Nepal timber analysis

Michele Nguyen

23/8/2020

Introduction

In this report, we thin the Nepal damage data so as to compare how fragility curves fitted with ordinal regression and separate probit regressions differ with different amounts of available data. We focus on the results for buildings with the timber superstructure.

The purpose is to illustrate the advantages of borrowing information across damage states and the PGA range. The expected benefits from an ordinal regression include:

- fragility curves which automatically do not overlap between damage states;
- and lower uncertainty in curves because of the use of more data.

```
# Read in Nepal buildings category data:

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

# Thinning proportions (20%-100% of data):
thin_prop <- seq(0.2, 1, by = 0.2)
PGA_list <- c((exp(min(ordinal_data$logPGA))/50)*(1:50),
              exp(min(ordinal_data$logPGA)) +
                ((exp(max(ordinal_data$logPGA))-exp(min(ordinal_data$logPGA)))/100)*(1:200))
# First 49 are extrapolation into lower values, next 101 correspond to data range.
# Following 100 are extrapolation.
estimates <- c("Slope", "Grade 0|Grade 1", "Grade 1|Grade 2", "Grade 2|Grade 3",
               "Grade 3|Grade 4", "Grade 4|Grade 5")
superstructures <- c("mud_mortar_stone", "cement_mortar_brick", "timber")
```

```
# Do the thinning experiment for each superstructure type:
thinning_results <- vector("list", length(superstructures))
names(thinning_results) <- superstructures

for (j in 1:length(superstructures)){

  superstructure_data <- ordinal_data[ordinal_data$superstructure == superstructures[j], ]
  superstructure_no <- nrow(superstructure_data)
  superstructure_se <- data.frame("Model" = NA, "Estimate" = NA, "Mean" = NA, "Standard_Error" = NA,
                                  "Data_percentage" = NA)
  superstructure_ci <- data.frame("Model" = NA, "PGA" = NA, "damage_grade" = NA, "Mean" = NA,
                                  "CI_Upper" = NA, "CI_Lower" = NA, "CI_width" = NA,
                                  "Data_percentage" = NA)

  for (i in 1:length(thin_prop)){

    set.seed(i)
    thinned_data <- superstructure_data %>% group_by(damage_grade) %>% sample_frac(thin_prop[i])
```

```r
thinned_data$damage_grade <-  ordered(thinned_data$damage_grade,
                                      levels = c("Grade 0", "Grade 1",
                                                 "Grade 2", "Grade 3",
                                                 "Grade 4", "Grade 5"))

## Ordinal model ##

ordinal_model <- polr(damage_grade ~ logPGA, data = thinned_data,
                      method = "probit", Hess = TRUE)
ordinal_vcov <- vcov(ordinal_model)

temp_se <- data.frame("Model" = rep("Ordinal", length(estimates)), "Estimate" = estimates,
                      "Mean" = c(ordinal_model$coefficients, ordinal_model$zeta),
                      "Standard_Error" = sqrt(diag(ordinal_vcov)),
                      "Data_percentage" = rep(thin_prop[i], length(estimates)))
superstructure_se <- rbind(superstructure_se, temp_se)

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.
lp_pred$b0_id <- as.numeric(lp_pred$damage_grade)
lp_pred$b0 <- ordinal_model$zeta[lp_pred$b0_id]

lp_mean <- lp_pred$b0 - ordinal_model$coefficients*lp_pred$logPGA
lp_se <- sqrt(diag(ordinal_vcov)[1+lp_pred$b0_id] + diag(ordinal_vcov)[1]*(lp_pred$logPGA^2)
              - 2*lp_pred$logPGA*ordinal_vcov[1, ][1+lp_pred$b0_id])
ordinal_mean <- pnorm(lp_mean, lower.tail = TRUE) # Currently, 1- exceedance probability.
temp_upper <- pnorm(qnorm(0.975)*lp_se + lp_mean)
temp_lower <- pnorm(qnorm(0.025)*lp_se + lp_mean)
# Convert to fragility curve scale: Exceedance probabilities
ordinal_mean <- 1 - ordinal_mean; ordinal_upper <- 1 - temp_lower;
ordinal_lower <- 1 - temp_upper
ordinal_width <- ordinal_upper - ordinal_lower

temp_ci <- data.frame("Model" = rep("Ordinal", (length(estimates)-1)*length(PGA_list)),
                      "PGA" = rep(PGA_list, length(estimates)-1),
                      "damage_grade" = rep(c("Grade 0", "Grade 1", "Grade 2",
                                             "Grade 3","Grade 4"), each = length(PGA_list)),
                      "Mean" = ordinal_mean, "CI_Upper" = ordinal_upper,
                      "CI_Lower" = ordinal_lower, "CI_width" = ordinal_width,
                      "Data_percentage" = rep(thin_prop[i],
                                              (length(estimates)-1)*length(PGA_list)))
superstructure_ci <- rbind(superstructure_ci, temp_ci)

## Probit models ##

superstructure_data_05 <- thinned_data
superstructure_data_05$Damage <- 1 # Success = exceed state
superstructure_data_05$Damage[superstructure_data_05$damage_grade == "Grade 0"] <- 0
multinom_05 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_05)
multinom_05_vcov <- vcov(multinom_05)
```

```r
superstructure_data_15 <- thinned_data
superstructure_data_15$Damage <- 1
superstructure_data_15$Damage[superstructure_data_15$damage_grade %in%
                                c("Grade 0", "Grade 1")] <- 0
multinom_15 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_15)
multinom_15_vcov <- vcov(multinom_15)

superstructure_data_25 <- thinned_data
superstructure_data_25$Damage <- 1
superstructure_data_25$Damage[superstructure_data_25$damage_grade %in%
                                c("Grade 0", "Grade 1", "Grade 2")] <- 0
multinom_25 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_25)
multinom_25_vcov <- vcov(multinom_25)

superstructure_data_35 <- thinned_data
superstructure_data_35$Damage <- 1
superstructure_data_35$Damage[superstructure_data_35$damage_grade %in%
                                c("Grade 0", "Grade 1", "Grade 2", "Grade 3")] <- 0
multinom_35 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_35)
multinom_35_vcov <- vcov(multinom_35)

superstructure_data_45 <- thinned_data
superstructure_data_45$Damage <- 1
superstructure_data_45$Damage[superstructure_data_45$damage_grade %in%
                                c("Grade 0", "Grade 1", "Grade 2", "Grade 3", "Grade 4")] <- 0
multinom_45 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_45)
multinom_45_vcov <- vcov(multinom_45)

temp_se <- data.frame("Model" = rep(c("Probit_0", "Probit_1", "Probit_2",
                                      "Probit_3", "Probit_4"), 2),
                      "Estimate" = c(rep(estimates[1], 4), estimates),
                      "Mean" = c(multinom_05$coefficients["logPGA"],
                                 multinom_15$coefficients["logPGA"],
                                 multinom_25$coefficients["logPGA"],
                                 multinom_35$coefficients["logPGA"],
                                 multinom_45$coefficients["logPGA"],
                                 -multinom_05$coefficients[1],
                                 -multinom_15$coefficients[1],
                                 -multinom_25$coefficients[1],
                                 -multinom_35$coefficients[1],
                                 -multinom_45$coefficients[1]),
                      "Standard_Error" = sqrt(c(diag(multinom_05_vcov)["logPGA"],
                                               diag(multinom_15_vcov)["logPGA"],
                                               diag(multinom_25_vcov)["logPGA"],
                                               diag(multinom_35_vcov)["logPGA"],
                                               diag(multinom_45_vcov)["logPGA"],
                                               diag(multinom_05_vcov)[1],
                                               diag(multinom_15_vcov)[1],
                                               diag(multinom_25_vcov)[1],
```

```
                                             diag(multinom_35_vcov)[1],
                                             diag(multinom_45_vcov)[1])),
                          "Data_percentage" = rep(thin_prop[i], (length(estimates)-1)*2))
    superstructure_se <- rbind(superstructure_se, temp_se)

    temp_se <- data.frame("Model" = rep(c("Probit_0", "Probit_1", "Probit_2",
                                          "Probit_3", "Probit_4"), 2),
                          "Estimate" = c(rep(estimates[1], 4), estimates),
                          "Mean" = c(multinom_05$coefficients["logPGA"],
                                     multinom_15$coefficients["logPGA"],
                                     multinom_25$coefficients["logPGA"],
                                     multinom_35$coefficients["logPGA"],
                                     multinom_45$coefficients["logPGA"],
                                     -multinom_05$coefficients[1],
                                     -multinom_15$coefficients[1],
                                     -multinom_25$coefficients[1],
                                     -multinom_35$coefficients[1],
                                     -multinom_45$coefficients[1]),
                          "Standard_Error" = sqrt(c(diag(multinom_05_vcov)["logPGA"],
                                             diag(multinom_15_vcov)["logPGA"],
                                             diag(multinom_25_vcov)["logPGA"],
                                             diag(multinom_35_vcov)["logPGA"],
                                             diag(multinom_45_vcov)["logPGA"],
                                             diag(multinom_05_vcov)[1],
                                             diag(multinom_15_vcov)[1],
                                             diag(multinom_25_vcov)[1],
                                             diag(multinom_35_vcov)[1],
                                             diag(multinom_45_vcov)[1])),
                          "Data_percentage" = rep(thin_prop[i], (length(estimates)-1)*2))
    superstructure_se <- rbind(superstructure_se, temp_se)

    lp_mean <- c(predict.glm(multinom_05, newdata = data.frame("logPGA" = log(PGA_list)),
                             type = "link"),
                 predict.glm(multinom_15, newdata = data.frame("logPGA" = log(PGA_list)),
                             type = "link"),
                 predict.glm(multinom_25, newdata = data.frame("logPGA" = log(PGA_list)),
                             type = "link"),
                 predict.glm(multinom_35, newdata = data.frame("logPGA" = log(PGA_list)),
                             type = "link"),
                 predict.glm(multinom_45, newdata = data.frame("logPGA" = log(PGA_list)),
                             type = "link"))

    lp_se <- sqrt(rep(c(diag(multinom_05_vcov)[1], diag(multinom_15_vcov)[1],
                        diag(multinom_25_vcov)[1],
                        diag(multinom_35_vcov)[1], diag(multinom_45_vcov)[1]),
                      each = length(PGA_list))
                  + rep(c(diag(multinom_05_vcov)[2], diag(multinom_15_vcov)[2],
                          diag(multinom_25_vcov)[2],
                          diag(multinom_35_vcov)[2], diag(multinom_45_vcov)[2]),
                        each = length(PGA_list))*(lp_pred$logPGA^2)
                  - 2*lp_pred$logPGA*rep(c(multinom_05_vcov[1, ][2], multinom_15_vcov[1, ][2],
                                           multinom_25_vcov[1, ][2], multinom_35_vcov[1, ][2],
                                           multinom_45_vcov[1, ][2]), each = length(PGA_list)))
```

```r
    multinom_mean <- pnorm(lp_mean, lower.tail = TRUE) # Currently, exceedance probability.
    temp_upper <- pnorm(qnorm(0.975)*lp_se + lp_mean)
    temp_lower <- pnorm(qnorm(0.025)*lp_se + lp_mean)
    multinom_upper <- temp_upper; multinom_lower <- temp_lower
    multinom_width <- multinom_upper - multinom_lower

    temp_ci <- data.frame("Model" = rep(c("Probit_0", "Probit_1", "Probit_2", "Probit_3", "Probit_4"),
                                   each = length(PGA_list)),
                          "PGA" = rep(PGA_list, length(estimates)-1),
                          "damage_grade" = rep(c("Grade 0", "Grade 1", "Grade 2",
                                                 "Grade 3","Grade 4"), each = length(PGA_list)),
                          "Mean" = multinom_mean, "CI_Upper" = multinom_upper,
                          "CI_Lower" = multinom_lower, "CI_width" = multinom_width,
                          "Data_percentage" = rep(thin_prop[i],
                                                  (length(estimates)-1)*length(PGA_list)))
    superstructure_ci <- rbind(superstructure_ci, temp_ci)

    print(paste("Superstructure ", j, "/", length(superstructures), ", Round ", i,
                "/", length(thin_prop), " done.", sep = ""))

  }

  # Remove dummy first rows.
  superstructure_se <- superstructure_se[-1, ]
  superstructure_ci <- superstructure_ci[-1, ]
  thinning_results[[j]] <- list("se" = superstructure_se, "ci" = superstructure_ci)

}

save(thinning_results, file = "D:/Documents/Ordinal_Fragility_Curves/Data/thinning_results.RData")
```

Results for buildings with the timber superstructure

Figure 1 compares the standard errors of the model estimates from the ordinal and separate probit regressions and how they vary according to the amount of data used for model fitting for the timber superstructure. We see that the ordinal model consistently provides estimates with lower standard errors and the standard errors decrease when more data is used for model fitting as expected.

```r
timber_se <- thinning_results[[3]][[1]]
timber_plot_1 <- list()

for (i in 1:length(estimates)){
  se_subset <- timber_se[timber_se$Estimate == estimates[i], ]
  upper_lim <- quantile(se_subset$Standard_Error, 0.75)*1.75
  timber_plot_1[[i]] <- ggplot(data = se_subset) +
    geom_line(aes(x = Data_percentage, y = Standard_Error, color = Model)) +
    ylim(0, upper_lim) + ggtitle(estimates[i]) +
    labs(y = "Standard error", x = "Data proportion") + theme_bw() +
    theme(plot.title = element_text(hjust = 0.5))
}

do.call("grid.arrange", c(timber_plot_1, ncol = 3))
```

```
## Warning: Removed 10 rows containing missing values (geom_path).
```

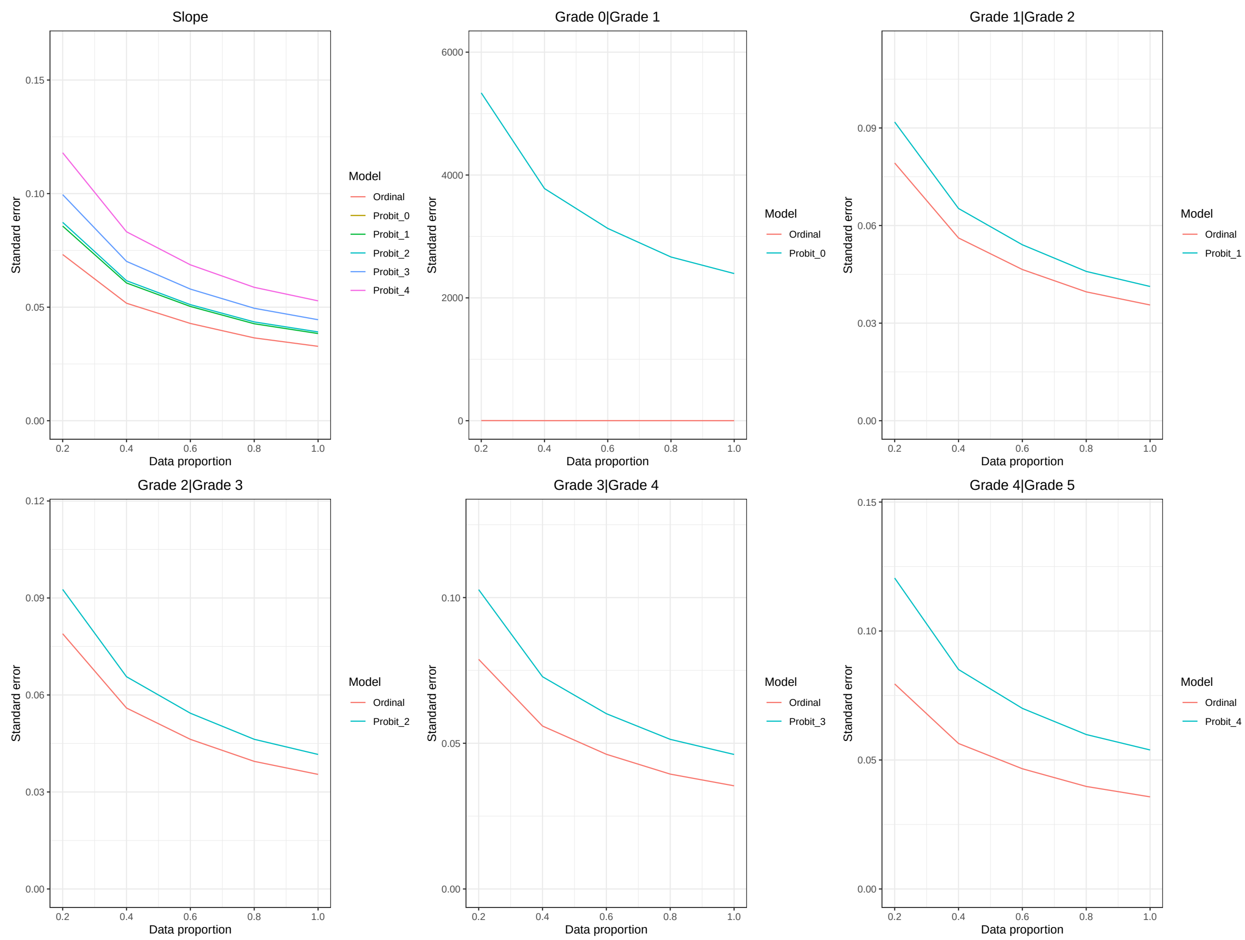

Figure 1: Timber: Standard errors of the model estimates against the proportion of data used for model fitting. Note that the results for Probit_0 (Grade 0 against Grades 1-5) are not included for the first plot because the standard errors are too large compared to the rest.

Figure 2 shows the widths of the 95% confidence intervals (CIs) at selected PGA values of the fitted fragility curves for the timber superstructure. We see that the CIs corresponding to the ordinal model are narrower than that obtained from the separate probit regressions at each selected PGA value. As expected, the CI widths decrease with increasing amounts of data used for fitting.

```r
timber_ci <- thinning_results[[3]][[2]]
selected_PGA <- PGA_list[49 + c(26, 51, 76)]
damage_grades <- c("Grade 0", "Grade 1", "Grade 2", "Grade 3", "Grade 4", "Grade 5")

timber_plot_2 <- list()

for (i in 1:(length(damage_grades)-1)){

  ci_subset <- timber_ci[timber_ci$damage_grade == damage_grades[i] & timber_ci$PGA %in%
                          selected_PGA, ]
  ci_subset$PGA <- as.character(round(ci_subset$PGA, 3))
  upper_lim <- quantile(ci_subset$CI_width, 0.75)*2
```

```
  timber_plot_2[[i]] <- ggplot(data = ci_subset) +
    geom_line(aes(x = Data_percentage, y = CI_width, lty = Model, color = PGA)) +
    ggtitle(paste(damage_grades[i], "exceedance")) +
    labs(y = "CI width", x = "Data proportion") + theme_bw() +
        theme(plot.title = element_text(hjust = 0.5),
          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), size = guide_legend(order = 2)) +
    coord_cartesian(ylim = c(0, upper_lim))

}

do.call("grid.arrange", c(timber_plot_2, ncol = 3))
```

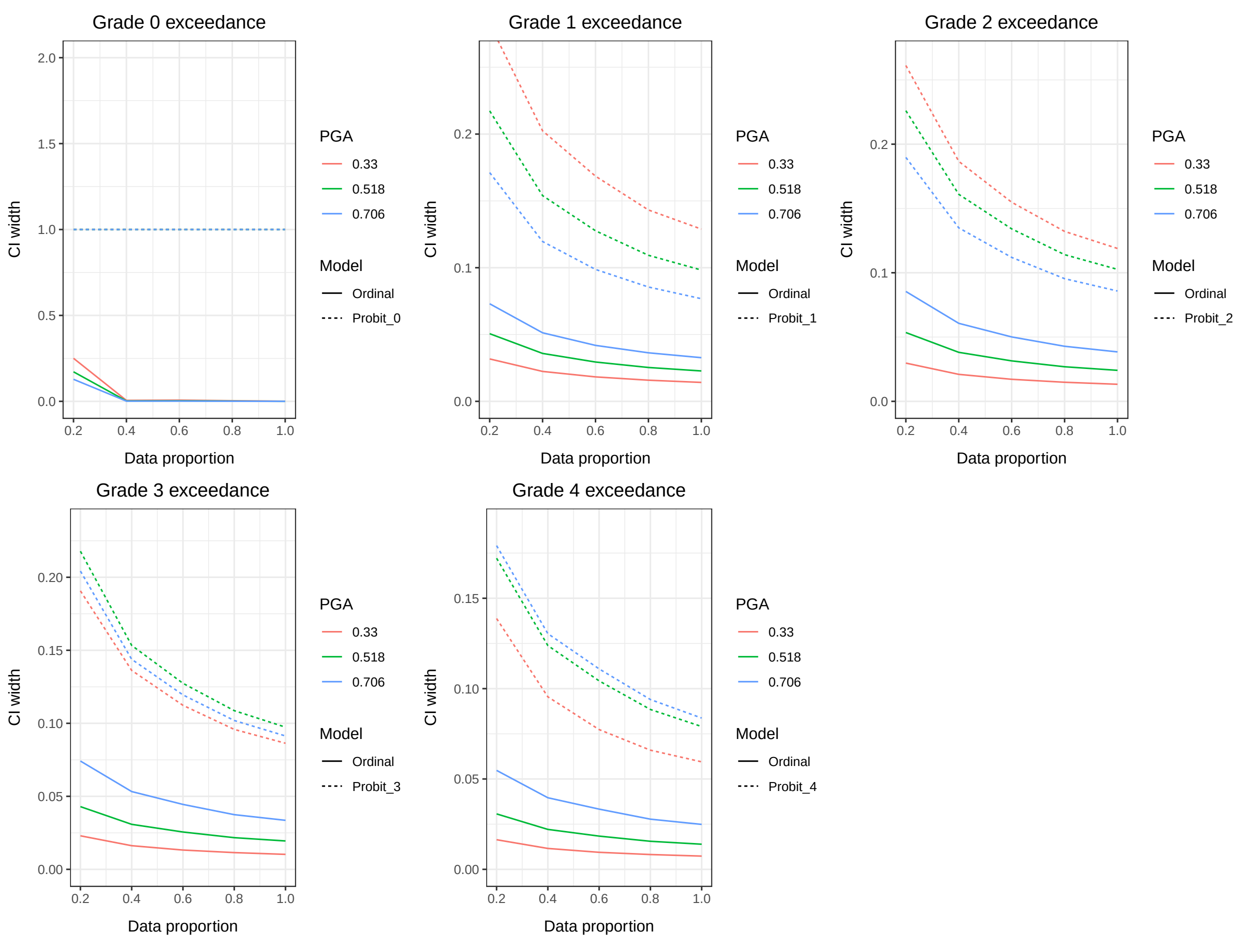

Figure 2: Timber: 95% confidence interval (CI) widths of the fitted fragility curves at selected peak ground acceleration (PGA) values against the proportion of data used for model fitting.

Figure 3 shows the fragility curves obtained from fitting the ordinal and separate probit regressions to different proportions of the data. From the plots corresponding to the probit regressions on the right column, we see that the CIs narrow with increasing data as mentioned earlier. We also see that the ordinal model gives narrower CIs than the separate probit regressions in general. When we extend the PGA range, we see that the fragility curves from the probit regressions which correspond to the exceedance of damage states 2 and

3 cross while those from the ordinal regression remain distinct.

```
selected_data_prop <- c(0.2, 0.6, 1)

timber_plot_3 <- list()

for (i in 1:length(selected_data_prop)){

  ci_subset <- timber_ci[timber_ci$Data_percentage > selected_data_prop[i]-0.01 &
                          timber_ci$Data_percentage < selected_data_prop[i]+0.01, ]

  # First 49 PGA are extrapolation into lower values, next 101 correspond to data range. Following 100

  timber_plot_3[[(2*(i-1)+1)]] <- ggplot(data = ci_subset[ci_subset$Model == "Ordinal", ]) +
    geom_line(aes(x = PGA, y = Mean, color = damage_grade)) + ylim(0, 1) +
    ggtitle(paste("Data proportion = ", selected_data_prop[i], ", Ordinal", sep = "")) +
    labs(y = "Probability of exceedance", x = "PGA", color = "Damage grade", fill = "95% CI") +
    theme_bw() +
    theme(plot.title = element_text(hjust = 0.5),
          axis.title.x = element_text(margin=margin(10,0,0,0)),
          axis.title.y = element_text(margin=margin(0,10,0,0))) +
    geom_ribbon(aes(x = PGA, ymin = CI_Lower, ymax = CI_Upper, fill = damage_grade),
                alpha = 0.25, show.legend = FALSE) +
    geom_vline(aes(xintercept = PGA_list[50]), lty = 2) +
    geom_vline(aes(xintercept = PGA_list[150]), lty = 2)

  timber_plot_3[[2*i]] <- ggplot(data = ci_subset[ci_subset$Model != "Ordinal", ]) +
    geom_line(aes(x = PGA, y = Mean, color = damage_grade)) + ylim(0, 1) +
    ggtitle(paste("Data proportion = ", selected_data_prop[i], ", Probit", sep = "")) +
    labs(y = "Probability of exceedance", x = "PGA", color = "Damage grade", fill = "95% CI") +
    theme_bw() +
    theme(plot.title = element_text(hjust = 0.5),
          axis.title.x = element_text(margin=margin(10,0,0,0)),
          axis.title.y = element_text(margin=margin(0,10,0,0))) +
    geom_ribbon(aes(x = PGA, ymin = CI_Lower, ymax = CI_Upper, fill = damage_grade),
                alpha = 0.25, show.legend = FALSE) +
    geom_vline(aes(xintercept = PGA_list[50]), lty = 2) +
    geom_vline(aes(xintercept = PGA_list[150]), lty = 2)

  if(i == 1){
    timber_plot_3[[2*i]] <- timber_plot_3[[2*i]] +
      geom_segment(mapping=aes(x=1.225, y=0.5, xend=1.225, yend=0.6),
                   arrow=arrow(length = unit(0.125, "inches")), size=0.75, color="blue") +
      annotate("text", x = 1.25, y = 0.475, label = "Curves cross", size = 3.5)
  }

}

grid_arrange_shared_legend(timber_plot_3[[1]], timber_plot_3[[2]], timber_plot_3[[3]],
                          timber_plot_3[[4]], timber_plot_3[[5]], timber_plot_3[[6]], nrow = 3, ncol =
```

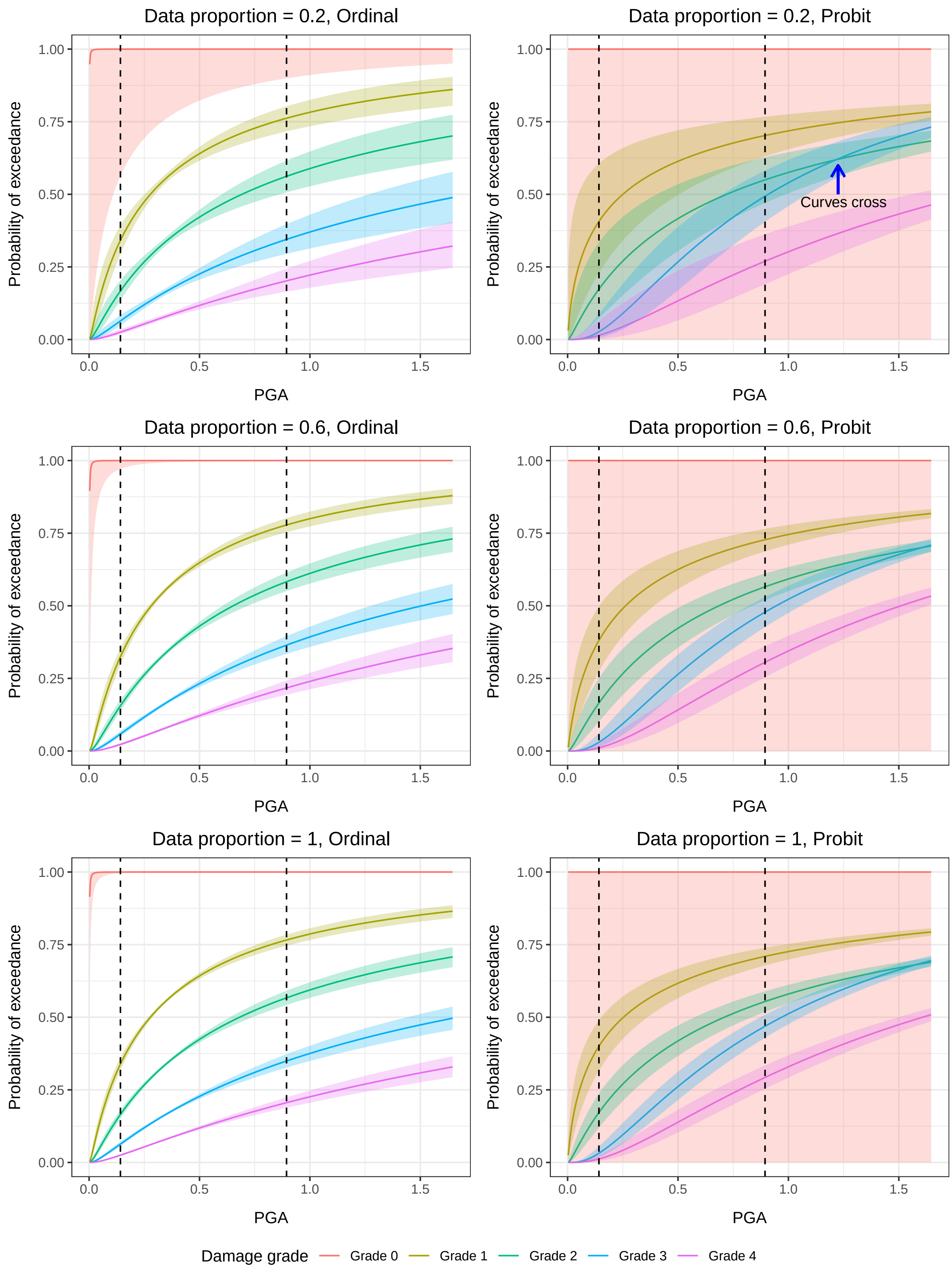

Figure 3: Timber: Fragility curves obtained from fitting ordinal and separate probit regressions to 20%, 60% and 100% of the data. The coloured bands correspond to the 95% confidence intervals (CIs). The black vertical dotted lines denote the PGA range covered in the data.

Porter method II:

In this subsection, we implement the second method mentioned by Porter et al. (2007) for fixing crossing fragility curves. This involves adjusting the slope and cutoff estimates. We use the thinned data for the timber superstructure (data proportion = 0.2) and calculate the Akaike Information Criterion (AIC) values for the ordinal model and the adjusted probit model for the exceedance of Grades 1-4. Grade 0 was omitted because there are no corresponding buildings and we cannot adjust the model estimates sensibly using the method suggested by Porter et al.

```
# Original model estimates:
superstructure_data <- ordinal_data[ordinal_data$superstructure == "timber", ]
superstructure_no <- nrow(superstructure_data)

i <- 1 # Data proportion = 0.2:
set.seed(i)
thinned_data <- superstructure_data %>% group_by(damage_grade) %>% sample_frac(thin_prop[i])
thinned_data$damage_grade <-  ordered(thinned_data$damage_grade,
                                      levels = c("Grade 0", "Grade 1",
                                                 "Grade 2", "Grade 3",
                                                 "Grade 4", "Grade 5"))

## Ordinal model ##

ordinal_model <- polr(damage_grade ~ logPGA, data = thinned_data,
                          method = "probit", Hess = TRUE)
```

```
# Likelihood Ratio Test and AIC based on ordinal likelihood (polr):

# Function to compute the ordinal likelihood by hand:
ordinal_nll <- function(betas, cutoffs, damage_grades, data){

  master.logPGA.list <- sort(unique(data$logPGA), decreasing = FALSE)

  nll <- 0

  for (i in 1:length(master.logPGA.list)){
    # i <- 1
    data.subset <- data[data$logPGA == master.logPGA.list[i], ]
    temp_nll <- 0

    for (j in 1:(length(cutoffs)+1)){
      # j <- 1
      n_j <- sum(data.subset$damage_grade == damage_grades[j])
      if (j==1){
        damage_prob <- pnorm(cutoffs[j] - betas[j]*master.logPGA.list[i], lower.tail = TRUE)
      }else{
        if(j==(length(cutoffs)+1)){
          damage_prob <- pnorm(cutoffs[j-1] - betas[j-1]*master.logPGA.list[i], lower.tail = FALSE)
        }else{
          lower_prob <- pnorm(cutoffs[j-1] - betas[j-1]*master.logPGA.list[i], lower.tail = TRUE)
          upper_prob <- pnorm(cutoffs[j] - betas[j]*master.logPGA.list[i], lower.tail = TRUE)
          damage_prob <- upper_prob - lower_prob
        }
      }
```

```r
      temp_nll <- temp_nll - n_j*log(damage_prob)
    }

    nll <- nll + temp_nll
  }
  return(nll)
}
```

```r
## Probit models ##

superstructure_data_05 <- thinned_data
superstructure_data_05$Damage <- 1 # Success = exceed state
superstructure_data_05$Damage[superstructure_data_05$damage_grade == "Grade 0"] <- 0
multinom_05 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_05)
```

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

```r
superstructure_data_15 <- thinned_data
superstructure_data_15$Damage <- 1
superstructure_data_15$Damage[superstructure_data_15$damage_grade %in%
                                c("Grade 0", "Grade 1")] <- 0
multinom_15 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_15)

superstructure_data_25 <- thinned_data
superstructure_data_25$Damage <- 1
superstructure_data_25$Damage[superstructure_data_25$damage_grade %in%
                                c("Grade 0", "Grade 1", "Grade 2")] <- 0
multinom_25 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_25)

superstructure_data_35 <- thinned_data
superstructure_data_35$Damage <- 1
superstructure_data_35$Damage[superstructure_data_35$damage_grade %in%
                                c("Grade 0", "Grade 1", "Grade 2", "Grade 3")] <- 0
multinom_35 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_35)

superstructure_data_45 <- thinned_data
superstructure_data_45$Damage <- 1
superstructure_data_45$Damage[superstructure_data_45$damage_grade %in%
                              c("Grade 0", "Grade 1", "Grade 2", "Grade 3", "Grade 4")] <- 0
multinom_45 <- glm(Damage ~ logPGA, family = binomial(link = "probit"),
                   data = superstructure_data_45)
```

```r
# Adjust model estimates for damage states 1-4 only (model for state 0 is a dummy):

bj <- 1/c(multinom_15$coefficients["logPGA"], multinom_25$coefficients["logPGA"],
          multinom_35$coefficients["logPGA"],
          multinom_45$coefficients["logPGA"])
bj_prime <- mean(bj)
```

```r
xmj <- exp(-c(multinom_15$coefficients[1], multinom_25$coefficients[1],
          multinom_35$coefficients[1],
          multinom_45$coefficients[1])*bj)
xmj_prime <- exp(1.28*(bj_prime - bj) + log(xmj))



multinom_15_prime <- multinom_15; multinom_25_prime <- multinom_25; multinom_35_prime <- multinom_35;
multinom_45_prime <- multinom_45;
multinom_15_prime$coefficients["logPGA"] <- multinom_25_prime$coefficients["logPGA"] <-
multinom_35_prime$coefficients["logPGA"] <- multinom_45_prime$coefficients["logPGA"] <- 1/bj_prime
multinom_15_prime$coefficients[1] <- -log(xmj_prime[1])/bj_prime
multinom_25_prime$coefficients[1] <- -log(xmj_prime[2])/bj_prime
multinom_35_prime$coefficients[1] <- -log(xmj_prime[3])/bj_prime
multinom_45_prime$coefficients[1] <- -log(xmj_prime[4])/bj_prime

lp_mean <- c(predict.glm(multinom_15_prime, newdata = data.frame("logPGA" = log(PGA_list)),
                         type = "link"),
             predict.glm(multinom_25_prime, newdata = data.frame("logPGA" = log(PGA_list)),
                         type = "link"),
             predict.glm(multinom_35_prime, newdata = data.frame("logPGA" = log(PGA_list)),
                         type = "link"),
             predict.glm(multinom_45_prime, newdata = data.frame("logPGA" = log(PGA_list)),
                         type = "link"))
multinom_mean <- pnorm(lp_mean, lower.tail = TRUE) # Currently, exceedance probability.
```

```r
# Plot adjusted curves vs original:

temp_curve <- data.frame("Model" = rep("Adjusted_Probit", length(PGA_list)*(length(estimates)-2)),
                      "PGA" = rep(PGA_list, length(estimates)-2),
                      "damage_grade" = rep(c("Grade 1", "Grade 2",
                                           "Grade 3","Grade 4"), each = length(PGA_list)),
                      "Mean" = multinom_mean)

ci_subset <- timber_ci[timber_ci$Data_percentage > 0.2-0.01 & timber_ci$Data_percentage < 0.2+0.01
                       & timber_ci$Model != "Ordinal" & timber_ci$damage_grade != "Grade 0",
                       colnames(temp_curve)]
ci_subset$Model <- "Probit"
curve_data <- rbind(ci_subset, temp_curve)

ggplot(data = curve_data) + geom_line(aes(x = PGA, y = Mean, color = damage_grade, lty = Model)) +
  ylim(0, 1) + ggtitle(paste("Data proportion = 0.2:", " Adjusted Probit", sep = "")) +
  labs(y = "Probability of exceedance", x = "PGA", color = "Damage grade", fill = "95% CI") +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5),
        axis.title.x = element_text(margin=margin(10,0,0,0)),
        axis.title.y = element_text(margin=margin(0,10,0,0))) +
  geom_hline(aes(yintercept = 0.1), lty = 2)
```

Figure 4 shows the estimated fragility curves from the adjusted probit model and the original probit-derived curves. As intended, the new curves match the original ones at 10% failure probability. The AIC of the combined adjusted probit models is $12995.06 + 5 = 13000.06 > 11246.25 + 5 = 11251.25$, the AIC of the ordinal model. This means that the ordinal model provides a better fit to the data.

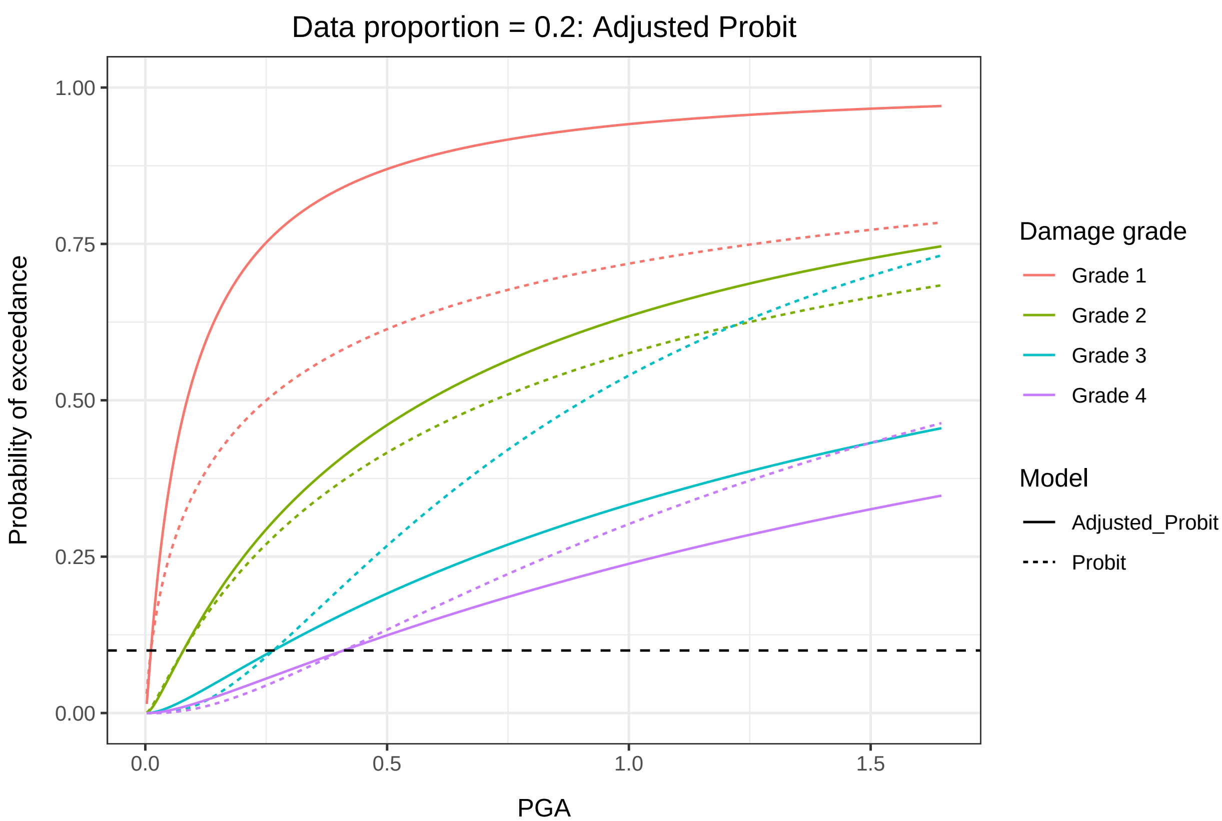

Figure 4: Timber: Fragility curves for Grades 1-4 obtained from the adjusted and original probit regressions. The horiztonal black dotted line denotes 10% failure probability.

```r
# Exclude Grade 0.
dev_porter <- 2*ordinal_nll(betas = c(multinom_15_prime$coefficients[2],
                                       multinom_25_prime$coefficients[2],
                                       multinom_35_prime$coefficients[2],
                                       multinom_45_prime$coefficients[2]),
                             cutoffs = -c(multinom_15_prime$coefficients[1],
                                          multinom_25_prime$coefficients[1],
                                          multinom_35_prime$coefficients[1],
                                          multinom_45_prime$coefficients[1]),
                             damage_grades = c("Grade 1", "Grade 2",
                                               "Grade 3", "Grade 4", "Grade 5"),
                             data = thinned_data)

dev_ordinal <- 2*ordinal_nll(betas = rep(ordinal_model$coefficients, 4),
                              cutoffs = ordinal_model$zeta[-1],
                              damage_grades = c("Grade 1", "Grade 2",
                                                "Grade 3", "Grade 4", "Grade 5"),
                              data = thinned_data)

#Deviance: Porter = 12995.06; Ordinal = 11246.25.
```