Read in and store data files; define NA values

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
XYZ_A <- read_csv("XYZ_calA.csv", na = "1000")
```

```
## New names:
## Rows: 2093 Columns: 25
## -- Column specification
## -------------------------------------------------------- Delimiter: "," dbl
## (23): Month, Day, Year, Hour, Minute, Second, Milliseconds, Sample Time ... lgl
## (2): Lattitude, Longitude
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `gyro_Y` -> `gyro_Y...16`
## * `gyro_Y` -> `gyro_Y...17`
```

```
head(XYZ_A)
```

```
## # A tibble: 6 x 25
##   Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)`  Acc_X
##   <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl>  <dbl>
## 1    10     4  2023    17     20     47          943                608 -0.266
## 2    10     4  2023    17     20     47          963                 20 -0.266
## 3    10     4  2023    17     20     47          983                 20 -0.266
## 4    10     4  2023    17     20     47            3                 20 -0.266
## 5    10     4  2023    17     20     47           23                 20 -0.266
## 6    10     4  2023    17     20     47           43                 20 -0.266
## # i 16 more variables: Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>
```

```
XYZ_B <- read_csv("XYZ_calB.csv", na = "1000")
```

```
## New names:
## Rows: 2084 Columns: 25
## -- Column specification
## -------------------------------------------------------- Delimiter: "," dbl
## (23): Month, Day, Year, Hour, Minute, Second, Milliseconds, Sample Time ... lgl
## (2): Lattitude, Longitude
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `gyro_Y` -> `gyro_Y...16`
## * `gyro_Y` -> `gyro_Y...17`
```

```r
head(XYZ_B)
```

```
## # A tibble: 6 x 25
##   Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)` Acc_X
##   <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl> <dbl>
## 1    10     4  2023    17     21     33          138                655 -31.1
## 2    10     4  2023    17     21     33          158                 20 -31.1
## 3    10     4  2023    17     21     33          238                 20 -31.1
## 4    10     4  2023    17     21     33          258                 20 -31.1
## 5    10     4  2023    17     21     33          278                 20 -31.1
## 6    10     4  2023    17     21     33          298                 20 -32.6
## # i 16 more variables: Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>
```

```r
XYZ_C <- read_csv("XYZ_calC.csv", na = "1000")
```

```
## New names:
## Rows: 2077 Columns: 25
## -- Column specification
## -------------------------------------------------------- Delimiter: "," dbl
## (23): Month, Day, Year, Hour, Minute, Second, Milliseconds, Sample Time ... lgl
## (2): Lattitude, Longitude
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `gyro_Y` -> `gyro_Y...16`
## * `gyro_Y` -> `gyro_Y...17`
```

```r
head(XYZ_C)
```

```
## # A tibble: 6 x 25
##   Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)`  Acc_X
##   <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl>  <dbl>
## 1    10     4  2023    17     22     20          972                663 -0.727
## 2    10     4  2023    17     22     20          992                 20 -0.727
## 3    10     4  2023    17     22     20           12                 20 -0.727
## 4    10     4  2023    17     22     20           32                 20 -0.766
## 5    10     4  2023    17     22     20           52                 20 -0.766
## 6    10     4  2023    17     22     20           72                 20 -0.766
## # i 16 more variables: Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>
```

```r
XYZ_D <- read_csv("XYZ_calD.csv", na = "1000")
```

```
## New names:
## Rows: 2078 Columns: 25
## -- Column specification
## -------------------------------------------------------- Delimiter: "," dbl
## (23): Month, Day, Year, Hour, Minute, Second, Milliseconds, Sample Time ... lgl
## (2): Lattitude, Longitude
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `gyro_Y` -> `gyro_Y...16`
## * `gyro_Y` -> `gyro_Y...17`
```

```r
head(XYZ_D)
```

```
## # A tibble: 6 x 25
##   Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)` Acc_X
##   <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl> <dbl>
## 1    10     4  2023    17     23      5          184                653 0.285
## 2    10     4  2023    17     23      5          204                 20 0.285
## 3    10     4  2023    17     23      5          224                 20 0.285
## 4    10     4  2023    17     23      5          244                 20 0.285
## 5    10     4  2023    17     23      5          264                 20 0.285
## 6    10     4  2023    17     23      5          284                 20 0.285
## # i 16 more variables: Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>
```

```r
XYZ_E <- read_csv("XYZ_calE.csv", na = "1000")
```

```
## New names:
## Rows: 2085 Columns: 25
## -- Column specification
## -------------------------------------------------------- Delimiter: "," dbl
## (23): Month, Day, Year, Hour, Minute, Second, Milliseconds, Sample Time ... lgl
## (2): Lattitude, Longitude
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `gyro_Y` -> `gyro_Y...16`
## * `gyro_Y` -> `gyro_Y...17`
```

```r
head(XYZ_E)
```

```
## # A tibble: 6 x 25
##   Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)`  Acc_X
##   <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl>  <dbl>
## 1    10     4  2023    17     23     50          582                650 -0.672
## 2    10     4  2023    17     23     51          602                 20 -0.672
## 3    10     4  2023    17     23     51          622                 20 -0.672
## 4    10     4  2023    17     23     51          642                 20 -0.672
## 5    10     4  2023    17     23     51          662                 20 -0.672
## 6    10     4  2023    17     23     51          682                 20 -0.672
## # i 16 more variables: Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>
```

```r
XYZ_F <- read_csv("XYZ_calF.csv", na = "1000")
```

```
## New names:
## Rows: 2083 Columns: 25
## -- Column specification
## -------------------------------------------------------- Delimiter: "," dbl
## (23): Month, Day, Year, Hour, Minute, Second, Milliseconds, Sample Time ... lgl
## (2): Lattitude, Longitude
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `gyro_Y` -> `gyro_Y...16`
## * `gyro_Y` -> `gyro_Y...17`
```

```
head(XYZ_F)
```

```
## # A tibble: 6 x 25
##   Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)`  Acc_X
##   <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl>  <dbl>
## 1    10     4  2023    17     24     36          143                647 -0.633
## 2    10     4  2023    17     24     36          163                 20 -0.633
## 3    10     4  2023    17     24     36          183                 20 -0.633
## 4    10     4  2023    17     24     36          203                 20 -0.633
## 5    10     4  2023    17     24     36          223                 20 -0.633
## 6    10     4  2023    17     24     36          243                 20 -0.633
## # i 16 more variables: Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>
```

Merge the data frames together with rbind() function

```
XYZ_CAL <- rbind(XYZ_A, XYZ_B, XYZ_C, XYZ_D, XYZ_E, XYZ_F)
print(XYZ_CAL)
```

```
## # A tibble: 12,500 x 25
##    Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)`  Acc_X
##    <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl>  <dbl>
##  1    10     4  2023    17     20     47          943                608 -0.266
##  2    10     4  2023    17     20     47          963                 20 -0.266
##  3    10     4  2023    17     20     47          983                 20 -0.266
##  4    10     4  2023    17     20     47            3                 20 -0.266
##  5    10     4  2023    17     20     47           23                 20 -0.266
##  6    10     4  2023    17     20     47           43                 20 -0.266
##  7    10     4  2023    17     20     47           63                 20 -0.266
##  8    10     4  2023    17     20     47           83                 20 -0.266
##  9    10     4  2023    17     20     47          103                 20 -0.266
## 10    10     4  2023    17     20     47          123                 20 -0.266
## # i 12,490 more rows
## # i 16 more variables: Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>
```

```
#check for error by making sure number of rows match
nrow(XYZ_CAL) #12500
```

```
## [1] 12500
```

```
nrow(XYZ_A) + nrow(XYZ_B) + nrow(XYZ_C) + nrow(XYZ_D) + nrow(XYZ_E) + nrow(XYZ_F) #12500
```

```
## [1] 12500
```

Add row ID column

```
XYZ_CAL <- tibble::rowid_to_column(XYZ_CAL, "ID")
head(XYZ_CAL)
```

```
## # A tibble: 6 x 26
##      ID Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)`
##   <int> <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl>
## 1     1    10     4  2023    17     20     47          943                608
## 2     2    10     4  2023    17     20     47          963                 20
## 3     3    10     4  2023    17     20     47          983                 20
## 4     4    10     4  2023    17     20     47            3                 20
## 5     5    10     4  2023    17     20     47           23                 20
## 6     6    10     4  2023    17     20     47           43                 20
## # i 17 more variables: Acc_X <dbl>, Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>
```

Create new dataframe with datetime column using make_datetime() function from Lubridate package

```
#library(lubridate)

XYZ_CAL1 <- XYZ_CAL %>%
  mutate(datetime = make_datetime(Year, Month, Day, Hour, Minute, Second))
```

Create a datetime value from string using ymd_hms() function

```
DT <- ymd_hms("1997-06-10 12:06:50")
```

Visually check new column for errors

```
head(XYZ_CAL1$datetime)
```

```
## [1] "2023-10-04 17:20:47 UTC" "2023-10-04 17:20:47 UTC"
## [3] "2023-10-04 17:20:47 UTC" "2023-10-04 17:20:47 UTC"
## [5] "2023-10-04 17:20:47 UTC" "2023-10-04 17:20:47 UTC"
```

```
head(XYZ_CAL1)
```

```
## # A tibble: 6 x 27
##      ID Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)`
##   <int> <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl>
## 1     1    10     4  2023    17     20     47          943                608
## 2     2    10     4  2023    17     20     47          963                 20
## 3     3    10     4  2023    17     20     47          983                 20
## 4     4    10     4  2023    17     20     47            3                 20
## 5     5    10     4  2023    17     20     47           23                 20
## 6     6    10     4  2023    17     20     47           43                 20
## # i 18 more variables: Acc_X <dbl>, Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>, datetime <dttm>
```

Create a row for just the times using hms package and parse_hms() function

```
library(hms)
```

```
##
## Attaching package: 'hms'

## The following object is masked from 'package:lubridate':
##
##     hms
```

```
#set format
times <- parse_hms(c("00:00:00.25", "00:00:01", "00:01:30", "01:00:00"))
```

```r
times
```

```
## 00:00:00.25
## 00:00:01.00
## 00:01:30.00
## 01:00:00.00
```

```r
#set value to numeric
times_num <- as.numeric(times)
times_num
```

```
## [1]    0.25    1.00   90.00 3600.00
```

```r
#format inputs
as_hms(times_num)
```

```
## 00:00:00.25
## 00:00:01.00
## 00:01:30.00
## 01:00:00.00
```

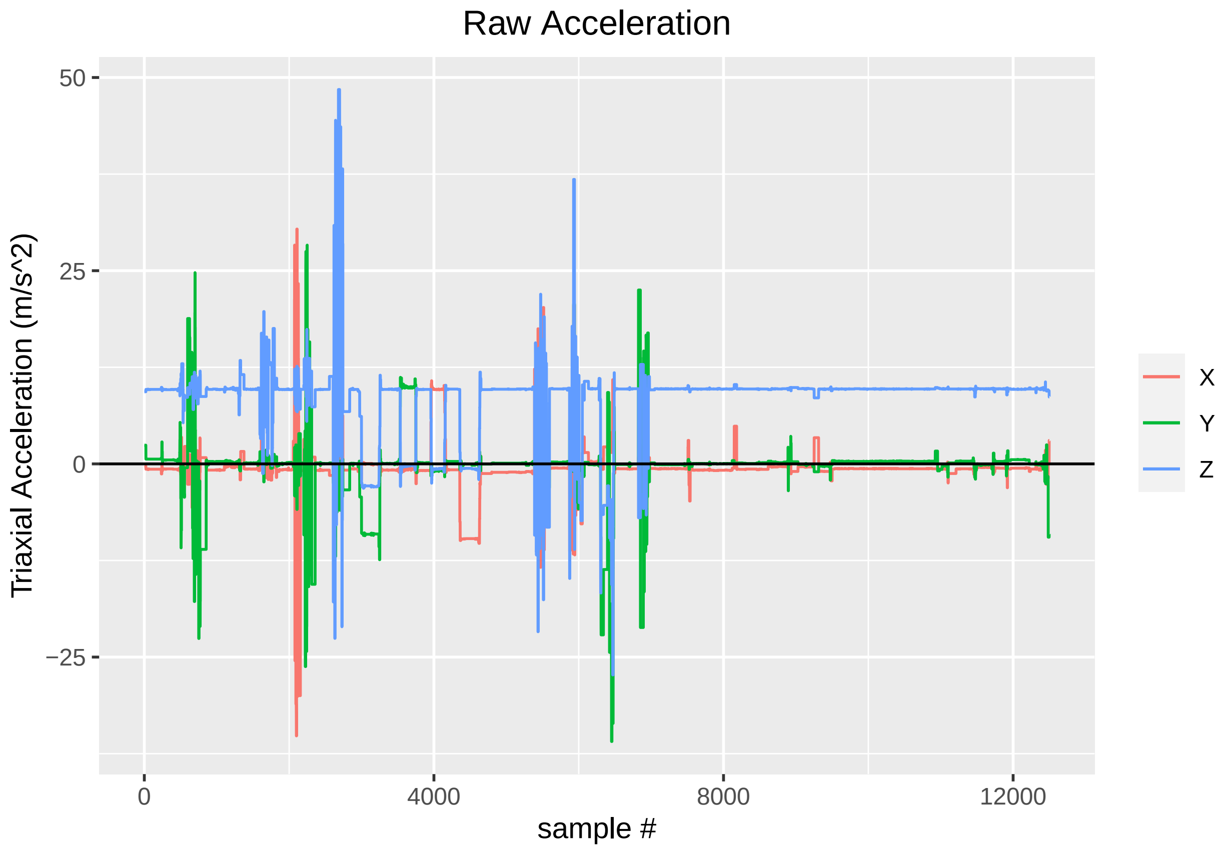
```r
#add column to dataframe
XYZ_CAL1$time <- as_hms(paste(XYZ_CAL1$Hour, XYZ_CAL1$Minute, XYZ_CAL1$Second, sep = ":"))
head(XYZ_CAL1)
```

```
## # A tibble: 6 x 28
##      ID Month   Day  Year  Hour Minute Second Milliseconds `Sample Time (ms)`
##   <int> <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>        <dbl>              <dbl>
## 1     1    10     4  2023    17     20     47          943                608
## 2     2    10     4  2023    17     20     47          963                 20
## 3     3    10     4  2023    17     20     47          983                 20
## 4     4    10     4  2023    17     20     47            3                 20
## 5     5    10     4  2023    17     20     47           23                 20
## 6     6    10     4  2023    17     20     47           43                 20
## # i 19 more variables: Acc_X <dbl>, Acc_Y <dbl>, Acc_Z <dbl>, linAcc_X <dbl>,
## #   linAcc_Y <dbl>, linAcc_Z <dbl>, gyro_X <dbl>, gyro_Y...16 <dbl>,
## #   gyro_Y...17 <dbl>, mag_X <dbl>, mag_Y <dbl>, mag_Z <dbl>, quatI <dbl>,
## #   quatJ <dbl>, quatK <dbl>, Lattitude <lgl>, Longitude <lgl>,
## #   datetime <dttm>, time <time>
```

RAW DATA PLOTS ___________________________________

Plot raw acceleration data from all 3 Acc XYZ columns

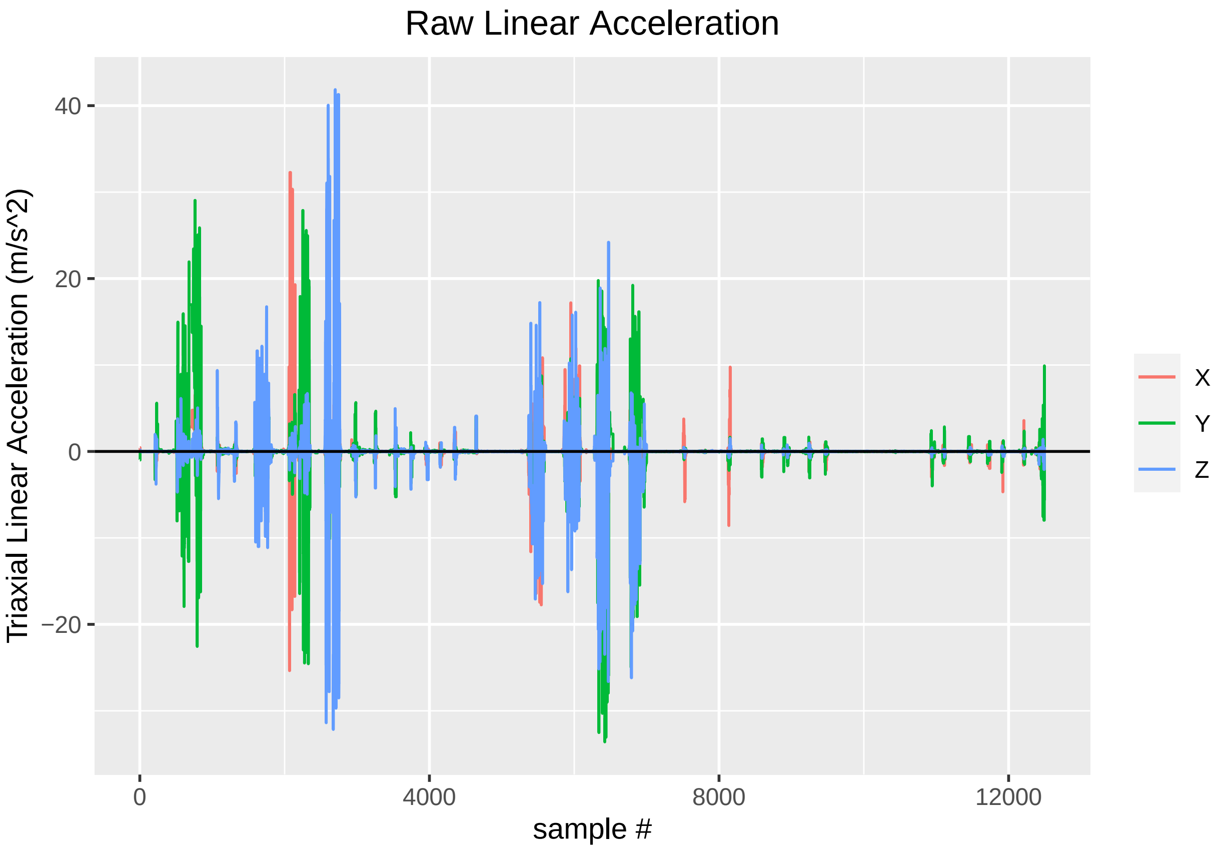
```r
rawXYZaccplot <- ggplot() +
  geom_line(data = XYZ_CAL1, mapping = aes(x = ID, y = Acc_X, color = "X")) +
  geom_line(data = XYZ_CAL1, mapping = aes(x = ID, y = Acc_Y, color = "Y")) +
  geom_line(data = XYZ_CAL1, mapping = aes(x = ID, y = Acc_Z, color = "Z")) +
  geom_hline(yintercept = 0) +
  labs(title = "Raw Acceleration", x = "sample #", y = "Triaxial Acceleration (m/s^2)") +
  theme(plot.title = element_text(hjust = 0.5))
rawXYZaccplot + theme(legend.title = element_blank())
```



```
#ggsave("/cloud/project/projects/mcgreal.pkg_project/XYZ_cal/rawXYZaccplot.png")
```

Plot raw linear acceleration data from all 3 linAcc XYZ columns

```
rawXYZlinaccplot <- ggplot() +
  geom_line(data = XYZ_CAL1, mapping = aes(x = ID, y = linAcc_X, color = "X")) +
  geom_line(data = XYZ_CAL1, mapping = aes(x = ID, y = linAcc_Y, color = "Y")) +
  geom_line(data = XYZ_CAL1, mapping = aes(x = ID, y = linAcc_Z, color = "Z")) +
  geom_hline(yintercept = 0) +
  labs(title = "Raw Linear Acceleration", x = "sample #", y = "Triaxial Linear Acceleration (m/s^2)") +
  theme(plot.title = element_text(hjust = 0.5))
rawXYZlinaccplot + theme(legend.title = element_blank())
```



```
#ggsave("/cloud/project/projects/mcgreal.pkg_project/XYZ_cal/rawXYZaccplot.png")
```

SMOOTHED DATA PLOTS ________________________________

Take the average of X/Y/Z measurements over 1 second; create dataframe for each

```
#Acceleration columns
Xacc <- XYZ_CAL1 %>%
  select(time, Acc_X) %>%
  group_by(time) %>%
  summarize(Xacc_mean = mean(Acc_X)) %>%
  group_by(time)
head(Xacc)
```

```
## # A tibble: 6 x 2
## # Groups:   time [6]
##   time     Xacc_mean
##   <time>       <dbl>
## 1 17:20:47    -0.427
## 2 17:20:48    -0.691
## 3 17:20:49    -0.688
## 4 17:20:50    -0.688
## 5 17:20:51    -0.688
## 6 17:20:52    -0.724
```

```
Yacc <- XYZ_CAL1 %>%
  select(time, Acc_Y) %>%
  group_by(time) %>%
  summarize(Yacc_mean = mean(Acc_Y)) %>%
  group_by(time)
head(Yacc)
```

```
## # A tibble: 6 x 2
## # Groups:   time [6]
##   time     Yacc_mean
##   <time>       <dbl>
## 1 17:20:47     1.73
## 2 17:20:48     0.621
## 3 17:20:49     0.621
## 4 17:20:50     0.621
## 5 17:20:51     0.621
## 6 17:20:52     0.769
```

```
Zacc <- XYZ_CAL1 %>%
  select(time, Acc_Z) %>%
  group_by(time) %>%
  summarize(Zacc_mean = mean(Acc_Z)) %>%
  group_by(time)
head(Zacc)
```

```
## # A tibble: 6 x 2
## # Groups:   time [6]
##   time     Zacc_mean
##   <time>       <dbl>
## 1 17:20:47      9.45
## 2 17:20:48      9.63
## 3 17:20:49      9.64
## 4 17:20:50      9.64
## 5 17:20:51      9.64
## 6 17:20:52      9.66
```

```
#Linear Acceleration columns
Xlinacc <- XYZ_CAL1 %>%
  select(time, linAcc_X) %>%
  group_by(time) %>%
  summarize(Xlinacc_mean = mean(linAcc_X)) %>%
  group_by(time)
head(Xlinacc)
```

```
## # A tibble: 6 x 2
## # Groups:   time [6]
##   time     Xlinacc_mean
##   <time>          <dbl>
## 1 17:20:47     0.0428
## 2 17:20:48     0.00736
## 3 17:20:49    -0.000300
## 4 17:20:50     0.00158
## 5 17:20:51     0.0858
## 6 17:20:52     0.0788
```

```
Ylinacc <- XYZ_CAL1 %>%
  select(time, linAcc_Y) %>%
  group_by(time) %>%
  summarize(Ylinacc_mean = mean(linAcc_Y)) %>%
  group_by(time)
head(Ylinacc)
```

```
## # A tibble: 6 x 2
## # Groups:   time [6]
##   time     Ylinacc_mean
##   <time>          <dbl>
## 1 17:20:47    -0.0919
## 2 17:20:48    -0.004
## 3 17:20:49    -0.004
## 4 17:20:50    -0.000100
## 5 17:20:51    -0.546
## 6 17:20:52     0.754
```

```
Zlinacc <- XYZ_CAL1 %>%
  select(time, linAcc_Z) %>%
  group_by(time) %>%
  summarize(Zlinacc_mean = mean(linAcc_Z)) %>%
  group_by(time)
head(Zlinacc)
```

```
## # A tibble: 6 x 2
## # Groups:   time [6]
##   time     Zlinacc_mean
##   <time>          <dbl>
## 1 17:20:47      0.0150
## 2 17:20:48     -0.00386
## 3 17:20:49     -0.0107
## 4 17:20:50     -0.00682
## 5 17:20:51     -0.0760
## 6 17:20:52      0.0200
```

Add row ID column to each new dataframe

```
Xacc <- tibble::rowid_to_column(Xacc, "ID")
head(Xacc)
```

```
## # A tibble: 6 x 3
## # Groups:   time [6]
##      ID time     Xacc_mean
##   <int> <time>       <dbl>
## 1     1 17:20:47    -0.427
## 2     2 17:20:48    -0.691
## 3     3 17:20:49    -0.688
## 4     4 17:20:50    -0.688
## 5     5 17:20:51    -0.688
## 6     6 17:20:52    -0.724
```

```
Yacc <- tibble::rowid_to_column(Yacc, "ID")
head(Yacc)
```

```
## # A tibble: 6 x 3
## # Groups:   time [6]
##      ID time     Yacc_mean
##   <int> <time>       <dbl>
## 1     1 17:20:47      1.73
## 2     2 17:20:48     0.621
## 3     3 17:20:49     0.621
## 4     4 17:20:50     0.621
## 5     5 17:20:51     0.621
## 6     6 17:20:52     0.769
```

```
Zacc <- tibble::rowid_to_column(Zacc, "ID")
head(Zacc)
```

```
## # A tibble: 6 x 3
## # Groups:   time [6]
##      ID time     Zacc_mean
##   <int> <time>       <dbl>
## 1     1 17:20:47      9.45
## 2     2 17:20:48      9.63
## 3     3 17:20:49      9.64
## 4     4 17:20:50      9.64
## 5     5 17:20:51      9.64
## 6     6 17:20:52      9.66
```

```
Xlinacc <- tibble::rowid_to_column(Xlinacc, "ID")
head(Xlinacc)
```

```
## # A tibble: 6 x 3
## # Groups:   time [6]
##      ID time     Xlinacc_mean
##   <int> <time>          <dbl>
## 1     1 17:20:47     0.0428
## 2     2 17:20:48     0.00736
## 3     3 17:20:49    -0.000300
## 4     4 17:20:50     0.00158
## 5     5 17:20:51     0.0858
## 6     6 17:20:52     0.0788
```

```
Ylinacc <- tibble::rowid_to_column(Ylinacc, "ID")
head(Ylinacc)
```

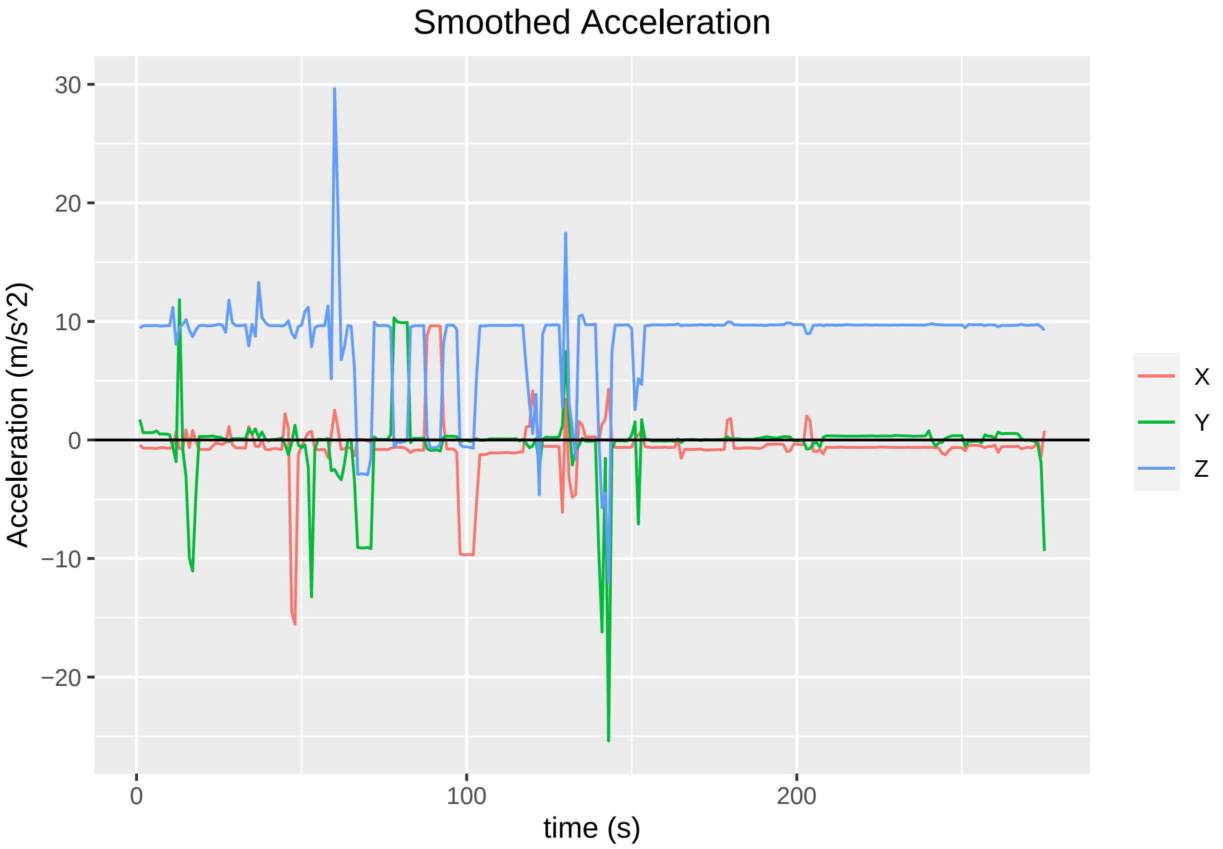
```
## # A tibble: 6 x 3
## # Groups:   time [6]
##      ID time     Ylinacc_mean
##   <int> <time>          <dbl>
## 1     1 17:20:47    -0.0919
## 2     2 17:20:48    -0.004
## 3     3 17:20:49    -0.004
## 4     4 17:20:50    -0.000100
## 5     5 17:20:51    -0.546
## 6     6 17:20:52     0.754
```

```
Zlinacc <- tibble::rowid_to_column(Zlinacc, "ID")
head(Zlinacc)
```

```
## # A tibble: 6 x 3
## # Groups:   time [6]
##      ID time     Zlinacc_mean
##   <int> <time>          <dbl>
## 1     1 17:20:47      0.0150
## 2     2 17:20:48     -0.00386
## 3      3 17:20:49      -0.0107
## 4      4 17:20:50      -0.00682
## 5      5 17:20:51      -0.0760
## 6      6 17:20:52       0.0200
```

Plot all acc axes together

```
XYZaccplot <- ggplot() +
  geom_line(data = Xacc, mapping = aes(x = ID, y = Xacc_mean, color = "X")) +
  geom_line(data = Yacc, mapping = aes(x = ID, y = Yacc_mean, color = "Y")) +
  geom_line(data = Zacc, mapping = aes(x = ID, y = Zacc_mean, color = "Z")) +
  geom_hline(yintercept = 0) +
  labs(title = "Smoothed Acceleration", x = "time (s)", y = "Acceleration (m/s^2)") +
  theme(plot.title = element_text(hjust = 0.5))
XYZaccplot + theme(legend.title = element_blank())
```



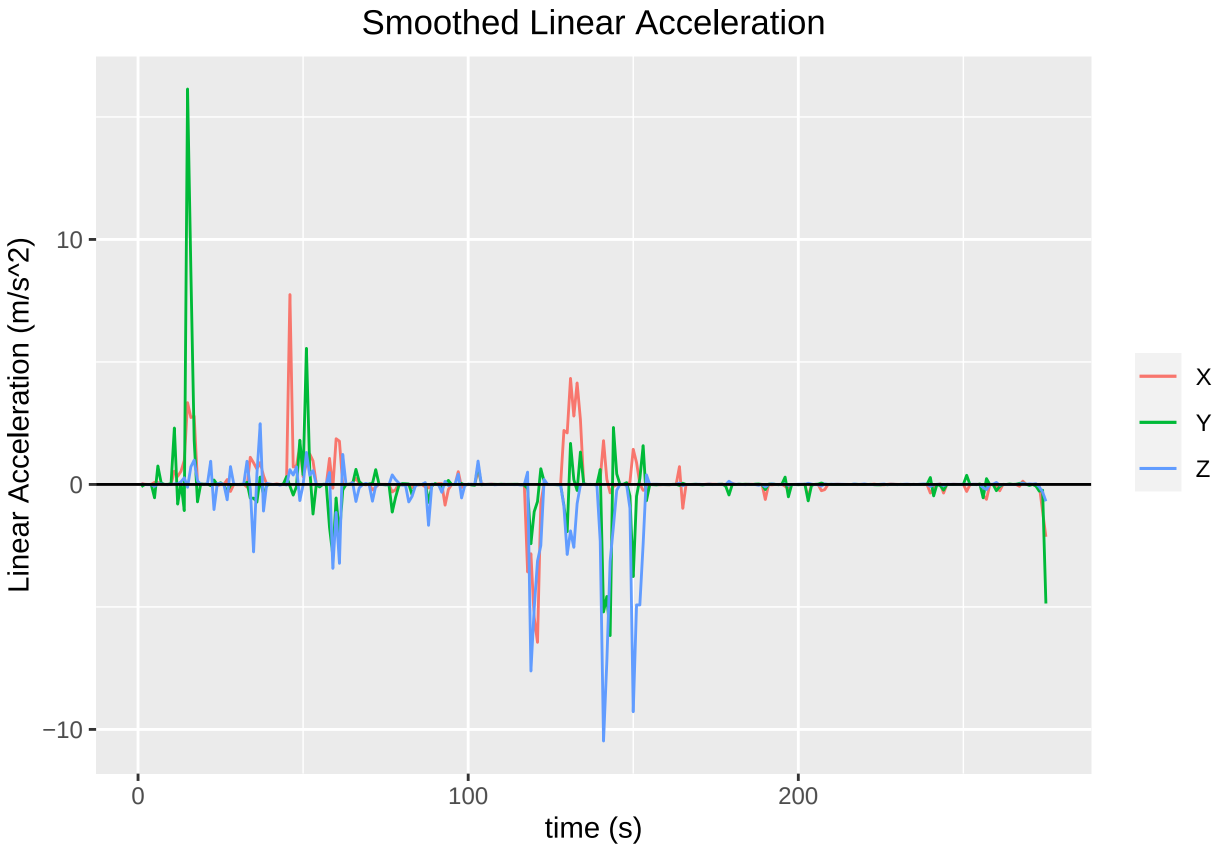
```
ggsave("/cloud/project/projects/mcgreal.pkg_project/XYZ_cal/XYZaccplot.png")
```

```
## Saving 6.5 x 4.5 in image
```

Plot all linear acc together

```
XYZlinaccplot <- ggplot() +
  geom_line(data = Xlinacc, mapping = aes(x = ID, y = Xlinacc_mean, color = "X")) +
  geom_line(data = Ylinacc, mapping = aes(x = ID, y = Ylinacc_mean, color = "Y")) +
  geom_line(data = Zlinacc, mapping = aes(x = ID, y = Zlinacc_mean, color = "Z")) +
  geom_hline(yintercept = 0) +
  labs(title = "Smoothed Linear Acceleration", x = "time (s)", y = "Linear Acceleration (m/s^2)") +
  theme(plot.title = element_text(hjust = 0.5))
```

```
XYZlinaccplot + theme(legend.title = element_blank())
```

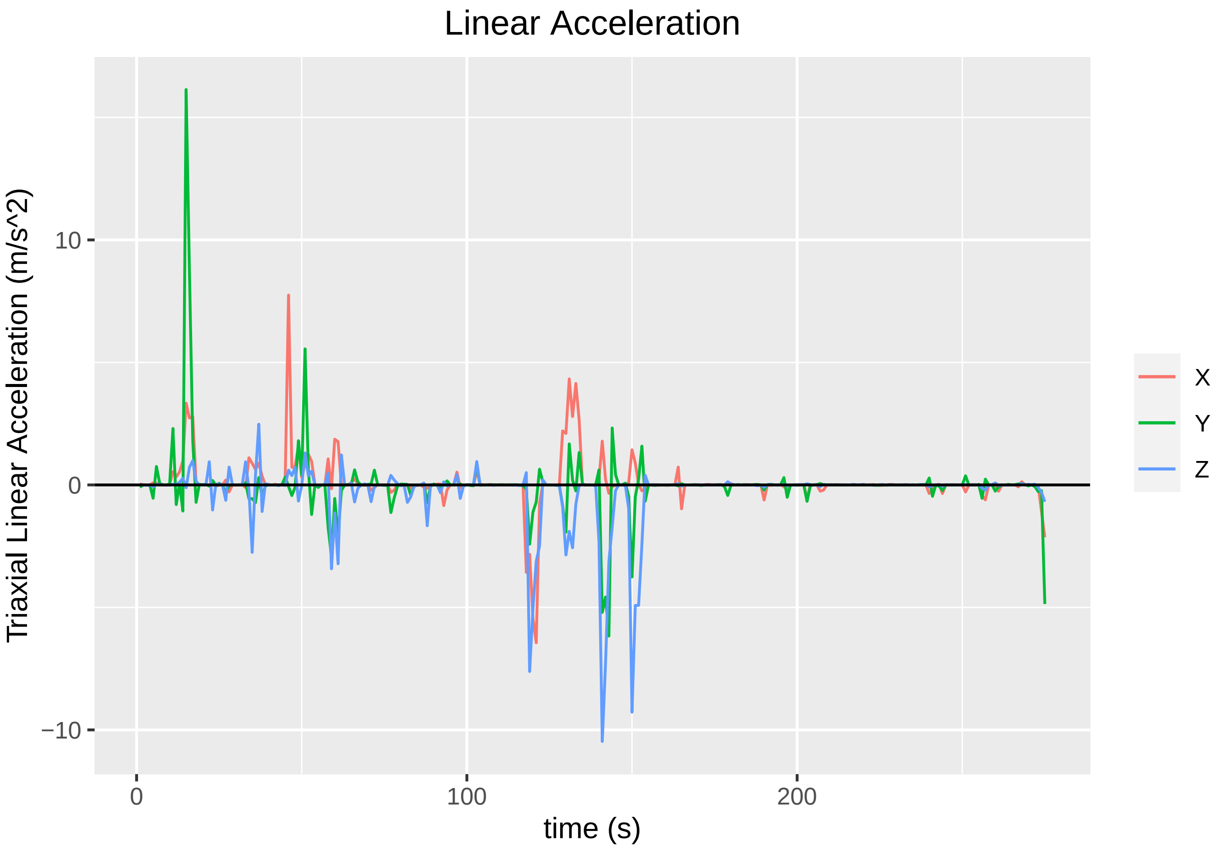


```
ggsave("/cloud/project/projects/mcgreal.pkg_project/XYZ_cal/XYZlinaccplot.png")
```

```
## Saving 6.5 x 4.5 in image
```

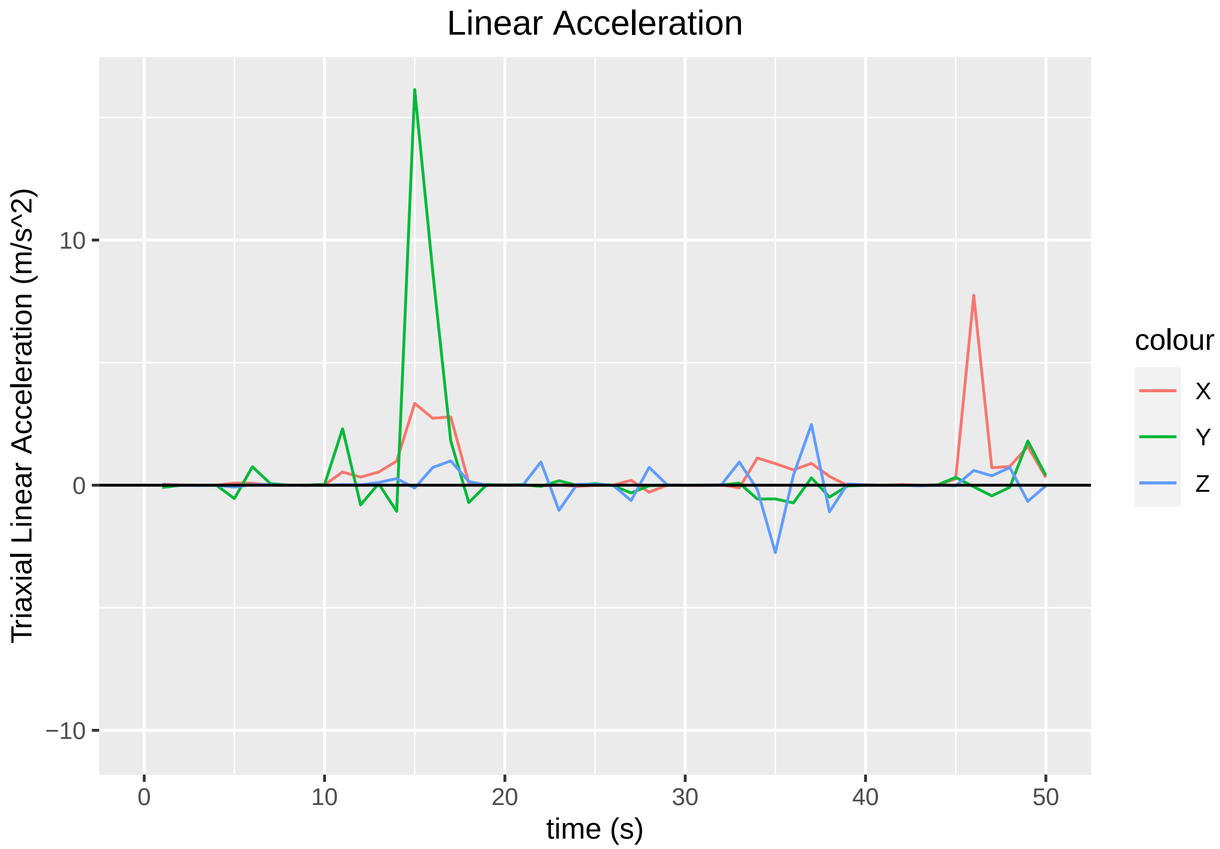
Inspect linear acc######

```
XYZlinaccplot <- ggplot() +
  geom_line(data = Xlinacc, mapping = aes(x = ID, y = Xlinacc_mean, color = "X")) +
  geom_line(data = Ylinacc, mapping = aes(x = ID, y = Ylinacc_mean, color = "Y")) +
  geom_line(data = Zlinacc, mapping = aes(x = ID, y = Zlinacc_mean, color = "Z")) +
  geom_hline(yintercept = 0) +
  labs(title = "Linear Acceleration", x = "time (s)", y = "Triaxial Linear Acceleration (m/s^2)") +
  theme(plot.title = element_text(hjust = 0.5))
XYZlinaccplot + theme(legend.title = element_blank())
```



```
XYZlinaccplot + xlim(0, 50)
```

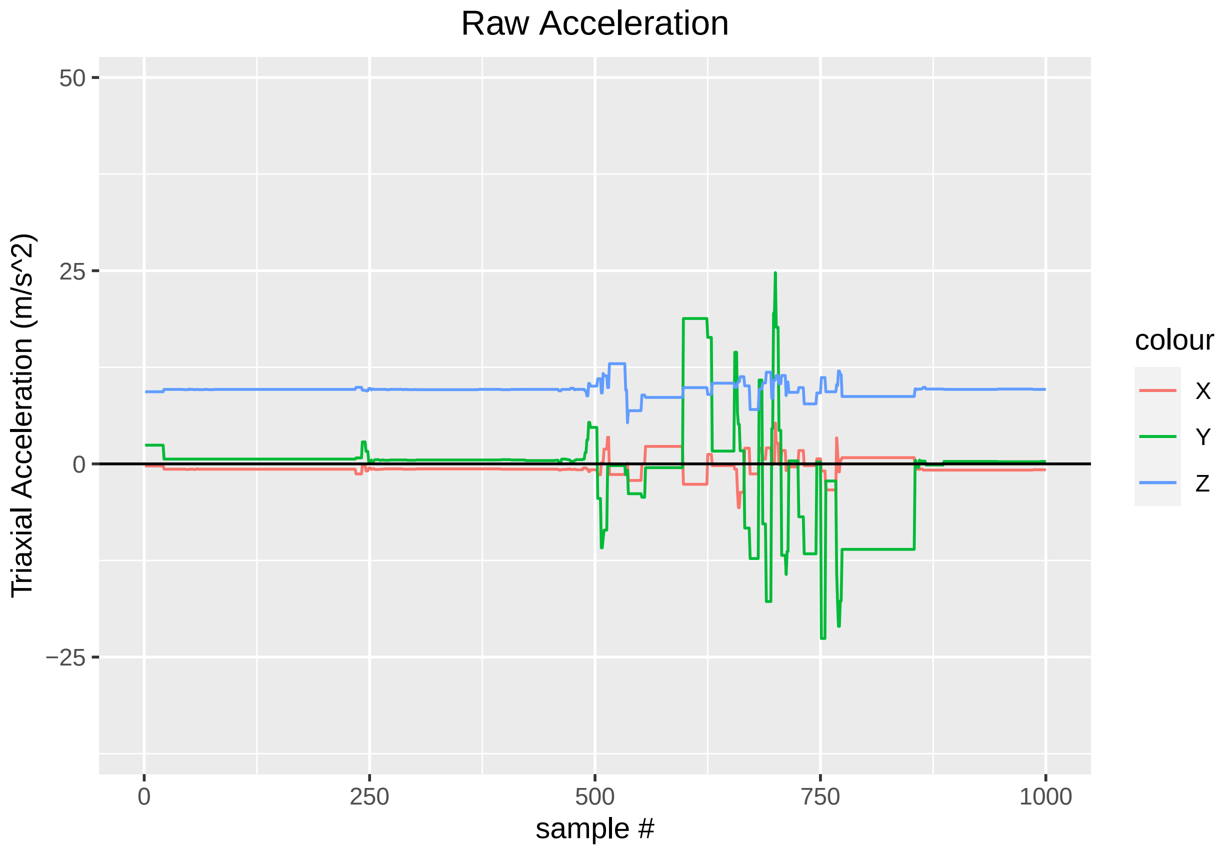
```
## Warning: Removed 225 rows containing missing values (`geom_line()`).
## Removed 225 rows containing missing values (`geom_line()`).
## Removed 225 rows containing missing values (`geom_line()`).
```



inspect raw acceleration ___________ raw acceleration

```
rawXYZaccplot + xlim(0, 1000)
```

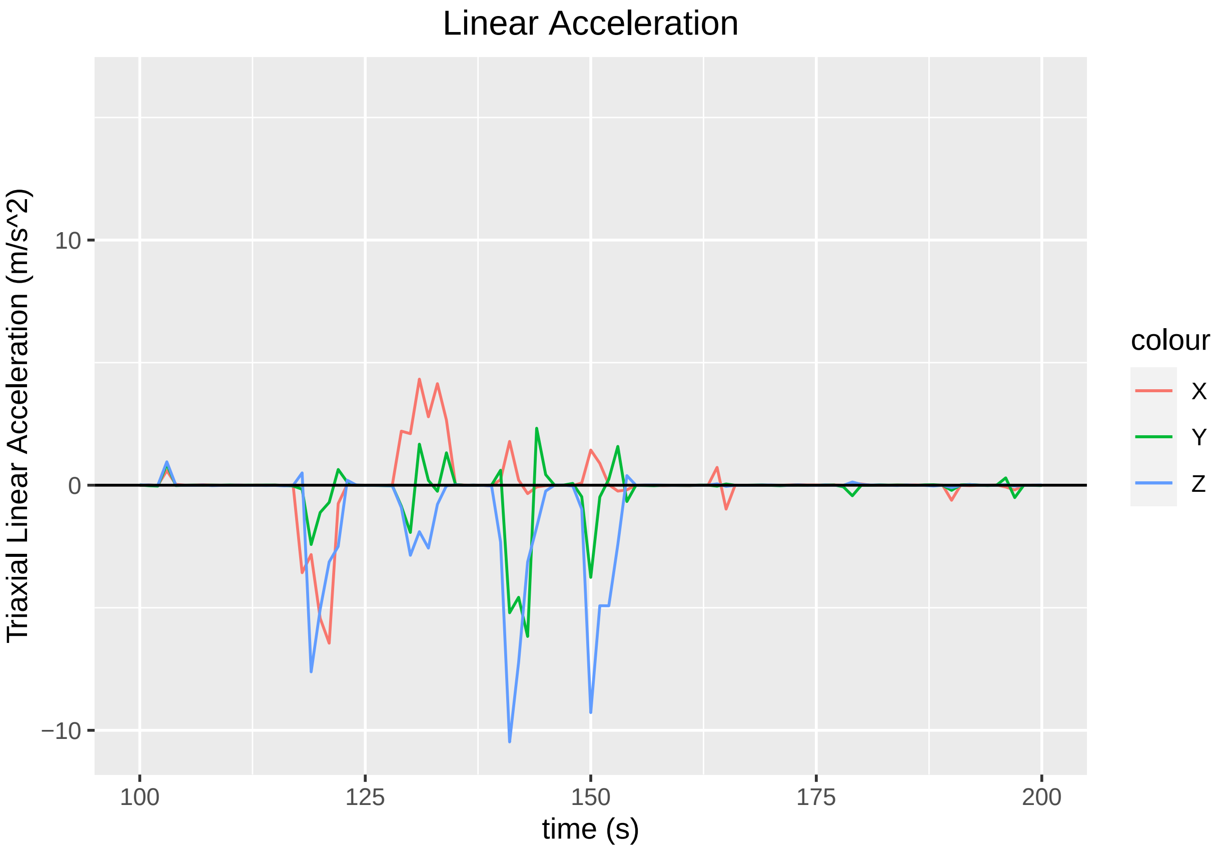
```
## Warning: Removed 11500 rows containing missing values (`geom_line()`).
## Removed 11500 rows containing missing values (`geom_line()`).
## Removed 11500 rows containing missing values (`geom_line()`).
```



```
XYZlinaccplot + xlim(100, 200)
```

```
## Warning: Removed 174 rows containing missing values (`geom_line()`).
```

```
## Warning: Removed 174 rows containing missing values (`geom_line()`).
## Removed 174 rows containing missing values (`geom_line()`).
```


Linear Acceleration
Triaxial Linear Acceleration (m/s^2)
10
0
−10
100
125
150
175
200
time (s)
colour
X
Y
Z