# Longitudinal G-Formula

A. Keil and M.Kamenetsky

2025-02-26

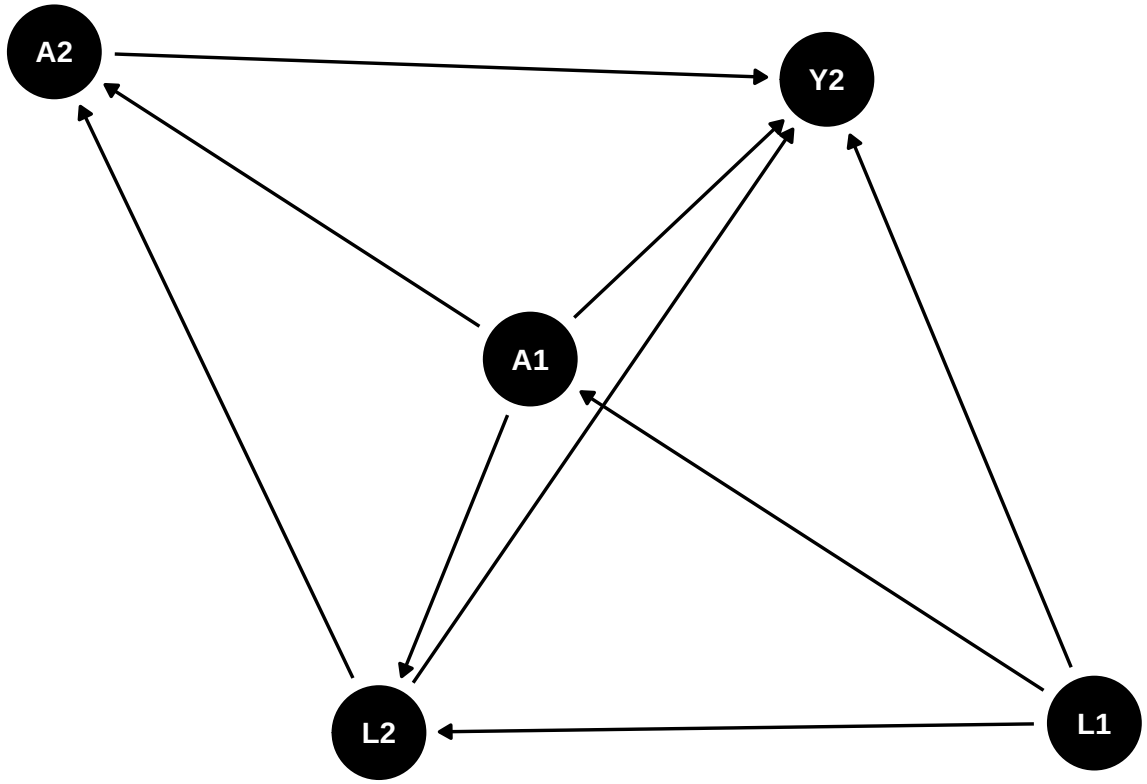
```
#Load in libraries
library(ggplot2) #for plotting
library(dplyr) #for data cleaning
library(survival) #for survival analysis
library(boot) #for bootstrapping
```

## Research Questions:

To estimate the risk difference of four different treatment regimes of radon reduction on uranium mine workers.

```
library(ggdag)
longdag <- ggdag::dagify(L2 ~ L1,
                         A1 ~ L1,
                         L2 ~ A1,
                         A2 ~ A1,
                         A2 ~ L2,
                         Y2 ~ A2,
                         Y2 ~ A1,
                         Y2 ~ L1,
                         Y2 ~ L2)

#set.seed(2)
set.seed(18)
ggdag::ggdag(longdag)+
  theme_void() +
  ggtitle("Example: Longitudinal DAG")
```

Example: Longitudinal DAG




## Learning Objectives:

By the end of this tutorial, you will be able to:

- Complete a parametric g-formula analysis for longitudinal data using `R` software
- Estimate different joint effects under different reduction scenarios in a longitudinal data set

## Key Points:

- Causal assumptions are assumed to be met. They are:
  - 1) Exchangeability
  - 2) Consistency
  - 3) Positivity
  - 4) No interference

## Key Points:

The Monte Carlo algorithm can be used to estimate $E(Y_2^g)$ or $p(y_2^g)$:

1) Sample with replacement from the target population at time $k = 0$, $\hat{p}(l_1) = \hat{p}_n(l_1)$ (**pseudo-population**, $N^g$)
2) Set $A_1$ equal to $g$ for everyone in the pseudo-population
3) Simulate values from $L_2$ from $p(l_2|g, l_1)$
   - For example, a Bernoulli distribution with $\mu = p(l_2|g, l_1)$
4) Simulate values of $Y_2$ from $p(Y_2|g, l_1, l_2)$
5) $E(Y^g)$ is the mean of $Y_2$ in the simulated pseudo-population

# Uranium Mine Workers Data Example: Longitudinal G-Formula

Consider a longitudinal study where you have two data measured at two (or more) time points. If we want to estimate joint effects, we can explore different realizations under different regimes, such as "always exposed", "never exposed", "natural course". To do so, we can simulate individuals through time using the directed acyclic graph (DAG)

The target population consists of uranium miners. We would like to estimate the risk (or cumulative incidence) from 0-20 years. We want to estimate the following regimes:

- No intervention ("natural course")
- 0.3 x 1000 pCi/L-year limit ("slightly below current standard")
- 0.1 x 1000 pCi/L-year limit ("well below current standard")
- No exposure ("attributable risk")

These are considered to be *dynamic regimes*, in that at work, exposure will be below the limit and unexposed otherwise.

We load in the data set `miners` using `read.csv()` function. We pipe (`%>%`) that data to the `mutate()` function to create a new variable (`lograd`), which we will use later. This variable takes the natural logarithm of the radon values when participants were at work (`atwork==1`) and is missing otherwise, as stated in the `ifelse()` function. We briefly explore a summary of the data:

```
miners <- read.csv("data/miners.csv") %>%
  mutate(lograd = ifelse(atwork==1, log(rad), NA))
str(miners)
```

```
## 'data.frame':    3601 obs. of  14 variables:
##  $ id          : int  1 1 1 1 1 2 3 3 3 4 ...
##  $ intime      : int  0 1 2 3 4 0 0 1 2 0 ...
##  $ outtime     : num  1 2 3 4 4.21 ...
##  $ dead        : int  0 0 0 0 1 1 0 0 1 0 ...
##  $ rad         : num  0.0326 0 0 0 0 ...
##  $ rad_lag1    : num  0 0.0326 0 0 0 ...
##  $ cum_rad     : num  0.0326 0.0326 0.0326 0.0326 0.0326 ...
##  $ cum_rad_lag1: num  0 0.0326 0.0326 0.0326 0.0326 ...
##  $ atwork      : int  1 0 0 0 0 1 1 1 0 1 ...
##  $ leavework   : int  0 1 0 0 0 0 0 0 1 0 ...
##  $ durwork_lag1: int  0 1 1 1 1 0 0 1 2 0 ...
##  $ durwork     : int  1 1 1 1 1 1 1 2 2 1 ...
##  $ smoker      : int  0 0 0 0 0 0 1 1 1 0 ...
##  $ lograd      : num  -3.42 NA NA NA NA ...
```

By using the `str()` command, we have the following 13 variables:

- `id`: a unique identifier for each individual
- `intime`: time during which participant is at work
- `outtime`: time during which participant is not at work
- `dead`: binary indicator for dead (1) or not dead (0)
- `rad`: radon measurement
- `rad_lag1`: radon measurement lagged by 1 time period
- `cum_rad`: cumulative radon exposure
- `cum_rad_lag1`: cumulative radon exposure lagged by 1 time period
- `atwork`: binary indicator of if exposure took place at work (1) or not at work (0)
- `leavework`: binary indicator of if exposure was on leave from work (1) or not on leave from work (0) **TODO**
- `durwork`: binary indicator of if exposure occurred during work (1) or not (0)

- `durwork_lag1`: binary indicator of if exposure occurred during work (1) or not (0) lagged by 1 time period
- `smoker`: binary indicator of individual was a smoker (1) or not a smoker (0)
- `lograd`: natural logarithm of the radon measurement when at work

```
summary(miners[, 2:13])
```

```
##      intime          outtime              dead             rad
##  Min.   : 0.000   Min.   : 0.004421   Min.   :0.0000   Min.   :0.00000
##  1st Qu.: 1.000   1st Qu.: 1.553428   1st Qu.:0.0000   1st Qu.:0.01743
##  Median : 2.000   Median : 3.000000   Median :0.0000   Median :0.06554
##  Mean   : 3.014   Mean   : 3.895930   Mean   :0.2216   Mean   :0.12015
##  3rd Qu.: 4.000   3rd Qu.: 5.000000   3rd Qu.:0.0000   3rd Qu.:0.14915
##  Max.   :19.000   Max.   :20.000000   Max.   :1.0000   Max.   :5.24338
##     rad_lag1          cum_rad          cum_rad_lag1         atwork
##  Min.   :0.00000   Min.   :0.002892   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.00000   1st Qu.:0.116539   1st Qu.:0.02805   1st Qu.:1.0000
##  Median :0.03905   Median :0.306124   Median :0.17251   Median :1.0000
##  Mean   :0.09282   Mean   :0.444990   Mean   :0.32484   Mean   :0.7837
##  3rd Qu.:0.12222   3rd Qu.:0.633812   3rd Qu.:0.49060   3rd Qu.:1.0000
##  Max.   :2.40830   Max.   :5.243377   Max.   :3.17834   Max.   :1.0000
##    leavework        durwork_lag1      durwork           smoker
##  Min.   :0.00000   Min.   : 0.00   Min.   : 1.000   Min.   :0.00000
##  1st Qu.:0.00000   1st Qu.: 1.00   1st Qu.: 1.000   1st Qu.:0.00000
##  Median :0.00000   Median : 2.00   Median : 2.000   Median :0.00000
##  Mean   :0.06693   Mean   : 2.44   Mean   : 3.224   Mean   :0.06443
##  3rd Qu.:0.00000   3rd Qu.: 4.00   3rd Qu.: 4.000   3rd Qu.:0.00000
##  Max.   :1.00000   Max.   :15.00   Max.   :16.000   Max.   :1.00000
```

First, extract baseline data at `intime=0` using the `filter()` function. We also create a new variable called `lograd` using the `mutate()` function which takes the natural logarithm of the radon measurement where participants were at work (`atwork==1`) and are missing (`NA`) otherwise. We set `N` to be equal to the number of observations (or `nrow()`):

```
baseline_data <- filter(miners, intime==0)
N = nrow(baseline_data)
```

Sample with replacement over 20000 realizations (`mc_iter`). To do this, we first set the seed so the analysis can be reproduced (`set.seed(12325)`). We then use the `slice()` function from the `dplyr` package to perform the sampling `mc_iter` times, with replacement (`replace=TRUE`).

```
# large sample from observed baseline data
mc_iter = 20000
set.seed(12325)
mcdata <- slice(baseline_data, sample(1:N, size = mc_iter, replace=TRUE))
```

We develop three different models:

1) Pooled linear model for (log) radon exposure while at work. This is model is the annual log-radon exposure: $rad_i = \beta_0 + \beta_1 outtime_i + \beta_2 cum\_rad\_lag1_i + \beta_3 smoker_i + \varepsilon_i$
2) Pooled logistic model for leaving work (at the end of the year) while at work (where $p_i$ is the probability of leaving work). This model the time-varying confounder of employment: $\log(\frac{p_i}{1-p_i}) = \beta_0 + \beta_1 outtime_i + \beta_2 outtime_i^2 + \beta_3 rad\_lag1_i + \beta_4 cum\_rad\_lag1 + \beta_5 smoker_i$
3) Pooled logistic model for death (where $p_i$ is the probability of death). This models the time-varying outcome of death: $\log(\frac{p_i}{1-p_i}) = \beta_0 + \beta_1 outtime_i + \beta_2 atwork_i + \beta_3 cum\_rad_i + \beta_4 cum\_rad_i^2 + \beta_4 smoker_i + \beta_5 cum\_rad_i * smoker_i$

For model 1, we filter the data such that we are only looking at when participants were at work

(filter(miners, atwork==1))

```
#Model 1: pooled linear model for log(exposure), while at work
mod_e = lm(lograd ~ outtime + cum_rad_lag1 + smoker, data=filter(miners, atwork==1))
```

For model 2, we filter the data to where participants where either at work or were on leave from work (filter(miners, atwork==1 | leavework==1)):

```
# pooled logistic model for leaving work (at the end of the year), while at work
mod_w = glm(leavework ~ outtime + I(outtime^2) + rad_lag1 + cum_rad_lag1 + smoker, data=filter(miners, a
```

For model 3, we do not filter the data and use the whole data set of miners:

```
mod_d = glm(dead ~ outtime + atwork + cum_rad + I(cum_rad*cum_rad) + smoker + smoker*cum_rad, data=miner
```

# Treatment Regimes

## Natural Course

Now that we have our three models, we will use them to create predictions under the different treatment regimes. First, we explore the natural course progression. We will do this manually first, and then create a function to make this more reproducible. We start with the pseudo-cohort at baseline we created above (mcdata). We set the end time for our hypothetical cohort to be 20 years, which is the follow-up time for which we want to follow this cohort. For each pseudo-person in our new pseudo-cohort, we create a new identifier, gfid. For each of the variables in the original data set, we set initial values.

For the linear exposure model, we also set the error variance as the sum of the squared residuals divided by the residual degrees of freedom. In order to prevent prediction outside of the observed range, we set a limit on the log exposure as max_e = 7.321276.

```
N <- nrow(mcdata)
ncols <- ncol(mcdata)
pseudo_cohort <- slice(mcdata,rep(1:N, each=20))
#create a new identifier
pseudo_cohort$gfid <- rep(seq(1, N, 1), each=20) # new id for each pseudo individual
# initialize values
pseudo_cohort$intime <- rep(seq(0, (20-1), 1), times=N)
pseudo_cohort$outtime <- pseudo_cohort$intime+1
pseudo_cohort$atwork <- 1
pseudo_cohort$durwork <- 1
pseudo_cohort$leavework <- 0
pseudo_cohort$rad <- 0
pseudo_cohort$rad_lag1 <- 0
pseudo_cohort$cum_rad_lag1 <- 0
pseudo_cohort$durwork_lag1 <- 0
pseudo_cohort$dead <- 0
pseudo_cohort$cum_hazard <- 0 # new cumulative hazard
pseudo_cohort$cum_incidence <- 0 # new cumulative incidence

# set error variance for exposure model
var_e = sum(mod_e$residuals^2)/mod_e$df.residual


# limits on log exposure (keep simulated values in range of observed)
max_e = 7.321276
```

```
#for the natural course, there is no intervention so we set it to NULL
intervention <- NULL

head(pseudo_cohort)#check out the first 6 observations of the pseudo-cohort and confirm that initial va
```

```
##    id intime outtime dead rad rad_lag1   cum_rad cum_rad_lag1 atwork leavework
## 1 717      0       1    0   0        0 0.3783546            0      1         0
## 2 717      1       2    0   0        0 0.3783546            0      1         0
## 3 717      2       3    0   0        0 0.3783546            0      1         0
## 4 717      3       4    0   0        0 0.3783546            0      1         0
## 5 717      4       5    0   0        0 0.3783546            0      1         0
## 6 717      5       6    0   0        0 0.3783546            0      1         0
##   durwork_lag1 durwork smoker     lograd gfid cum_hazard cum_incidence
## 1            0       1      0 -0.9719233    1          0             0
## 2            0       1      0 -0.9719233    1          0             0
## 3            0       1      0 -0.9719233    1          0             0
## 4            0       1      0 -0.9719233    1          0             0
## 5            0       1      0 -0.9719233    1          0             0
## 6            0       1      0 -0.9719233    1          0             0
```

Now that the parameters have been set, we simulate time-varying values using a series of for-loops. The outer loop loops over each `t` time period in time periods 1 through 20.

Depending on which model (leaving work) or if exploring lagged variables, we perform appropriate accounting (ex: starting everyone at work in first period). Then we use the respective three models above (`mod_e`, `mod_w`, `mod_d`) to predict and generate random variables using `rbinom()` or `rnorm()` functions to simulate responses based on those predicted models.

```
set.seed(12325)
for(t in 1:20){
  # index that keeps track of time
  idx = which(pseudo_cohort$outtime == t)
  #update all values of lagged variables (easy to forget!)
  if(t > 1){
    #lagging the data because of disease latency
    pseudo_cohort[idx,'cum_rad_lag1'] = pseudo_cohort[idx-1,'cum_rad']
    pseudo_cohort[idx,'rad_lag1'] = pseudo_cohort[idx-1,'rad']
  }
  #############
  # leaving work (time varying confounder)
  #############
  if(t==1){
    widx = 1:length(idx) # everyone is at work at first time point
  }
  if(t > 1){
    #simulation below is forward in time, did someone leave work during this time point?
    widx = which(pseudo_cohort[idx-1,'atwork']==1) #work index
    # index keeping track of which workers still work
    ## simulate here from known probability distribution
    pseudo_cohort[idx[widx],'leavework'] <- rbinom(n=length(widx), size=1,
                                                   prob=predict(mod_w, newdata = pseudo_cohort[idx[widx]
                                                                type = 'response'))
    #for everyone else still alive at time t, do they leave work at this time?
    # if worker didn't leave, then assume stay at work
    pseudo_cohort[idx,'atwork'] <- (pseudo_cohort[idx,'leavework']==0 & pseudo_cohort[idx-1,'atwork']==
```

```
  # update at work index to account for workers who left
  widx = which(pseudo_cohort[idx,'atwork']==1)
  #durwork: keeping track of work duration
  pseudo_cohort[idx,'durwork'] <- pseudo_cohort[idx-1,'durwork'] + pseudo_cohort[idx,'atwork']
  pseudo_cohort[idx,'durwork_lag1'] <- pseudo_cohort[idx-1,'durwork']
}
#############
# exposure/interventions
#############
# exposure is assumed log-normally distributed (=predicted value plus draw from the residuals)
# these are predicted values from the upstream process
meanlogr = predict(mod_e, newdata = pseudo_cohort[idx[widx],])
logr = meanlogr + rnorm(n=length(widx), mean=0, sd=sqrt(var_e))
pseudo_cohort[idx[widx],'rad'] <- exp(pmin(log(max_e), logr))
# exposure under different interventions/regimes
if(typeof(intervention) != "NULL"){
  if(is.numeric(intervention)){
    # static, deterministic intervention, e.g. set exposure = 0
    pseudo_cohort[idx,'rad'] <- intervention
  } else{
    if(is.character(intervention)){
      # dynamic interventions are defined by character strings: e.g. 'rad>0.3' means we intervene to
      #  this line creates an index of rows at the current time which are in violation of the interve
      #  exposure will be redrawn for these observations until they are in compliance.
      #  This is equivalent to drawing from a truncated log-normal distribution
      #
      #  Below is rejection sampling; if value is above the regime threshold,
      #  then reject it and sample again. This is more or less equivalent to sampling
      #  from a truncated distribution at regime limit
      viol_idx <- with(pseudo_cohort[idx,], which(eval(parse(text=intervention))))
      while(length(viol_idx)>0){
        # accept/rejection sampling to get draws from truncated log-normal
        # this is how we (I) assume an intervention would be implemented, but
        # we could imagine other ways (e.g. a hard cap on exposure)
        meanlogr = predict(mod_e, newdata = pseudo_cohort[idx[viol_idx],])
        lograd = meanlogr + rnorm(n=length(viol_idx), mean=0, sd=sqrt(var_e))
        pseudo_cohort[idx[viol_idx],'rad'] <- exp(lograd)
        # check whether any are still in violation of the distribution, if so, repeat loop
        viol_idx <- with(pseudo_cohort[idx,], which(eval(parse(text=intervention))))
      }}} # end dynamic intervention
} # end all interventions on exposure
if(t > 1){
  pseudo_cohort[idx,'cum_rad'] = pseudo_cohort[idx-1,'cum_rad'] + pseudo_cohort[idx,'rad']
} else pseudo_cohort[idx,'cum_rad'] = pseudo_cohort[idx,'rad']
#############
# death (simulate discrete hazard as in Taubman et al 2009, rather than 1/0 as in Keil et al 2014)
#  see note at bottom of program about late entry - in the case of late entry
#  the typical approach is to simulate actual death (1/0 variable) and then
#  estimate survival in the pseudo-population using a survival curve estimator
#  that can account for late entry
#############
#below is final prediction of death or not (if not, then keep calculating
#cumulative incidence)
```

```
  #here, we are making mortality predictions from the updated data. All people
  #alive at time t and making predictions based on the simulated data
  pseudo_cohort[idx,'dead'] = predict(mod_d, newdata = pseudo_cohort[idx,], type = 'response')
  if(t > 1){
    # product limit estimator of cumulative incidence (note this is applied on the individual basis)
    pseudo_cohort[idx,'cum_incidence'] = pseudo_cohort[idx-1,'cum_incidence'] +
      (1-pseudo_cohort[idx-1,'cum_incidence'])*pseudo_cohort[idx,'dead']
  } else{
    pseudo_cohort[idx,'cum_incidence'] = pseudo_cohort[idx,'dead']
  }
  #alternative estimator of the cumulative incidence: Breslow estimator
  #if(t > 1){
  #  # Kaplan-meier estimator of cumulative incidence (note this is applied on the individual basis)
  #  pseudo_cohort[idx,'cum_hazard'] = pseudo_cohort[idx-1,'cum_hazard'] + pseudo_cohort[idx,'dead']
  #} else{
  #  pseudo_cohort[idx,'cum_hazard'] = pseudo_cohort[idx,'dead']
  #}
  #  pseudo_cohort[idx,'cum_incidence'] = 1-exp(-pseudo_cohort[idx,'cum_hazard'])
  # could also use Breslow estimator, but need aalen-johansen estimator if there are competing risks
  # the 'cum_incidence' variable is an estimate of the individual risk. We then
  # average that over the population below to get the marginal risk
} # end time loop
head(pseudo_cohort)#check out first 6 observations with new predicted values
```

```
##    id intime outtime      dead        rad   rad_lag1    cum_rad cum_rad_lag1
## 1 717      0       1 0.1425864 0.09787482 0.00000000 0.09787482   0.00000000
## 2 717      1       2 0.4997362 1.30914720 0.09787482 1.40702202   0.09787482
## 3 717      2       3 0.4949424 0.10071520 1.30914720 1.50773722   1.40702202
## 4 717      3       4 0.5250058 0.21758813 0.10071520 1.72532535   1.50773722
## 5 717      4       5 0.5302889 0.14324742 0.21758813 1.86857277   1.72532535
## 6 717      5       6 0.5161137 0.07935105 0.14324742 1.94792382   1.86857277
##   atwork leavework durwork_lag1 durwork smoker     lograd gfid cum_hazard
## 1      1         0            0       1      0 -0.9719233    1          0
## 2      1         0            1       2      0 -0.9719233    1          0
## 3      1         0            2       3      0 -0.9719233    1          0
## 4      1         0            3       4      0 -0.9719233    1          0
## 5      1         0            4       5      0 -0.9719233    1          0
## 6      1         0            5       6      0 -0.9719233    1          0
##   cum_incidence
## 1     0.1425864
## 2     0.5710670
## 3     0.7833641
## 4     0.8970992
## 5     0.9516664
## 6     0.9766120
```

From our pseudo-cohort under the natural course regime, we want to calculate the cumulative incidence for each year, 1 through 20. To do so, we take the mean cumulative incidence in each year group (`outtime`) across all individuals in the pseudo-cohort. We print our results using the `kable()` function from the `knitr` package

```
cuminc_nc0 <- with(pseudo_cohort, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean)
knitr::kable(cuminc_nc0, caption = "Cumulative incidence: unexposed regime (hard code)")
```

Table 1: Cumulative incidence: unexposed regime (hard code)

| time | ci |
|---|---|
| 0 | 0.0000000 |
| 1 | 0.1853386 |
| 2 | 0.3524044 |
| 3 | 0.4949088 |
| 4 | 0.6114532 |
| 5 | 0.7043842 |
| 6 | 0.7759172 |
| 7 | 0.8296103 |
| 8 | 0.8693592 |
| 9 | 0.8985161 |
| 10 | 0.9199909 |
| 11 | 0.9358033 |
| 12 | 0.9475717 |
| 13 | 0.9564408 |
| 14 | 0.9632063 |
| 15 | 0.9684409 |
| 16 | 0.9725237 |
| 17 | 0.9757296 |
| 18 | 0.9782872 |
| 19 | 0.9803374 |
| 20 | 0.9819894 |

For the other 3 regimes, we must do the same thing. In order to make things more reproducible, we will create a function called `gformula_risks()` which will allow us to perform the same procedure. The parameters to the function are:

- @param `intervention` Intervention or treatment regime. Options are `NULL`, 0, `rad>0.3`, or `rad>0.1`, as defined above.
- @param `pseudo_cohort_bl` Pseudo-cohort at baseline. We create this object above and have called it `mcdata`.
- @param `endtime` Number of total years for the pseudo-cohort to be followed. In this study, we set `endtime=20`
- @param `mod_e` Pooled linear model for (log) radon exposure while at work. This is model is the annual log-radon exposure (defined above). This model is also created above and stored as the object `mod_e`.
- @param `mod_w` Pooled logistic model for leaving work (at the end of the year) while at work. This models the time-varying confounder of employment. This model is also created above and stored as the object `mod_w`.
- @param `mod_d` Pooled logistic model for death. This models the time-varying outcome of death. This model is also created above and stored as the object `mod_d`.
- @param `seed` Seed value we input so that the results are reproducible.

```
gformula_risks <- function(intervention=NULL, pseudo_cohort_bl,
                           endtime, mod_e, mod_w, mod_d, seed=NULL){
  N <- nrow(pseudo_cohort_bl)
  ncols <- ncol(pseudo_cohort_bl)
  pseudo_cohort <- slice(pseudo_cohort_bl,rep(1:N, each=endtime))
  pseudo_cohort$gfid <- rep(seq(1, N, 1), each=endtime)
  pseudo_cohort$intime <- rep(seq(0, (endtime-1), 1), times=N)
  pseudo_cohort$outtime <- pseudo_cohort$intime+1
  pseudo_cohort$atwork <- 1
  pseudo_cohort$durwork <- 1
```

```
pseudo_cohort$leavework <- 0
pseudo_cohort$rad <- 0
pseudo_cohort$rad_lag1 <- 0
pseudo_cohort$cum_rad_lag1 <- 0
pseudo_cohort$durwork_lag1 <- 0
pseudo_cohort$dead <- 0
pseudo_cohort$cum_hazard <- 0
pseudo_cohort$cum_incidence <- 0
var_e = mean(mod_e$residuals^2)
set.seed(seed)
max_e = 7.321276
for(t in seq(1, endtime, 1)){
  idx = which(pseudo_cohort$outtime == t)
  if(t > 1){
    pseudo_cohort[idx,'cum_rad_lag1'] = pseudo_cohort[idx-1,'cum_rad']
    pseudo_cohort[idx,'rad_lag1'] = pseudo_cohort[idx-1,'rad']
  }
  if(t==1) widx = 1:length(idx)
  if(t > 1){
    widx = which(pseudo_cohort[idx-1,'atwork']==1)
    pseudo_cohort[idx[widx],'leavework'] <- rbinom(n=length(widx), size=1,
                                                   prob=predict(mod_w, newdata = pseudo_cohort[idx[wi
                                                                type = 'response'))
    pseudo_cohort[idx,'atwork'] <- (pseudo_cohort[idx,'leavework']==0 & pseudo_cohort[idx-1,'atwork']=
    widx = which(pseudo_cohort[idx,'atwork']==1)
    pseudo_cohort[idx,'durwork'] <- pseudo_cohort[idx-1,'durwork'] + pseudo_cohort[idx,'atwork']
    pseudo_cohort[idx,'durwork_lag1'] <- pseudo_cohort[idx-1,'durwork']
  }
  meanlogr = predict(mod_e, newdata = pseudo_cohort[idx[widx],])
  logr = meanlogr + rnorm(n=length(widx), mean=0, sd=sqrt(var_e))
  pseudo_cohort[idx[widx],'rad'] <- exp(pmin(log(max_e), logr))
  if(typeof(intervention) != "NULL"){
    if(is.numeric(intervention)){
      pseudo_cohort[idx,'rad'] <- intervention
    } else{
      if(is.character(intervention)){
        viol_idx <- with(pseudo_cohort[idx,], which(eval(parse(text=intervention))))
        while(length(viol_idx)>0){
          meanlogr = predict(mod_e, newdata = pseudo_cohort[idx[viol_idx],])
          lograd = meanlogr + rnorm(n=length(viol_idx), mean=0, sd=sqrt(var_e))
          pseudo_cohort[idx[viol_idx],'rad'] <- exp(lograd)
          viol_idx <- with(pseudo_cohort[idx,], which(eval(parse(text=intervention))))
        }}}
  }
  if(t > 1){
    pseudo_cohort[idx,'cum_rad'] = pseudo_cohort[idx-1,'cum_rad'] + pseudo_cohort[idx,'rad']
  } else pseudo_cohort[idx,'cum_rad'] = pseudo_cohort[idx,'rad']
  pseudo_cohort[idx,'dead'] = predict(mod_d, newdata = pseudo_cohort[idx,], type = 'response')
  if(t > 1){
    pseudo_cohort[idx,'cum_incidence'] = pseudo_cohort[idx-1,'cum_incidence'] +
      (1-pseudo_cohort[idx-1,'cum_incidence'])*pseudo_cohort[idx,'dead']
    #individual-level risks calculated using Kaplan-Meier formula and then summed and averaged
  } else{
```

```
      pseudo_cohort[idx,'cum_incidence'] = pseudo_cohort[idx,'dead']
    }
  }
  pseudo_cohort
}
```

We perform the simulation under the natural course again using the `gformula_risks()` function, and store the results from the function call into the object `nc` (or natural course).

```
nc = gformula_risks(intervention=NULL, pseudo_cohort_bl=mcdata, endtime=20,
                    mod_e=mod_e, mod_w=mod_w, mod_d=mod_d, seed=12325)
gf_cuminc_nc <- with(nc, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean))))
knitr::kable(gf_cuminc_nc, caption = "Cumulative incidence: natural course regime")
```

Table 2: Cumulative incidence: natural course regime

| time | ci |
|---|---|
| 0 | 0.0000000 |
| 1 | 0.1853118 |
| 2 | 0.3523480 |
| 3 | 0.4948295 |
| 4 | 0.6114228 |
| 5 | 0.7044357 |
| 6 | 0.7762090 |
| 7 | 0.8301504 |
| 8 | 0.8700183 |
| 9 | 0.8992587 |
| 10 | 0.9206480 |
| 11 | 0.9363762 |
| 12 | 0.9480581 |
| 13 | 0.9568467 |
| 14 | 0.9635285 |
| 15 | 0.9687199 |
| 16 | 0.9727688 |
| 17 | 0.9759461 |
| 18 | 0.9784746 |
| 19 | 0.9804996 |
| 20 | 0.9821380 |

We perform the same analysis under the remaining three regimes:

```
unex = gformula_risks(intervention=0,
                      pseudo_cohort_bl=mcdata,
                      endtime=20,
                      mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                      seed=12325)
gf_cuminc_unex <- with(unex, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean))))
knitr::kable(gf_cuminc_unex, caption = "Cumulative incidence: unexposed regime")
```

Table 3: Cumulative incidence: unexposed regime

| time | ci |
|---|---|
| 0 | 0.0000000 |
| 1 | 0.1551826 |
| 2 | 0.2809765 |
| 3 | 0.3833794 |
| 4 | 0.4674376 |
| 5 | 0.5379277 |
| 6 | 0.5965654 |
| 7 | 0.6449063 |
| 8 | 0.6843107 |
| 9 | 0.7160998 |
| 10 | 0.7416395 |
| 11 | 0.7620995 |
| 12 | 0.7785237 |
| 13 | 0.7917648 |
| 14 | 0.8025146 |
| 15 | 0.8113073 |
| 16 | 0.8185484 |
| 17 | 0.8245475 |
| 18 | 0.8295444 |
| 19 | 0.8337274 |
| 20 | 0.8372428 |

```
exp3 = gformula_risks(intervention='rad>0.3',
                      pseudo_cohort_bl=mcdata,
                      endtime=20,
                      mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                      seed=12325)
gf_cuminc_exp3 <- with(exp3, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean))))
knitr::kable(gf_cuminc_exp3, caption = "Cumulative incidence: radon limit = 0.3 exposure regime")
```

Table 4: Cumulative incidence: radon limit = 0.3 exposure regime

| time | ci |
|---|---|
| 0 | 0.0000000 |
| 1 | 0.1726130 |
| 2 | 0.3234703 |
| 3 | 0.4517287 |
| 4 | 0.5588723 |
| 5 | 0.6471637 |
| 6 | 0.7181688 |
| 7 | 0.7742639 |
| 8 | 0.8177422 |
| 9 | 0.8512567 |
| 10 | 0.8769662 |
| 11 | 0.8967229 |
| 12 | 0.9120497 |
| 13 | 0.9241015 |
| 14 | 0.9337187 |
| 15 | 0.9414689 |

| time | ci |
|---|---|
| 16 | 0.9477903 |
| 17 | 0.9529907 |
| 18 | 0.9572977 |
| 19 | 0.9608905 |
| 20 | 0.9639062 |

```
exp1 = gformula_risks(intervention='rad>0.1',
                      pseudo_cohort_bl=mcdata,
                      endtime=20,
                      mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                      seed=12325)
gf_cuminc_exp1 <- with(exp1, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean))))
knitr::kable(gf_cuminc_exp1, caption = "Cumulative incidence: radon limit = 0.1 exposure regime")
```

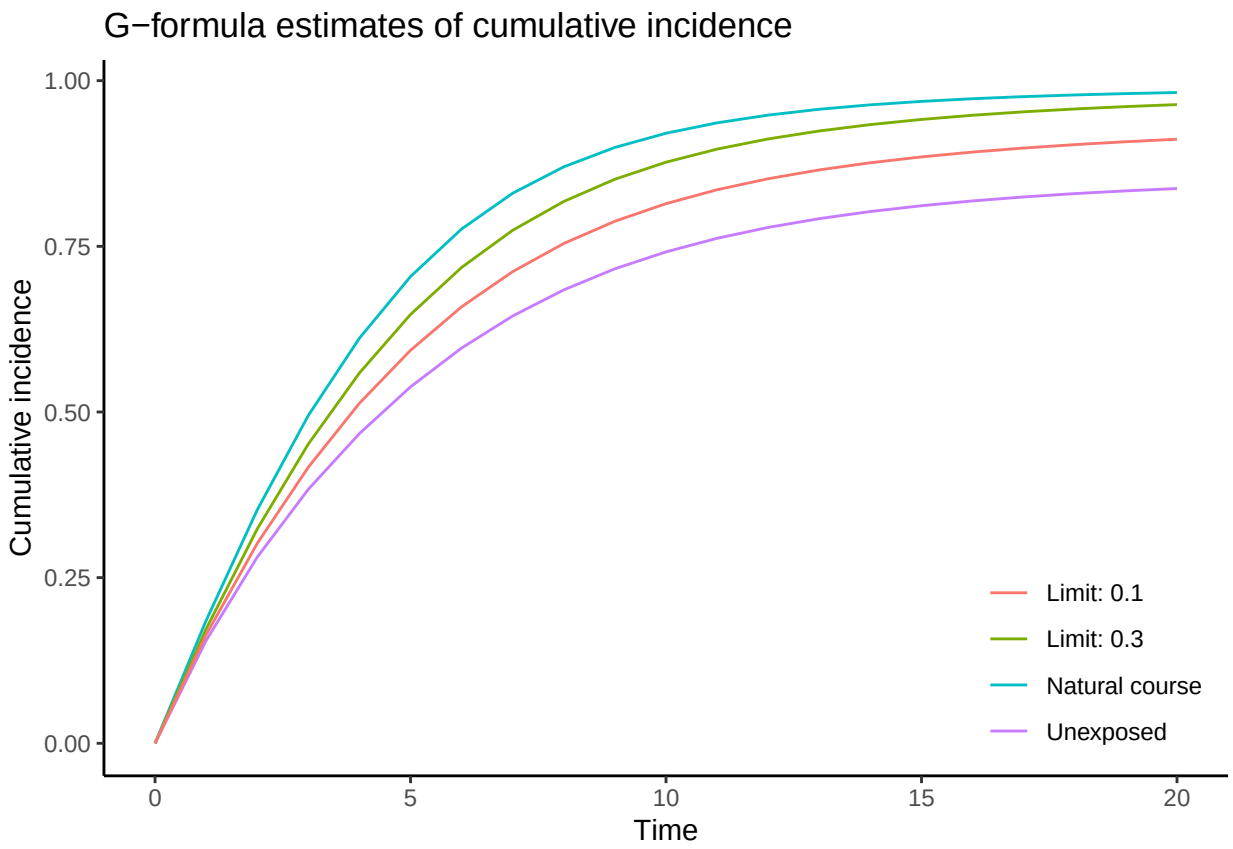
Table 5: Cumulative incidence: radon limit = 0.1 exposure regime

| time | ci |
|---|---|
| 0 | 0.0000000 |
| 1 | 0.1637380 |
| 2 | 0.3017197 |
| 3 | 0.4166945 |
| 4 | 0.5132191 |
| 5 | 0.5931417 |
| 6 | 0.6587804 |
| 7 | 0.7119056 |
| 8 | 0.7542917 |
| 9 | 0.7878810 |
| 10 | 0.8144093 |
| 11 | 0.8354030 |
| 12 | 0.8520334 |
| 13 | 0.8653507 |
| 14 | 0.8761327 |
| 15 | 0.8849591 |
| 16 | 0.8922597 |
| 17 | 0.8983618 |
| 18 | 0.9035053 |
| 19 | 0.9078728 |
| 20 | 0.9116041 |

Instead of using a table, we can visualize these results:

```
ggplot() +
  geom_line(aes(x=time, y=ci, color="Natural course"), data=gf_cuminc_nc) +
  geom_line(aes(x=time, y=ci, color="Unexposed"), data=gf_cuminc_unex ) +
  geom_line(aes(x=time, y=ci, color="Limit: 0.3"), data=gf_cuminc_exp3 ) +
  geom_line(aes(x=time, y=ci, color="Limit: 0.1"), data=gf_cuminc_exp1 ) +
  scale_y_continuous(name="Cumulative incidence") +
  scale_x_continuous(name="Time", limits=c(0,20)) +
  scale_color_discrete(name="") +
  theme_classic() +
```

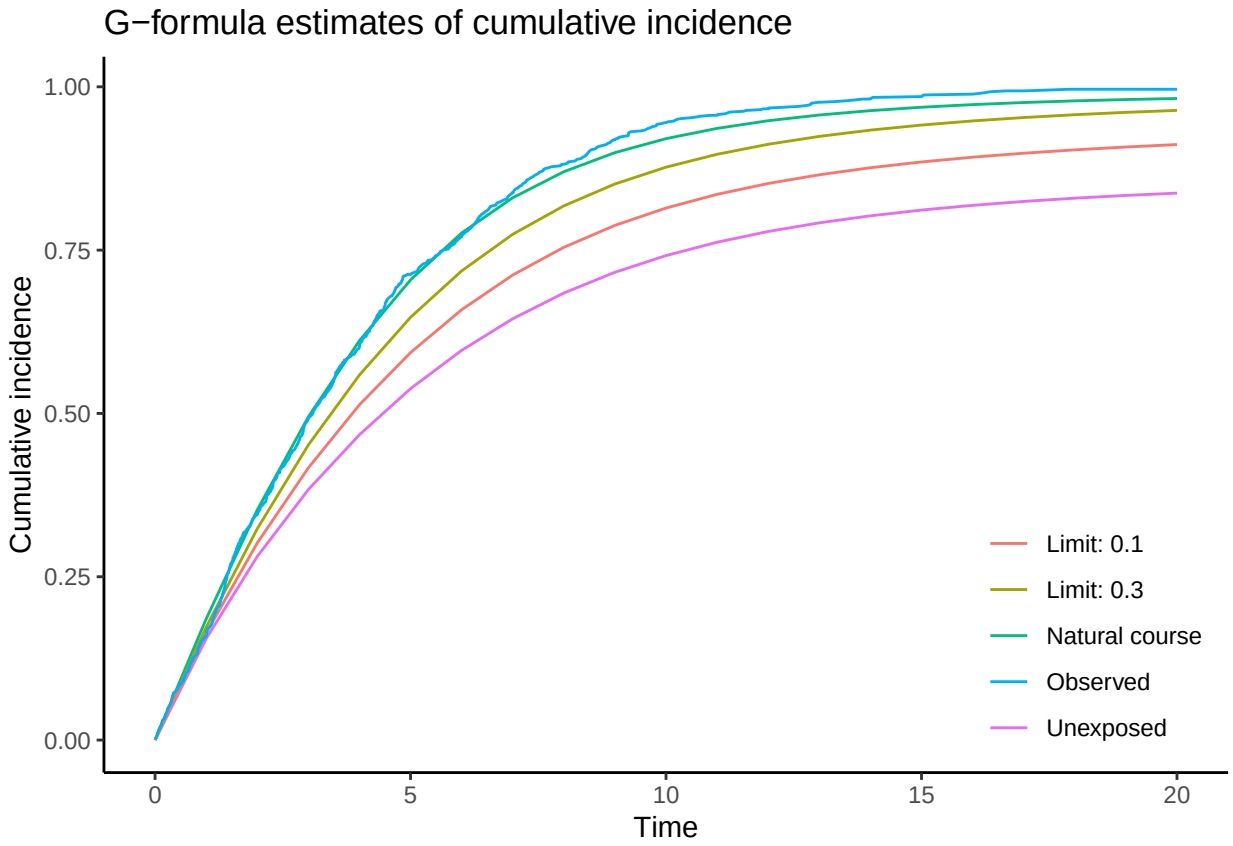
```
  theme(legend.position = c(0.99,0.01),legend.justification = c(1,0))+
  ggtitle(expression(paste("G-formula estimates of cumulative incidence")))
```




The observed risk estimates can be calculated using the Kaplain-Meier formula. These will be approximately similar to the natural course under correct model specification and in a large ennough sample size. To compare to observed incidence, we can use the `survfit()` function from the `survival` package:

```
obscifit <- survfit(Surv(intime, outtime, dead)~1, data=miners)
obsnc <- tibble(time=obscifit$time, ci=1-obscifit$surv)
```

```
ggplot() +
  geom_line(aes(x=time, y=ci, color="Natural course"), data=gf_cuminc_nc) +
  geom_line(aes(x=time, y=ci, color="Unexposed"), data=gf_cuminc_unex ) +
  geom_line(aes(x=time, y=ci, color="Limit: 0.3"), data=gf_cuminc_exp3 ) +
  geom_line(aes(x=time, y=ci, color="Limit: 0.1"), data=gf_cuminc_exp1 ) +
  geom_line(aes(x=time, y=ci, color="Observed"), data=obsnc) +
  scale_y_continuous(name="Cumulative incidence") +
  scale_x_continuous(name="Time", limits=c(0,20)) +
  scale_color_discrete(name="") +
  theme_classic() +
  theme(legend.position = c(0.99,0.01),legend.justification = c(1,0))+
  ggtitle(expression(paste("G-formula estimates of cumulative incidence")))
```




## Bootstrapped variance

In order to estimate confidence intervals, we will need to get the sample variance using bootstrapping. To do so, we first need to create a function that will perform the bootstrapping. Then we will use the `boot()` function from the `boot` package to perform the bootstrapping $R$ times, based on that function.

The function `gformula_effectestimates()` takes the following parameters:

- @param `baseline_data` Baseline data set at `intime=0` (created above)
- @param `index` If `NULL`, then original data will be used. If not null, then bootstrapped samples fro the baseline will be taken using the vector `index`. The default is `NULL`.
- @param `fdata` Functional data if `index` is not NULL. The default is the full `miners` data set
- @param `seed` Seed value we input so that the results are reproducible.

```
# now we turn the g-formula algorithm to get risk difference into a function in order to get bootstrap
gformula_effectestimates <- function(data=baseline_data, index=NULL, fdata=miners, endtime=10, seed=NULL
  # for bootstrap samples, when needed: first take a random sample of the
  #  study IDs, with replacement
  # 0) preprocessing: take bootstrap sample if bootstrapping, otherwise use the observed data
  if(is.null(index)){
    # if index is NULL, then just use original data and do nothing here
  } else{
    # bootstrap sample of the baseline data using the vector 'index'
    data = slice(data, index)
    # resample from the full data according to the id variable in the baseline data
    idindex = tibble(reps = table(data$id), id = as.numeric(names(table(data$id))))
    fdata = merge(idindex, fdata, by='id')
    fdata = slice(fdata, rep(1:dim(fdata)[1], times=fdata$reps))
```

```
  }
  # 1) fit models to full data/bootstrap sample of the full data
  mod_e = glm(lograd ~ outtime + cum_rad_lag1 + smoker, data=filter(data, atwork==1))
  # pooled logistic model for leaving work (at the end of the year), while at work
  mod_w = glm(leavework ~ outtime + I(outtime^2) + rad_lag1 + cum_rad_lag1 + smoker,
              data=filter(fdata, atwork==1 | leavework==1), family=binomial(link=logit))
  # pooled logistic model for death
  mod_d = glm(dead ~ outtime + atwork + cum_rad + I(cum_rad*cum_rad) + smoker + smoker*cum_rad,
              data=fdata, family=binomial(link=logit))

  # 2) take large MC sample from baseline data
  mc_iter = 20000
  set.seed(seed)
  mcdata <- slice(data, sample(1:N, size = mc_iter, replace=TRUE))
  # 3) simulate probability distributions in large MC sample using model fits from step 1
  nc = gformula_risks(intervention=NULL,
                      pseudo_cohort_bl=mcdata,
                      endtime=endtime,
                      mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                      seed=seed)
  unex = gformula_risks(intervention=0,
                        pseudo_cohort_bl=mcdata,
                        endtime=endtime,
                        mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                        seed=seed)
  exp3 = gformula_risks(intervention='rad>0.3',
                        pseudo_cohort_bl=mcdata,
                        endtime=endtime,
                        mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                        seed=seed)
  exp1 = gformula_risks(intervention='rad>0.1',
                        pseudo_cohort_bl=mcdata, endtime=endtime,
                        mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                        seed=seed)
  # 4) summarize data
  ci_nc = as.numeric(with(nc, tapply(cum_incidence, outtime, mean)))
  ci_unex = as.numeric(with(unex, tapply(cum_incidence, outtime, mean)))
  ci_exp3 = as.numeric(with(exp3, tapply(cum_incidence, outtime, mean)))
  ci_exp1 = as.numeric(with(exp1, tapply(cum_incidence, outtime, mean)))
  # return a vector of risk differences and log-risk ratios at time t=endtime
  # could also calculate for all times
  c(
    rd_exp3_10=c(ci_exp3-ci_nc)[endtime],
    rd_exp1_10=c(ci_exp1-ci_nc)[endtime],
    rd_unex_10=c(ci_unex-ci_nc)[endtime],
    lrr_exp3_10=log(c(ci_exp3/ci_nc)[endtime]),
    lrr_exp1_10=log(c(ci_exp1/ci_nc)[endtime]),
    lrr_unex_10=log(c(ci_unex/ci_nc)[endtime])
  )
}
```

Prior to performing the bootstrapping, we will use the function to get point estimates for the risk difference at `t=10`:

```r
gf_est <- gformula_effectestimates(data=baseline_data, fdata=miners, endtime=10, seed=NULL)
knitr::kable(gf_est, caption="1 Bootstrapped Resample: Risk Differences")
```

Table 6: 1 Bootstrapped Resample: Risk Differences

| | x |
|---|---|
| rd_exp3_10 | -0.0285547 |
| rd_exp1_10 | -0.0801665 |
| rd_unex_10 | -0.1394153 |
| lrr_exp3_10 | -0.0326347 |
| lrr_exp1_10 | -0.0944668 |
| lrr_unex_10 | -0.1705078 |

In order to get confidence intervals, we will use the non-parametric bootstrap. To note: to get valid standard errors, we should use at least 200 bootstrap samples, and at least 1000 for percentile-based confidence intervals. In the interest of computation time, we will only perform 10 bootstrap samples here.

```r
nbootsamples = 10
boot_samples <- boot(data = baseline_data, fdata=miners, statistic = gformula_effectestimates, R = nboo
knitr::kable(boot_samples$t0, caption="10 Bootstrapped Resamples: Risk Differences")
```

Table 7: 10 Bootstrapped Resamples: Risk Differences

| | x |
|---|---|
| rd_exp3_10 | -0.0296165 |
| rd_exp1_10 | -0.0800448 |
| rd_unex_10 | -0.1443737 |
| lrr_exp3_10 | -0.0337427 |
| lrr_exp1_10 | -0.0939539 |
| lrr_unex_10 | -0.1764315 |

Finally, we calculate the 95% bootstrapped confidence intervals:

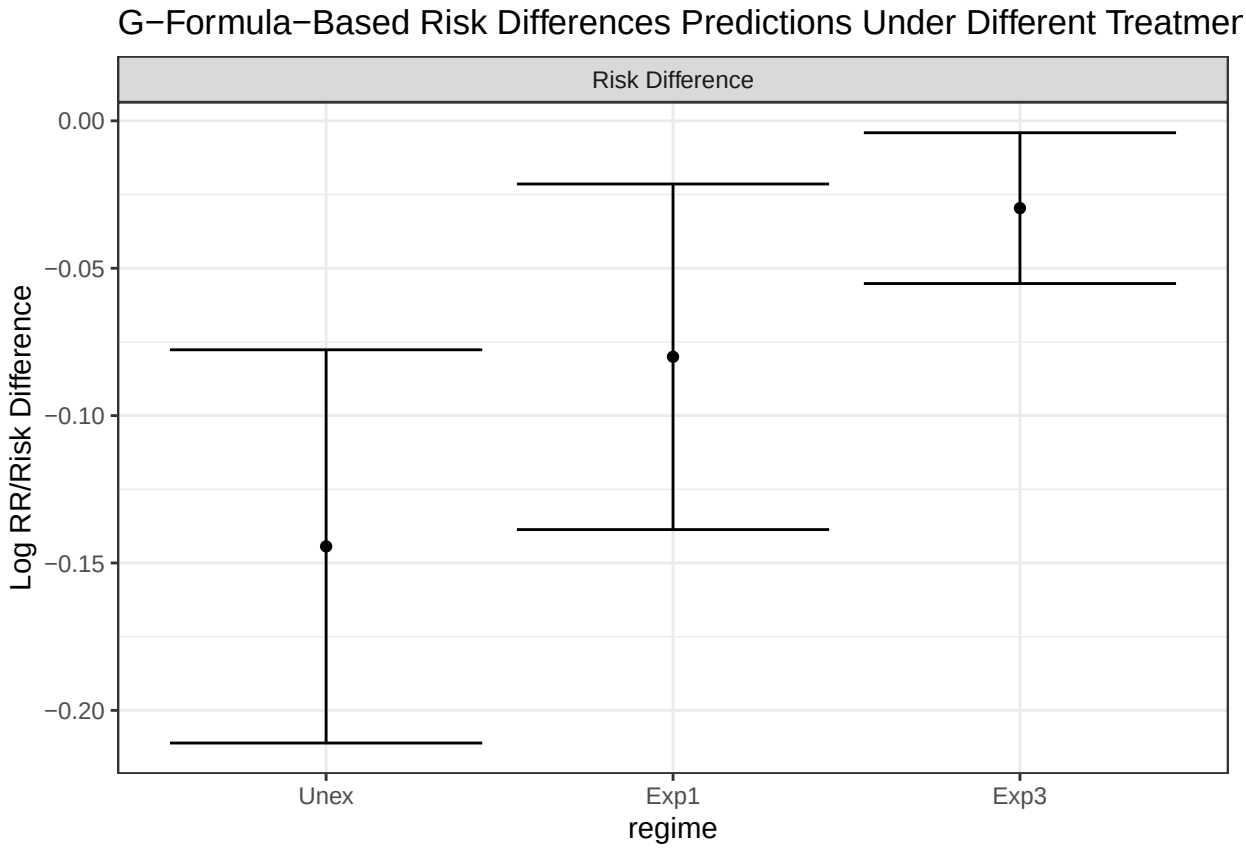
```r
est = boot_samples$t0 #the risk differences
lci =  boot_samples$t0 - 1.96*apply(boot_samples$t, 2, sd)
uci =  boot_samples$t0 + 1.96*apply(boot_samples$t, 2, sd)

dat1 <- cbind.data.frame(regime = c("Exp3","Exp1","Unex","Exp3","Exp1","Unex"),
                         type = c(rep("Risk Difference",3), rep("Log RR",3)),
                         RD = est, LB=lci, UB = uci) %>%
  mutate(regime = factor(regime, levels=c("Unex", "Exp1","Exp3")))

dat1 %>%
  knitr::kable(caption = "Risk Differences/Log Relative Risks and 95% Bootstrapped Confidence Intervals
```

Table 8: Risk Differences/Log Relative Risks and 95% Bootstrapped Confidence Intervals (R=10) Under Different Treatment Regimes at t=10 Using G-Formula

| | regime | type | RD | LB | UB |
|---|---|---|---|---|---|
| rd_exp3_10 | Exp3 | Risk Difference | -0.0296165 | -0.0551958 | -0.0040373 |

| | regime | type | RD | LB | UB |
|---|---|---|---|---|---|
| rd_exp1_10 | Exp1 | Risk Difference | -0.0800448 | -0.1386676 | -0.0214221 |
| rd_unex_10 | Unex | Risk Difference | -0.1443737 | -0.2110813 | -0.0776661 |
| lrr_exp3_10 | Exp3 | Log RR | -0.0337427 | -0.0600071 | -0.0074783 |
| lrr_exp1_10 | Exp1 | Log RR | -0.0939539 | -0.1565175 | -0.0313904 |
| lrr_unex_10 | Unex | Log RR | -0.1764315 | -0.2500566 | -0.1028063 |

```
#plot risk difference
ggplot(data=subset(dat1, type=="Risk Difference")) +
  geom_point(aes(x=regime, y=RD)) +
  geom_errorbar(aes(ymin = LB, ymax=UB, x=regime))+
  facet_wrap(.~type) +
  ylab("Log RR/Risk Difference") +
  theme_bw() +
  ggtitle("G-Formula-Based Risk Differences Predictions Under Different Treatment Regimes") +
  guides(color='none')
```



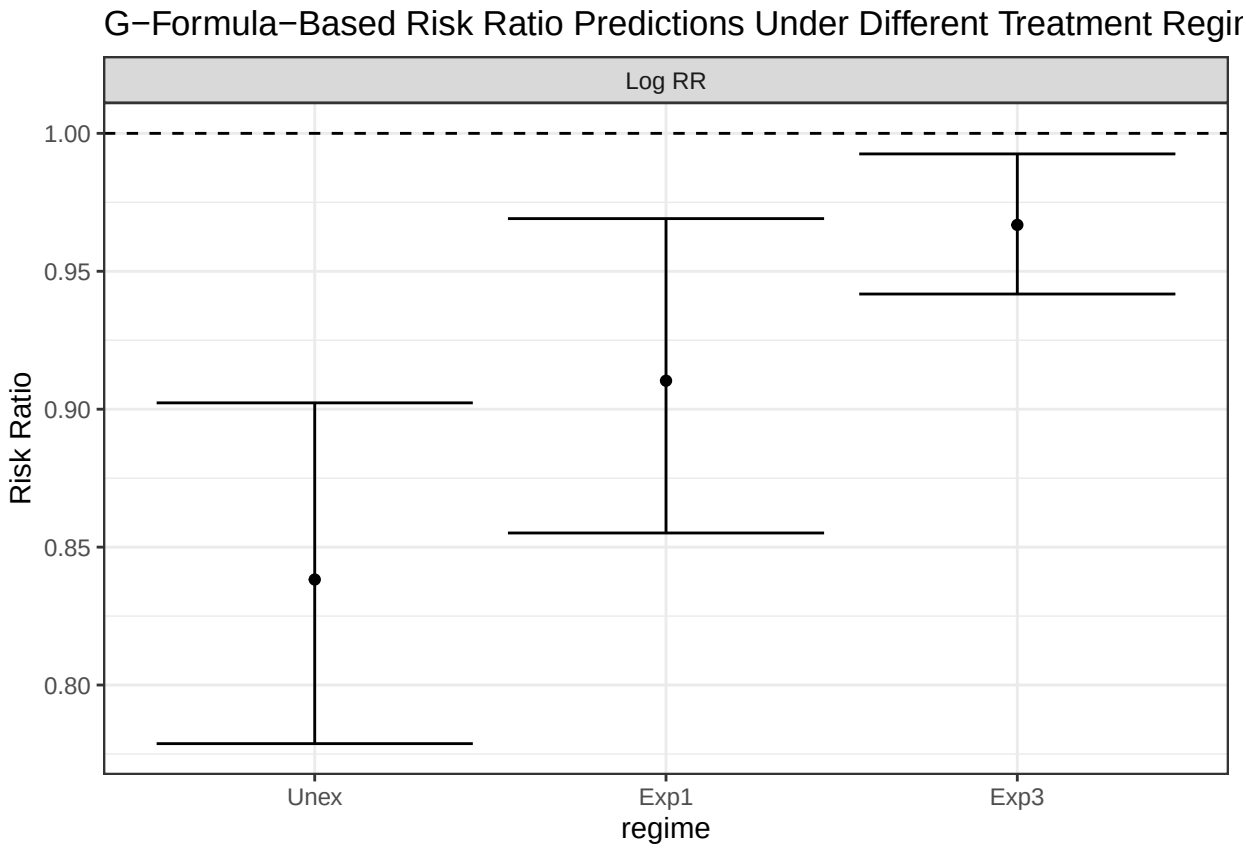

Under the unexposed regime, the risk difference is -0.14. By reducing all radon exposure for miners to 0, we can expect to reduce 14 deaths per 100 miners as attributable to radon exposure.

Under the regime of limiting the exposure to 0.1, the risk difference is -0.07. By reducing radon exposure for miners to only 0.1 x 1000 pCi/L-year limit ("well below current standard"), we can expect to reduce 7 deaths per 100 miners as attributable to radon exposure under this regime.

Under the regime of limiting the exposure to 0.3, the risk difference is -0.03. By reducing radon exposure for miners to 0.3 x 1000 pCi/L-year limit ("slightly below current standard"), we can expect to only reduce 3 deaths per 100 miners as attributable to radon exposure under this regime.

```r
#make table
#exponentiate the log RR to get RRs
dat1 %>%
  filter(type=="Log RR") %>%
  dplyr::rename(RR = RD) %>%
  mutate(across(c(RR,LB,UB), exp)) %>%
  print() %>%
ggplot() +
  geom_point(aes(x=regime, y=RR)) +
  geom_errorbar(aes(ymin = LB, ymax=UB, x=regime))+
  facet_wrap(.~type) +
  ylab("Risk Ratio") +
  theme_bw() +
  ggtitle("G-Formula-Based Risk Ratio Predictions Under Different Treatment Regimes") +
  guides(color='none') +
  geom_hline(yintercept = 1, linetype="dashed")
```

```
##              regime   type        RR        LB        UB
## lrr_exp3_10    Exp3 Log RR 0.9668202 0.9417579 0.9925496
## lrr_exp1_10    Exp1 Log RR 0.9103247 0.8551166 0.9690972
## lrr_unex_10    Unex Log RR 0.8382562 0.7787567 0.9023017
```




Under the unexposed regime, the risk ratio is 0.84. We expect that reducing all radon exposure for miners to 0 will reduce mortality by 16%.

Under the regime of limiting the exposure to 0.1, the risk ratio is 0.92. By reducing radon exposure for miners to only 0.1 x 1000 pCi/L-year limit ("well below current standard"), we can expect to reduce radon deaths by 8% as attributable to radon exposure under this regime.

Under the regime of limiting the exposure to 0.3, the risk ratio is 97%. By reducing radon exposure for miners to 0.3 x 1000 pCi/L-year limit ("slightly below current standard"), we can expect to only reduce radon deaths by 3% as attributable to radon exposure under this regime.

## Exercise

Perform the bootstrap sampling for $R = 200$ resamples. Plot your results and interpret them:

```
boot_samples200 <- boot(data = baseline_data, fdata=miners, statistic = gformula_effectestimates, R = 20

est200 = boot_samples200$t0 #the risk differences
lci200 =  boot_samples200$t0 - 1.96*apply(boot_samples200$t, 2, sd)
uci200 =  boot_samples200$t0 + 1.96*apply(boot_samples200$t, 2, sd)

dat <- cbind.data.frame(regime = c("Exp3","Exp1","Unex","Exp3","Exp1","Unex"),
                        type = c(rep("Risk Difference",3), rep("Log RR",3)),
                        RD = est200, LB=lci200, UB = uci200)
knitr::kable(dat, caption = "Risk Differences/Log Relative Risks and 95% Bootstrapped Confidence Interva

ggplot(data=dat) +
  geom_point(aes(x=regime, y=RD, color=type)) +
  geom_errorbar(aes(ymin = LB, ymax=UB, x=regime, color=type))+
  facet_wrap(.~type) +
  ylab("Log RR/Risk Difference") +
  theme_bw() +
  ggtitle("G-Formula-Based Risk Differences/Log Relative Risk Predictions Under Different Treatment Regi
  theme(legend.position = "bottom")
```

## Appendix - R Code

```
knitr::opts_chunk$set(echo = TRUE, warning = FALSE, message = FALSE,cache=TRUE)
#Load in libraries
library(ggplot2) #for plotting
library(dplyr) #for data cleaning
library(survival) #for survival analysis
library(boot) #for bootstrapping

library(ggdag)
longdag <- ggdag::dagify(L2 ~ L1,
                         A1 ~ L1,
                         L2 ~ A1,
                         A2 ~ A1,
                         A2 ~ L2,
                         Y2 ~ A2,
                         Y2 ~ A1,
                         Y2 ~ L1,
                         Y2 ~ L2)

#set.seed(2)
set.seed(18)
ggdag::ggdag(longdag)+
  theme_void() +
  ggtitle("Example: Longitudinal DAG")
miners <- read.csv("data/miners.csv") %>%
  mutate(lograd = ifelse(atwork==1, log(rad), NA))
str(miners)

summary(miners[, 2:13])


baseline_data <- filter(miners, intime==0)
N = nrow(baseline_data)

# large sample from observed baseline data
mc_iter = 20000
set.seed(12325)
mcdata <- slice(baseline_data, sample(1:N, size = mc_iter, replace=TRUE))

#Model 1: pooled linear model for log(exposure), while at work
mod_e = lm(lograd ~ outtime + cum_rad_lag1 + smoker, data=filter(miners, atwork==1))

# pooled logistic model for leaving work (at the end of the year), while at work
mod_w = glm(leavework ~ outtime + I(outtime^2) + rad_lag1 + cum_rad_lag1 + smoker, data=filter(miners, a


mod_d = glm(dead ~ outtime + atwork + cum_rad + I(cum_rad*cum_rad) + smoker + smoker*cum_rad, data=miner

N <- nrow(mcdata)
ncols <- ncol(mcdata)
pseudo_cohort <- slice(mcdata,rep(1:N, each=20))
#create a new identifier
pseudo_cohort$gfid <- rep(seq(1, N, 1), each=20) # new id for each pseudo individual
```

```r
# initialize values
pseudo_cohort$intime <- rep(seq(0, (20-1), 1), times=N)
pseudo_cohort$outtime <- pseudo_cohort$intime+1
pseudo_cohort$atwork <- 1
pseudo_cohort$durwork <- 1
pseudo_cohort$leavework <- 0
pseudo_cohort$rad <- 0
pseudo_cohort$rad_lag1 <- 0
pseudo_cohort$cum_rad_lag1 <- 0
pseudo_cohort$durwork_lag1 <- 0
pseudo_cohort$dead <- 0
pseudo_cohort$cum_hazard <- 0 # new cumulative hazard
pseudo_cohort$cum_incidence <- 0 # new cumulative incidence

# set error variance for exposure model
var_e = sum(mod_e$residuals^2)/mod_e$df.residual


# limits on log exposure (keep simulated values in range of observed)
max_e = 7.321276

#for the natural course, there is no intervention so we set it to NULL
intervention <- NULL

head(pseudo_cohort)#check out the first 6 observations of the pseudo-cohort and confirm that initial va

set.seed(12325)
for(t in 1:20){
  # index that keeps track of time
  idx = which(pseudo_cohort$outtime == t)
  #update all values of lagged variables (easy to forget!)
  if(t > 1){
    #lagging the data because of disease latency
    pseudo_cohort[idx,'cum_rad_lag1'] = pseudo_cohort[idx-1,'cum_rad']
    pseudo_cohort[idx,'rad_lag1'] = pseudo_cohort[idx-1,'rad']
  }
  ############
  # leaving work (time varying confounder)
  ############
  if(t==1){
    widx = 1:length(idx) # everyone is at work at first time point
  }
  if(t > 1){
    #simulation below is forward in time, did someone leave work during this time point?
    widx = which(pseudo_cohort[idx-1,'atwork']==1) #work index
    # index keeping track of which workers still work
    ## simulate here from known probability distribution
    pseudo_cohort[idx[widx],'leavework'] <- rbinom(n=length(widx), size=1,
                                                   prob=predict(mod_w, newdata = pseudo_cohort[idx[widx]
                                                                type = 'response'))
    #for everyone else still alive at time t, do they leave work at this time?
    # if worker didn't leave, then assume stay at work
    pseudo_cohort[idx,'atwork'] <- (pseudo_cohort[idx,'leavework']==0 & pseudo_cohort[idx-1,'atwork']==
```

```
  # update at work index to account for workers who left
  widx = which(pseudo_cohort[idx,'atwork']==1)
  #durwork: keeping track of work duration
  pseudo_cohort[idx,'durwork'] <- pseudo_cohort[idx-1,'durwork'] + pseudo_cohort[idx,'atwork']
  pseudo_cohort[idx,'durwork_lag1'] <- pseudo_cohort[idx-1,'durwork']
}
#############
# exposure/interventions
#############
# exposure is assumed log-normally distributed (=predicted value plus draw from the residuals)
# these are predicted values from the upstream process
meanlogr = predict(mod_e, newdata = pseudo_cohort[idx[widx],])
logr = meanlogr + rnorm(n=length(widx), mean=0, sd=sqrt(var_e))
pseudo_cohort[idx[widx],'rad'] <- exp(pmin(log(max_e), logr))
# exposure under different interventions/regimes
if(typeof(intervention) != "NULL"){
  if(is.numeric(intervention)){
    # static, deterministic intervention, e.g. set exposure = 0
    pseudo_cohort[idx,'rad'] <- intervention
  } else{
    if(is.character(intervention)){
      # dynamic interventions are defined by character strings: e.g. 'rad>0.3' means we intervene to
      #  this line creates an index of rows at the current time which are in violation of the interve
      #  exposure will be redrawn for these observations until they are in compliance.
      #  This is equivalent to drawing from a truncated log-normal distribution
      #
      #  Below is rejection sampling; if value is above the regime threshold,
      #  then reject it and sample again. This is more or less equivalent to sampling
      #  from a truncated distribution at regime limit
      viol_idx <- with(pseudo_cohort[idx,], which(eval(parse(text=intervention))))
      while(length(viol_idx)>0){
        # accept/rejection sampling to get draws from truncated log-normal
        # this is how we (I) assume an intervention would be implemented, but
        # we could imagine other ways (e.g. a hard cap on exposure)
        meanlogr = predict(mod_e, newdata = pseudo_cohort[idx[viol_idx],])
        lograd = meanlogr + rnorm(n=length(viol_idx), mean=0, sd=sqrt(var_e))
        pseudo_cohort[idx[viol_idx],'rad'] <- exp(lograd)
        # check whether any are still in violation of the distribution, if so, repeat loop
        viol_idx <- with(pseudo_cohort[idx,], which(eval(parse(text=intervention))))
      }}} # end dynamic intervention
} # end all interventions on exposure
if(t > 1){
  pseudo_cohort[idx,'cum_rad'] = pseudo_cohort[idx-1,'cum_rad'] + pseudo_cohort[idx,'rad']
} else pseudo_cohort[idx,'cum_rad'] = pseudo_cohort[idx,'rad']
#############
# death (simulate discrete hazard as in Taubman et al 2009, rather than 1/0 as in Keil et al 2014)
#  see note at bottom of program about late entry - in the case of late entry
#  the typical approach is to simulate actual death (1/0 variable) and then
#  estimate survival in the pseudo-population using a survival curve estimator
#  that can account for late entry
#############
#below is final prediction of death or not (if not, then keep calculating
#cumulative incidence)
```

```
  #here, we are making mortality predictions from the updated data. All people
  #alive at time t and making predictions based on the simulated data
  pseudo_cohort[idx,'dead'] = predict(mod_d, newdata = pseudo_cohort[idx,], type = 'response')
  if(t > 1){
    # product limit estimator of cumulative incidence (note this is applied on the individual basis)
    pseudo_cohort[idx,'cum_incidence'] = pseudo_cohort[idx-1,'cum_incidence'] +
      (1-pseudo_cohort[idx-1,'cum_incidence'])*pseudo_cohort[idx,'dead']
  } else{
    pseudo_cohort[idx,'cum_incidence'] = pseudo_cohort[idx,'dead']
  }
  #alternative estimator of the cumulative incidence: Breslow estimator
  #if(t > 1){
  #  # Kaplan-meier estimator of cumulative incidence (note this is applied on the individual basis)
  #  pseudo_cohort[idx,'cum_hazard'] = pseudo_cohort[idx-1,'cum_hazard'] + pseudo_cohort[idx,'dead']
  #} else{
  #  pseudo_cohort[idx,'cum_hazard'] = pseudo_cohort[idx,'dead']
  #}
  #  pseudo_cohort[idx,'cum_incidence'] = 1-exp(-pseudo_cohort[idx,'cum_hazard'])
  # could also use Breslow estimator, but need aalen-johansen estimator if there are competing risks
  # the 'cum_incidence' variable is an estimate of the individual risk. We then
  # average that over the population below to get the marginal risk
} # end time loop
head(pseudo_cohort)#check out first 6 observations with new predicted values

cuminc_nc0 <- with(pseudo_cohort, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean)
knitr::kable(cuminc_nc0, caption = "Cumulative incidence: unexposed regime (hard code)")


gformula_risks <- function(intervention=NULL, pseudo_cohort_bl,
                           endtime, mod_e, mod_w, mod_d, seed=NULL){
  N <- nrow(pseudo_cohort_bl)
  ncols <- ncol(pseudo_cohort_bl)
  pseudo_cohort <- slice(pseudo_cohort_bl,rep(1:N, each=endtime))
  pseudo_cohort$gfid <- rep(seq(1, N, 1), each=endtime)
  pseudo_cohort$intime <- rep(seq(0, (endtime-1), 1), times=N)
  pseudo_cohort$outtime <- pseudo_cohort$intime+1
  pseudo_cohort$atwork <- 1
  pseudo_cohort$durwork <- 1
  pseudo_cohort$leavework <- 0
  pseudo_cohort$rad <- 0
  pseudo_cohort$rad_lag1 <- 0
  pseudo_cohort$cum_rad_lag1 <- 0
  pseudo_cohort$durwork_lag1 <- 0
  pseudo_cohort$dead <- 0
  pseudo_cohort$cum_hazard <- 0
  pseudo_cohort$cum_incidence <- 0
  var_e = mean(mod_e$residuals^2)
  set.seed(seed)
  max_e = 7.321276
  for(t in seq(1, endtime, 1)){
    idx = which(pseudo_cohort$outtime == t)
    if(t > 1){
      pseudo_cohort[idx,'cum_rad_lag1'] = pseudo_cohort[idx-1,'cum_rad']
```

```
      pseudo_cohort[idx,'rad_lag1'] = pseudo_cohort[idx-1,'rad']
    }
    if(t==1) widx = 1:length(idx)
    if(t > 1){
      widx = which(pseudo_cohort[idx-1,'atwork']==1)
      pseudo_cohort[idx[widx],'leavework'] <- rbinom(n=length(widx), size=1,
                                                     prob=predict(mod_w, newdata = pseudo_cohort[idx[wi
                                                                  type = 'response'))
      pseudo_cohort[idx,'atwork'] <- (pseudo_cohort[idx,'leavework']==0 & pseudo_cohort[idx-1,'atwork']=
      widx = which(pseudo_cohort[idx,'atwork']==1)
      pseudo_cohort[idx,'durwork'] <- pseudo_cohort[idx-1,'durwork'] + pseudo_cohort[idx,'atwork']
      pseudo_cohort[idx,'durwork_lag1'] <- pseudo_cohort[idx-1,'durwork']
    }
    meanlogr = predict(mod_e, newdata = pseudo_cohort[idx[widx],])
    logr = meanlogr + rnorm(n=length(widx), mean=0, sd=sqrt(var_e))
    pseudo_cohort[idx[widx],'rad'] <- exp(pmin(log(max_e), logr))
    if(typeof(intervention) != "NULL"){
      if(is.numeric(intervention)){
        pseudo_cohort[idx,'rad'] <- intervention
      } else{
        if(is.character(intervention)){
          viol_idx <- with(pseudo_cohort[idx,], which(eval(parse(text=intervention))))
          while(length(viol_idx)>0){
            meanlogr = predict(mod_e, newdata = pseudo_cohort[idx[viol_idx],])
            lograd = meanlogr + rnorm(n=length(viol_idx), mean=0, sd=sqrt(var_e))
            pseudo_cohort[idx[viol_idx],'rad'] <- exp(lograd)
            viol_idx <- with(pseudo_cohort[idx,], which(eval(parse(text=intervention))))
          }}}
    }
    if(t > 1){
      pseudo_cohort[idx,'cum_rad'] = pseudo_cohort[idx-1,'cum_rad'] + pseudo_cohort[idx,'rad']
    } else pseudo_cohort[idx,'cum_rad'] = pseudo_cohort[idx,'rad']
    pseudo_cohort[idx,'dead'] = predict(mod_d, newdata = pseudo_cohort[idx,], type = 'response')
    if(t > 1){
      pseudo_cohort[idx,'cum_incidence'] = pseudo_cohort[idx-1,'cum_incidence'] +
        (1-pseudo_cohort[idx-1,'cum_incidence'])*pseudo_cohort[idx,'dead']
      #individual-level risks calculated using Kaplan-Meier formula and then summed and averaged
    } else{
      pseudo_cohort[idx,'cum_incidence'] = pseudo_cohort[idx,'dead']
    }
  }
  pseudo_cohort
}



nc = gformula_risks(intervention=NULL, pseudo_cohort_bl=mcdata, endtime=20,
                    mod_e=mod_e, mod_w=mod_w, mod_d=mod_d, seed=12325)
gf_cuminc_nc <- with(nc, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean))))
knitr::kable(gf_cuminc_nc, caption = "Cumulative incidence: natural course regime")
```

```
unex = gformula_risks(intervention=0,
                      pseudo_cohort_bl=mcdata,
                      endtime=20,
                      mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                      seed=12325)
gf_cuminc_unex <- with(unex, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean))))
knitr::kable(gf_cuminc_unex, caption = "Cumulative incidence: unexposed regime")

exp3 = gformula_risks(intervention='rad>0.3',
                      pseudo_cohort_bl=mcdata,
                      endtime=20,
                      mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                      seed=12325)
gf_cuminc_exp3 <- with(exp3, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean))))
knitr::kable(gf_cuminc_exp3, caption = "Cumulative incidence: radon limit = 0.3 exposure regime")

exp1 = gformula_risks(intervention='rad>0.1',
                      pseudo_cohort_bl=mcdata,
                      endtime=20,
                      mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                      seed=12325)
gf_cuminc_exp1 <- with(exp1, tibble(time=seq(0,20, 1), ci=c(0, tapply(cum_incidence, outtime, mean))))
knitr::kable(gf_cuminc_exp1, caption = "Cumulative incidence: radon limit = 0.1 exposure regime")


ggplot() +
  geom_line(aes(x=time, y=ci, color="Natural course"), data=gf_cuminc_nc) +
  geom_line(aes(x=time, y=ci, color="Unexposed"), data=gf_cuminc_unex ) +
  geom_line(aes(x=time, y=ci, color="Limit: 0.3"), data=gf_cuminc_exp3 ) +
  geom_line(aes(x=time, y=ci, color="Limit: 0.1"), data=gf_cuminc_exp1 ) +
  scale_y_continuous(name="Cumulative incidence") +
  scale_x_continuous(name="Time", limits=c(0,20)) +
  scale_color_discrete(name="") +
  theme_classic() +
  theme(legend.position = c(0.99,0.01),legend.justification = c(1,0))+
  ggtitle(expression(paste("G-formula estimates of cumulative incidence")))


obscifit <- survfit(Surv(intime, outtime, dead)~1, data=miners)
obsnc <- tibble(time=obscifit$time, ci=1-obscifit$surv)


ggplot() +
  geom_line(aes(x=time, y=ci, color="Natural course"), data=gf_cuminc_nc) +
  geom_line(aes(x=time, y=ci, color="Unexposed"), data=gf_cuminc_unex ) +
  geom_line(aes(x=time, y=ci, color="Limit: 0.3"), data=gf_cuminc_exp3 ) +
  geom_line(aes(x=time, y=ci, color="Limit: 0.1"), data=gf_cuminc_exp1 ) +
  geom_line(aes(x=time, y=ci, color="Observed"), data=obsnc) +
  scale_y_continuous(name="Cumulative incidence") +
  scale_x_continuous(name="Time", limits=c(0,20)) +
  scale_color_discrete(name="") +
  theme_classic() +
```

```
  theme(legend.position = c(0.99,0.01),legend.justification = c(1,0))+
  ggtitle(expression(paste("G-formula estimates of cumulative incidence")))
# now we turn the g-formula algorithm to get risk difference into a function in order to get bootstrap
gformula_effectestimates <- function(data=baseline_data, index=NULL, fdata=miners, endtime=10, seed=NULL
  # for bootstrap samples, when needed: first take a random sample of the
  #  study IDs, with replacement
  # 0) preprocessing: take bootstrap sample if bootstrapping, otherwise use the observed data
  if(is.null(index)){
    # if index is NULL, then just use original data and do nothing here
  } else{
    # bootstrap sample of the baseline data using the vector 'index'
    data = slice(data, index)
    # resample from the full data according to the id variable in the baseline data
    idindex = tibble(reps = table(data$id), id = as.numeric(names(table(data$id))))
    fdata = merge(idindex, fdata, by='id')
    fdata = slice(fdata, rep(1:dim(fdata)[1], times=fdata$reps))
  }
  # 1) fit models to full data/bootstrap sample of the full data
  mod_e = glm(lograd ~ outtime + cum_rad_lag1 + smoker, data=filter(data, atwork==1))
  # pooled logistic model for leaving work (at the end of the year), while at work
  mod_w = glm(leavework ~ outtime + I(outtime^2) + rad_lag1 + cum_rad_lag1 + smoker,
              data=filter(fdata, atwork==1 | leavework==1), family=binomial(link=logit))
  # pooled logistic model for death
  mod_d = glm(dead ~ outtime + atwork + cum_rad + I(cum_rad*cum_rad) + smoker + smoker*cum_rad,
              data=fdata, family=binomial(link=logit))

  # 2) take large MC sample from baseline data
  mc_iter = 20000
  set.seed(seed)
  mcdata <- slice(data, sample(1:N, size = mc_iter, replace=TRUE))
  # 3) simulate probability distributions in large MC sample using model fits from step 1
  nc = gformula_risks(intervention=NULL,
                      pseudo_cohort_bl=mcdata,
                      endtime=endtime,
                      mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                      seed=seed)
  unex = gformula_risks(intervention=0,
                        pseudo_cohort_bl=mcdata,
                        endtime=endtime,
                        mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                        seed=seed)
  exp3 = gformula_risks(intervention='rad>0.3',
                        pseudo_cohort_bl=mcdata,
                        endtime=endtime,
                        mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                        seed=seed)
  exp1 = gformula_risks(intervention='rad>0.1',
                        pseudo_cohort_bl=mcdata, endtime=endtime,
                        mod_e=mod_e, mod_w=mod_w, mod_d=mod_d,
                        seed=seed)
  # 4) summarize data
  ci_nc = as.numeric(with(nc, tapply(cum_incidence, outtime, mean)))
  ci_unex = as.numeric(with(unex, tapply(cum_incidence, outtime, mean)))
```

```
  ci_exp3 = as.numeric(with(exp3, tapply(cum_incidence, outtime, mean)))
  ci_exp1 = as.numeric(with(exp1, tapply(cum_incidence, outtime, mean)))
  # return a vector of risk differences and log-risk ratios at time t=endtime
  # could also calculate for all times
  c(
    rd_exp3_10=c(ci_exp3-ci_nc)[endtime],
    rd_exp1_10=c(ci_exp1-ci_nc)[endtime],
    rd_unex_10=c(ci_unex-ci_nc)[endtime],
    lrr_exp3_10=log(c(ci_exp3/ci_nc)[endtime]),
    lrr_exp1_10=log(c(ci_exp1/ci_nc)[endtime]),
    lrr_unex_10=log(c(ci_unex/ci_nc)[endtime])
  )
}
gf_est <- gformula_effectestimates(data=baseline_data, fdata=miners, endtime=10, seed=NULL)
knitr::kable(gf_est, caption="1 Bootstrapped Resample: Risk Differences")

nbootsamples = 10
boot_samples <- boot(data = baseline_data, fdata=miners, statistic = gformula_effectestimates, R = nboo
knitr::kable(boot_samples$t0, caption="10 Bootstrapped Resamples: Risk Differences")

est = boot_samples$t0 #the risk differences
lci =  boot_samples$t0 - 1.96*apply(boot_samples$t, 2, sd)
uci =  boot_samples$t0 + 1.96*apply(boot_samples$t, 2, sd)

dat1 <- cbind.data.frame(regime = c("Exp3","Exp1","Unex","Exp3","Exp1","Unex"),
                         type = c(rep("Risk Difference",3), rep("Log RR",3)),
                         RD = est, LB=lci, UB = uci) %>%
  mutate(regime = factor(regime, levels=c("Unex", "Exp1","Exp3")))

dat1 %>%
  knitr::kable(caption = "Risk Differences/Log Relative Risks and 95% Bootstrapped Confidence Intervals

#plot risk difference
ggplot(data=subset(dat1, type=="Risk Difference")) +
  geom_point(aes(x=regime, y=RD)) +
  geom_errorbar(aes(ymin = LB, ymax=UB, x=regime))+
  facet_wrap(.~type) +
  ylab("Log RR/Risk Difference") +
  theme_bw() +
  ggtitle("G-Formula-Based Risk Differences Predictions Under Different Treatment Regimes") +
  guides(color='none')



#make table
#exponentiate the log RR to get RRs
dat1 %>%
  filter(type=="Log RR") %>%
  dplyr::rename(RR = RD) %>%
  mutate(across(c(RR,LB,UB), exp)) %>%
  print() %>%
ggplot() +
```

```
  geom_point(aes(x=regime, y=RR)) +
  geom_errorbar(aes(ymin = LB, ymax=UB, x=regime))+
  facet_wrap(.~type) +
  ylab("Risk Ratio") +
  theme_bw() +
  ggtitle("G-Formula-Based Risk Ratio Predictions Under Different Treatment Regimes") +
  guides(color='none') +
  geom_hline(yintercept = 1, linetype="dashed")



boot_samples200 <- boot(data = baseline_data, fdata=miners, statistic = gformula_effectestimates, R = 2

est200 = boot_samples200$t0 #the risk differences
lci200 =  boot_samples200$t0 - 1.96*apply(boot_samples200$t, 2, sd)
uci200 =  boot_samples200$t0 + 1.96*apply(boot_samples200$t, 2, sd)

dat <- cbind.data.frame(regime = c("Exp3","Exp1","Unex","Exp3","Exp1","Unex"),
                        type = c(rep("Risk Difference",3), rep("Log RR",3)),
                        RD = est200, LB=lci200, UB = uci200)
knitr::kable(dat, caption = "Risk Differences/Log Relative Risks and 95% Bootstrapped Confidence Interva

ggplot(data=dat) +
  geom_point(aes(x=regime, y=RD, color=type)) +
  geom_errorbar(aes(ymin = LB, ymax=UB, x=regime, color=type))+
  facet_wrap(.~type) +
  ylab("Log RR/Risk Difference") +
  theme_bw() +
  ggtitle("G-Formula-Based Risk Differences/Log Relative Risk Predictions Under Different Treatment Regi
  theme(legend.position = "bottom")

knitr::purl(input = "longitudinal_gformula.Rmd", output = "longitudinal_gformula.R",documentation = 0)
```

```
knitr::purl(input = "longitudinal_gformula.Rmd", output = "longitudinal_gformula.R",documentation = 0)
```

```
##   |                                                                      |
```

```
## [1] "longitudinal_gformula.R"
```