Blistering speed

Is Rcpp really so scary?

Zachary Waller

zwaller01@qub.ac.uk

QUEEN'S UNIVERSITY BELFAST

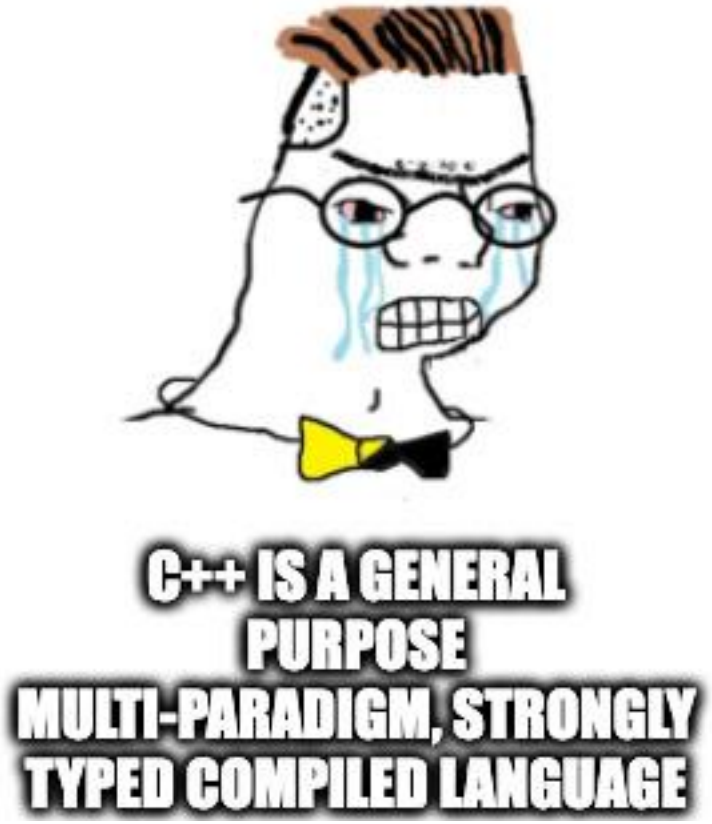
C-what-what?

- C++ is a…
  - General purpose
  - Multi-paradigm
  - Strongly typed
  - Compiled language

……

C-what-what?

C-what-what?


C++ IS A GENERAL PURPOSE MULTI-PARADIGM, STRONGLY TYPED COMPILED LANGUAGE


HAHA, C++ GO BRRRR

C-what-what?

- C++ is a lower-level language than R
- It can go *really fast*
- It's not too hard to learn if you ignore the complicated stuff and save it for simple tasks

R under the hood...

- Lots of R is written in C, C++ and Fortran

- For example `dplyr::filter` calls a C++ function at the end

```
> dplyr:::dplyr_reconstruct.data.frame
function (data, template)
{
    .Call(ffi_dplyr_reconstruct, data, template)
}
<bytecode: 0x12884b7c0>
<environment: namespace:dplyr>
>
```

R under the hood...

- Inside `dplyr::ffi_dplyr_reconstruct`

```
SEXP ffi_dplyr_reconstruct(SEXP data, SEXP template_) {
  if (TYPEOF(data) != VECSXP) {
    Rf_errorcall(R_NilValue, "Internal error: `data` must be a list.");
  }
  if (TYPEOF(template_) != VECSXP) {
    Rf_errorcall(R_NilValue, "Internal error: `template` must be a list.");
  }
  if (!OBJECT(data)) {
    Rf_errorcall(R_NilValue, "Internal error: `data` must be an object.");
  }
  if (!OBJECT(template_)) {
    Rf_errorcall(R_NilValue, "Internal error: `template` must be an object.");
  }

  bool seen_names = false;
  bool seen_row_names = false;
```

R under the hood...

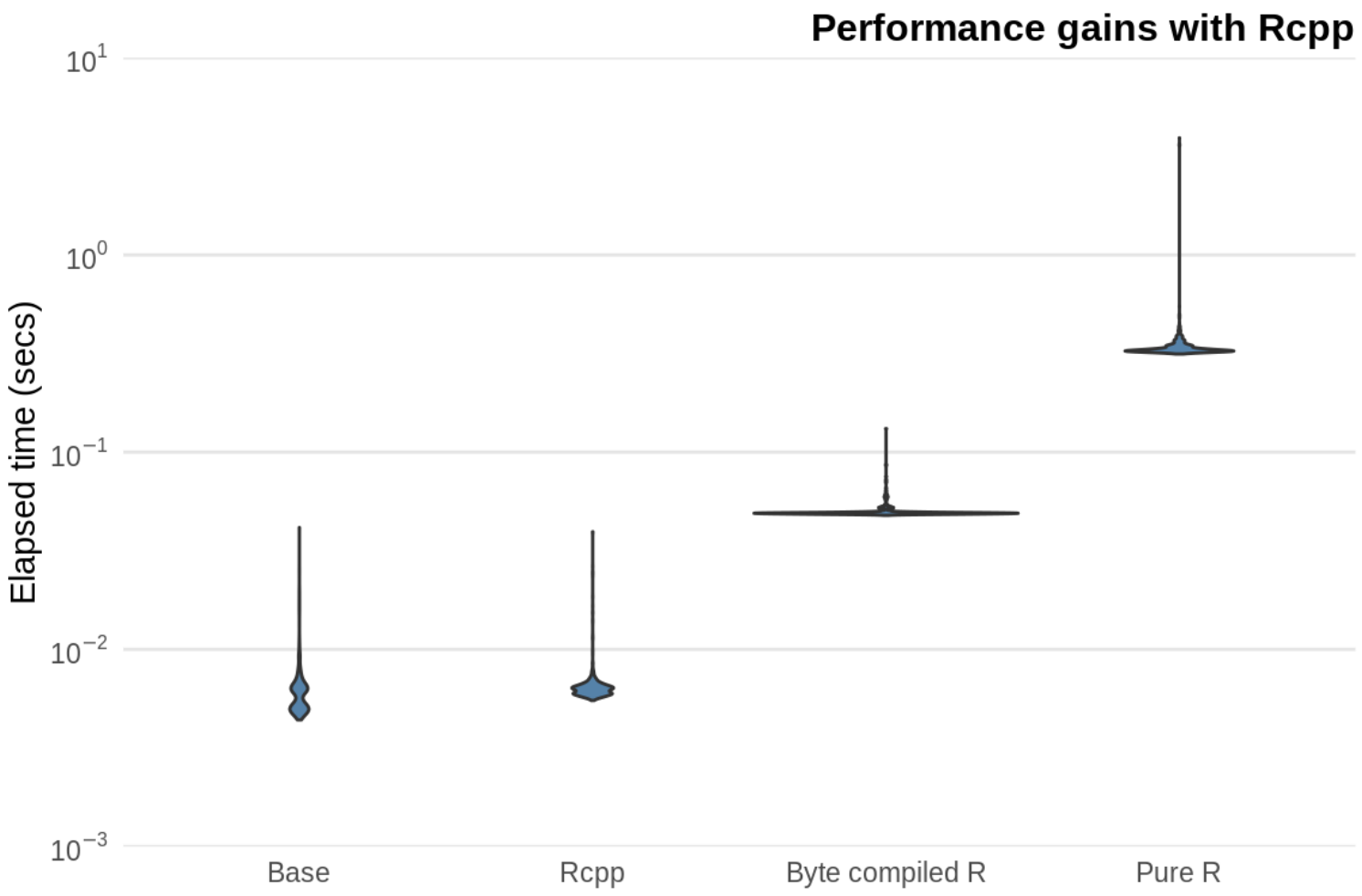
Rcpp

- Rcpp is an R package
  - Eddelbuettel et al. (2011)
- Let's us call C++ code from R
- You can do this other ways, but Rcpp is a lot easier
- Rcpp also adds functionality to make the transition easier

Motivation

- PhD student in mathematics at Queen's University Belfast
- Developing methods for analysing survival data with interval-censored event times
  - You only know an event happened between two observation times, but not exactly when it happened
- Non-parametric maximum likelihood estimation
- No analytical solution
- Need iterative algorithms
  - I want them to be fast

How fast we talkin?



Source: Efficient R Programming, Gillespie and Lovelace

Small example…

Best solutions to slow code

- Be patient
- Use a faster computer
- See if someone has solved the same problem already

HOW LONG CAN YOU WORK ON MAKING A ROUTINE TASK MORE EFFICIENT BEFORE YOU'RE SPENDING MORE TIME THAN YOU SAVE?

(ACROSS FIVE YEARS)

HOW MUCH TIME YOU SHAVE OFF	HOW OFTEN YOU DO THE TASK: 50/DAY	5/DAY	DAILY	WEEKLY	MONTHLY	YEARLY
1 SECOND	1 DAY	2 HOURS	30 MINUTES	4 MINUTES	1 MINUTE	5 SECONDS
5 SECONDS	5 DAYS	12 HOURS	2 HOURS	21 MINUTES	5 MINUTES	25 SECONDS
30 SECONDS	4 WEEKS	3 DAYS	12 HOURS	2 HOURS	30 MINUTES	2 MINUTES
1 MINUTE	8 WEEKS	6 DAYS	1 DAY	4 HOURS	1 HOUR	5 MINUTES
5 MINUTES	9 MONTHS	4 WEEKS	6 DAYS	21 HOURS	5 HOURS	25 MINUTES
30 MINUTES		6 MONTHS	5 WEEKS	5 DAYS	1 DAY	2 HOURS
1 HOUR		10 MONTHS	2 MONTHS	10 DAYS	2 DAYS	5 HOURS
6 HOURS				2 MONTHS	2 WEEKS	1 DAY
1 DAY					8 WEEKS	5 DAYS

https://xkcd.com/1205

C++ basics

- You have to declare variables **before** you use them
- Loops start at 0
  - This **will** trip you up
- By default C++ doesn't have vector/matrix multiplication

C++ basics

Simple example

Getting started

- You can copy and paste into R and compile with Rcpp

```
Rcpp::cppFunction(
'double sumC(NumericVector x) {
  int n = x.size();
  double total = 0;
  for(int i = 0; i < n; ++i) {
    total += x[i];
  }
  return total;
}')
```

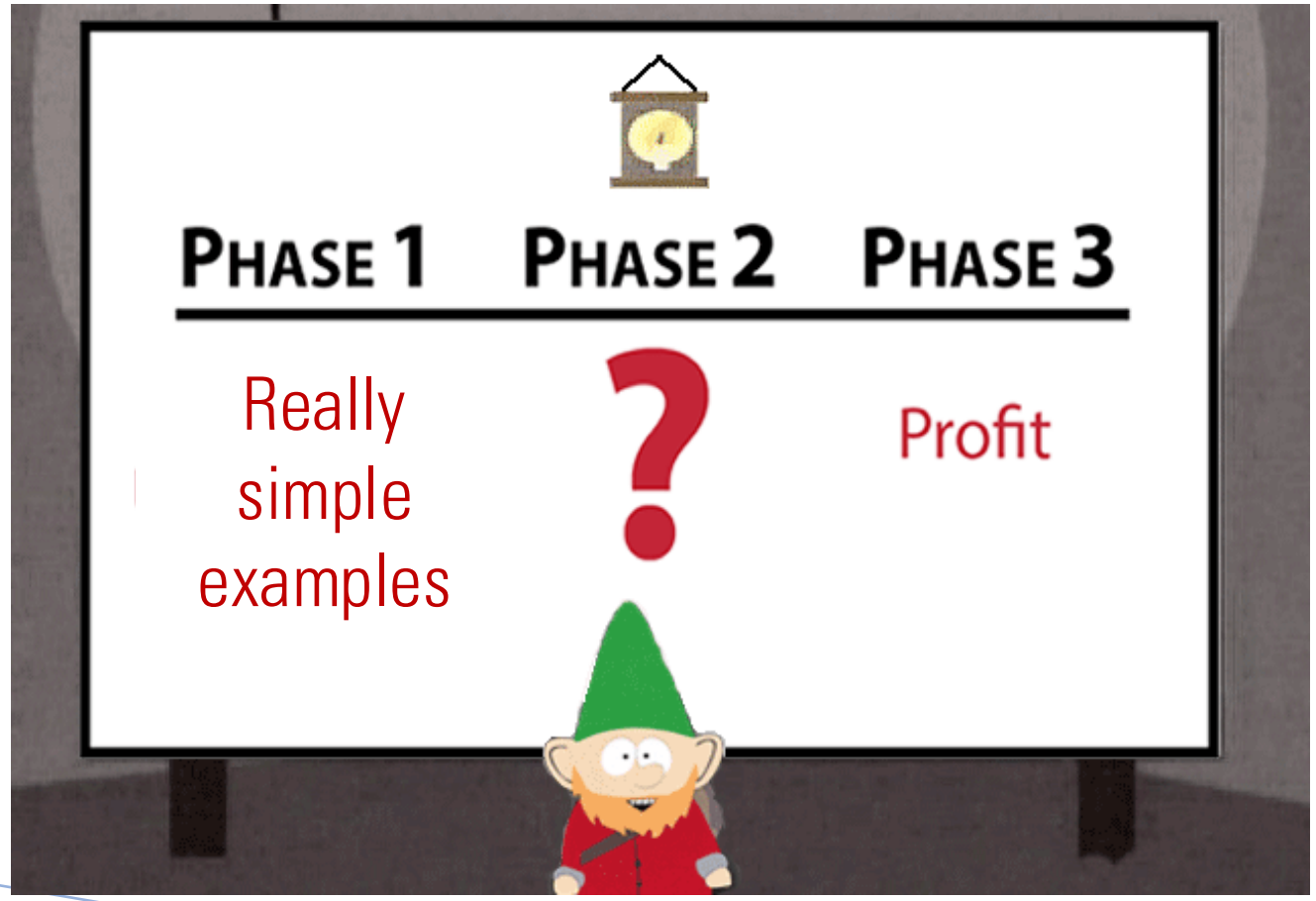
Getting started

- Use another script

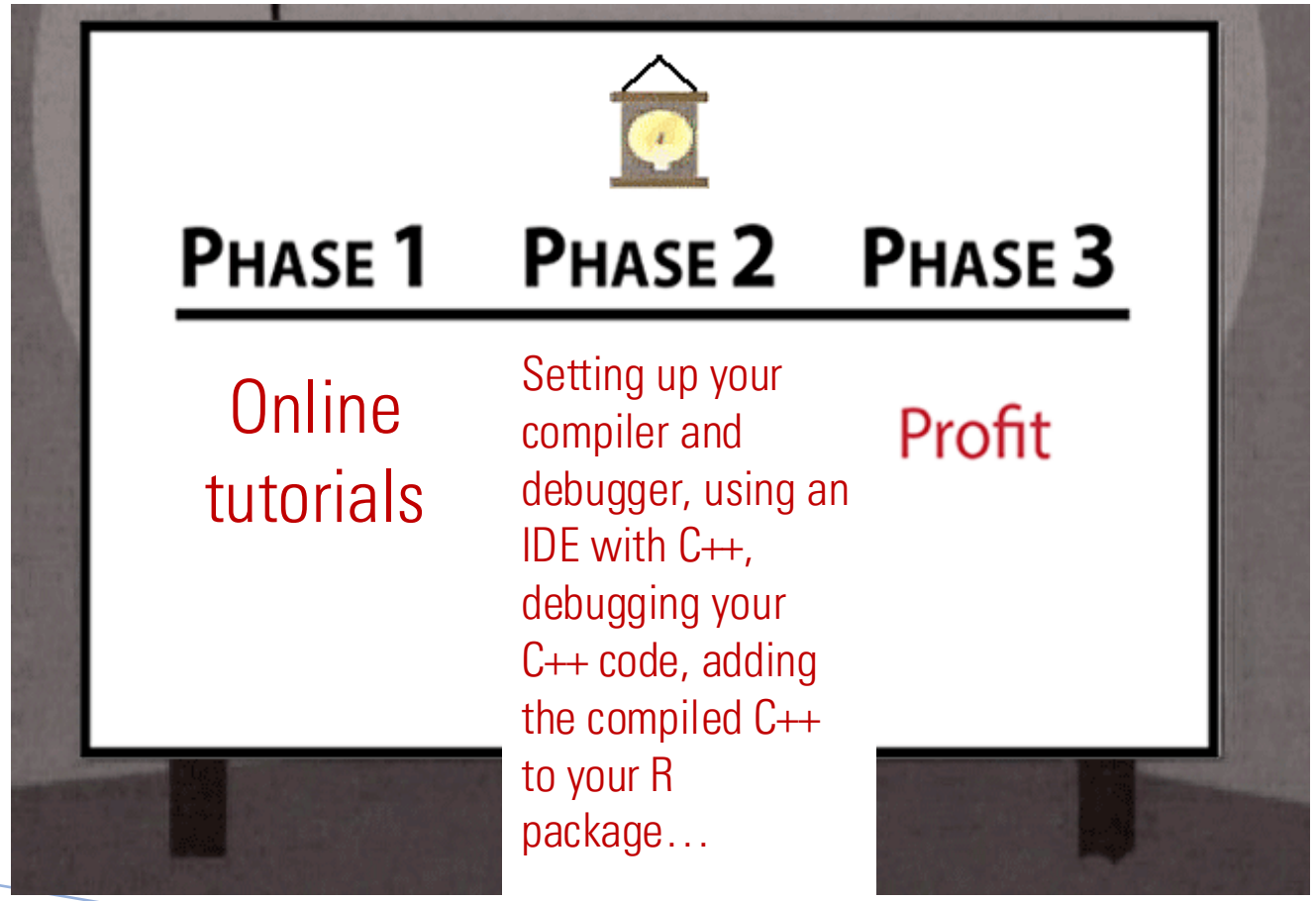
```
#include <Rcpp.h>
using namespace Rcpp;

// [[Rcpp::export]]
double sumC(NumericVector x) {
  int n = x.size();
  double total = 0;
  for(int i = 0; i < n; ++i) {
    total += x[i];
  }
  return total;
}
```

Bridging the gap



Bridging the gap



You **can** *do this in an afternoon*

Getting more complicated

- Use an IDE with C++ support

- I recommend using Visual Studio Code
  - You can run and debug both R and C++

Getting more complicated

- You need…
  - C++ compiler
  - C++ debugger
  - Rcpp package
  - Your R code as package
  - Patience

Debug example

Recommendation

- Do as little as possible in C++
- Focus on the most repetitive part of your code
- Make sure it's actually the slow bit!
  - Use the `microbenchmark` and `profvis` package in R to find the slow parts

Thank you for listening

Zachary Waller

zwaller01@qub.ac.uk

QUEEN'S UNIVERSITY BELFAST

Resources

- Set-up with VSCode
  - link

- Actually writing code:

- Part 1
- Part 2
- Part 3
- Part 4
- Part 5
- Part 6

Helpful books

- Efficient R – Book on optimizing both your approach to coding and your code itself
- Advanced R – Book covering the ins and outs of R development with a good chapter on optimization and Rcpp
- R Inferno – Book covering general "cardinal sins" in R programming. Slightly old school, but a lot of great nuggets of information

Annoying things...

Getting VSCode to recognize all my libraries for C++ code is a little temperamental

```
#include <RcppEigen.h>
#include <iostream>
#include "cov.h"
#include "monotone.h"
using namespace Rcpp;
using namespace Eigen;
```

Annoying things…

Viewing objects from C++ libraries when debugging is a little annoying.

I often have to resort to printing individual elements to the debug console.

There must be a better way!

But I don't know it…