STAT 33B Lab Workbook WK 8

Won Shil Park (3033452021)

Mar 7, 2021

This workbook is due **Mar 11, 2021** by 9:00am or, if you attend lab, by Mar 12 11:59pm PT.

Workbooks are graded for completeness, so as long as you make a clear effort to solve each problem, you'll get full credit. That said, make sure you understand the concepts here, because they're likely to reappear in homeworks, quizzes, and later lectures.

As you work, write your answers in this notebook. Answer questions with complete sentences, and put code in code chunks. You can make as many new code chunks as you like.

- Knit and submit the generated PDF file on Gradescope.

Default Arguments

Watch the "Default Arguments" lecture video.

Exercise 1

Write a function `to_kelvin()` to convert temperatures in degrees Celsius to temperatures in degrees Kelvin (you can find the formula online). Your function should have a parameter `celsius` for the temperature to convert.

Make sure that your function is vectorized in `celsius`, so that it can convert an entire vector of temperatures in one call.

Show that your function works correctly for a few test cases.

YOUR ANSWER GOES HERE:

```
to_kelvin = function(celsius) {
  celsius + 273.15
}

to_kelvin(numeric(0))
```

```
## numeric(0)
```

```
to_kelvin(0)
```

```
## [1] 273.15
```

```
to_kelvin(c(0, 20, 100))
```

```
## [1] 273.15 293.15 373.15
```

Function Example

The "Function Example" lecture video is optional. It provides a more realistic example of writing a function.

No exercises for this section.

Variable Scope & Lookup

Watch the "Variable Scope & Lookup" lecture video.

No exercises for this section.

Lazy Evaluation

Watch the "Lazy Evaluation" lecture video.

```
# simplified the body since it's simple
# default is something that doesn't exit = masking
# error out since jasdfl doesn't exist when x = jasdfl in parameters
func = function(y = x, x) {
  x = 2
  y
}

func(x = 1)
```

```
## [1] 2
```

Exercise 2

Extend your `to_kelvin()` function to be able to convert degrees Fahrenheit to degrees Kelvin as well. Add a parameter `fahrenheit` for the Fahrenheit temperature to convert.

Leave the code in the body of your function unchanged. Instead, provide a default argument for `celsius` that converts the `fahrenheit` argument.

Show that your function works correctly for a few test cases (for both Celsius and Fahrenheit temperatures).

What happens if you call your function with arguments to the `celsius` parameter and the `fahrenheit` parameter at the same time?

$$celsius = \frac{5}{9}(fahrenheit - 32)$$

YOUR ANSWER GOES HERE:

```r
to_kelvin = function(celsius = (fahrenheit - 32) * 5/9, fahrenheit) {
  celsius + 273.15
}

to_kelvin(celsius = c(0, 20, 30, 100))
```

```
## [1] 273.15 293.15 303.15 373.15
```

```r
to_kelvin(fahrenheit = c(32, 68, 86, 212))
```

```
## [1] 273.15 293.15 303.15 373.15
```

```r
to_kelvin(0, 10)
```

```
## [1] 273.15
```

```r
to_kelvin(celsius = 0, fahrenheit = asdfa)
```

```
## [1] 273.15
```

Calling the version of the `to_kelvin()` function above with both arguments converts only the `celsius` argument. The `fahrenheit` argument is completely ignored.

Depending on how you wrote `to_kelvin()`, your version might have different results.

Exercise 3

Rather than using a separate parameter in `to_kelvin()` for each source unit, a better design is to have a `temperature` parameter and a `unit` parameter. The function can then check `unit` and choose an appropriate conversion for `temperature`.

The `unit` parameter should be restricted to supported units, which in this case are `"celsius"` and `"fahrenheit"`. You can use the special `match.arg()` function to check that an argument matches against a set of candidate arguments. For instance, `match.arg(x, c("red", "blue"))` checks that the argument to `x` is either `"red"` or `"blue"`. The function also allows for partial matches, so for instance `"r"` matches `"red"`. See the documentation for full details.

Rewrite the `to_kelvin()` function with a `temperature` and `unit` parameter. Make sure the function is still vectorized in the `temperature` parameter.

As always, show that your function works correctly for a few test cases.

YOUR ANSWER GOES HERE:

```r
to_kelvin = function(temperature, unit) {
  unit = match.arg(unit, c("celsius", "fahrenheit"))
  if (unit == 'fahrenheit') {
    temperature = (temperature - 32) * 5/9
  }
  temperature + 273.15
}

to_kelvin(c(0, 20, 30, 100), "c")
```

```
## [1] 273.15 293.15 303.15 373.15
```

```
to_kelvin(c(32, 68, 86, 212), "fahrenheit")
```

```
## [1] 273.15 293.15 303.15 373.15
```

The Dots Parameter

Watch the "The Dots Parameter" lecture video.

No exercises for this section.

Using Functions

Watch the "Using Functions" lecture video.

No exercises for this section.