

*Introduction to*

# FUNCTIONS

*Part 2*

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

```
ID <- 20
```

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

```
ID <- 20
```

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


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()  
}
```



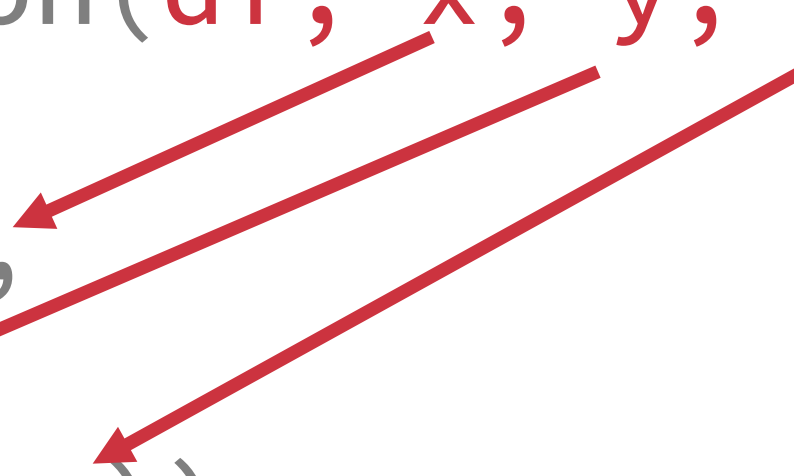


```
gg_conc_time <- function(df) {  
  df %>%  
  ggplot(aes(x = Time,  
             y = conc,  
             group = Subject)) +  
  geom_line() + geom_point()  
}
```

```
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()  
}
```



A diagram consisting of three red arrows pointing from the function arguments 'x', 'y', and 'g' in the first line of the code to the corresponding parameters 'x', 'y', and 'group' in the 'aes()' function call on the fourth line. This illustrates how the data frame columns are mapped to the plot aesthetics.

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

*Introduction to*

# Non-Standard Evaluation (NSE)

<b>function</b>	<b>purpose</b>
<b>lazyeval::interp()</b>	take in a code with template variables, and the mappings for how to replace the template variables
<b>lazyeval::lazy_eval()</b>	evaluate the generated code expression from interp()

```
my_nse_func <- function(param_1, param_2) {  
  ptemplate <-  
    lazyeval::interp(~ <code w/ template_vars>,  
      template_var1 = as.name(param_1),  
      template_var2 = as.name(param_2)  
    )  
  return(lazyeval::lazy_eval(p_template))  
}
```

```
lazyeval::interp(~ paste(tvar1, tvar2),  
                  tvar1 = "hello",  
                  tvar2 = "world"  
)
```



```
lazyeval::interp(~ paste(tvar1, tvar2),  
                  tvar1 = "hello",  
                  tvar2 = "world"  
)
```



```
~paste("hello", "world")
```

```
texpr <- lazyeval::interp(~ paste(tvar1, tvar2),  
  tvar1 = "hello",  
  tvar2 = "world"  
)
```

```
lazyeval::lazy_eval(texpr)
```



“hello world”

```
my_nse_func <- function(param_1, param_2) {  
  ptemplate <-  
    lazyeval::interp(~ <code w/ template_vars>,  
      template_var1 = as.name(param_1),  
      template_var2 = as.name(param_2)  
    )  
  return(lazyeval::lazy_eval(p_template))  
}
```

```
my_nse_func <- function(param_1, param_2) {  
  ptemplate <-  
    lazyeval::interp(~ <code w/ template_vars>,  
      template_var1 = as.name(param_1),  
      template_var2 = as.name(param_2)  
    )  
  return(lazyeval::lazy_eval(p_template))  
}
```

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

```
gg_conc_time <- function(df, x, y, g){  
  ptemplate <-  
  lazyeval::interp(~ df %>%  
    ggplot(aes(x = xtemplate,  
              y = ytemplate,  
              group = gtemplate)) +  
    geom_line() + geom_point(),  
  xtemplate = as.name(x),  
  ytemplate = as.name(y),  
  gtemplate = as.name(g)  
  return(lazyeval::lazy_eval(ptemplate))  
}
```

```
my_nse_func <- function(param_1, param_2) {  
  ptemplate <-  
    lazyeval::interp(~ <code w/ template_vars>,  
      template_var1 = as.name(param_1),  
      template_var2 = as.name(param_2)  
    )  
  return(lazyeval::lazy_eval(p_template))  
}
```

```
my_nse_func <- function(param_1, param_2) {  
  ptemplate <-  
    lazyeval::interp(~ <code w/ template_vars>,  
      template_var1 = as.name(param_1),  
      template_var2 = as.name(param_2)  
    )  
  return(lazyeval::lazy_eval(p_template))  
}
```



```
my_nse_func <- function(param_1, param_2) {  
  ptemplate <-  
    lazyeval::interp(~ <code w/ template_vars>,  
    template_var1 = as.name(param_1),  
    template_var2 = as.name(param_2)  
  )  
  return(lazyeval::lazy_eval(p_template))  
}
```

```
my_nse_func <- function(param_1, param_2) {  
  ptemplate <-  
    lazyeval::interp(~ <code w/ template_vars>,  
    template_var1 = as.name(param_1),  
    template_var2 = as.name(param_2)  
  )  
  return(lazyeval::lazy_eval(p_template))  
}
```

```
my_nse_func <- function(param_1, param_2) {  
  ptemplate <-  
    lazyeval::interp(~ <code w/ template_vars>,  
    template_var1 = as.name(param_1),  
    template_var2 = as.name(param_2)  
  )  
  return(lazyeval::lazy_eval(p_template))  
}
```