Introduction to

FUNCTIONS

Part 2

```
ggplot(df, aes(x = time,
               y = conc,
               group = ID))
```

```
ggplot(df, aes(x = “time”,
          y = “conc”,
             group = “ID”))
```

```
grp <- “ID”

ggplot(df, aes(x = time,
               y = conc,
               group = grp))
```

```
Theoph %>%
    ggplot(aes(x = Time,
               y = conc,
               group = Subject)) +
    geom_line() + geom_point()
```

```
gg_conc_time <- function() {
  Theoph %>%
      ggplot(aes(x = Time,
                 y = conc,
                 group = Subject)) +
      geom_line() + geom_point()
  }
```

```
gg_conc_time <- function() {
  Theoph %>%
      ggplot(aes(x = Time,
                 y = conc,
                 group = Subject)) +
      geom_line() + geom_point()
   }
```

```
gg_conc_time <- function(df) {
  Theoph %>%
      ggplot(aes(x = Time,
                 y = conc,
                 group = Subject)) +
      geom_line() + geom_point()
  }
```

```
gg_conc_time <- function(df) {
        df %>%
        ggplot(aes(x = Time,
                   y = conc,
                   group = Subject)) +
        geom_line() + geom_point()
    }
```

```
gg_conc_time <- function(df) {
        df %>%
        ggplot(aes(x = Time,
                   y = conc,
                   group = Subject)) +
        geom_line() + geom_point()
    }
```

```
gg_conc_time <- function(df, x, y, g){
        df %>%
        ggplot(aes(x = x,
                   y = y,
                   group = g)) +
        geom_line() + geom_point()
    }
```

Will either of these work?

```
gg_conc_time(Theoph,
        “Time”,
        “conc”,
        “Subject”)
```

```
gg_conc_time(Theoph,
             Time,
             conc,
             Subject)
```

Introduction to

Non-Standard Evaluation (NSE)

Key rlang functions for general NSE/SE use

function	purpose
!!/!!!	Unquote and immediately evaluate the argument in the surrounding context
enquo()	Turn an argument name into an expression, without evaluating it (like substitute)
sym()/syms()	Turn a string into a symbol (like as.name)

Great for interactive functions

```
gg_conc_time(Theoph,
             Time,
             conc,
             Subject)
```

```
gg_conc_time <- function(df, x, y, g){
   x <- enquo(x)
   y <- enquo(y)
   g <- enquo(g)
   res <- df %>%
    ggplot(aes(x = !!x, y = !!y,
               group = !!g)) +
       geom_line() + geom_point()
    return(res)
    }
```

But what about programmatic

```
xvar <- “Time”
yvar <- “conc”
group <- “Subject”

gg_conc_time(Theoph,
      xvar, yvar, group)
```

```
gg_conc_time <- function(df, x, y, g){
   x <- enquo(x)
   y <- enquo(y)
   g <- enquo(g)
   res <- df %>%
    ggplot(aes(x = !!x, y = !!y,
               group = !!g)) +
       geom_line() + geom_point()
    return(res)
    }
```

```
gg_conc_time <- function(df, x, y, g){
   x <- enquo(x)
   y <- enquo(y)
   g <- enquo(g)
   res <- df %>%
    ggplot(aes(x = !!x, y = !!y,
               group = !!g)) +
        geom_line() + geom_point()
    return(res)
    }
```

```
gg_conc_time2 <- function(df, x, y, g){
    res <- df %>%
    ggplot(aes(x = !!sym(x), y = !!sym(y),
               group = !!sym(g))) +
     geom_line() +
    geom_point()
    return(res)
    }
```

Programmatic “problems”

```
gg_conc_time(Theoph,
  Time/24, conc, Subject)

gg_conc_time2(Theoph,
  xvar/24, yvar, group) ??
```

Programmatic “problems”

```
gg_conc_time(Theoph,
  Time/24, conc, Subject)

gg_conc_time(Theoph,
  sym(xvar)/24, sym(yvar),
group)
```

```r
min_max <- function(df, dv, grps) {
  dv <- enquo(dv)
  df %>%
    group_by(!!!grps) %>%
    summarize(min = min(!!dv),
              max = max(!!dv))
}

Theoph %>% min_max(conc, vars(Subject, Time))

grps <- c("Subject", "Time")
Theoph %>% min_max(conc, syms(grps))
```

```r
min_max2 <- function(df, dv, grps) {
  dv <- enquo(dv)
  dv_nm <- quo_name(dv)
  df %>%
    group_by(!!!grps) %>%
    summarize(!!sprintf("min_%s", dv_nm) := min(!!dv),
              !!sprintf("max_%s", dv_nm) := max(!!dv))
}

Theoph %>% min_max2(conc, vars(Subject))
```

Take advantage of pipelines

```
min_max3 <- function(df, dv, grps) {
  dv <- enquo(dv)
  dv_nm <- quo_name(dv)
  df %>%
    group_by(!!!grps) %>%
    summarize(!!sprintf("min_%s", dv_nm) := min(!!dv),
              !!sprintf("max_%s", dv_nm) := max(!!dv))
}

Theoph %>% group_by(Subject) %>% min_max2(conc)
```

Take advantage of pipelines

```
min_max <- function(df, dv) {
  dv <- enquo(dv)
  dv_nm <- quo_name(dv)
  df %>%
    mutate(!!sprintf("min_%s", dv_nm) := min(!!dv),
           !!sprintf("max_%s", dv_nm) := max(!!dv))
}

df %>%
   group_by(Subject) %>%
   min_max(conc) %>%
   min_max(Time)
```

```r
Theoph %>%
  group_by(Subject) %>%
  min_max(conc) %>%
  min_max(Time) %>%
  distinct(Subject, .keep_all = TRUE)
```

```
#> # A tibble: 12 x 9#> # Groups:   Subject [12]
#>    Subject    Wt  Dose  Time  conc min_conc max_conc min_Time max_Time
#>    <ord>   <dbl> <dbl> <dbl> <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
#>  1 1        79.6  4.02     0  0.74     0.74    10.5         0     24.4
#>  2 2        72.4  4.4      0  0        0        8.33        0     24.3
#>  3 3        70.5  4.53     0  0        0        8.2         0     24.2
#>  4 4        72.7  4.4      0  0        0        8.6         0     24.6
#>  5 5        54.6  5.86     0  0        0       11.4         0     24.4
#>  6 6        80    4        0  0        0        6.44        0     23.8
#>  7 7        64.6  4.95     0  0.15     0.15     7.09        0     24.2
#>  8 8        70.5  4.53     0  0        0        7.56        0     24.1
#>  9 9        86.4  3.1      0  0        0        9.03        0     24.4
#> 10 10       58.2  5.5      0  0.24     0.24    10.2         0     23.7
#> 11 11       65    4.92     0  0        0        8           0     24.1
#> 12 12       60.5  5.3      0  0        0        9.75        0     24.2
```