CT1100: Computer Systems

Topic 6: Running Scripts and using pipes (magrittr)

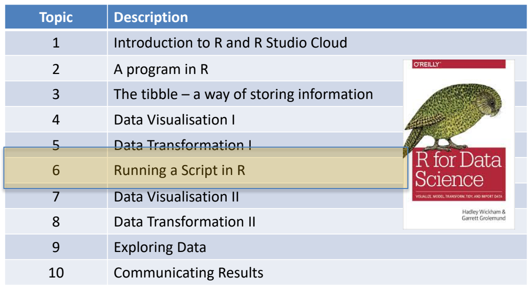
Prof. Jim Duggan,
School of Engineering & Informatics
National University of Ireland Galway.

Overview

- select() from dplyr
- Running Scripts
- The Pipe Operator

dplyr – "A grammar of data manipulation"

Function	Purpose
filter()	Pick observations by their values
arrange()	Reorder the rows
select()	Pick variables by their names
mutate()	*Create new variables with functions of existing variables*
summarise()	*Collapse many values down to a single summary*



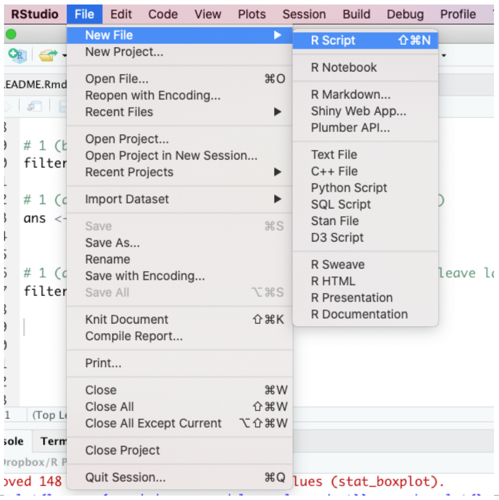
Topic	Description
1	Introduction to R and R Studio Cloud
2	A program in R
3	The tibble – a way of storing information
4	Data Visualisation I
5	Data Transformation I
6	Running a Script in R
7	Data Visualisation II
8	Data Transformation II
9	Exploring Data
10	Communicating Results

O'REILLY
R for Data Science
Hadley Wickham & Garrett Grolemund

https://r4ds.had.co.nz

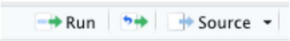
(2) Running Scripts

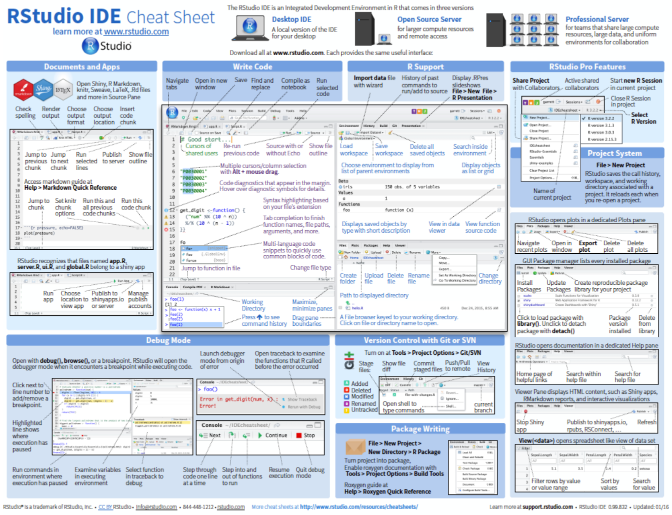
- The console is very useful, but it has limitations
- To provide more room for work, the script editor should be used.
- The script should contain code that you care about
- For a new script:
  - New File
  - R Script
- Save your scripts regularly!



Tips & Shortcuts (1)

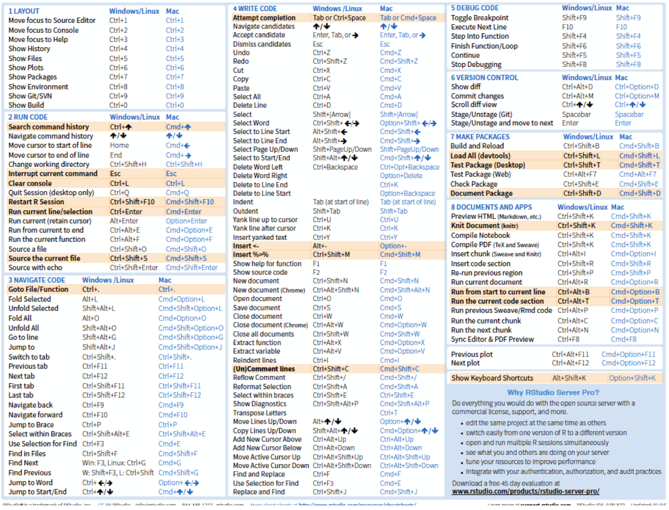
- Run a line Cmd/Ctrl-Enter
- Run a file Cms/Ctrl-Shift-S
- Or use the R Studio buttons





Tips & Shortcuts (2)

- Always start your script with the packages you need
- That way, you can clearly show other coders what is needed to run the script.
- Avoid calling install.packages() or setwd() in a script, as you will be changing someone elses settings!



1 LAYOUT	Windows/Linux	Mac
Move focus to Source Editor	Ctrl+1	Ctrl+1
Move focus to Console	Ctrl+2	Ctrl+2
Move focus to Help	Ctrl+3	Ctrl+3
Show History	Ctrl+4	Ctrl+4
Show Files	Ctrl+5	Ctrl+5
Show Plots	Ctrl+6	Ctrl+6
Show Packages	Ctrl+7	Ctrl+7
Show Environment	Ctrl+8	Ctrl+8
Show Git/SVN	Ctrl+9	Ctrl+9
Show Build	Ctrl+0	Ctrl+0

2 RUN CODE	Windows/Linux	Mac
Search command history	**Ctrl+↑**	**Cmd+↑**
Navigate command history	↑/↓	↑/↓
Move cursor to start of line	Home	Cmd+←
Move cursor to end of line	End	Cmd+→
Change working directory	Ctrl+Shift+H	Ctrl+Shift+H
Interrupt current command	**Esc**	**Esc**
Clear console	**Ctrl+L**	**Ctrl+L**
Quit Session (desktop only)	Ctrl+Q	Cmd+Q
Restart R Session	**Ctrl+Shift+F10**	**Cmd+Shift+F10**
Run current line/selection	**Ctrl+Enter**	**Cmd+Enter**
Run current (retain cursor)	Alt+Enter	Option+Enter
Run from current to end	Ctrl+Alt+E	Cmd+Option+E
Run the current function	Ctrl+Alt+F	Cmd+Option+F
Source a file	Ctrl+Shift+O	Cmd+Shift+O
Source the current file	**Ctrl+Shift+S**	**Cmd+Shift+S**
Source with echo	Ctrl+Shift+Enter	Cmd+Shift+Enter

3 NAVIGATE CODE	Windows /Linux	Mac
Goto File/Function	**Ctrl+.**	**Ctrl+.**
Fold Selected	Alt+L	Cmd+Option+L
Unfold Selected	Shift+Alt+L	Cmd+Shift+Option+L
Fold All	Alt+O	Cmd+Option+O
Unfold All	Shift+Alt+O	Cmd+Shift+Option+O
Go to line	Shift+Alt+G	Cmd+Shift+Option+G
Jump to	Shift+Alt+J	Cmd+Shift+Option+J
Switch to tab	Ctrl+Shift+.	Ctrl+Shift+.
Previous tab	Ctrl+F11	Ctrl+F11
Next tab	Ctrl+F12	Ctrl+F12
First tab	Ctrl+Shift+F11	Ctrl+Shift+F11
Last tab	Ctrl+Shift+F12	Ctrl+Shift+F12
Navigate back	Ctrl+F9	Cmd+F9
Navigate forward	Ctrl+F10	Cmd+F10
Jump to Brace	Ctrl+P	Ctrl+P
Select within Braces	Ctrl+Shift+Alt+E	Ctrl+Shift+Alt+E
Use Selection for Find	Ctrl+F3	Cmd+E
Find in Files	Ctrl+Shift+F	Cmd+Shift+F
Find Next	Win: F3, Linux: Ctrl+G	Cmd+G
Find Previous	W: Shift+F3, L: Ctrl+Shift	Cmd+Shift+G
Jump to Word	Ctrl+ ←/→	Option+ ←/→
Jump to Start/End	Ctrl+↑/↓	Cmd+↑/↓

4 WRITE CODE	Windows /Linux	Mac
Attempt completion	**Tab or Ctrl+Space**	**Tab or Cmd+Space**
Navigate candidates	↑/↓	↑/↓
Accept candidate	Enter, Tab, or →	Enter, Tab, or →
Dismiss candidates	Esc	Esc
Undo	Ctrl+Z	Cmd+Z
Redo	Ctrl+Shift+Z	Cmd+Shift+Z
Cut	Ctrl+X	Cmd+X
Copy	Ctrl+C	Cmd+C
Paste	Ctrl+V	Cmd+V
Select All	Ctrl+A	Cmd+A
Delete Line	Ctrl+D	Cmd+D
Select	Shift+[Arrow]	Shift+[Arrow]
Select Word	Ctrl+Shift+ ←/→	Option+Shift+ ←/→
Select to Line Start	Alt+Shift+←	Cmd+Shift+←
Select to Line End	Alt+Shift+→	Cmd+Shift+→
Select Page Up/Down	Shift+PageUp/Down	Shift+PageUp/Down
Select to Start/End	Shift+Alt+↑/↓	Cmd+Shift+↑/↓
Delete Word Left	Ctrl+Backspace	Ctrl+Opt+Backspace
Delete Word Right		Option+Delete
Delete to Line End		Ctrl+K
Delete to Line Start		Option+Backspace
Indent	Tab (at start of line)	Tab (at start of line)
Outdent	Shift+Tab	Shift+Tab
Yank line up to cursor	Ctrl+U	Ctrl+U
Yank line after cursor	Ctrl+K	Ctrl+K
Insert yanked text	Ctrl+Y	Ctrl+Y
Insert <-	**Alt+-**	**Option+-**
Insert %>%	**Ctrl+Shift+M**	**Cmd+Shift+M**
Show help for function	F1	F1
Show source code	F2	F2
New document	Ctrl+Shift+N	Cmd+Shift+N
New document (Chrome)	Ctrl+Alt+Shift+N	Cmd+Shift+Alt+N
Open document	Ctrl+O	Cmd+O
Save document	Ctrl+S	Cmd+S
Close document	Ctrl+W	Cmd+W
Close document (Chrome)	Ctrl+Alt+W	Cmd+Option+W
Close all documents	Ctrl+Shift+W	Cmd+Shift+W
Extract function	Ctrl+Alt+X	Cmd+Option+X
Extract variable	Ctrl+Alt+V	Cmd+Option+V
Reindent lines	Ctrl+I	Cmd+I
(Un)Comment lines	**Ctrl+Shift+C**	**Cmd+Shift+C**
Reflow Comment	Ctrl+Shift+/	Cmd+Shift+/
Reformat Selection	Ctrl+Shift+A	Cmd+Shift+A
Select within braces	Ctrl+Shift+E	Ctrl+Shift+E
Show Diagnostics	Ctrl+Shift+Alt+P	Cmd+Shift+Alt+P
Transpose Letters		Ctrl+T
Move Lines Up/Down	Alt+↑/↓	Option+↑/↓
Copy Lines Up/Down	Shift+Alt+↑/↓	Cmd+Option+↑/↓
Add New Cursor Above	Ctrl+Alt+Up	Ctrl+Alt+Up
Add New Cursor Below	Ctrl+Alt+Down	Ctrl+Alt+Down
Move Active Cursor Up	Ctrl+Alt+Shift+Up	Ctrl+Alt+Shift+Up
Move Active Cursor Down	Ctrl+Alt+Shift+Down	Ctrl+Alt+Shift+Down
Find and Replace	Ctrl+F	Cmd+F
Use Selection for Find	Ctrl+F3	Cmd+E
Replace and Find	Ctrl+Shift+J	Cmd+Shift+J

5 DEBUG CODE	Windows/Linux	Mac
Toggle Breakpoint	Shift+F9	Shift+F9
Execute Next Line	F10	F10
Step Into Function	Shift+F4	Shift+F4
Finish Function/Loop	Shift+F6	Shift+F6
Continue	Shift+F5	Shift+F5
Stop Debugging	Shift+F8	Shift+F8

6 VERSION CONTROL	Windows/Linux	Mac
Show diff	Ctrl+Alt+D	Ctrl+Option+D
Commit changes	Ctrl+Alt+M	Ctrl+Option+M
Scroll diff view	Ctrl+↑/↓	Ctrl+↑/↓
Stage/Unstage (Git)	Spacebar	Spacebar
Stage/Unstage and move to next	Enter	Enter

7 MAKE PACKAGES	Windows/Linux	Mac
Build and Reload	Ctrl+Shift+B	Cmd+Shift+B
Load All (devtools)	**Ctrl+Shift+L**	**Cmd+Shift+L**
Test Package (Desktop)	**Ctrl+Shift+T**	**Cmd+Shift+T**
Test Package (Web)	Ctrl+Alt+F7	Cmd+Alt+F7
Check Package	Ctrl+Shift+E	Cmd+Shift+E
Document Package	**Ctrl+Shift+D**	**Cmd+Shift+D**

8 DOCUMENTS AND APPS	Windows/Linux	Mac
Preview HTML (Markdown, etc.)	Ctrl+Shift+K	Cmd+Shift+K
Knit Document (knitr)	**Ctrl+Shift+K**	**Cmd+Shift+K**
Compile Notebook	Ctrl+Shift+K	Cmd+Shift+K
Compile PDF (TeX and Sweave)	Ctrl+Shift+K	Cmd+Shift+K
Insert chunk (Sweave and Knitr)	Ctrl+Alt+I	Cmd+Option+I
Insert code section	Ctrl+Shift+R	Cmd+Shift+R
Re-run previous region	Ctrl+Shift+P	Cmd+Shift+P
Run current document	Ctrl+Alt+R	Cmd+Option+R
Run from start to current line	**Ctrl+Alt+B**	**Cmd+Option+B**
Run the current code section	**Ctrl+Alt+T**	**Cmd+Option+T**
Run previous Sweave/Rmd code	Ctrl+Alt+P	Cmd+Option+P
Run the current chunk	Ctrl+Alt+C	Cmd+Option+C
Run the next chunk	Ctrl+Alt+N	Cmd+Option+N
Sync Editor & PDF Preview	Ctrl+F8	Cmd+F8

Previous plot	Ctrl+Alt+F11	Cmd+Option+F11
Next plot	Ctrl+Alt+F12	Cmd+Option+F12

Show Keyboard Shortcuts	Alt+Shift+K	Option+Shift+K

Why RStudio Server Pro?

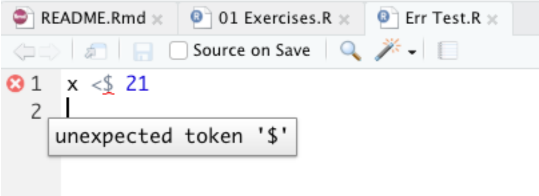
Do everything you would do with the open source server with a commercial license, support, and more.

- edit the same project at the same time as others
- switch easily from one version of R to a different version
- open and run multiple R sessions simultaneously
- see what you and others are doing on your server
- tune your resources to improve performance
- integrate with your authentication, authorization, and audit practices

Download a free 45 day evaluation at **www.rstudio.com/products/rstudio-server-pro/**

Tips & Shortcuts (3)

- The script editor will highlight syntax errors with a red squiggly line and a cross on the sidebar.

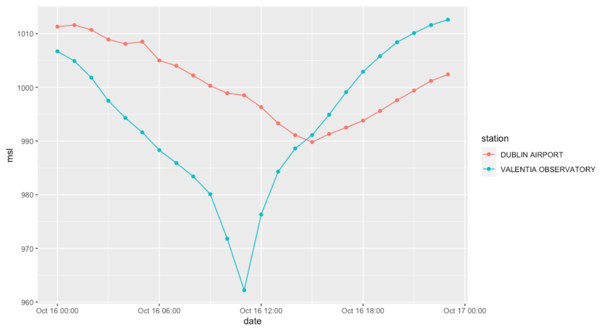


```
x <$ 21
```

unexpected token '$'

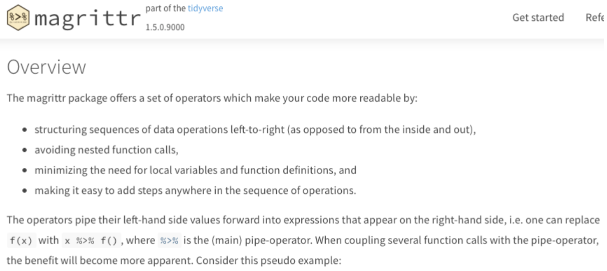
Challenge 6.2 – Place in a script and run

- Create tibble one that has the columns month, hour, day, date, station and msl
- Filter the tibble to a second tibble for October 16th, and for “VALENTIA OBSERVATORY” and “DUBLIN AIRPORT”
- Display the hourly values on a time series (x axis is date) using ggplot2 with the aesthetic set to station



(3) Combining operations with the Pipe

- The pipe %>% comes from the magrittr package (Stefan Milton Bache)
- Helps to write code that is easier to read and understand
- x %>% f(y) turns into f(x, y)
- x %>% f(y) %>% g(z) turns into g(f(x, y),z)



Overview

The magrittr package offers a set of operators which make your code more readable by:

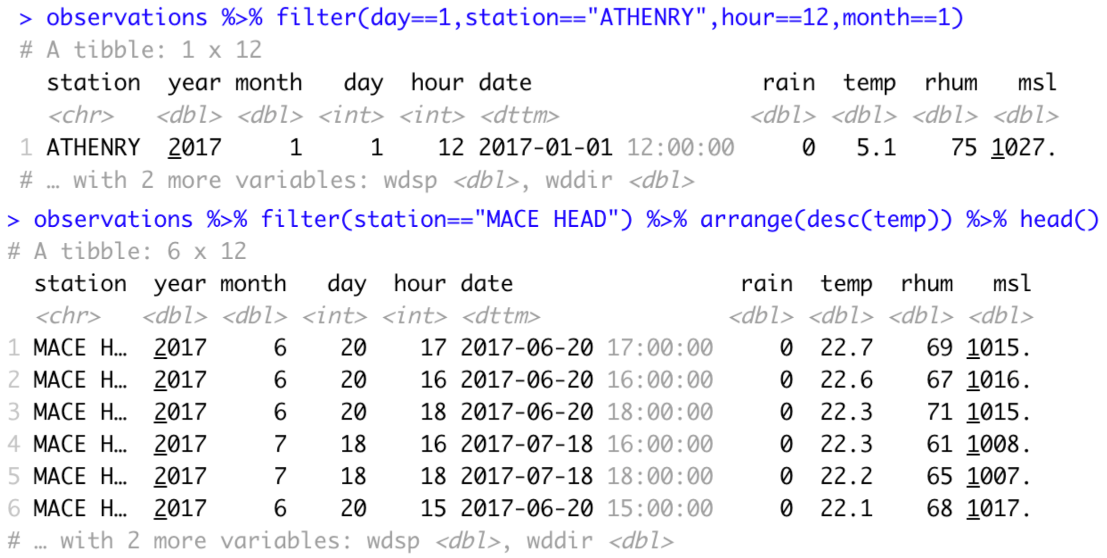
- structuring sequences of data operations left-to-right (as opposed to from the inside and out),
- avoiding nested function calls,
- minimizing the need for local variables and function definitions, and
- making it easy to add steps anywhere in the sequence of operations.

The operators pipe their left-hand side values forward into expressions that appear on the right-hand side, i.e. one can replace `f(x)` with `x %>% f()`, where `%>%` is the (main) pipe-operator. When coupling several function calls with the pipe-operator, the benefit will become more apparent. Consider this pseudo example:

https://magrittr.tidyverse.org

```
> sqrt(1:5)
[1] 1.000000 1.414214 1.732051 2.000000 2.236068
> 1:5 %>% sqrt()
[1] 1.000000 1.414214 1.732051 2.000000 2.236068
```

Examples



```
> observations %>% filter(day==1,station=="ATHENRY",hour==12,month==1)
# A tibble: 1 x 12
  station  year month   day  hour date                 rain  temp  rhum   msl
  <chr>   <dbl> <dbl> <int> <int> <dttm>              <dbl> <dbl> <dbl> <dbl>
1 ATHENRY  2017     1     1    12 2017-01-01 12:00:00     0   5.1    75 1027.
# … with 2 more variables: wdsp <dbl>, wddir <dbl>
> observations %>% filter(station=="MACE HEAD") %>% arrange(desc(temp)) %>% head()
# A tibble: 6 x 12
  station  year month   day  hour date                 rain  temp  rhum   msl
  <chr>   <dbl> <dbl> <int> <int> <dttm>              <dbl> <dbl> <dbl> <dbl>
1 MACE H…  2017     6    20    17 2017-06-20 17:00:00     0  22.7    69 1015.
2 MACE H…  2017     6    20    16 2017-06-20 16:00:00     0  22.6    67 1016.
3 MACE H…  2017     6    20    18 2017-06-20 18:00:00     0  22.3    71 1015.
4 MACE H…  2017     7    18    16 2017-07-18 16:00:00     0  22.3    61 1008.
5 MACE H…  2017     7    18    18 2017-07-18 18:00:00     0  22.2    65 1007.
6 MACE H…  2017     6    20    15 2017-06-20 15:00:00     0  22.1    68 1017.
# … with 2 more variables: wdsp <dbl>, wddir <dbl>
```

Challenge 6.2

- Organise the following into a pipeline command
  - Subset all observations from aimsir17 from October 2017
  - Select all those from “ROCHES POINT”
  - Sort the observations by wind speed (descending)

Summary: 3 of the 5 verbs

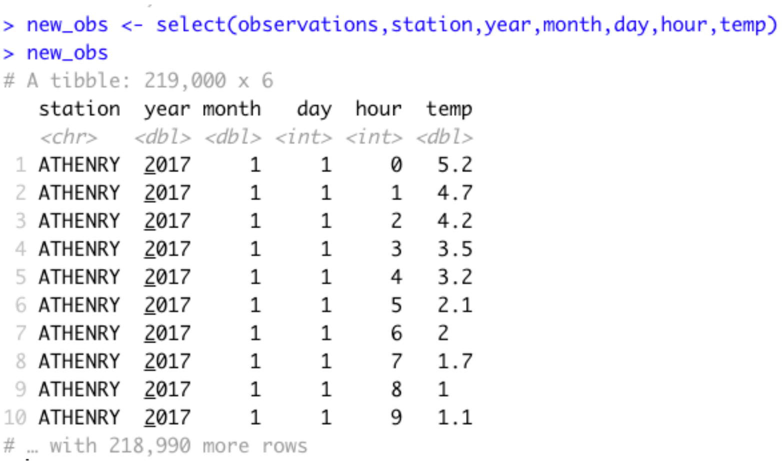
Function	Purpose
filter()	Pick observations by their values
arrange()	Reorder the rows
select()	Pick variables by their names
mutate()	*Create new variables with functions of existing variables*
summarise()	*Collapse many values down to a single summary*

- All verbs (functions) work similarly
  - The first argument is a data frame/tibble
  - The subsequent arguments decide what to do with the data frame/tibble
  - The result (data frame/tibble) supports chaining of steps – NOTE the “pipe operator” which we will cover later.

https://dplyr.tidyverse.org

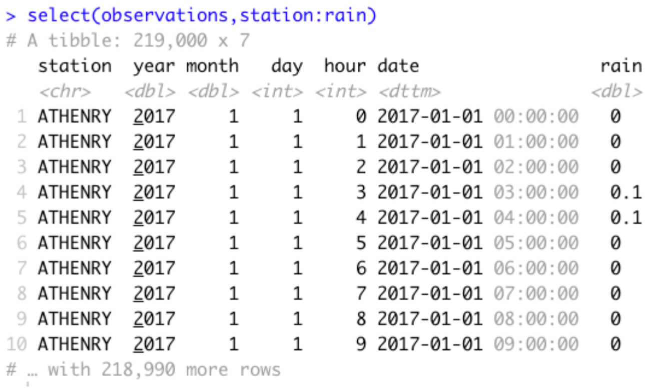
3. select()

- It is not uncommon to get datasets with hundreds, or even thousands, of variables
- A challenge is to narrow down on the variables of you’re interested in
- select() allows you to rapidly zoom in on a useful subset using operations based on the variable names
- Number of rows does not change

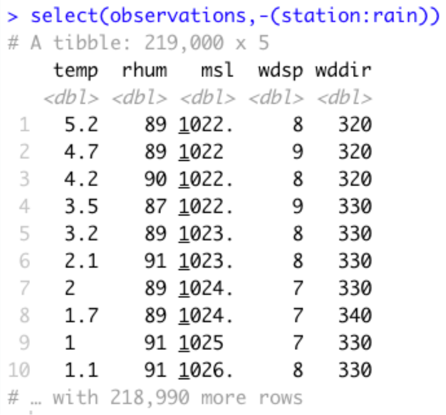


```
> new_obs <- select(observations,station,year,month,day,hour,temp)
> new_obs
# A tibble: 219,000 x 6
   station  year month   day  hour  temp
   <chr>   <dbl> <dbl> <int> <int> <dbl>
 1 ATHENRY  2017     1     1     0   5.2
 2 ATHENRY  2017     1     1     1   4.7
 3 ATHENRY  2017     1     1     2   4.2
 4 ATHENRY  2017     1     1     3   3.5
 5 ATHENRY  2017     1     1     4   3.2
 6 ATHENRY  2017     1     1     5   2.1
 7 ATHENRY  2017     1     1     6   2  
 8 ATHENRY  2017     1     1     7   1.7
 9 ATHENRY  2017     1     1     8   1  
10 ATHENRY  2017     1     1     9   1.1
# … with 218,990 more rows
```

Useful options with select()

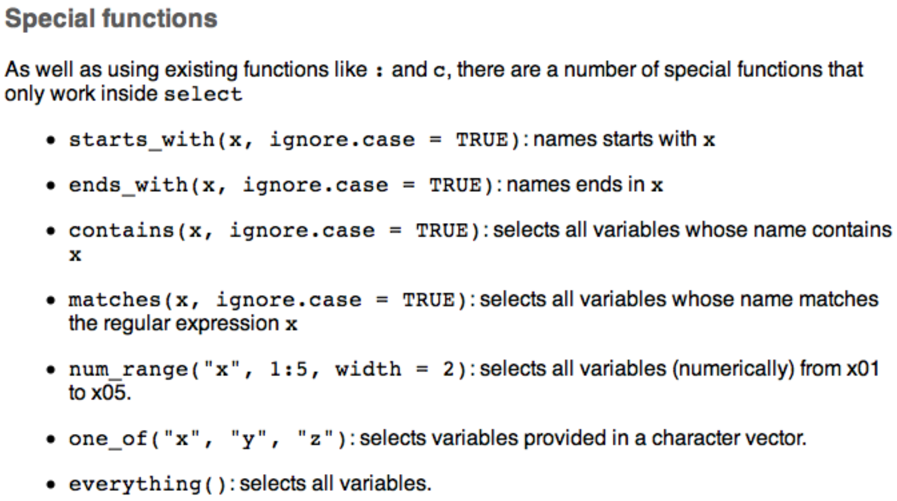


```
> select(observations,station:rain)
# A tibble: 219,000 x 7
   station  year month   day  hour date                 rain
   <chr>   <dbl> <dbl> <int> <int> <dttm>              <dbl>
 1 ATHENRY  2017     1     1     0 2017-01-01 00:00:00   0  
 2 ATHENRY  2017     1     1     1 2017-01-01 01:00:00   0  
 3 ATHENRY  2017     1     1     2 2017-01-01 02:00:00   0  
 4 ATHENRY  2017     1     1     3 2017-01-01 03:00:00   0.1
 5 ATHENRY  2017     1     1     4 2017-01-01 04:00:00   0.1
 6 ATHENRY  2017     1     1     5 2017-01-01 05:00:00   0  
 7 ATHENRY  2017     1     1     6 2017-01-01 06:00:00   0  
 8 ATHENRY  2017     1     1     7 2017-01-01 07:00:00   0  
 9 ATHENRY  2017     1     1     8 2017-01-01 08:00:00   0  
10 ATHENRY  2017     1     1     9 2017-01-01 09:00:00   0  
# … with 218,990 more rows
```



```
> select(observations,-(station:rain))
# A tibble: 219,000 x 5
    temp  rhum   msl  wdsp wddir
   <dbl> <dbl> <dbl> <dbl> <dbl>
 1   5.2    89 1022.     8   320
 2   4.7    89 1022      9   320
 3   4.2    90 1022.     8   320
 4   3.5    87 1022.     9   330
 5   3.2    89 1023.     8   330
 6   2.1    91 1023.     8   330
 7   2      89 1024.     7   330
 8   1.7    89 1024.     7   340
 9   1      91 1025      7   330
10   1.1    91 1026.     8   330
# … with 218,990 more rows
```

Special functions with select()

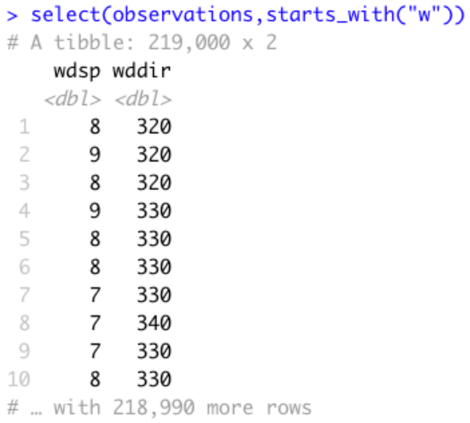


Special functions

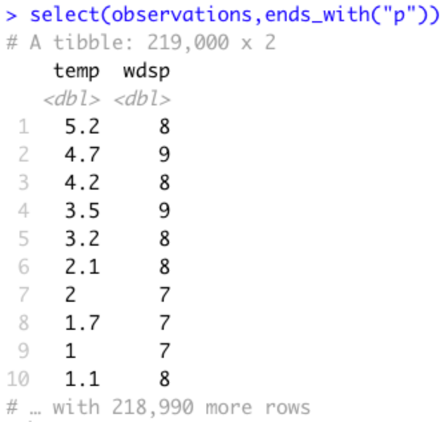
As well as using existing functions like `:` and `c`, there are a number of special functions that only work inside `select`

- `starts_with(x, ignore.case = TRUE)`: names starts with `x`
- `ends_with(x, ignore.case = TRUE)`: names ends in `x`
- `contains(x, ignore.case = TRUE)`: selects all variables whose name contains `x`
- `matches(x, ignore.case = TRUE)`: selects all variables whose name matches the regular expression `x`
- `num_range("x", 1:5, width = 2)`: selects all variables (numerically) from x01 to x05.
- `one_of("x", "y", "z")`: selects variables provided in a character vector.
- `everything()`: selects all variables.

Examples



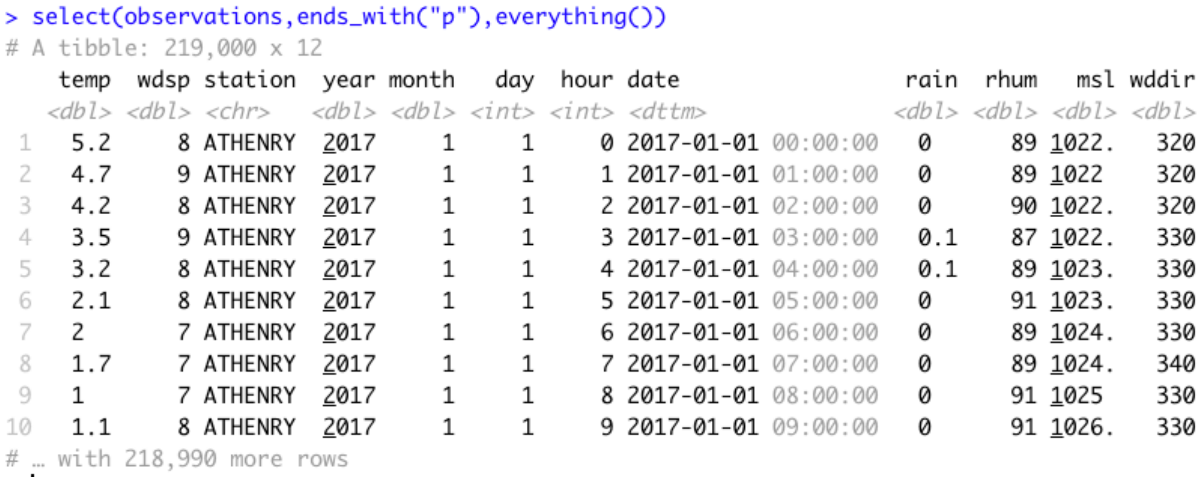
```
> select(observations,starts_with("w"))
# A tibble: 219,000 x 2
    wdsp wddir
   <dbl> <dbl>
 1     8   320
 2     9   320
 3     8   320
 4     9   330
 5     8   330
 6     8   330
 7     7   330
 8     7   340
 9     7   330
10     8   330
# … with 218,990 more rows
```



```
> select(observations,ends_with("p"))
# A tibble: 219,000 x 2
    temp  wdsp
   <dbl> <dbl>
 1   5.2     8
 2   4.7     9
 3   4.2     8
 4   3.5     9
 5   3.2     8
 6   2.1     8
 7   2       7
 8   1.7     7
 9   1       7
10   1.1     8
# … with 218,990 more rows
```

15

everything()

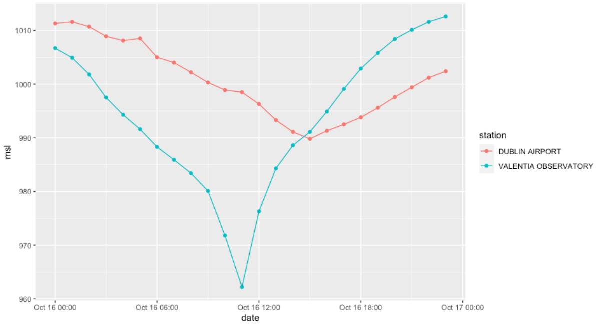


```
> select(observations,ends_with("p"),everything())
# A tibble: 219,000 x 12
    temp  wdsp station  year month   day  hour date                 rain  rhum   msl wddir
   <dbl> <dbl> <chr>   <dbl> <dbl> <int> <int> <dttm>              <dbl> <dbl> <dbl> <dbl>
 1   5.2     8 ATHENRY  2017     1     1     0 2017-01-01 00:00:00   0      89 1022.   320
 2   4.7     9 ATHENRY  2017     1     1     1 2017-01-01 01:00:00   0      89 1022    320
 3   4.2     8 ATHENRY  2017     1     1     2 2017-01-01 02:00:00   0      90 1022.   320
 4   3.5     9 ATHENRY  2017     1     1     3 2017-01-01 03:00:00   0.1    87 1022.   330
 5   3.2     8 ATHENRY  2017     1     1     4 2017-01-01 04:00:00   0.1    89 1023.   330
 6   2.1     8 ATHENRY  2017     1     1     5 2017-01-01 05:00:00   0      91 1023.   330
 7   2       7 ATHENRY  2017     1     1     6 2017-01-01 06:00:00   0      89 1024.   330
 8   1.7     7 ATHENRY  2017     1     1     7 2017-01-01 07:00:00   0      89 1024.   340
 9   1       7 ATHENRY  2017     1     1     8 2017-01-01 08:00:00   0      91 1025    330
10   1.1     8 ATHENRY  2017     1     1     9 2017-01-01 09:00:00   0      91 1026.   330
# … with 218,990 more rows
```

16

Challenge 6.1

- Create tibble one that has the columns month, hour, day, date, station and msl
- Filter the tibble to a second tibble for October 16th, and for “VALENTIA OBSERVATORY” and “DUBLIN AIRPORT”
- Display the hourly values on a time series (x axis is date) using ggplot2 with the aesthetic set to station


msl
Oct 16 00:00
Oct 16 06:00
Oct 16 12:00
Oct 16 18:00
Oct 17 00:00
date
station
DUBLIN AIRPORT
VALENTIA OBSERVATORY