Adjusting for Random Measurement Error and Misclassification Using SIMEX

Jose Pina-Sánchez and Albert Varela

22 December, 2023

Introduction

This is the second practical exercise from day 2 of the short course 'Adjustment Methods for Data Quality Problems: Missing Data, Measurement Error, and Misclassification.' You can download the associated lecture slides and data here: https://github.com/jmpinasanchez/measurement

In this workshop we will practice using SIMEX (Cook & Stefanski, 1994) and MC-SIMEX (Küchenhoff et al., 2006) to adjust for different types of random measurement error and misclassification. To do so we will employ the package `simex` (Lederer et al., 2019), and a couple of datasets derived from the Cyber Security Breaches Survey and the Magistrates Court Sentencing Survey.

The former of those datasets is a survey conducted with managers from businesses, charities and education establishments across the UK. It records all forms of cyberattacks to which these firms have been exposed, together with participants' knowledge of the problem, preventive strategies they have considered, and some other general characteristics of the company. We will use this data to explore two research questions: i) Which firms are more exposed to phishing attacks? And, ii) What makes their managers more willing to adopt new measures to prevent future cyber security breaches? One of the key controls that we will employ is the company's size. This variable, just like the number of phishing attacks experienced in the last twelve months, are likely affected by memory failures taking the form of multiplicative random errors. We will see how the standard SIMEX method can be used to adjust for these types of measurement errors one by one, or simultaneously. We will also see how SIMEX can be used to adjust for different types of outcome models, such as linear and logit models.

In a follow up to this first exercise you will be asked to replicate the SIMEX adjustments we have explored up to this point on your own, using a new dataset. Answers to this exercise will be made available at the end of the workshop. The exercise is based on the Chronic Bronchitis and Dust Concentration Study, a dataset that has been used extensively on the biomedic literature to illustrate the impact and adjustment of measurement error and misclassifications (Gossl & Kuchenhoff, 2001; Kuchenhoff, Carroll, 1997). The study was conducted between 1960 and 1977 in a Munich workspace to estimate the effect of dust concentration on the probability of developing chronic bronchitis, where the concentration of dust is likely affected by classical measurement error as a result of the unreliability of the measurement process, while the duration of the exposure is likely affected by similar recall errors to those seen in the retrospective questions of the cybersecurity survey.

The second exercise is based on a dataset capturing sentences imposed in the magistrates court against shoplifting offenders. Many of the key aggravating and mitigating factors taken into consideration by magistrates in deciding whether to impose a custodial sentence are recorded in this dataset, amongst them is the mitigating factor of the offender showing genuine remorse. However, there are reasons to believe this and other mitigating and aggravating factors recorded in the dataset are misclassified, as the questionnaire is designed in such a way that potential problems of item-missingness (e.g. the magistrate that imposed the sentence skimming over that section of the questionnaire) are recorded as the aggravating/mitigating factor not featuring in the case, i.e. a potential false negative. There is a rich literature exploring the weight that sentencers attribute to different aggravating and mitigating factors (for the case of remorse see Dhami &

Belton, 2023, or Maslen, 2015). Here, we will explore how robust is the estimated effect of remorse on the probability of receiving a custodial sentence in the presence of different types of misclassification problems.

This exercise will also be followed up with an unsupervised practical using the Bronchitis study. Here you will be asked to adapt your previous adjustments by taking into account a potential problem of misclassification affecting one of the control variables, whether the participant was a smoker, which is is likely subject to social desirability bias. As before, answers for this follow up exercise will be made available at the end of the workshop.

Exercise 1A. Adjusting for recall errors in the Cyber Security Breaches Survey

Here we are going to use a simplified version of the 2023 Cyber Security Breaches Survey, but to do so we first need to sign the Access Agreement for Teaching, so please do so before we proceed. This form is required by the UK Data Service to be able to use their data. In addition, we need to check that we comply with the requirements established by the UK Data Service. In our case, this entails ensuring that the data does not leave this room and is deleted when we complete the course. Because of such data security protocols it won't be possible to reproduce this exercise if you are following the practical remotely.

```
#Importing the data.
cyber = read.csv('cyber.csv')
#Remember to use the address of the folder where you saved the dataset.
```

In this simplified version of the Cyber Security Survey we have just five variables describing 1315 firms that were subject to phishing attacks. These five variables represent: i) a broad definition of the company's *sector* (private sector, charity, or education establishment); ii) the *size* of the firm in terms of number of employees; iii) whether they have a *policy* in place to protect against cyber crime; iv) whether the interviewee considers that new measures are needed to prevent future breaches/attacks (*response*); and v) the number of phishing attacks experienced over the last twelve months (*phishcount*). Let's take a quick look at the dataset.

```
head(cyber)
```

```
##           sector size policy response phishcount
## 1 private sector    3      1        0          1
## 2 private sector    2      0        0          5
## 3 private sector    4      0        0          1
## 4 private sector    2      0        0          2
## 5 private sector    8      0        0          2
## 6 private sector    3      0        0         20
```

```
table(cyber$sector, useNA="ifany")
```

```
##
##        charity      education private sector
##            353            271            691
```

```
summary(cyber)[,2:5]
```

```
##       size           policy        response      phishcount
##  Min.   :  -97   Min.   :0.00   Min.   :0.00   Min.   :  -99
##  1st Qu.:    5   1st Qu.:0.00   1st Qu.:0.00   1st Qu.:    2
##  Median :   40   Median :1.00   Median :0.00   Median :    6
##  Mean   :  490   Mean   :0.62   Mean   :0.26   Mean   :  603
##  3rd Qu.:  200   3rd Qu.:1.00   3rd Qu.:1.00   3rd Qu.:   35
##  Max.   :90000   Max.   :1.00   Max.   :1.00   Max.   :100000
```

Notice how there are some strange negative cases in both *size* and *phishcount*. These reflect instances where

the participant declined to respond or when they responded that they did not know the answer. In this part of the workshop we will set them as missing cases and ignore them using listwise deletion, but when we explore Bayesian adjustments we will get back to this problem to demonstrate how we can adjust for missing data and measurement error simultaneously.

```
#Setting non-responses as missing data.
cyber$size = ifelse(cyber$size<0, NA, cyber$size)
cyber$phishcount = ifelse(cyber$phishcount<0, NA, cyber$phishcount)
```

We conclude our 'data cleaning' by undertaking log-transformations for *size* and *phishcount*. These two variables are heavily right-skewed, which makes their log-transformation a useful strategy to avoid problems with outliers exerting an undue influence. Moreover, taking the natural logarithms of those two variables will help us simplify the measurement error mechanisms to which they are exposed.

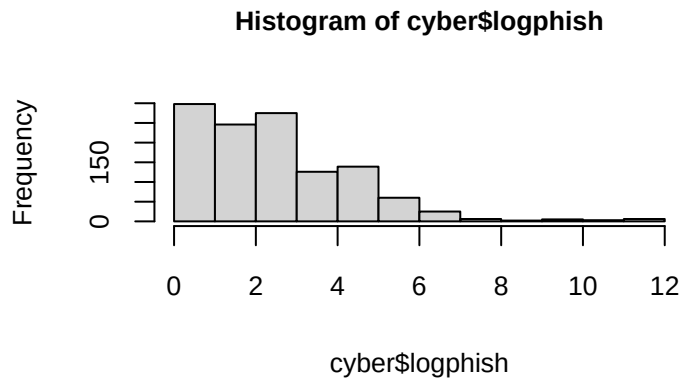
```
#Log-transforming two heavily right-skewed variables.
cyber$logsize = log(cyber$size)
cyber$logphish = log(cyber$phishcount)
```

We theorise that survey reports of both the number of employees in one's company, and the number of phishing attacks experienced, are subject to memory failures, which will likely take the form of random multiplicative errors ($X^* = X \cdot U$). This is because the bigger the actual value (X_i) the harder it is to recall it accurately, hence, the further away the observed value (X_i^*) will be.

The package `simex` has different built-in measurement error models that can be explored. Here we will consider classical errors and misclassifications. As far as I am aware this package does not have an option for multiplicative errors. However, by log-transforming *size* and *phishcount* we can transform their measurement error process into a classical additive error, since $log(X^*) = log(X \cdot U) = log(X) + log(U)$, which means that we will be able to use the standard form of SIMEX to undertake the measurement error adjustments that we anticipate.

We can now proceed to explore our first question: Which types of firms tend to experience more phishing attacks? To do so we will start with a *naive* model where we assume perfectly measured variables. We will use a linear model for convenience, however, notice that even after log-transforming *phishcount* it is still right-skewed. Hence, we should be mindful about the robustness of the standard errors from this model.

```
#Notice that logphish is not normally distributed.
hist(cyber$logphish, cex.lab=0.8, cex.axis=0.8, cex.main=0.8)
```




```
naive_phish = lm(logphish ~ logsize + sector + policy, data=cyber, x=TRUE)
#To use SIMEX we have to include the option 'x=TRUE' in our naive model.
#This is so the model matrix used in the fitting process is included as a component
#of the returned value in our object 'naive_phish'.
```

```r
summary(naive_phish)
```

```
##
## Call:
## lm(formula = logphish ~ logsize + sector + policy, data = cyber,
##     x = TRUE)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.450 -1.478 -0.237  1.155  9.105
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.2628     0.1732    7.29  5.9e-13 ***
## logsize                 0.1900     0.0318    5.98  3.0e-09 ***
## sectoreducation         0.0508     0.1738    0.29  0.77005
## sectorprivate sector    0.4942     0.1420    3.48  0.00052 ***
## policy                  0.3455     0.1277    2.71  0.00693 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.9 on 1116 degrees of freedom
##   (194 observations deleted due to missingness)
## Multiple R-squared:  0.0543, Adjusted R-squared:  0.0509
## F-statistic:   16 on 4 and 1116 DF,  p-value: 9.49e-13
```

We can see that company size is positively associated with the number of phishing attacks. The coefficient is rather small and lower than one, meaning a diminishing marginal rate of phishing attacks as companies get bigger. Education establishments and charities are targeted at roughly the same rate, but private sector firms are attacked at a statistically significant higher rate, a 64% higher rate to be precise (see in the code below how this figure is estimated), which is not really a substantial difference. Having an anti-cybercrime policy in place also has a positive effect. I suspect this is the result of a reverse causal path.

```r
#Calculating the effect of private sector firms in relative terms.
(exp(naive_phish$coefficients[4])-1)*100
```

```
## sectorprivate sector
##                   64
```

Ok, but we anticipated that both *logsize* and *logphish* are subject to classical measurement error. Let's explore the impact those types of errors could have on the estimates from our naive model. We can start by focusing on the errors present in *logsize* first. we will look at *logphish* later since we know that classical errors on the outcome variable will only affect our measures of uncertainty and therefore we can take it as a less problematic form of measurement error.

We anticipated that the errors will be random additive ($X^* = X + U$), but we do not know their standard deviation (σ_U), which means that we do not know the reliability of our error-prone variables (ρ_{X^*}). We could estimate them directly if we were to repeat the same question at different times for a subgroup of our sample. However, since we are using secondary data, we cannot undertake such assessment. The next step should be to consider studies in the literature which have carried out estimations of the reliability of similar questions. However, given the rather niche topic and recent publication of the survey, I do not think there are studies that have explored the validity or reliability of this specific question. We could still try to look for studies that have explored other questions that are nonetheless similar enough in terms of salience and recall timeframe. Here, we will simply try to come up with a range of 'sensible' estimates.

To run SIMEX we will need an estimate of the standard deviation of the error term. This task can be made more

intuitive if we consider instead the reliability ratio of the observed variable ($\rho_{X^*} = \sigma^2_X/\sigma^2_{X^*} = \sigma^2_X/(\sigma^2_X + \sigma^2_U)$), and then derive the standard error of the error term from there. We could also consider that recalling the number of employees should be markedly more accurate than recalling the number of phishing attacks, since the former is more memorable and stable across time, but also because it is a key characteristic of the company that is quite relevant to managers. As such, I suggest exploring reliability ratios for *logsize* between 0.9 and 0.95. To find out the standard deviation of error term that corresponds to each of those reliability ratios we can follow these three steps: i) rearrange the formula of the reliability ratio to derive the variance of the true value ($\sigma^2_X = \rho_{X^*}\sigma^2_{X^*}$); ii) rearrange the variance of the error prone variable to derive the variance of the error term ($\sigma^2_U = \sigma^2_{X^*} - \sigma^2_X$); iii) take the square root of the variance of the error term.

```
#Calculating the sd(U) for a reliability ratio of 0.95.
#VarX = RhoX*VarX*
varX_95 = 0.95 * var(cyber$logsize, na.rm = TRUE)
#VarU = VarX*-VarX
varU_95 = var(cyber$logsize, na.rm = TRUE) - varX_95
sdU_95 = sqrt(varU_95)
#You can use the reliability ratio formula again to check that we got the right
#sd(U) for a 0.95 reliability ratio.
varX_95 / (varX_95 + varU_95)
```

```
## [1] 0.95
```

```
#Calculating the sd(U) for a reliability ratio of 0.90
varX_90 = 0.9 * var(cyber$logsize, na.rm = TRUE)
varU_90 = var(cyber$logsize, na.rm = TRUE) - varX_90
sdU_90 = sqrt(varU_90)
```

We are now ready to undertake a couple of adjustments assuming that *logsize* is affected by classical errors with a reliability ratio of 0.95 and 0.90. Let's start with 0.95 first.

```
#install(simex)  #Make sure you have the simex package installed.
library(simex)
#We should also specify a seed so we can get the same results in our simulations
#of random errors.
set.seed=7
#To undertake the simex adjustment we need to indicate the model to be adjusted,
#the error prone variable, and the standard error of the measurement error term.
adj_size95 = simex(naive_phish, SIMEXvariable="logsize", measurement.error=sdU_95,
                   asymptotic=FALSE)
#We also declare asymptotic=FALSE, so we only get estimates of uncertainty
#derived empirically through the Jackknife method. These take longer to compute,
#but we can expect them to be more accurate since our outcome variable (and
#probably the residuals of the model too) are not normally distributed.
summary(adj_size95)
```

```
## Call:
## simex(model = naive_phish, SIMEXvariable = "logsize", measurement.error = sdU_95,
##     asymptotic = FALSE)
##
## Naive model:
## lm(formula = logphish ~ logsize + sector + policy, data = cyber,
##     x = TRUE)
##
## Simex variable :
##                      logsize
## Measurement error :    0.48
```

```
##
##
## Number of iterations:  100
##
## Residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    -3.5    -1.5    -0.3     0.0     1.1     9.1
##
## Coefficients:
##
## Jackknife variance:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.2180     0.1789    6.81  1.6e-11 ***
## logsize                 0.2039     0.0346    5.90  4.8e-09 ***
## sectoreducation         0.0402     0.1743    0.23  0.81756
## sectorprivate sector    0.5089     0.1427    3.57  0.00038 ***
## policy                  0.3256     0.1291    2.52  0.01182 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

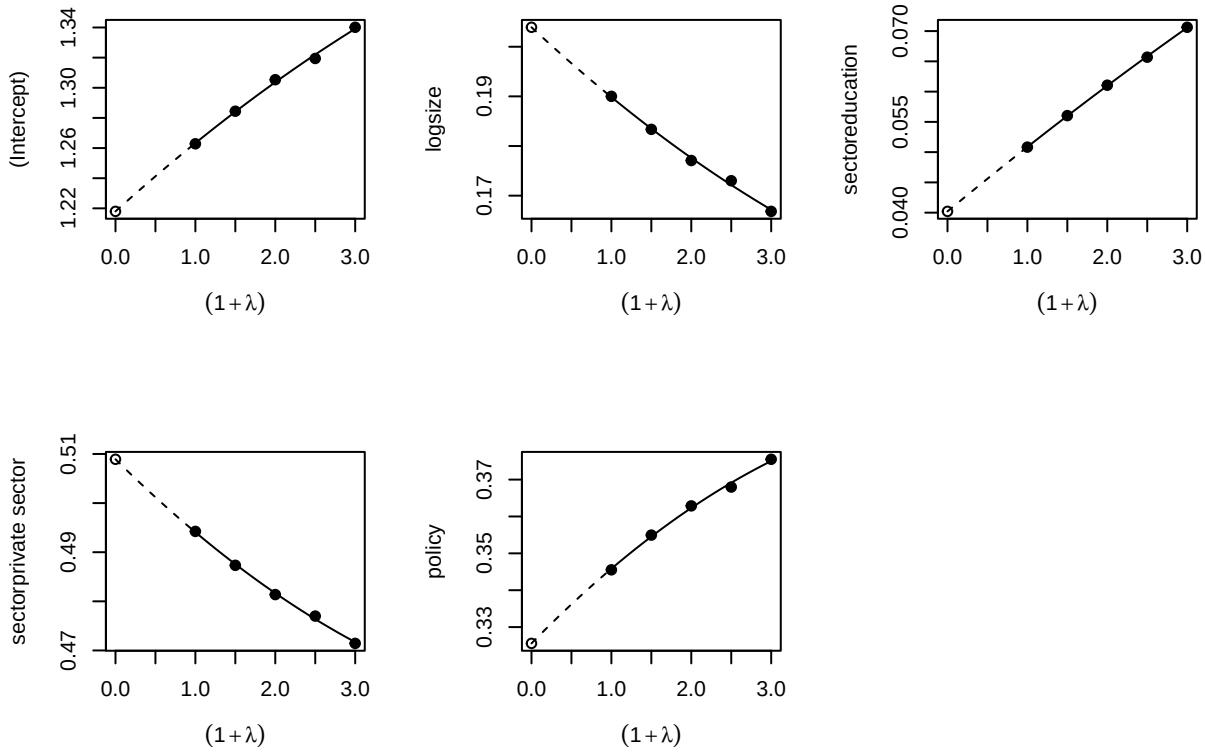
The estimated regression coefficients in our adjustment are relatively similar to those from the naive model. That makes sense since we are assuming a very small share of noise in *logsize*. To compare the two models more clearly we can bring together their `$coefficients` using a table.

```
#Creating a table with estimates from the naive model and the SIMEX adjustment.
results = cbind(naive_phish$coefficients, adj_size95$coefficients)
# Apply formatting to control decimal places.
results = round(results, digits = 3)
colnames(results) = c("naive_phish","adj_size95")
results
```

```
##                       naive_phish adj_size95
## (Intercept)                 1.263       1.22
## logsize                     0.190       0.20
## sectoreducation             0.051       0.04
## sectorprivate sector        0.494       0.51
## policy                      0.346       0.33
```

Before we move forward it is always good practice to check that the simulation-extrapolation process was carried out sensibly. To do so we can plot the simex curves for each of our regression coefficients.

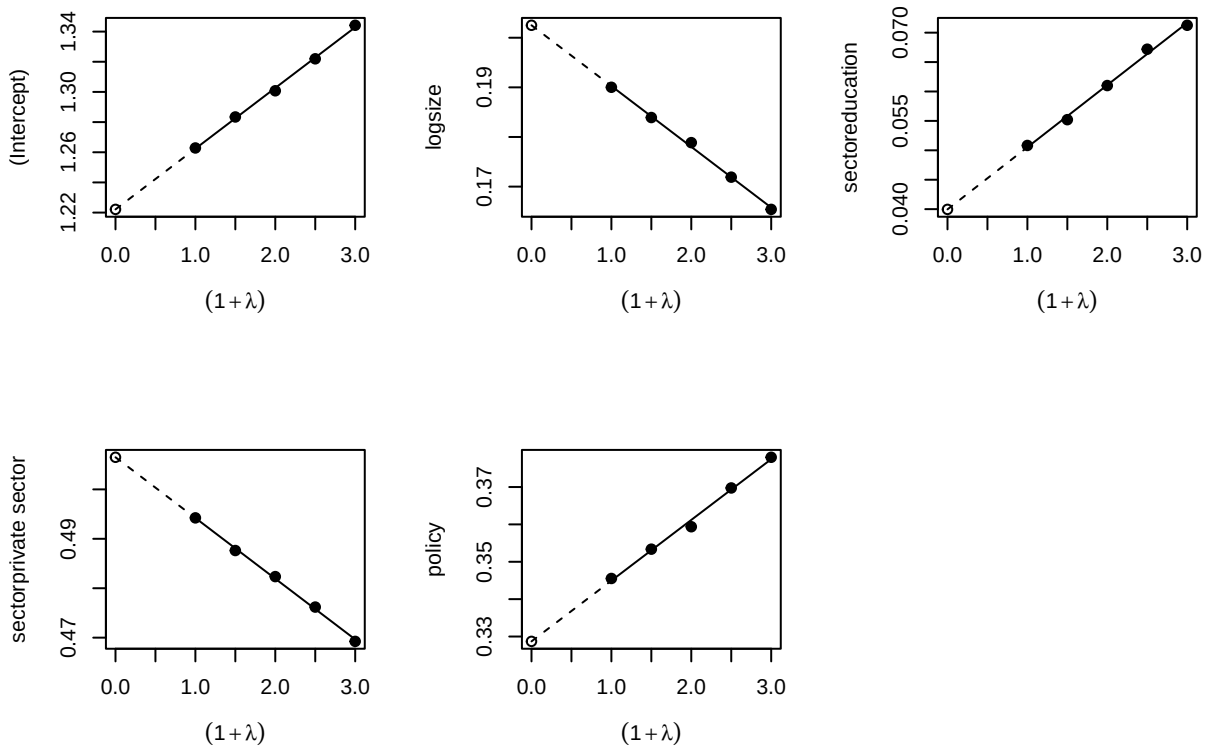
```
#The SIMEX curves for our first adjustment.
plot(adj_size95, mfrow= c(2,3))
```



We can see how for increasing levels of measurement error ($\sigma_U \cdot (1 + \lambda)$) there is a corresponding increasing level of bias, and that this relationship seems stable enough so we can estimate it precisely. In addition, we can see that for most of the model coefficients the relationship between errors and bias is either linear or slightly quadratic, which can be represented well enough using a quadratic function, the function used by default in the `simex` package.

If you wanted to specify a different extrapolation you can use the `fitting.method` option, which allows you to use a `linear` and a `non-linear` function, in addition to the default `quadratic` function. Let's repeat the adjustment considering now a `linear` function.

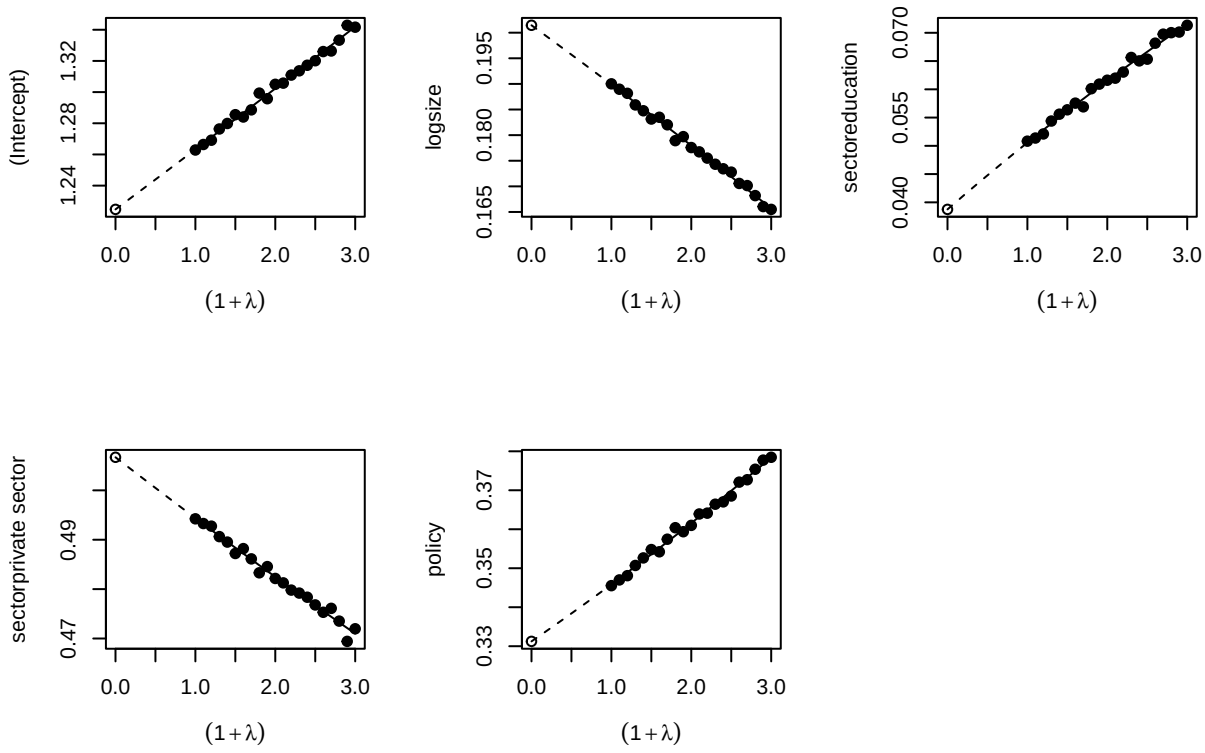
```
#The adjustment assuming a linear SIMEX function.
adj_size95_linear = simex(naive_phish, SIMEXvariable="logsize",
                          measurement.error=sdU_95, asymptotic=FALSE,
                          fitting.method="linear")
plot(adj_size95_linear, mfrow= c(2,3))
```



In our current example, using a `linear` function gives us practically the same results.

One more option worth knowing is `lambda`. We can use this option to increase the number of measurement-error-to-bias points, so we can estimate the SIMEX function more precisely, which should in turn lead to a more precise adjustment. The downside of doing this is the additional computation burden that we add, although for a model as simple, and a dataset as small, as the ones we are using here, it will not be a problem.

```
#The SIMEX adjustment requesting 20 levels of measurement error, instead of the
#standard five levels.
adj_size95 = simex(naive_phish, SIMEXvariable="logsize", measurement.error=sdU_95,
                   asymptotic=FALSE, lambda = seq(.1, 2, by = .1))
plot(adj_size95, mfrow= c(2,3))
```



Ok, let's now undertake the adjustment assuming a 0.90 reliability ratio for *logsize*. Remember that since we do not really know what the reliability ratio is, it is a good idea to use a range of values rather than just one. See if you can do this yourself following two steps: i) calling the SIMEX adjustment as we did before but changing the `measurement.error` option; ii) check that the SIMEX function used seems appropriate; and iii) try to compare results by bringing together the naive model and the two adjustments (with reliability 0.95 and 0.90) in a table.

################## REMOVE THIS FROM THE HANDOUT ##################

```
#The adjustment assuming a reliability ratio of 0.90 for logsize.
adj_size90 = simex(naive_phish, measurement.error=sdU_90,
                   SIMEXvariable = "logsize", asymptotic=FALSE)
summary(adj_size90)
plot(adj_size90, mfrow= c(2,3))
```

We can see how the SIMEX function is clearly quadratic, hence, keeping the default option is the best choice.

```
#Combining results for the naive model and the two adjustments.
results = cbind(naive_phish$coefficients, adj_size95$coefficients,
                adj_size90$coefficients)
results = round(results, digits = 3)
colnames(results) = c("naive_phish","adj_size95","adj_size90")
results
```

```
##                     naive_phish adj_size95 adj_size90
## (Intercept)               1.263      1.225      1.158
## logsize                   0.190      0.201      0.222
## sectoreducation           0.051      0.039      0.024
```

```
## sectorprivate sector       0.494       0.507       0.526
## policy                     0.346       0.331       0.303
```

We can see that for a 0.90 *logsize* reliability, the previously observed biases are accentuated, however their magnitude is still relatively negligible.

########################## UP TO HERE ##########################

To report the impact of measurement error more intuitively we can estimate the relative bias. To do so we first estimate the absolute bias ($Bias = \widehat{\beta_{naive}} - \beta_{true}$) then we express it in relative terms using the true estimate (i.e. our adjustment) as the denominator ($RBias = (Bias/\beta_{true})100\%$).[1] So, assuming our latest adjustment with a 0.90 reliability ratio for *logsize* leads to the true regression coefficients, we can estimate the relative biases in our naive model as follows:

```
#Expressing the impact of measurement error in terms of relative bias.
Bias = naive_phish$coefficients - adj_size90$coefficients
RBias = Bias / adj_size90$coefficients
RBias
```

```
##          (Intercept)              logsize      sectoreducation
##                0.091               -0.144                1.128
## sectorprivate sector               policy
##               -0.061                0.141
```

Leaving aside *sector* : *education* which was not statistically significant and hence unreliably estimated, we can see that the strongest biases are for *policy* and *logsize*, which effects were roughly 13% stronger and weaker, respectively (the exact figure will vary slightly as a result of the simulation process), than what was estimated in the naive model.

Ok, we can now move on to see how SIMEX can be used to adjust for multiple variables affected by measurement error, including the outcome variable. At the start of the exercise we mentioned how we would expect the recall of the number of phishing attacks to be less precise than that of the company's size, so we will consider reliability ratios of 0.8 for *logphish* and 0.90 for *logsize*.

```
#Calculating the sd(U) for logphish and a reliability ratio of 0.80.
#VarX = RhoX*VarX*
varX_80_phish = 0.8 * var(cyber$logphish, na.rm = TRUE)
#VarU = VarX*-VarX
varU_80_phish = var(cyber$logphish, na.rm = TRUE) - varX_80_phish
sdU_80_phish = sqrt(varU_80_phish)
```

```
#The SIMEX adjustment with reliability ratios 0.90 and 0.80 for company's size
#and number of phishing attacks.
adj_size90_phish80 = simex(naive_phish, SIMEXvariable = c("logsize","logphish"),
                     measurement.error = cbind(sdU_90, sdU_80_phish),
                     asymptotic=FALSE)
summary(adj_size90_phish80)
plot(adj_size90_phish80, mfrow= c(2,3))
```

```
#Combining results for the naive model and adjustments of one and two variables
results = cbind(naive_phish$coefficients, adj_size90$coefficients,
                adj_size90_phish80$coefficients)
results = round(results, digits = 3)
```

[1] Another measure commonly used to report the impact of measurement error, missing data, or other unmet assumptions is the root mean squared error. This combines the impact in terms of bias (accuracy) with the impact on the measures of uncertainty (the added variance or loss of precision), $RMSE = \sqrt{Bias^2 + Var}$. The RMSE is particularly useful to assess the impact of unmet assumptions in studies seeking to predict rather than to explain (e.g. what are the number of phishing attacks to be experienced by a given firm?). To explore causal questions (e.g. by how much are phishing attacks reduced following the implementation of a preventive policy?), I would use the RBias as I find it more intuitive.

```r
colnames(results) = c("naive_phish","adj_size90","adj_size90_phish80")
results
```

```
##                       naive_phish adj_size90 adj_size90_phish80
## (Intercept)                 1.263      1.158              1.147
## logsize                     0.190      0.222              0.219
## sectoreducation             0.051      0.024              0.063
## sectorprivate sector        0.494      0.526              0.539
## policy                      0.346      0.303              0.313
```

The adjusted regression coefficients when considering classical measurement error in the outcome variable are quite similar to those observed in our previous adjustment. This is to be expected since classical measurement error in the outcome variable is known to affect only the measures of uncertainty of the model. We can see if that is the case.

```r
#Combining results for the naive model and adjustments of one and two variables
SEs = cbind(summary(naive_phish)$coefficients[,2],
            summary(adj_size90)$coefficients$jackknife[,2],
            summary(adj_size90_phish80)$coefficients$jackknife[,2])
colnames(SEs) = c("naive_phish","adj_size90","adj_size90_phish80")
SEs
```

```
##                       naive_phish adj_size90 adj_size90_phish80
## (Intercept)                 0.173      0.182               0.18
## logsize                     0.032      0.036               0.04
## sectoreducation             0.174      0.174               0.16
## sectorprivate sector        0.142      0.143               0.14
## policy                      0.128      0.129               0.13
```

As expected, we can see larger standard errors when we consider random errors in the outcome variable, but their difference is rather small, so we can safely assume that the recall errors affecting *logphishing* can be safely ignored. Still, we should consider additional combinations of reliability ratios to provide a more complete picture of the potential impact of measurement error. For example, it might be worth reducing the reliability ratio for *logphishing* down to 0.70.

In the interest of time we are going to move on to illustrate another important feature of SIMEX that highlights its versatility; namely, how it can also be adopted to adjust for measurement error in non-linear models. We are going to use a logit model to explore which is the stronger predictor when it comes to convince managers that they need to put in place new preventive measures to protect their companies against cybercrime.

```r
#Estimating the naive logit model.
naive = glm(response ~ logsize + logphish + sector + policy, data=cyber, x=TRUE,
            family="binomial")
summary(naive)
```

```
##
## Call:
## glm(formula = response ~ logsize + logphish + sector + policy,
##     family = "binomial", data = cyber, x = TRUE)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -2.0601     0.2194   -9.39  < 2e-16 ***
## logsize                0.1746     0.0376    4.64  3.5e-06 ***
## logphish               0.0191     0.0353    0.54   0.5876
## sectoreducation       -0.0137     0.2001   -0.07   0.9454
```

```
## sectorprivate sector   0.0277     0.1694    0.16   0.8703
## policy                 0.4298     0.1596    2.69   0.0071 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1292.1  on 1120  degrees of freedom
## Residual deviance: 1242.6  on 1115  degrees of freedom
##   (194 observations deleted due to missingness)
## AIC: 1255
##
## Number of Fisher Scoring iterations: 4
```

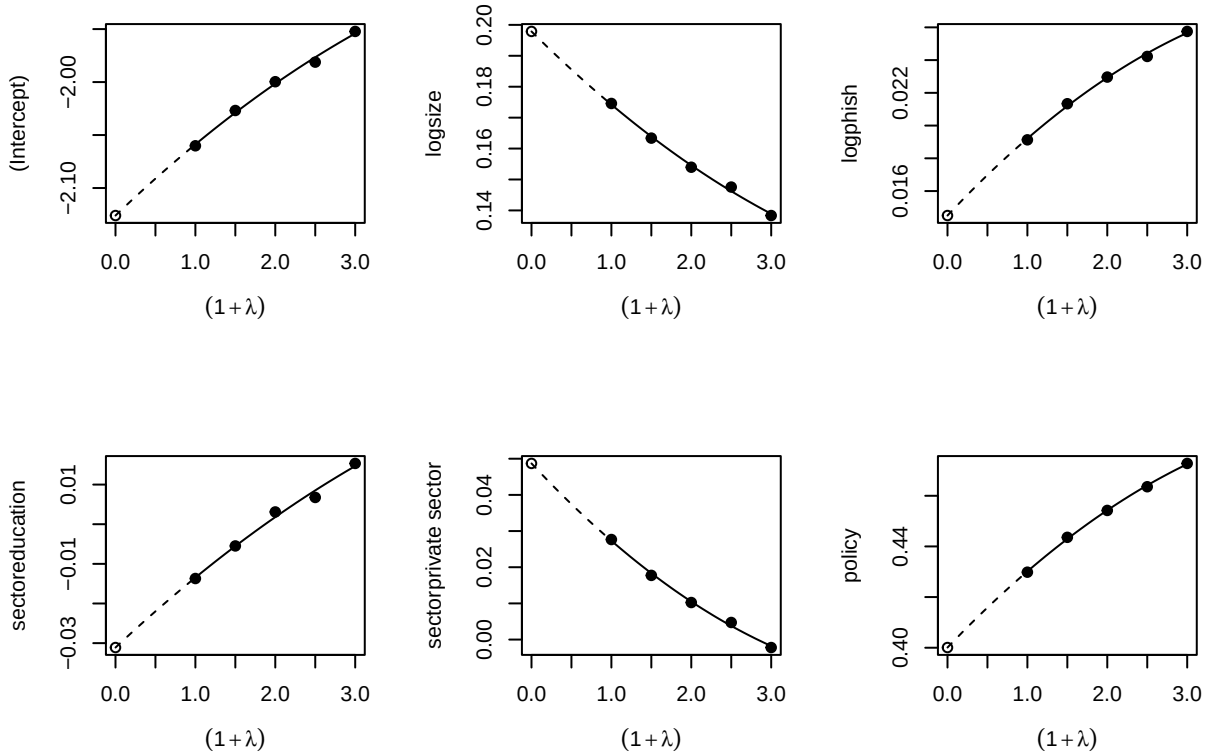
We find that the company's sector does not seem to affect the probability of considering new preventive measures, surprisingly, nor does the number of phishing attacks experienced. This could be potentially explained by having controlled for size, which is positively correlated with the probability of implementing protective measures, and as before, so does the fact of already having a protective policy in place.

```
#The SIMEX adjustment.
adj_size90 = simex(naive, SIMEXvariable = "logsize",
                   measurement.error = sdU_90, jackknife.estimation = FALSE)
#By specifying jackknife.estimation=FALSE we are indicating that we want to
#estimate the asymptotic variance instead, which is faster and should not be
#biased.
summary(adj_size90)
```

```
## Call:
## simex(model = naive, SIMEXvariable = "logsize", measurement.error = sdU_90,
##     jackknife.estimation = FALSE)
##
## Naive model:
## glm(formula = response ~ logsize + logphish + sector + policy,
##     family = "binomial", data = cyber, x = TRUE)
##
## Simex variable :
##                      logsize
## Measurement error :    0.68
##
##
## Number of iterations:  100
##
## Residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   -0.59   -0.29   -0.19    0.00    0.52    0.87
##
## Coefficients:
##
## Asymptotic variance:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -2.1260     0.2210   -9.62  < 2e-16 ***
## logsize                0.1979     0.0425    4.66  3.6e-06 ***
## logphish               0.0145     0.0381    0.38    0.704
## sectoreducation       -0.0311     0.1986   -0.16    0.876
## sectorprivate sector   0.0487     0.1670    0.29    0.771
```

```
## policy                 0.4001     0.1594    2.51    0.012 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

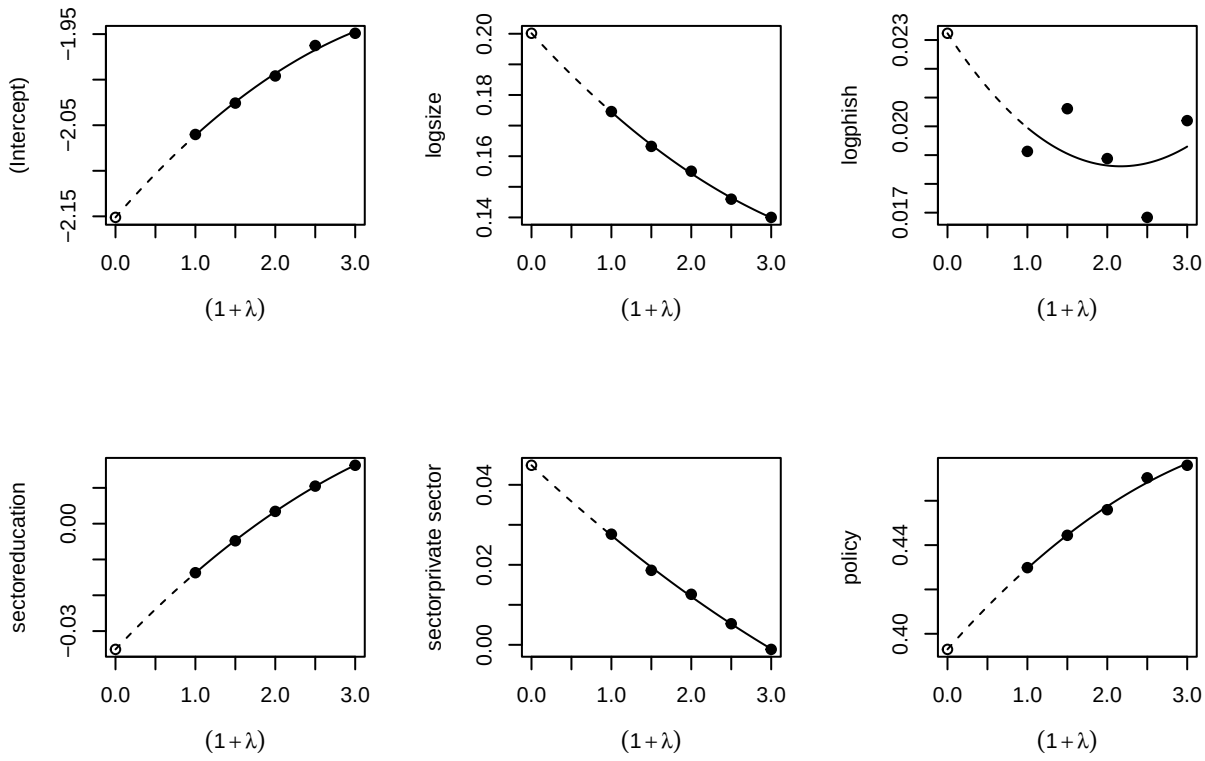
```
plot(adj_size90, mfrow= c(2,3))
```



When we adjust for measurement error in *logsize* we see that its effect had been artificially attenuated, while the effect of *policy* was unduly inflated. However, as before, none of the regression coefficients are substantially impacted to lead us to wrong inferences, hence we can state that our results are robust to measurement error in *logsize*. To be more throrough we can also explore the impact of the measurement error affecting *logphish*. See if you can do this by replicating the code that we used before to adjust for measurement error in multiple variables.

################## REMOVE THIS FROM THE HANDOUT ##################

```
#The SIMEX adjustment
adj_size90_phish80 = simex(naive, SIMEXvariable = c("logsize","logphish"),
                     measurement.error=cbind(sdU_90, sdU_80_phish),
                     jackknife.estimation = FALSE)
summary(adj_size90_phish80)
plot(adj_size90_phish80, mfrow= c(2,3))
```



We see that the errors in *logphish* have practically a null impact. In fact it is worth noticing here how the SIMEX adjustment did not 'work' for *logphish*, as there does not seem to be a clear relationship between increasing levels of measurement error and bias, hence we cannot estimate a reliable SIMEX function and extrapolate to a point of no measurement error. This is probably because the relationship between *logphish* and *response* was already quite noisy (it was not significant), so the addition of more noise does not modify that already non-existent relationship. This is a useful example that illustrate the importance of checking your SIMEX plots before trusting their adjustments blindly.

########################## UP TO HERE ##########################

Exercise 1B. Adjusting for recall errors in dust exposure

In this exercise you are requested to apply the SIMEX adjustments that we have seen so far to a different study exploring the relationship between dust concentration in the working place and the occurrence of chronic bronchitis.

Workplace exposure to toxic particles and their effect in the development of disease or disability represents a vast area of research in occupational studies and epidemiology more generally. Often these kind of studies seek to estimate a tolerable level of exposure to such toxic particles that can be experienced without developing adverse effects, i.e. before they lead to serious diseases or disability. Findings from these types of studies are then used to determine a threshold limiting value (TLV) to guide public policy or sector regulations. Providing accurate estimates is therefore essential. However, as you can expect, many of the variables used in these types of studies are prone to measurement error.

Using what you have learnt so far, see if you could enhance the robustness in the estimation of the effect of dust concentration in the workplace on the development of chronic bronchitis. Let's start by taking a quick look at the data.

```r
#Importing the data.
load(file='bronch.rda')
#Quick exploratory analysis.
head(bronch)
```

```
##   cbr dust smoking expo
## 1   0 0.18       1    5
## 2   0 0.22       1    4
## 3   0 0.22       1    4
## 4   0 0.22       1    4
## 5   0 0.22       1    8
## 6   0 0.22       1    8
```

```r
summary(bronch)
```

```
##       cbr            dust     smoking      expo
##  Min.   :0.00   Min.   :0.2   0:325   Min.   : 3
##  1st Qu.:0.00   1st Qu.:0.4   1:921   1st Qu.:16
##  Median :0.00   Median :0.9           Median :25
##  Mean   :0.23   Mean   :1.1           Mean   :25
##  3rd Qu.:0.00   3rd Qu.:1.8           3rd Qu.:33
##  Max.   :1.00   Max.   :3.2           Max.   :66
```

We can see that 23.4% of the sample developed chronic bronchitis (*cbr*). This will be used as our outcome variable. The main covariate of interest is dust concentration in miligrams per cubic metre (*dust*), which could be expected to be affected by classical measurement error since measurements could vary depending on random factors like - I am speculating here - whether the room was ventilated before the measurements were taken, whether it had been cleaned recently, or the number of people attending the workplace that day. We also have a couple of controls, whether the participant is a smoker (*smoking*), and the level of exposure (*expo*) measured by the reported number of years the participant has been working on that same workplace, which could be affected by similar memory failures to what we have seen in the previous exercise.

```r
#Estimating the naive model
naive_bronch = glm(cbr ~ dust + smoking + expo, x=T, family= binomial,
                   data=bronch)
summary(naive_bronch)
```

```
##
## Call:
## glm(formula = cbr ~ dust + smoking + expo, family = binomial,
##     data = bronch, x = T)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.1754     0.2611  -12.16  < 2e-16 ***
## dust          0.3592     0.0938    3.83  0.00013 ***
## smoking1      0.6815     0.1743    3.91  9.3e-05 ***
## expo          0.0401     0.0062    6.46  1.1e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1356.8  on 1245  degrees of freedom
## Residual deviance: 1279.1  on 1242  degrees of freedom
## AIC: 1287
```

```
##
## Number of Fisher Scoring iterations: 4
```

We can see that the three explanatory variables are statistically significant and pointing in the expected direction. Specifically, we estimate that the association between *dust* and *cbr*, expressed as an odds-ratio is 1.43. That is, everything else being constant, for every additional unit (mg/m^3) of dust in the workplace the odds of developing chronic bronchitis increase by 43%.

```
#The estimated relationship between dust and cbr, according to the naive model.
exp(naive_bronch$coefficients[2])
```

```
## dust
##  1.4
```

You are now requested to undertake SIMEX adjustments to explore the robustness of the above estimate. To do so consider the following four steps: i) calculate a range of sensible standard deviations for the measurement error in *dust*; ii) undertake SIMEX adjustments based on the different levels of measurement error in *dust* you have considered; iii) try to bring together your regression coefficients in a table to compare them more clearly, if you want even try to calculate the relative bias for the relationship of interest; iv) repeat the same process by considering measurement error in *expo* too.

################## REMOVE THIS FROM THE HANDOUT ##################

```
#Calculating the sd(U) in dust for a reliability ratio of 0.90.
#VarX = RhoX*VarX*
varX_90 = 0.9 * var(bronch$dust, na.rm = TRUE)
#VarU = VarX*-VarX
varU_90 = var(bronch$dust, na.rm = TRUE) - varX_90
sdU_90 = sqrt(varU_90)
#Calculating the sd(U) in dust for a reliability ratio of 0.80.
#VarX = RhoX*VarX*
varX_80 = 0.80 * var(bronch$dust, na.rm = TRUE)
#VarU = VarX*-VarX
varU_80 = var(bronch$dust, na.rm = TRUE) - varX_80
sdU_80 = sqrt(varU_80)
```

```
#SIMEX adjustment for a dust reliability ratio of 0.90.
adj_dust90 = simex(naive_bronch, SIMEXvariable="dust", measurement.error=sdU_90,
                   jackknife.estimation = FALSE)
#summary(adj_dust90)
#plot(adj_dust90, mfrow= c(2,2))
```

```
#SIMEX adjustment for a dust reliability ratio of 0.90.
adj_dust80 = simex(naive_bronch, SIMEXvariable="dust", measurement.error=sdU_80,
                   jackknife.estimation = FALSE)
#summary(adj_dust80)
#plot(adj_dust80, mfrow= c(2,2))
```

```
#Combining results for the naive model and adjustments of a 0.90 and a 0.80
#reliability ratio for dust.
results = cbind(naive_bronch$coefficients, adj_dust90$coefficients,
                adj_dust80$coefficients)
colnames(results) = c("naive_bronch","adj_dust90","adj_dust80")
results = round(results, digits = 3)
results
```

```
##             naive_bronch adj_dust90 adj_dust80
## (Intercept)        -3.17      -3.21      -3.25
```

```
## dust                0.36       0.39       0.43
## smoking1            0.68       0.68       0.68
## expo                0.04       0.04       0.04
```

```
#Calculating the relative bias when the reliability of dust is 0.80.
Bias = naive_bronch$coefficients - adj_dust80$coefficients
RBias = Bias / adj_dust80$coefficients
RBias
```

```
## (Intercept)        dust    smoking1        expo
##     -0.0243     -0.1622     -0.0044      0.0027
```

The effect of dust concentration is considerably stronger (14% stronger) than previously assumed under a naive model. Hence, tolerance levels should be reduced in any regulation that was adopted based on this study. How much should acceptable tolerance levels being lowered depends on how reliable dust measurements truly are. If we can safely assume that they are 80% reliable we could suggest roughly a 14% reduction in the tolerance level.

We can also examine the impact stemming from years of exposure being also affected by measurement error in the form of recall errors. To adjust for those errors we can follow the modelling strategy we undertook in the previous exercise and log-transform this variable before we introduce it in our model.

```
#Log-transforming exposure.
bronch$logexpo = log(bronch$expo)
#The naive model including the log-transformed version of exposure.
naive_bronch = glm(cbr ~ dust + smoking + logexpo, x=T, family= binomial,
                   data=bronch)
summary(naive_bronch)
```

```
##
## Call:
## glm(formula = cbr ~ dust + smoking + logexpo, family = binomial,
##     data = bronch, x = T)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -5.2818     0.5262  -10.04  < 2e-16 ***
## dust          0.3469     0.0939    3.69  0.00022 ***
## smoking1      0.6657     0.1742    3.82  0.00013 ***
## logexpo       1.0061     0.1510    6.66  2.7e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1356.8  on 1245  degrees of freedom
## Residual deviance: 1270.8  on 1242  degrees of freedom
## AIC: 1279
##
## Number of Fisher Scoring iterations: 4
```

I suggest considering a 90% reliability ratio since recalling the number of years working in the same place should be fairly accurate.

```
#Calculating the sd(U) for a reliability ratio of 0.90 for logexpo.
#VarX = RhoX*VarX*
varX_90_logexpo = 0.9 * var(bronch$logexpo, na.rm = TRUE)
#VarU = VarX*-VarX
```

```
varU_90_logexpo = var(bronch$logexpo, na.rm = TRUE) - varX_90_logexpo
sdU_90_logexpo = sqrt(varU_90_logexpo)
```

```
#SIMEX adjustment for a 0.80 reliability in dust and 0.90 in logexpo.
adj_dust80_expo90 = simex(naive_bronch, SIMEXvariable=c("dust","logexpo"),
                          measurement.error=cbind(sdU_80, sdU_90_logexpo),
                          jackknife.estimation = FALSE)
#summary(adj_dust80_expo90)
#plot(adj_dust80_expo90, mfrow= c(2,3))
```

```
#Combining results for the naive model and the adjustment with 0.80 and 0.90
#reliability ratios for dust and logexpo.
results = cbind(naive_bronch$coefficients, adj_dust80_expo90$coefficients)
colnames(results) = c("naive_bronch","adj_dust80_expo90")
results
```

```
##             naive_bronch adj_dust80_expo90
## (Intercept)        -5.28             -5.77
## dust                0.35              0.42
## smoking1            0.67              0.67
## logexpo             1.01              1.13
```

```
#Calculating the relative bias when the reliability of dust is 0.80
Bias = naive_bronch$coefficients - adj_dust80_expo90$coefficients
RBias = Bias / adj_dust80_expo90$coefficients
RBias
```

```
## (Intercept)        dust    smoking1     logexpo
##     -0.0850     -0.1789     -0.0053     -0.1110
```

We observe the expected attenuation effect for *logexpo*, but similar effects for all other variables, including *dust*. Hence, we can conclude that the attenuation effect is slightly less pronounced if we take into account the possible recall errors in exposure, but only marginally so. The effect of dust concentration on developing chronic bronchitis remains biased downwards by more than 10% if we ignore the problem of measurement error in our dataset.

########################## UP TO HERE ##########################

Exercise 2A. Assessing the robustness of the mitigating effect of remorse when it is subject to misclassification

In this exercise we are going to estimate the extent to which showing remorse mitigates sentence severity. To do so we are going to use the dataset 'theft_MCSS_simplified.csv'. This is a simplified version of the Magistrates Court Sentencing Survey, which was compiled by the Sentencing Council for England and Wales. Magistrates across different courts were sampled and asked to report the sentence imposed and the main characteristics of the case.

```
#Importing the data.
mcss = read.csv('theft_MCSS_simplified.csv')
```

The simplified dataset that we will be using is composed of 2116 cases and seven variables. The first three variables capture different aggravating factors listed in the sentencing guidelines. The following two variables represent mitigating factors also from the guidelines. Amongst them is *remorse*, our main variable of interest. In addition, we have a variable capturing the gender of the offender (*male*), and our outcome variable, whether the offender received a custodial sentence (*custody*).

```
head(mcss)
```

```
##   injury prev_cons on_bail mental_disorder remorse male custody
## 1      0         1       0               0       1    1       0
## 2      0         0       0               0       0    1       0
## 3      0         1       0               0       1    0       0
## 4      0         1       0               0       1    1       0
## 5      1         1       0               0       0    0       1
## 6      0         1       0               0       0    0       1
```

```
summary(mcss)
```

```
##      injury       prev_cons       on_bail     mental_disorder    remorse
##  Min.   :0.00   Min.   :0.00   Min.   :0.00   Min.   :0.00    Min.   :0.00
##  1st Qu.:0.00   1st Qu.:1.00   1st Qu.:0.00   1st Qu.:0.00    1st Qu.:0.00
##  Median :0.00   Median :1.00   Median :0.00   Median :0.00    Median :0.00
##  Mean   :0.01   Mean   :0.84   Mean   :0.17   Mean   :0.05    Mean   :0.16
##  3rd Qu.:0.00   3rd Qu.:1.00   3rd Qu.:0.00   3rd Qu.:0.00    3rd Qu.:0.00
##  Max.   :1.00   Max.   :1.00   Max.   :1.00   Max.   :1.00    Max.   :1.00
##       male         custody
##  Min.   :0.00   Min.   :0.00
##  1st Qu.:0.00   1st Qu.:0.00
##  Median :1.00   Median :0.00
##  Mean   :0.72   Mean   :0.29
##  3rd Qu.:1.00   3rd Qu.:1.00
##  Max.   :1.00   Max.   :1.00
```

When inspecting the data we can see that all the variables are binary. We can also see that 29% of sentences imposed are custodial sentences, 72% of offenders are male, and judges only took into account offenders' remorse in just 16% cases. This strikes me as a relatively low prevalence of remorse, especially if we consider that roughly two/thirds of offenders sentenced in the magistrates' court plead guilty (Ministry of Justice, 2023). I propose that the recording of this mitigating factor is susceptible to false negatives, a point that becomes clearer upon closer examination of the questionnaire used to collect the data.

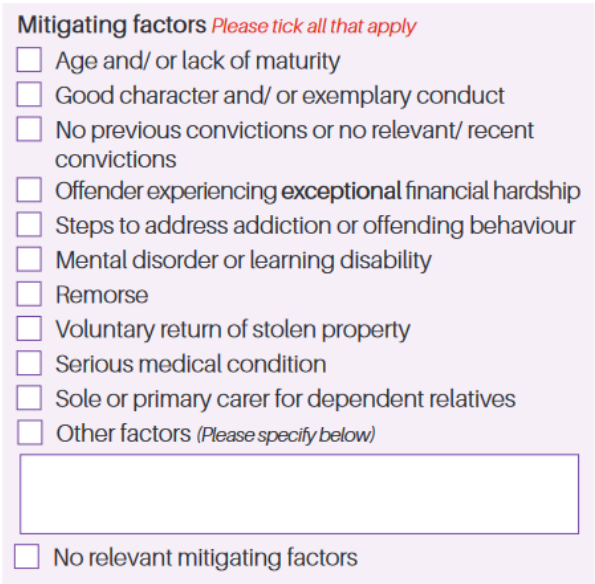


Figure 1: MCSS questionnaire: Mitigating factors

The MCSS questionnaire is available in the same webpage where the original data is stored, here. Below I have included a snapshot of the specific question used to retrieve the mitigating factors present in the case. As you can see the question follows the format of a 'multi select multiple choice question'. This means that unless the option 'No relevant mitigating factors' is ticked, we cannot be sure whether the absence of a tick in a given factor (including *remorse*) is due to that mitigating factor not being present in that case, or as a result of item-missingness (i.e. non-response), which could take place when the questionnaire is filled in a rush.

To explore the robustness of the association between *remorse* and the probability of receiving a custodial sentence, when false negatives are present in *remorse*, we should start - as always - estimating our naive model.

```
#To be able to run mc.simex we need to make sure the misclassified variable is
#of class factor.
mcss$remorse = as.factor(mcss$remorse)
#I use the symbol '.' to take all other variables in the dataset as explanatory
#variables in the model, which simplifies the code.
naive = glm(custody ~ ., x=T, family= binomial, data=mcss)
summary(naive)
```

```
##
## Call:
## glm(formula = custody ~ ., family = binomial, data = mcss, x = T)
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -3.340      0.311  -10.75  < 2e-16 ***
## injury             1.992      0.517    3.85  0.00012 ***
## prev_cons          2.265      0.303    7.49  7.0e-14 ***
## on_bail            1.644      0.129   12.75  < 2e-16 ***
## mental_disorder   -1.464      0.336   -4.35  1.3e-05 ***
## remorse1          -0.777      0.175   -4.44  9.0e-06 ***
## male               0.223      0.123    1.81  0.06983 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2552.5  on 2115  degrees of freedom
## Residual deviance: 2137.6  on 2109  degrees of freedom
## AIC: 2152
##
## Number of Fisher Scoring iterations: 6
```

We can see that all the factors included in the sentencing guidelines point in the expected direction, aggravating factors increase the probability of receiving a custodial sentence and mitigating factors reduce that probability. In addition, we can also see how the *male* effect is non-sginificant, which suggests that judges do not take the offender's gender into account, i.e. there are no unwarranted gender disparities.

To assess the proportion of cases where *remorse* is wrongly coded as a 0, when in fact should be considered a missing value, we can look at 'single-choice questions' included in the same questionnaire that allow for the recording of a non-response. A good example is the question on 'Culpability and Harm category'. These are not included in the simplified dataset that we are using, but if you download the original data you will find that 7.2% of questions about culpability and 12.7% about harm were left unanswered.

The lower response rate for harm is probably a result of appearing after culpability in the questionnaire. If so, the presence of missing cases for *remorse* could be placed at a minimum of 20%, possibly more, given how

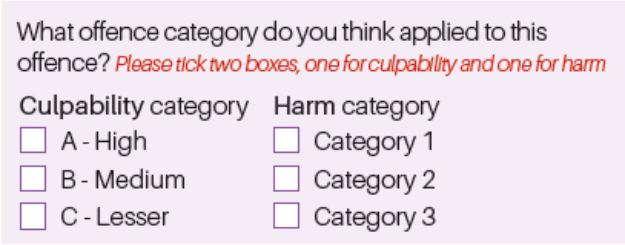


Figure 2: MCSS questionnaire: Culpability and Harm category

buried it is in the questionnaire. However, only a fraction of those missing cases should be interpreted as indicating the presence remorse; the remainder will be missing when remorse was truly absent, leading to a correct classification, even if accidentally. So, I would suggest considering 5% and 10% false negative rates.

Let's try then two adjustments, one assuming a 90% sensitivity (i.e. 10% false negatives), and another for an 80% sensitivity (i.e. 20% false negatives). We need to specify these two matrices so we can determine the MC.SIMEX adjustment for the expected level of misclassification in *remorse*.

This can be a bit confusing at first, but it gets clearer once you practice it a few times. It helps if we think first how a general 2X2 misclassification matrix looks like, which we can then use as a template for our own adjustments.

```
##      X=0              X=1
## X*=0 "Specificity"    "False negative"
## X*=1 "False positive" "Sensitivity"
```

```
#Specifying the misclassification matrix for a 90% specificity.
mc.remorse_90SN = matrix(c(1,0,0.10,0.90),nrow=2)
#To run mcsimex we need the same labels from the misclassified factor used as
#row and column labels in the misclassification matrix.
dimnames(mc.remorse_90SN) = list(levels(mcss$remorse), levels(mcss$remorse))
#We should always check that we got the misclassification matrix that we wanted.
mc.remorse_90SN
```

```
##   0   1
## 0 1 0.1
## 1 0 0.9
```

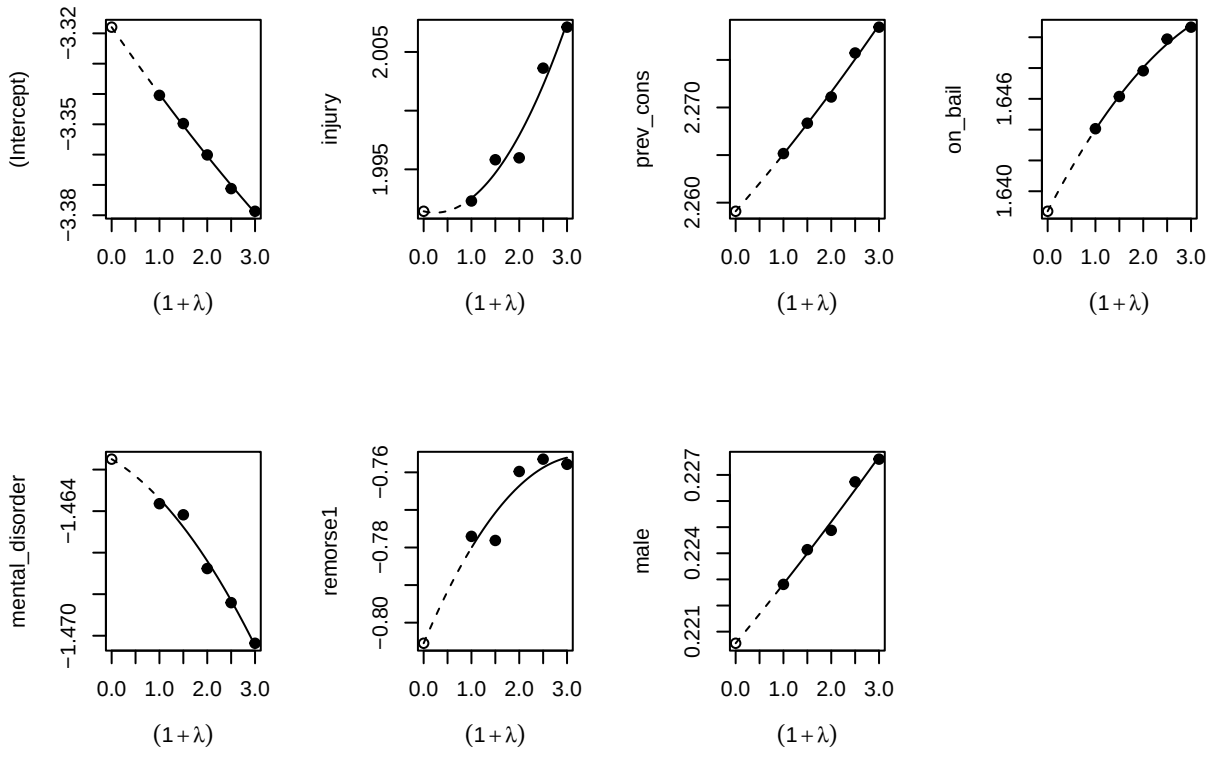
Now we are ready to run the MC-SIMEX adjustment. The only difference with the standard SIMEX adjustments that we have already undertaken is that we now use the command `mcsimex` instead of `simex`, and the option `mc.matrix` instead of `measurement.error`.

```
adj_rem90SN  = mcsimex(naive, mc.matrix = mc.remorse_90SN, SIMEXvariable = "remorse",
                       jackknife.estimation = FALSE)
summary(adj_rem90SN)
```

```
## Call:
## mcsimex(model = naive, SIMEXvariable = "remorse", mc.matrix = mc.remorse_90SN,
##     jackknife.estimation = FALSE)
##
## Naive model:
## glm(formula = custody ~ ., family = binomial, data = mcss, x = T)
##
## Simex variable : remorse
## Misclassification matrix:
##   0   1
```

```
## 0 1 0.1
## 1 0 0.9
##
## Number of iterations:  100
##
## Residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   -0.76   -0.30   -0.16    0.00    0.31    0.99
##
## Coefficients:
##
## Asymptotic variance:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -3.318      0.315  -10.55  < 2e-16 ***
## injury             1.991      0.468    4.25  2.2e-05 ***
## prev_cons          2.259      0.309    7.30  4.0e-13 ***
## on_bail            1.639      0.128   12.80  < 2e-16 ***
## mental_disorder   -1.461      0.355   -4.12  4.0e-05 ***
## remorse1          -0.805      0.176   -4.58  4.9e-06 ***
## male               0.221      0.120    1.83    0.067 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(adj_rem90SN , mfrow= c(2,4))
```



We can see that the effect of *remorse* is only marginally attenuated when contemplating a 90% specificity in that variable. Furthermore, we can see that there is not a clear SIMEX function, which suggests the

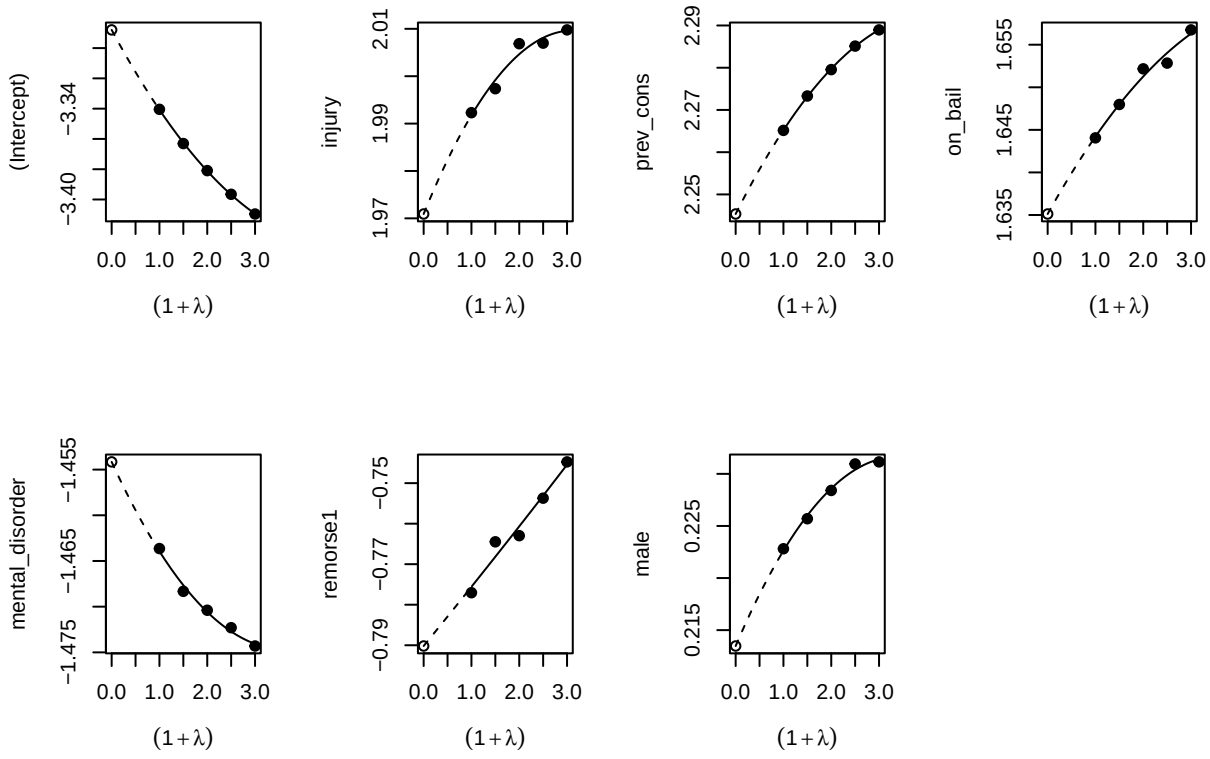
adjustment isunreliable. It is possible that you got a rather different adjustment than the one shown here because of that unreliability and the uncertainty inherent to the simulation process. Can you now run another adjustment on your own for the case of a 80% sensitivity in *remorse*?

################## REMOVE THIS FROM THE HANDOUT ##################

```
#The misclassification matrix for a 80% sensitivity.
mc.remorse_80SN = matrix(c(1,0,0.20,0.80),nrow=2)
dimnames(mc.remorse_80SN) = list(levels(mcss$remorse), levels(mcss$remorse))
mc.remorse_80SN
```

```
##   0   1
## 0 1 0.2
## 1 0 0.8
```

```
adj_rem80SN = mcsimex(naive, mc.matrix = mc.remorse_80SN, SIMEXvariable = "remorse",
              jackknife.estimation = FALSE)
summary(adj_rem80SN )
plot(adj_rem80SN , mfrow= c(2,4))
```



Again, the adjustment is not too reliable, and in any case it only points at a marginal attenuation bias in the effect of *remorse* on custody. Hence, we can conclude that the effect we have attributed to *remorse* on the probability of receiving a custodial sentence is robust to a potential problem of false negatives in the recording of *remorse*.

########################## UP TO HERE ##########################

Let's now proceed by assuming that the variable *male* is also misclassified. This is a hypothetical problem. As far as I am aware this variable is accurately recorded in the MCSS. However, when that information is

not available, researchers have sought to derive demographic characteristics like gender or ethnicity using subject's names (see for example Pina-Sánchez et al., 2019). We could expect such measures to be prone to some degree of misclassification. For example, Alex, Jordan, Jamie, are names commonly used for men and women in the UK.

If the above rationale seems plausible, we could explore the impact of both *male* and *remorse* being misclassified. For example, we could consider the impact of a 90% specificity and sensitivity in *male*, while keeping the previously considered 90% sensitivity for *remorse*. However, before we can conduct the adjustment we need to turn *male* into a factor and re-estimate our naive model.

```
mcss$male = as.factor(mcss$male)
naive = glm(custody ~ ., x=T, family= binomial, data=mcss)
summary(naive)
```

```
##
## Call:
## glm(formula = custody ~ ., family = binomial, data = mcss, x = T)
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -3.340      0.311  -10.75  < 2e-16 ***
## injury             1.992      0.517    3.85  0.00012 ***
## prev_cons          2.265      0.303    7.49  7.0e-14 ***
## on_bail            1.644      0.129   12.75  < 2e-16 ***
## mental_disorder   -1.464      0.336   -4.35  1.3e-05 ***
## remorse1          -0.777      0.175   -4.44  9.0e-06 ***
## male1              0.223      0.123    1.81  0.06983 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2552.5  on 2115  degrees of freedom
## Residual deviance: 2137.6  on 2109  degrees of freedom
## AIC: 2152
##
## Number of Fisher Scoring iterations: 6
```

We can then specify our new misclassification matrix for *male*.

```
#The misclassification matrix for a 90% sensitivity and 90% specificity in male.
mc.male_90SN90SP = matrix(c(0.90,0.10,0.10,0.90),nrow=2)
dimnames(mc.male_90SN90SP) = list(levels(mcss$male), levels(mcss$male))
mc.male_90SN90SP
```

```
##     0   1
## 0 0.9 0.1
## 1 0.1 0.9
```

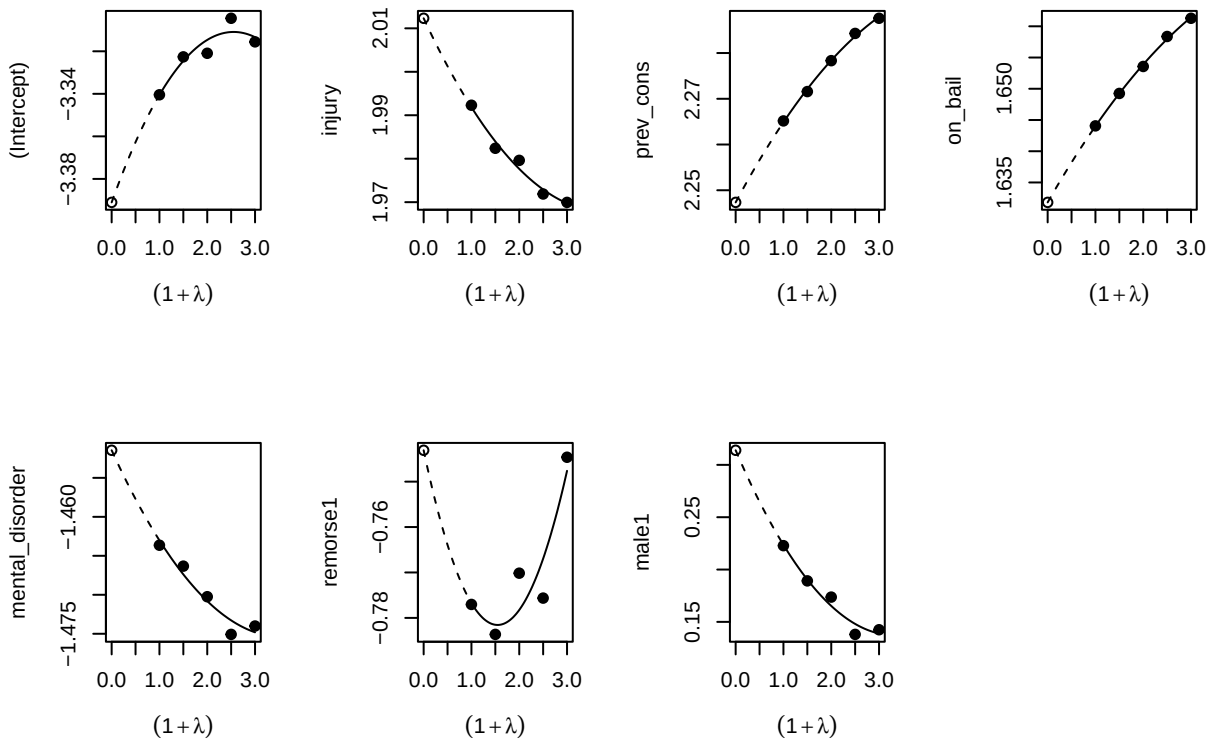
At this point we can undertake the adjustment considering two misclassified variables.

```
adj = mcsimex(naive, mc.matrix = list(remorse = mc.remorse_90SN,
              male = mc.male_90SN90SP), SIMEXvariable = c("remorse","male"),
              jackknife.estimation = FALSE)
summary(adj)
```

```
## Call:
## mcsimex(model = naive, SIMEXvariable = c("remorse", "male"),
```

```
##     mc.matrix = list(remorse = mc.remorse_90SN, male = mc.male_90SN90SP),
##     jackknife.estimation = FALSE)
##
## Naive model:
## glm(formula = custody ~ ., family = binomial, data = mcss, x = T)
##
## Simex variable : remorse male
## Misclassification matrix:
##   0   1
## 0 1 0.1
## 1 0 0.9
##     0   1
## 0 0.9 0.1
## 1 0.1 0.9
##
## Number of iterations:  100
##
## Residuals:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   -0.77   -0.30   -0.15    0.00    0.31    0.99
##
## Coefficients:
##
## Asymptotic variance:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -3.391      0.325  -10.43  < 2e-16 ***
## injury             2.012      0.479    4.20  2.8e-05 ***
## prev_cons          2.247      0.310    7.26  5.5e-13 ***
## on_bail            1.632      0.128   12.70  < 2e-16 ***
## mental_disorder   -1.451      0.354   -4.10  4.3e-05 ***
## remorse1          -0.743      0.177   -4.20  2.8e-05 ***
## male1              0.314      0.169    1.86    0.063 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(adj, mfrow= c(2,4))
```



While the estimated effect of *remorse* remains unaffected, we now observe that even a minor degree of misclassification in *male* can introduce a substantial attenuation bias in its estimated effect. Nevertheless, it is important to note that, in the results obtained here, the adjusted *male* effect does not reach statistical significance. However, because of the uncertainty associated with the simulation process you might have found that *male* became significant in your adjustment, which would be pointing at unwarranted disparities in sentencing. If you want to be more certain about this result you can increase the number of lambda points, as we did in Exercise 1A.

Exercise 2B. Misclassification in the dust exposure study.

In this last exercise you are asked to go back to the bronchitis study to contemplate the impact of potentially misclassified variables. Lederer and Kuchenhoff (2006) refer to studies in the literature suggesting that about 8% of smokers self-report them as non-smokers. Would you be able to adjust for this type of misclassification using the naive model that we estimated in Exercise 1B and the `mcsimex` adjustments we have learnt in Exercise 2B? Can you conclude that the association between dust concentration and development of chronic bronchitis is robust to such a problem of misclassification in self-reported smoking?

Next, we can consider that the outcome variable *cbr* is also subject to misclassification, as it is often the case when diagnosing any kind of complex disease. If you had access to a validation subsample (e.g. a more thorough medical examination undertaken for 10% of the participants in the sample), which informs your that the sensitivity and specificity rates for diagnostics of chronic bronchitis based on the methods used in the bronchitis study are 0.90 and 0.80 respectively, would you be able to assess the robustness of the effect of dust concentration in the presence of misclassified *cbr* and *smoking* simultaneously?

If possible see if you can express the biases you detect in terms of relative bias so you can report their impact more clearly.

```r
#Importing the data
load(file='bronch.rda')
summary(bronch)
#I set cbr as a factor so it can be adjusted using mcsimex
bronch$cbr = as.factor(bronch$cbr)
#The naive model
naive = glm(cbr ~ dust + smoking + expo, x=T, family= binomial, data=bronch)
```

################## REMOVE THIS FROM THE HANDOUT ##################

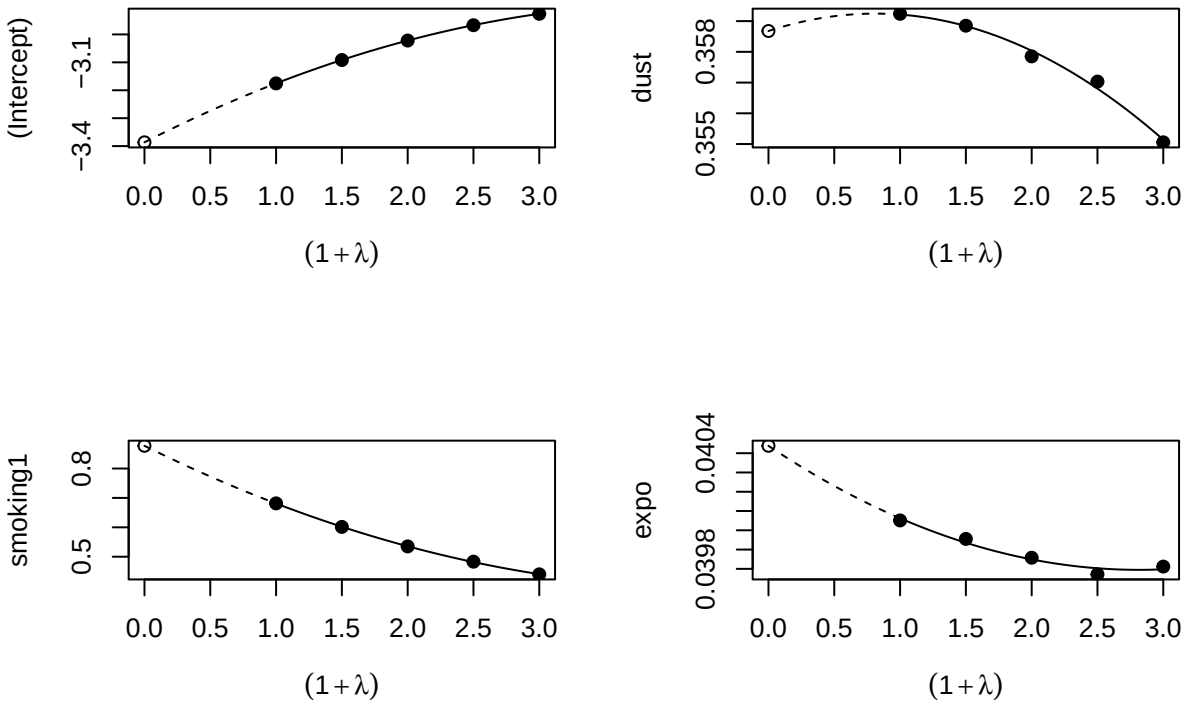
```r
#I find it useful to start by drawing the general 2x2 misclassfication matrix.
#So I can match the right probability in each cell more clearly.
matrix_data = matrix(c("Specificity", "False positive", "False negative",
                       "Sensitivity"), nrow = 2)
colnames(matrix_data) = c("X=0", "X=1")
rownames(matrix_data) = c("X*=0", "X*=1")
matrix_data
```

```
##      X=0              X=1
## X*=0 "Specificity"    "False negative"
## X*=1 "False positive" "Sensitivity"
```

```r
#Then check that my misclassification matrix is ordered correctly.
mc.s = matrix(c(1,0,0.08,0.92),nrow=2)
dimnames(mc.s) = list(levels(bronch$smoking), levels(bronch$smoking))
mc.s
```

```
##   0    1
## 0 1 0.08
## 1 0 0.92
```

```r
#The mcsimex adjustment with misclassification in smoking.
adj_smoking92 = mcsimex(naive, mc.matrix = mc.s, SIMEXvariable = "smoking")
#summary(mod.smoking)
plot(adj_smoking92, mfrow= c(2,2))
```



The effect of *smoking* appears to be severely attenuated, but the effect of *dust* remains practically unchanged. Hence, the association of interest is robust to false negatives in *smoking* behaviour. Good news.

```
#Expressing the impact of measurement error in terms of relative bias.
Bias = naive$coefficients - adj_smoking92$coefficients
RBias = Bias / adj_smoking92$coefficients
RBias
```

```
## (Intercept)        dust    smoking1        expo
##     -0.0623      0.0016     -0.2232     -0.0096
```

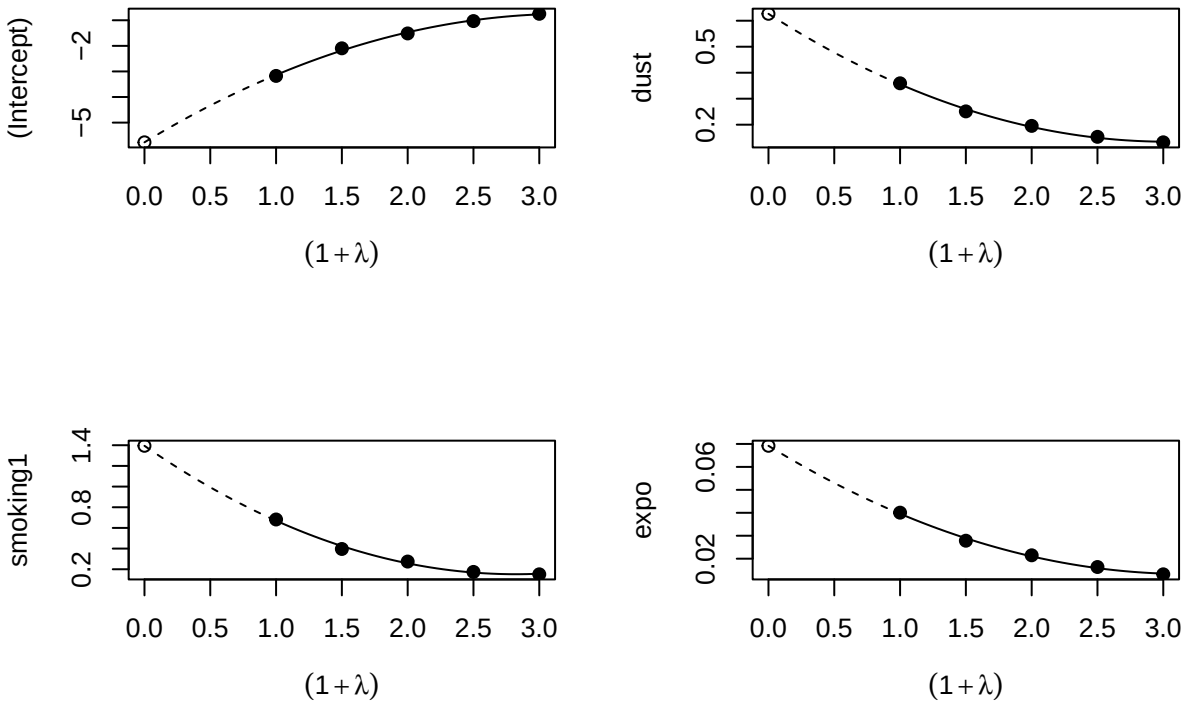
```
#The effect of smoking is attenuated by 22%.
```

If we also want to contemplate misclassification in the outcome first we need to specify its misclassification matrix.

```
mc.cbr = matrix(c(0.8,0.2,0.1,0.9), nrow=2)
dimnames(mc.cbr) = list(levels(bronch$cbr), levels(bronch$cbr))
mc.cbr
```

```
##     0   1
## 0 0.8 0.1
## 1 0.2 0.9
```

```
adj_smoking90cbr9080 = mcsimex(naive, mc.matrix = list(smoking = mc.s, cbr = mc.cbr),
          SIMEXvariable = c("cbr", "smoking"))
#summary(mod.smoking)
plot(adj_smoking90cbr9080, mfrow= c(2,2))
```




```
#Expressing the impact of measurement error in terms of relative bias.
Bias = naive$coefficients - adj_smoking90cbr9080$coefficients
RBias = Bias / adj_smoking90cbr9080$coefficients
RBias
```

```
## (Intercept)        dust    smoking1        expo
##       -0.45       -0.43       -0.51       -0.42
```

This provides a far more worrying picture. If *cbr* is not correctly recorded, we could be considering a strong attenuation bias in its relationship with *dust*. Namely, more than a 40% attenuation when *cbr* is recorded with 90% sensitivity and 80% specifcity.

########################## UP TO HERE ##########################

Concluding remarks

We have seen how SIMEX is another highly flexible adjustment method that could be employed to adjust for a wide range of measurement error and misclassification problems across different outcome models. Crucially, SIMEX allows us to consider random errors, which is something that we could not simply simulate and use it to adjust the error-prone variable directly. However, SIMEX is not without its limitations.

In terms of applicability, and as far as I am aware, the `simex` package cannot be used to adjust for models including variables affected by both measurement error and misclassification. This is a clear limitation, as we have just seen for the case of the bronchitis study. In Exercise 1A we also encountered another limitation in the form of outcome models that could be adjusted. There we used a linear model but given the right-skewed distribution a model for count or duration data might have been more appropriate.

There are R packages that have been created to address these gaps. See for example `augSIMEX` (Zhang &

Yi, 2022) to adjust for measurement error and misclassification simultaneously, or `simexaft` (Xiong & He, 2022) to adjust for measurement error in the outcome variable of survival models. However, if what you want is complete flexibility in the modelling of your outcome variable and/or the measurement error and misclassification mechanisms present in your data, the best option is to use Bayesian statistics. Let's move on to the next practical, where we are going to demonstrate the potential of Bayesian adjustments.

References

Cook, J. R., and Stefanski, L. A. (1994). 'Simulation-extrapolation estimation in parametric measurement error models'. *Journal of the American Statistical Association*, 89(428), 1314-1328.

Dhami, M. K., and Belton, I. K. (2023). 'The role of character-based personal mitigation in sentencing judgments'. *Journal of Empirical Legal Studies*.

Gossl, C., and Kuchenhoff, H. (2001). 'Bayesian analysis of logistic regression with an unknown change point and covariate measurement error. *Statistics in Medicine*, 20, 3109-3121.

Kuchenhoff, H., and Carroll, R.J. (1997). 'Segmented regression with errors in predictors: semiparametric and parametric methods'. *Statistics in Medicine*, 16, 169-188.

Küchenhoff, H., Mwalili, S. M., and Lesaffre, E. (2006). 'A general method for dealing with misclassification in regression: the misclassification SIMEX'. *Biometrics*, 62(1), 85-96.

Lederer, W., and Kuchenhoff, H. (2006). 'A short introduction to the SIMEX and MCSIMEX'. *The Newsletter of the R Project*, 6(4), 26–31.

Maslen, H. (2015). 'Penitence and persistence: How should sentencing factors interact?'. In: *Exploring sentencing practice in England and Wales* (pp. 173-193). London: Palgrave Macmillan UK.

Ministry of Justice (2023). 'Criminal Court Statistics Quarterly: April to June 2023'.

Pina-Sánchez, J., Roberts, J., and Sferopoulos, D. (2019). 'Does the Crown Court discriminate against Muslim-named offenders? a novel investigation based on text mining techniques'. *The British Journal of Criminology*, 59(3), 718–736

Seibold, H., and Lederer, W. (2022). 'SIMEX- and MCSIMEX-algorithm for measurement error models. The simex package: Version 1.8'. *CRAN*.

Xiong, J., and He, W. (2022). 'Implement of the Simulation-Extrapolation (SIMEX) algorithm for the accelerated failure time (AFT) with covariates subject to measurement error. The simexaft package: Version 1.0.7.1'. *CRAN*.

Zhang, Q., and Yi, G. (2022). 'Analysis of data with mixed measurement error and misclassification in covariates. The augSIMEX package: Version 3.7.4'. *CRAN*.