Code Appendix

```
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# #############################     Libraries    ##############################
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

## General
library(tidyverse)
library(kableExtra)
library(latex2exp)
library(scales)

## Methods
library(nloptr)
```

```
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# #####################   Direct Discrimination Example   #####################
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

set.seed(74)

## Generating Biased Data
generate_data <- function(n, d_effect){

  # E[X] = a_x
  a_x = 0
  X <- rnorm(n, a_x, 1)

  # P(A|X)
  b_0 = 0.5
  b_x = 0.5
  p_A1 <- exp(b_0 + b_x*X)/(1 + exp(b_0 + b_x*X))
  A <- rbinom(n, size=1, prob=p_A1)

  # P(M|A,X)
  c_0 = 0.5
  c_x = 1
  c_a = 0.5
  p_M1 <- exp(c_0 + c_x*X + c_a*A)/(1 + exp(c_0 + c_x*X + c_a*A))
  M <- rbinom(n, size=1, prob=p_M1)

  # P(Y|A,M,X)
  d_0 = 1
  d_x = 1
  d_a = d_effect # direct discrimination effect
  d_m = 1
  eps <- rnorm(n, 0, 1)
  Y <- d_0 + d_x*X + d_a*A + d_m*M + eps

  # Data
  dat = data.frame(1, X, A, M, Y)
  colnames(dat) = c("(Intercept)", "X", "A", "M", "Y")
```

```r
  # Coefficients
  beta = c(b_0, b_x,
           c_0, c_x, c_a,
           d_0, d_x, d_a, d_m)
  names(beta) = c("A_0", "A_X",
                  "M_0", "M_X", "M_A",
                  "Y_0", "Y_X", "Y_A", "Y_M")

  return(list(data=dat,
              beta=beta))
}

## Data Processing (Interventions)
process_data <- function(dat, a, m){
  dat$A = a
  dat$M = m
  return(dat)
}

## Objective Function
neg_log_lik <- function(beta, data){

  # add names
  names(beta) = c("A_0", "A_X",
                  "M_0", "M_X", "M_A",
                  "Y_0", "Y_X", "Y_A", "Y_M")

  # estimators
  exp_A = exp(beta["A_0"]+data$X*beta["A_X"])
  exp_M = exp(beta["M_0"]+data$X*beta["M_X"]+data$A*beta["M_A"])
  A_hat = exp_A/(1+exp_A) # f(X)
  M_hat = exp_M/(1+exp_M) # f(X, A)
  Y_hat = beta["Y_0"]+data$X*beta["Y_X"]+data$A*beta["Y_A"]+data$M*beta["Y_M"] # f(X, A, M)

  # densities
  p_A = data$A*A_hat + (1-data$A)*(1-A_hat) # Bernoulli PMF
  p_M = data$M*M_hat + (1-data$M)*(1-M_hat) # Bernoulli PMF
  p_Y = dnorm(data$Y, Y_hat, 1) # Normal PDF

  # objective function F (negative log-likelihood)
  nll = -sum(log(p_A) + log(p_M) + log(p_Y) + log(1/nrow(data))) # include px??

  return(nll)
}

## PSE Computation (NDE)
compute_NDE <- function(beta, data){

  # add names
  names(beta) = c("A_0", "A_X",
                  "M_0", "M_X", "M_A",
```

```r
                  "Y_0", "Y_X", "Y_A", "Y_M")

  # all combinations of a={0, 1}, m={0, 1}
  data_a0m0 = process_data(data, a=0, m=0)
  data_a0m1 = process_data(data, a=0, m=1)
  data_a1m0 = process_data(data, a=1, m=0)
  data_a1m1 = process_data(data, a=1, m=1)

  # all combinations of a={0, 1}, m=m
  data_a0mm = process_data(data, a=0, m=data$M)
  data_a1mm = process_data(data, a=1, m=data$M)

  # P(M|A,X)
  p_m1a0 = exp(beta["M_0"]+data_a0mm$X*beta["M_X"]+data_a0mm$A*beta["M_A"])
  p_m1a0 = p_m1a0/(1+p_m1a0)
  p_m0a0 = 1-p_m1a0
  p_m1a1 = exp(beta["M_0"]+data_a1mm$X*beta["M_X"]+data_a1mm$A*beta["M_A"])
  p_m1a1 = p_m1a1/(1+p_m1a1)
  p_m0a1 = 1-p_m1a1

  # P(Y|M,A,X)
  y_a0m0 = beta["Y_0"]+data_a0m0$X*beta["Y_X"]+data_a0m0$A*beta["Y_A"]+
    data_a0m0$M*beta["Y_M"]
  y_a1m0 = beta["Y_0"]+data_a1m0$X*beta["Y_X"]+data_a1m0$A*beta["Y_A"]+
    data_a1m0$M*beta["Y_M"]
  y_a0m1 = beta["Y_0"]+data_a0m1$X*beta["Y_X"]+data_a0m1$A*beta["Y_A"]+
    data_a0m1$M*beta["Y_M"]
  y_a1m1 = beta["Y_0"]+data_a1m1$X*beta["Y_X"]+data_a1m1$A*beta["Y_A"]+
    data_a1m1$M*beta["Y_M"]

  # NDE Comptation (G-Formula)
  NDE = sum(((y_a1m0-y_a0m0)*p_m0a0 + (y_a1m1-y_a0m1)*p_m1a0))/nrow(data)

  return(NDE)
}

## New Penalized Objective
eval_f <- function(beta, data, lambda){
  nll <- neg_log_lik(beta, data)
  penalty <- lambda*abs(compute_NDE(beta, data))
  return(nll+penalty)
}
```

```r
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# ####################   Simulating Unfair World Data   #####################
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

set.seed(74)

## Example for Direct Effect = 5
example_5 <- generate_data(n=1000, d_effect=5)
```

```r
## Simulated Dataset
data_5 <- example_5$data

## Initial Estimates
beta_start_5 <- example_5$beta

## Unfair Outcome Distributions by Group
data_5_dist <- ggplot(data_5, aes(x=Y, fill=as.factor(A))) +
  geom_density(alpha=0.7) +
  geom_vline(xintercept=mean(data_5[data_5$A==0,]$Y), lty=2, color="red") +
  geom_vline(xintercept=mean(data_5[data_5$A==1,]$Y), lty=2, color="red") +
  annotate("text", x=0.4, y=0.28,
           label=round(mean(data_5[data_5$A==0,]$Y), 2), color="black", size=4) +
  annotate("text", x=7.8, y=0.28,
           label=round(mean(data_5[data_5$A==1,]$Y), 2), color="black", size=4) +
  labs(x="Y", fill="A") +
  theme(plot.title=element_blank(),
        axis.title.y=element_blank(),
        axis.title.x=element_text(size=12),
        axis.text.y=element_text(size=13),
        axis.text.x=element_text(size=13),
        legend.title=element_text(size=12),
        legend.text=element_text(size=12),
        legend.title.align=0.25) +
  xlim(-5, 12)

## Discriminatory Effect Estimates
unfair_fit1 <- lm(Y ~ X + A + M, data=data_5) # adjusted for covariates (~ 4.98)
summary(unfair_fit1)
coef(unfair_fit1)["A"]
unfair_fit2 <- lm(Y ~ A, data=data_5) # unadjusting for covariates (~ 5.69)
summary(unfair_fit2)
coef(unfair_fit2)["A"]
```

```r
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# #################      Optimizing for Changing Lambda      #################
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

# set.seed(74)
#
# ## Changing Lambda
# lambda_sols <- function(beta_start, data){
#   lambda_vals <- seq(0, 1500, 10)
#   results <- data.frame(nde=rep(0, length(lambda_vals)),
#                         neg_log_lik=rep(0, length(lambda_vals)),
#                         lambda=lambda_vals)
#   for (i in 1:length(lambda_vals)){
#     res <- nloptr(x0=beta_start,
#                   eval_f=eval_f,
#                   opts=list("algorithm"="NLOPT_LN_COBYLA",
#                             "xtol_rel"=1.0e-8,
#                             "maxeval"=10000),
```

```r
#                    data=data,
#                    lambda=lambda_vals[i])
#     results$nde[i] <- compute_NDE(res$solution, data)
#     results$neg_log_lik[i] <- neg_log_lik(res$solution, data)
#   }
#   return(results)
# }
#
# ## Results for Changing Lambda
# lambda_results_5 <- lambda_sols(beta_start_5, data_5)
```

```r
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# ####################      Visualization for Lambda     ####################
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

## NLL vs. Lambda
results_5_plot1 <- ggplot(lambda_results_5) +
  geom_line(aes(x=lambda, y=neg_log_lik), color="purple") +
  geom_vline(xintercept=1090, color="red", lty=2) +
  annotate("text", x=990, y=12250,
           label=scales::comma(round(
             lambda_results_5[lambda_results_5$lambda==1090,]$neg_log_lik, 2)),
           color="black", size=4) +
  labs(title="A) Negative Log-Likelihood vs. Lambda",
       x=TeX(r"($\lambda$)"),
       y="Negative Log-Likelihood") +
  scale_y_continuous(labels=label_number(scale=1/1000, suffix="k")) +
  theme(axis.text.y=element_text(size=15),
        axis.text.x=element_text(size=14),
        axis.title.x=element_text(size=13),
        axis.title.y=element_text(size=13))

## NDE vs. Lambda
results_5_plot2 <- ggplot(lambda_results_5) +
  geom_line(aes(x=lambda, y=nde), color="deepskyblue") +
  geom_vline(xintercept=1090, color="red", lty=2) +
  annotate("text", x=980, y=0.05,
           label=format(round(lambda_results_5[lambda_results_5$lambda==1090,]$nde, 4),
                        scientific=FALSE),
           color="black", size=4) +
  labs(title="B) NDE vs. Lambda",
       x=TeX(r"($\lambda$)"),
       y="Natural Direct Effect") +
  theme(axis.text.y=element_text(size=17),
        axis.text.x=element_text(size=14),
        axis.title.x=element_text(size=13),
        axis.title.y=element_text(size=13))

## NLL vs. NDE
results_5_plot3 <- ggplot(lambda_results_5) +
  geom_line(aes(x=nde, y=neg_log_lik), color="black") +
  geom_hline(yintercept=lambda_results_5[lambda_results_5$lambda==1090,]$neg_log_lik,
```

```r
             color="red", lty=2) +
  annotate("text", x=0.15, y=12350,
           label=TeX(r"($\lambda$)"), color="red", size=4) +
  annotate("text", x=0.41, y=12350,
           label="=1,090", color="red", size=4) +
  labs(title="C) Negative Log-Likelihood vs. NDE",
       x="Natural Direct Effect",
       y="Negative Log-Likelihood") +
  scale_y_continuous(labels=label_number(scale=1/1000, suffix="k")) +
  theme(axis.text.y=element_text(size=15),
        axis.text.x=element_text(size=14),
        axis.title.x=element_text(size=13),
        axis.title.y=element_text(size=13))
```

```r
# +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# ##################      Optimization for Optimal Lambda    ###################
# +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

set.seed(74)

## Optimization with Lambda=1090
opt_res_5 <- nloptr(x0=beta_start_5,
                    eval_f=eval_f,
                    opts=list("algorithm"="NLOPT_LN_COBYLA",
                              "xtol_rel"=1.0e-8,
                              "maxeval"=10000),
                    data=data_5,
                    lambda=1090)

## Optimal Parameters
beta_opt_5 <- opt_res_5$solution
names(beta_opt_5) = c("A_0", "A_X",
                      "M_0", "M_X", "M_A",
                      "Y_0", "Y_X", "Y_A", "Y_M")

## Optimal Negative Log-Likelihood
NLL_start <- neg_log_lik(beta_start_5, data_5) # initial NLL: 9,520.069
NLL_opt <- neg_log_lik(beta_opt_5, data_5) # optimal NLL: 12,229.52

## Optimal NDE
NDE_start <- compute_NDE(beta_start_5, data_5) # initial NDE: 5
NDE_opt <- compute_NDE(beta_opt_5, data_5) # optimal NDE: 0.0006

# # same as:
# lambda_results_5[lambda_results_5$lambda==0,]$neg_log_lik
# lambda_results_5[lambda_results_5$lambda==1090,]$neg_log_lik
# lambda_results_5[lambda_results_5$lambda==0,]$nde
# lambda_results_5[lambda_results_5$lambda==1090,]$nde
```

```r
# +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# ####################      Sampling from Fair World     ####################
# +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```r
set.seed(74)

## Generating Fair Data
generate_fair_data <- function(n, beta_opt){

  beta <- as.vector(beta_opt)

  X <- data_5$X
  #X <- rep(0, n)

  # P(A|X)
  b_0 = beta[1]
  b_x = beta[2]
  p_A1 <- exp(b_0 + b_x*X)/(1 + exp(b_0 + b_x*X))
  A <- rbinom(n, size=1, prob=p_A1)

  # P(M|A,X)
  c_0 = beta[3]
  c_x = beta[4]
  c_a = beta[5] # indirect effect
  p_M1 <- exp(c_0 + c_x*X + c_a*A)/(1 + exp(c_0 + c_x*X + c_a*A))
  M <- rbinom(n, size=1, prob=p_M1)

  # P(Y|A,M,X)
  d_0 = beta[6]
  d_x = beta[7]
  d_a = beta[8] # direct discrimination effect
  d_m = beta[9]
  eps <- rnorm(n, 0, 1)
  Y <- d_0 + d_x*X + d_a*A + d_m*M + eps

  # Data
  dat = data.frame(1, X, A, M, Y)
  colnames(dat) = c("(Intercept)", "X", "A", "M", "Y")

  return(dat)
}

## Simulated Dataset
fair_data_5 <- generate_fair_data(n=1000, beta_opt=beta_opt_5)
```

```r
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
# ###########    Fair Prediction Distributions by Group (Plots)    ############
# ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

# Y_hat
fair_data_5_dist <- ggplot(fair_data_5, aes(x=Y, fill=as.factor(A))) +
  geom_density(alpha=0.7) +
  geom_vline(xintercept=mean(fair_data_5[fair_data_5$A==0,]$Y),
             lty=2, color="red") +
  geom_vline(xintercept=mean(fair_data_5[fair_data_5$A==1,]$Y),
```

```
             lty=2, color="red") +
  annotate("text", x=3.2, y=0.28,
           label=round(mean(fair_data_5[fair_data_5$A==0,]$Y), 2),
           color="black", size=4) +
  annotate("text", x=6.1, y=0.28,
           label=round(mean(fair_data_5[fair_data_5$A==1,]$Y), 2),
           color="black", size=4) +
  labs(x=TeX(r"($\hat{Y}$)"), fill="A") +
  theme(plot.title=element_blank(),
        axis.title.y=element_blank(),
        axis.title.x=element_text(size=12),
        axis.text.y=element_text(size=13),
        axis.text.x=element_text(size=13),
        legend.title=element_text(size=12),
        legend.text=element_text(size=12),
        legend.title.align=0.25) +
  xlim(-5, 12)

## Stratified Datasets wrt M and X
neg_X0 <- fair_data_5[fair_data_5$X<0 & fair_data_5$M==0,]
pos_X0 <- fair_data_5[fair_data_5$X>=0 & fair_data_5$M==0,]
neg_X1 <- fair_data_5[fair_data_5$X<0 & fair_data_5$M==1,]
pos_X1 <- fair_data_5[fair_data_5$X>=0 & fair_data_5$M==1,]

# Y_hat | M=0, X<0
neg_X0_dist <- ggplot(neg_X0, aes(x=Y, fill=as.factor(A))) +
  geom_density(alpha=0.7) +
  geom_vline(xintercept=mean(neg_X0[neg_X0$A==0,]$Y),
             lty=2, color="red") +
  geom_vline(xintercept=mean(neg_X0[neg_X0$A==1,]$Y),
             lty=2, color="red") +
  labs(x=TeX(r"($\hat{Y}|\hat{M}=0, X<0$)"), fill="A") +
  theme(plot.title=element_blank(),
        axis.title.y=element_blank(),
        axis.title.x=element_text(size=12),
        axis.text.y=element_text(size=13),
        axis.text.x=element_text(size=13),
        legend.title=element_text(size=12),
        legend.text=element_text(size=12),
        legend.title.align=0.25) +
  xlim(-5, 12) +
  scale_fill_brewer(palette="Accent", direction=-1)

# Y_hat | M=0, X>=0
pos_X0_dist <- ggplot(pos_X0, aes(x=Y, fill=as.factor(A))) +
  geom_density(alpha=0.7) +
  geom_vline(xintercept=mean(pos_X0[pos_X0$A==0,]$Y),
             lty=2, color="red") +
  geom_vline(xintercept=mean(pos_X0[pos_X0$A==1,]$Y),
             lty=2, color="red") +
  labs(x=TeX(r"($\hat{Y}|\hat{M}=0, X\geq 0$)"), fill="A") +
```

```
  theme(plot.title=element_blank(),
        axis.title.y=element_blank(),
        axis.title.x=element_text(size=12),
        axis.text.y=element_text(size=13),
        axis.text.x=element_text(size=13),
        legend.title=element_text(size=12),
        legend.text=element_text(size=12),
        legend.title.align=0.25) +
  xlim(-5, 12) +
  scale_fill_brewer(palette="Accent", direction=-1)

# Y_hat | M=1, X<0
neg_X1_dist <- ggplot(neg_X1, aes(x=Y, fill=as.factor(A))) +
  geom_density(alpha=0.7) +
  geom_vline(xintercept=mean(neg_X1[neg_X1$A==0,]$Y),
             lty=2, color="red") +
  geom_vline(xintercept=mean(neg_X1[neg_X1$A==1,]$Y),
             lty=2, color="red") +
  labs(x=TeX(r"($\hat{Y}|\hat{M}=1, X<0$)"), fill="A") +
  theme(plot.title=element_blank(),
        axis.title.y=element_blank(),
        axis.title.x=element_text(size=12),
        axis.text.y=element_text(size=13),
        axis.text.x=element_text(size=13),
        legend.title=element_text(size=12),
        legend.text=element_text(size=12),
        legend.title.align=0.25) +
  xlim(-5, 12) +
  scale_fill_brewer(palette="Accent", direction=-1)

# Y_hat | M=1, X>=0
pos_X1_dist <- ggplot(pos_X1, aes(x=Y, fill=as.factor(A))) +
  geom_density(alpha=0.7) +
  geom_vline(xintercept=mean(pos_X1[pos_X1$A==0,]$Y),
             lty=2, color="red") +
  geom_vline(xintercept=mean(pos_X1[pos_X1$A==1,]$Y),
             lty=2, color="red") +
  labs(x=TeX(r"($\hat{Y}|\hat{M}=1, X\geq 0$)"), fill="A") +
  theme(plot.title=element_blank(),
        axis.title.y=element_blank(),
        axis.title.x=element_text(size=12),
        axis.text.y=element_text(size=13),
        axis.text.x=element_text(size=13),
        legend.title=element_text(size=12),
        legend.text=element_text(size=12),
        legend.title.align=0.25) +
  xlim(-5, 12) +
  scale_fill_brewer(palette="Accent", direction=-1)

fair_data_5_dist
neg_X0_dist
```

```
pos_X0_dist
neg_X1_dist
pos_X1_dist
```

```
## Differences in Stratified Expected Prediction Values
mean(neg_X0[neg_X0$A==1,]$Y)-mean(neg_X0[neg_X0$A==0,]$Y)
mean(pos_X0[pos_X0$A==1,]$Y)-mean(pos_X0[pos_X0$A==0,]$Y)
mean(neg_X1[neg_X1$A==1,]$Y)-mean(neg_X1[neg_X1$A==0,]$Y)
mean(pos_X1[pos_X1$A==1,]$Y)-mean(pos_X1[pos_X1$A==0,]$Y)

# E[Y_hat|A=1,M,X] ~ 4.58
mean(c(mean(neg_X0[neg_X0$A==1,]$Y),
       mean(pos_X0[pos_X0$A==1,]$Y),
       mean(neg_X1[neg_X1$A==1,]$Y),
       mean(pos_X1[pos_X1$A==1,]$Y)))

# E[Y_hat|A=0,M,X] ~ 4.39
mean(c(mean(neg_X0[neg_X0$A==0,]$Y),
       mean(pos_X0[pos_X0$A==0,]$Y),
       mean(neg_X1[neg_X1$A==0,]$Y),
       mean(pos_X1[pos_X1$A==0,]$Y)))

## Discriminatory Effect Estimates
fair_fit1 <- lm(Y ~ X + A + M, data=fair_data_5) # adjusted for covariates (~ )
summary(fair_fit1)
coef(fair_fit1)["A"]
fair_fit2 <- lm(Y ~ A, data=fair_data_5) # unadjusting for covariates (~ )
summary(fair_fit2)
coef(fair_fit2)["A"]
```