
  test_set$Data <- "Test set"

    for (i in 1:nrow(pga_range)){

    probit_excp <- frag_probit3[frag_probit3$PGA == pga_range$PGA[i], ]
    ordinal_excp <- ds3_ordinal_frag[ds3_ordinal_frag$PGA == pga_range$PGA[i], ]
    probit_prob <- excp_to_prob(probit_excp)
    names(probit_prob) <- c("Grade 0", "Grade 1", "Grade 2", "Grade 3", "Grade 4", "Grade 5")
```

```
    ordinal_prob <- excp_to_prob(ordinal_excp)
    names(ordinal_prob) <- c("Grade 0", "Grade 1", "Grade 2", "Grade 3", "Grade 4", "Grade 5")


    test_subset <- test_set[test_set$PGA == pga_range$PGA[i], ]
    test_prob <- c(0, sum(test_subset$damage_grade == "Grade 1"),
                sum(test_subset$damage_grade == "Grade 2"),
                sum(test_subset$damage_grade == "Grade 3"),
                sum(test_subset$damage_grade == "Grade 4"),
                sum(test_subset$damage_grade == "Grade 5"))
    test_prob <- test_prob/sum(test_prob)

    valid_id <- which(test_prob>0 & probit_prob>0)
    valid_id_2 <- which(test_prob>0 & ordinal_prob>0)

    test_KL_probit[j, i] <- sum(test_prob[valid_id]*log2(test_prob[valid_id]/probit_prob[valid_id]))
    test_KL_ordinal[j, i] <- sum(test_prob[valid_id_2 ]*log2(test_prob[valid_id_2 ]/
                                                          ordinal_prob[valid_id_2]))
    # Note asymmetry: there are other options for KL.
}
}
```

```
kl_table <- data.frame("Model" = c("Probit", "Ordinal"),
                       "PGA_0.1" = round(c(mean(test_KL_probit[, 1]), mean(test_KL_ordinal[, 1])),
                                         digits = 3),
                       "PGA_0.4" = round(c(mean(test_KL_probit[, 2]), mean(test_KL_ordinal[, 2])),
                                         digits = 3),
                       "PGA_0.7" = round(c(mean(test_KL_probit[, 3]), mean(test_KL_ordinal[, 3])),
                                         digits = 3),
                       "Average" = round(c(mean(test_KL_probit), mean(test_KL_ordinal)),
                                         digits = 3))

kable(kl_table, caption = "Kullback-Leibler (KL) divergence values comparing the estimated
      damage state proportions from the ordinal and probit regressions to the empirical
      proportions from test data at the three peak ground acceleration (PGA) values. The
      mean results from 100 randomly generated test sets are shown.",
      col.names = gsub("[_]", " ", names(kl_table)))
```

Table 2: Kullback-Leibler (KL) divergence values comparing the estimated damage state proportions from the ordinal and probit regressions to the empirical proportions from test data at the three peak ground acceleration (PGA) values. The mean results from 100 randomly generated test sets are shown.

Model	PGA 0.1	PGA 0.4	PGA 0.7	Average
Probit	0.162	0.040	0.062	0.088
Ordinal	0.132	0.039	0.036	0.069

Figure 3: Kullback-Leibler (KL) divergence values corresponding to the probit and ordinal model predictions for different proportions of the data replaced by uniformly distributed damage states (x). The results are shown for the three peak ground acceleration (PGA) values and averaged over the PGA values. Each KL value is computed using 100 independent test sets.

Advantage 3: Non-crossing fragility curves

In this section, we illustrate the issue of crossing fragility curves seen in the nominal approach using Dataset 4. We fit probit regressions for each damage state and compare the results to that from the ordinal regressions. Figure 4 shows that the curves from the separate probit regressions cross while those from the ordinal regressions do not.

```
# Fit ordinal regression to Dataset 3.


dataset_4$damage_grade <- ordered(dataset_4$damage_grade,
                                  levels = c("Grade 0", "Grade 1",
                                             "Grade 2", "Grade 3",
                                             "Grade 4", "Grade 5"))
ds4_ordinal <- polr(damage_grade ~ logPGA, data = dataset_4,
                    method = "probit", Hess = TRUE)

summary(ds4_ordinal)
```

```
## Call:
## polr(formula = damage_grade ~ logPGA, data = dataset_4, Hess = TRUE,
##     method = "probit")
##
## Coefficients:
##         Value Std. Error t value
## logPGA 0.6317    0.08435   7.489
##
## Intercepts:
##                 Value   Std. Error t value
## Grade 0|Grade 1 -5.6878 13.7193    -0.4146
## Grade 1|Grade 2 -1.1515  0.1309    -8.7965
## Grade 2|Grade 3  0.1253  0.1184     1.0579
## Grade 3|Grade 4  0.2281  0.1194     1.9094
## Grade 4|Grade 5  0.7498  0.1312     5.7164
##
## Residual Deviance: 709.6669
## AIC: 721.6669
```

```
ds4_lp_pred <- expand.grid("logPGA" = log(PGA_list),
                           "damage_grade" = c("Grade 0", "Grade 1", "Grade 2",
                                              "Grade 3","Grade 4"))
# Exclude Grade 5 because fragility curve for probability of exceedance.
ds4_lp_pred$b0_id <- as.numeric(ds4_lp_pred$damage_grade)
ds4_lp_pred$b0 <- ds4_ordinal$zeta[ds4_lp_pred$b0_id]
ds4_ordinal_lp_mean <- ds4_lp_pred$b0 - ds4_ordinal$coefficients*ds4_lp_pred$logPGA
ds4_ordinal_mean <- pnorm(ds4_ordinal_lp_mean, lower.tail = TRUE)
```

```r
# Currently, 1- exceedance probability.
ds4_ordinal_mean <- 1 - ds4_ordinal_mean

ds4_ordinal_frag <- data.frame("PGA" = rep(PGA_list, nlevels(dataset_1$damage_grade)-1),
                               "damage_grade" = rep(c("Grade 0", "Grade 1", "Grade 2",
                                                      "Grade 3","Grade 4"), each = length(PGA_list)),
                               "Mean" = ds4_ordinal_mean)
```

```r
grid_arrange_shared_legend(plot3, plot4, nrow = 1, ncol = 2)
```

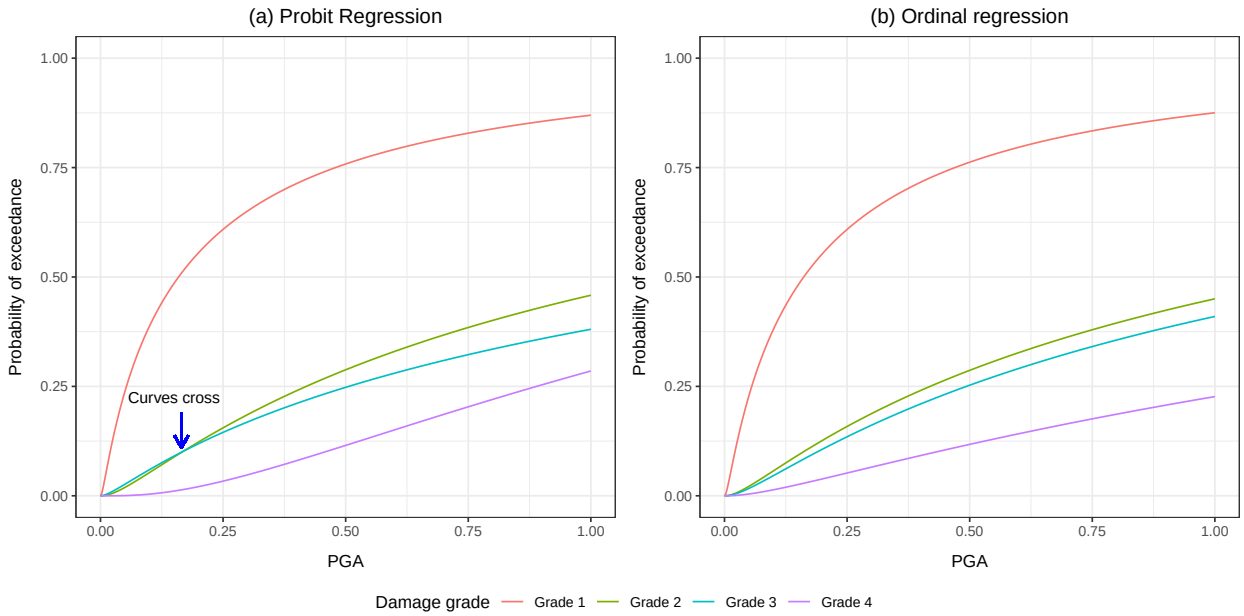

Figure 4: Comparing the fragility curves fitted to Dataset 4 via (a) probit and (b) ordinal regressions.