**An Introduction to Pipes in R**

R is a functional language, which means that your code often contains a lot of parenthesis, ( and ).

When you have complex code, this often means that you will have to nest those parentheses together. This makes your R code hard to read and understand.

In short, here are four reasons why you should be using pipes in R:

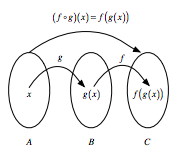
* You'll structure the sequence of your data operations from left to right, as opposed to from inside and out;
* You'll avoid nested function calls;
* You'll minimise the need for local variables and function definitions; And
* You'll make it easy to add steps anywhere in the sequence of operations.

In mathematics, a function is a relation between a set of inputs and a set of permissible outputs. Functions have the property that each input is related to exactly one output. For example, in the function any input for will give one output only.

"Function Composition" is applying one function to the results of another. If you have two functions, let's say and , you can chain these functions together by taking the output of one function and inserting it into the next.

The result of is sent through . It is written: . Which means:

Example: and , then .



In short, this means that functions that take one argument, function(argument), can be rewritten as follows: argument %>% function().

can be rewritten as %>%

Remember that "chaining" means that you invoke multiple method calls. As each method returns an object, you can actually allow the calls to be chained together in a single statement, without needing variables to store the intermediate results.

can be rewritten as %>% %>%