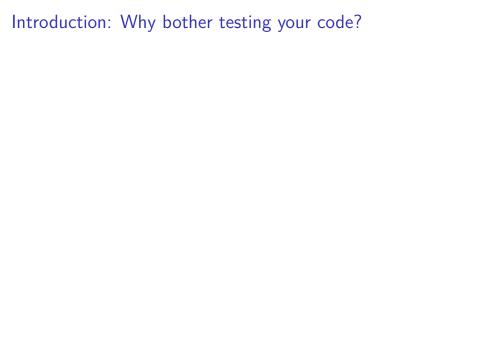
#### Introduction to testing R code

**Brad Duthie** 

1 May 2019



► You already do this informally (checking if a function works)

- You already do this informally (checking if a function works)
- Code that initially works might not later under different conditions

- You already do this informally (checking if a function works)
- Code that initially works might not later under different conditions
- Encourages better coding (writing shorter, more manageable functions)

- You already do this informally (checking if a function works)
- Code that initially works might not later under different conditions
- Encourages better coding (writing shorter, more manageable functions)
- Reassuring to double-check that everything works after changing something

- You already do this informally (checking if a function works)
- Code that initially works might not later under different conditions
- Encourages better coding (writing shorter, more manageable functions)
- Reassuring to double-check that everything works after changing something
- Gratifying to watch code pass multiple automated tests (it looks cool).

## Testing all of your code at once is satisfying

```
≔> devtools::test()
Loading GMSE
Testing GMSE
  I OK F W S I Context
               gmse apply tests [2.6 s]
              | Agent initialisation
              | Cost array initialisation
            | Main gmse function[1] "Initialising simulations ...
              Main gmse function[1] "Initialising simulations ...
              Main gmse function[1] "Initialising simulations ...
             Main gmse function[1] "Initialising simulations ...
              Main gmse function[1] "Initialising simulations ...
             Main gmse function[1] "Initialising simulations ... Main gmse function[1] "Initialising simulations ...
              Main gmse function[1] "Initialising simulations ...
            | Main gmse function[1] "Initialising simulations ...
              | Main gmse function [4.5 s]
              | Interaction array and table initialisation
              Landscape initialisation
              I Resource initialisation
              Action array initialisation
    11
              i Manager model
              | Observation model
              I Resource model
              Summary functions[1] "Initialising simulations ...
              | Summary functions [3.0 s]
              i User model
              Action and cost laver initialisation
= Results =
Duration: 10.3 s
Failed:
Warnings: 0
Skipped: 0
```

## Getting started: install the testthat package

Can install testthat from CRAN.

```
install.packages("testthat")
```

Or install from GitHub with the devtools R package.

```
devtools::install_github("r-lib/testthat");
```

Load testthat into Rstudio just like any other R package.

```
library(testthat);
```

#### Consider one R script (file with .R extension) with functions.

Function 1: converts a temperature from Fahrenheit to Celsius.

```
F_to_C <- function(F_temp){
    C_temp <- (F_temp - 32) * 5/9;
    return(C_temp);
}</pre>
```

#### Consider one R script (file with .R extension) with functions.

Function 1: converts a temperature from Fahrenheit to Celsius.

```
F_to_C <- function(F_temp){
    C_temp <- (F_temp - 32) * 5/9;
    return(C_temp);
}</pre>
```

Function 2: converts from Celsius to Fahrenheit.

```
C_to_F <- function(C_temp){
    F_temp <- (C_temp * 9/5) + 32;
    return(F_temp);
}</pre>
```

#### Consider one R script (file with .R extension) with functions.

Function 1: converts a temperature from Fahrenheit to Celsius.

```
F_to_C(50)
```

```
## [1] 10
```

#### Consider one R script (file with .R extension) with functions.

Function 1: converts a temperature from Fahrenheit to Celsius.

## [1] 10

Function 2: converts from Celsius to Fahrenheit.

```
C_to_F(10)
```

```
## [1] 50
```

## How the test\_that function works

Example of a testthat R script, < test-temp\_conversion.R >

```
library(testthat);
context("Temperature function testing");
source("temp conversion.R"); # Functions to test
test that("Fahrenheit to Celsius", {
  temp C \leftarrow F to C(50);
  # Test that the result is numeric
  expect_that( is.numeric(temp_C), equals(TRUE) );
  # Test that the result is the correct value
  expect that( temp C, equals(10) );
})
```

#### How to run many tests quickly

Check Rstudio is set to the same directory as the test R script(s)

```
test_dir("."); # Runs all of your tests
```

The above will run all files with the prefix test- and extension .R.

#### How to run many tests quickly

# Check Rstudio is set to the same directory as the test R script(s)

```
test_dir("."); # Runs all of your tests
```

The above will run all files with the prefix test- and extension .R.

#### Test output looks like this:

```
V | OK F M S | Context

X | 4 1 | Temperature function testing | 4 | Temperature function testing

test-temp_conversion.R:30: failure: Fahrenheit to Celsius wrong
'x' not equal to 'expected'.
1/1 mismatches
[1] 10 - 2 == 8

Results

OK: 4
Failed: 1
Warnings: 0
Skipped: 0
```

## Testing as part of your coding work flow

#### Hadley Wickham also makes several key points about testing

- ▶ Helps spot bugs in the code earlier and see what needs fixing
- ▶ Multiple unit tests encourages smaller, manageable functions
- Writing failing tests can be useful when fixing bugs

## Testing as part of your coding work flow

#### Hadley Wickham also makes several key points about testing

- ▶ Helps spot bugs in the code earlier and see what needs fixing
- ▶ Multiple unit tests encourages smaller, manageable functions
- Writing failing tests can be useful when fixing bugs

#### Additional resources

- ► Example of unit testing R code with testthat (John D. Cook)
- Testing R packages (Hadley Wickam)