Benchmarking and Performance Evaluations

Todd Mytkowicz
Microsoft Research

Let’s pole for an upcoming election

I ask 3 of my co-workers who they are voting for.

Let's pole for an upcoming election

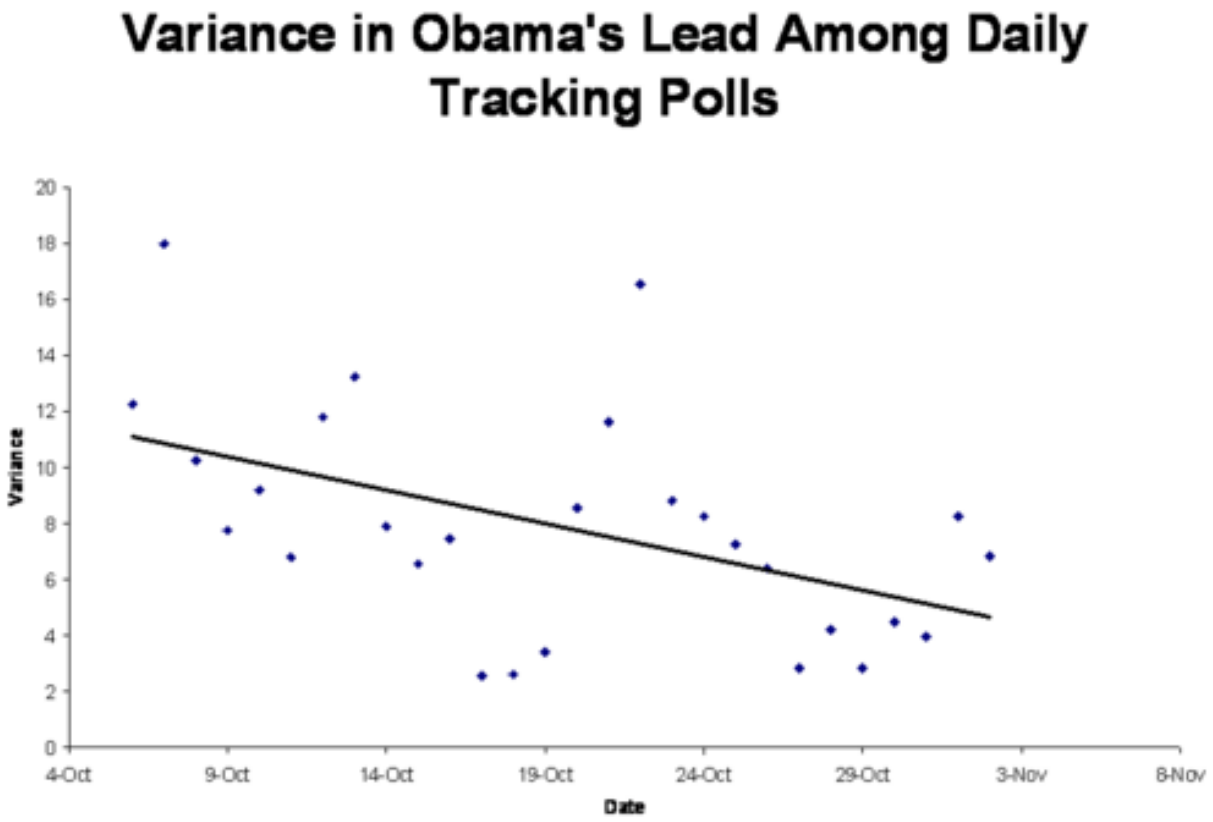
I ask 3 of my co-workers who they are voting for.

- **My approach does not deal with**
  - **Variability**
  - **Bias**

Issues with my approach

Variability

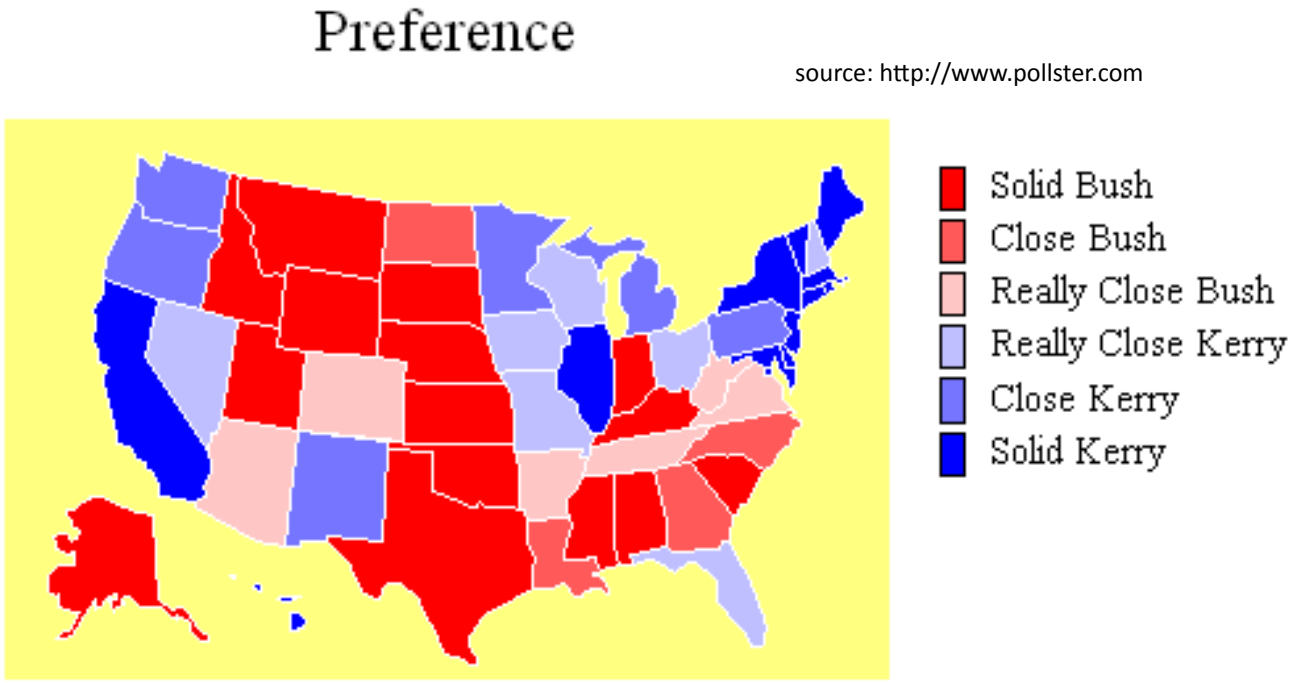
source: http://www.pollster.com



My approach is not reproducible

Issues with my approach(II)

Bias



My approach is not generalizable

Take Home Message

- Variability and Bias are two different things
  - Difference between reproducible and generalizable!

Take Home Message

- Variability and Bias are two different things
  - Difference between reproducible and generalizable!

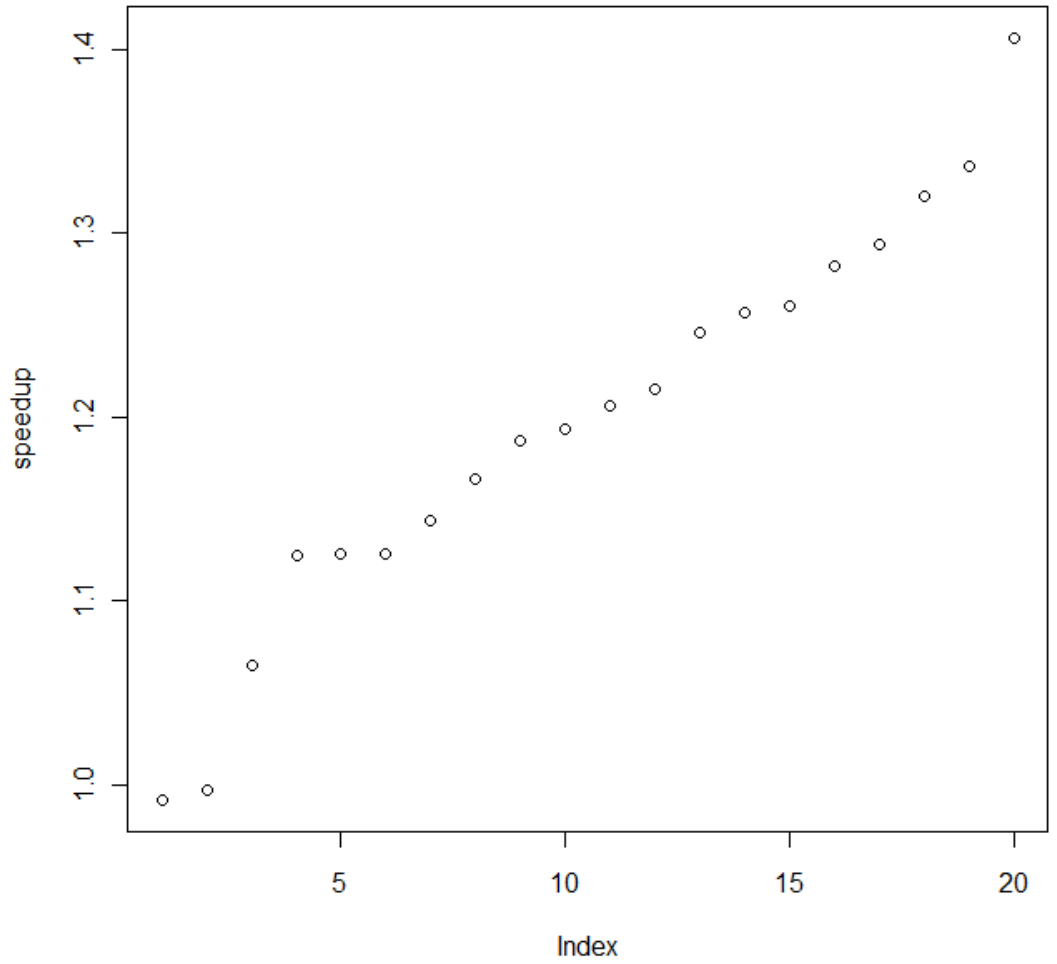
Do we have to worry about Variability and Bias when we benchmark?

Let’s evaluate the speedup of my whizbang idea

What do we do
about Variability?

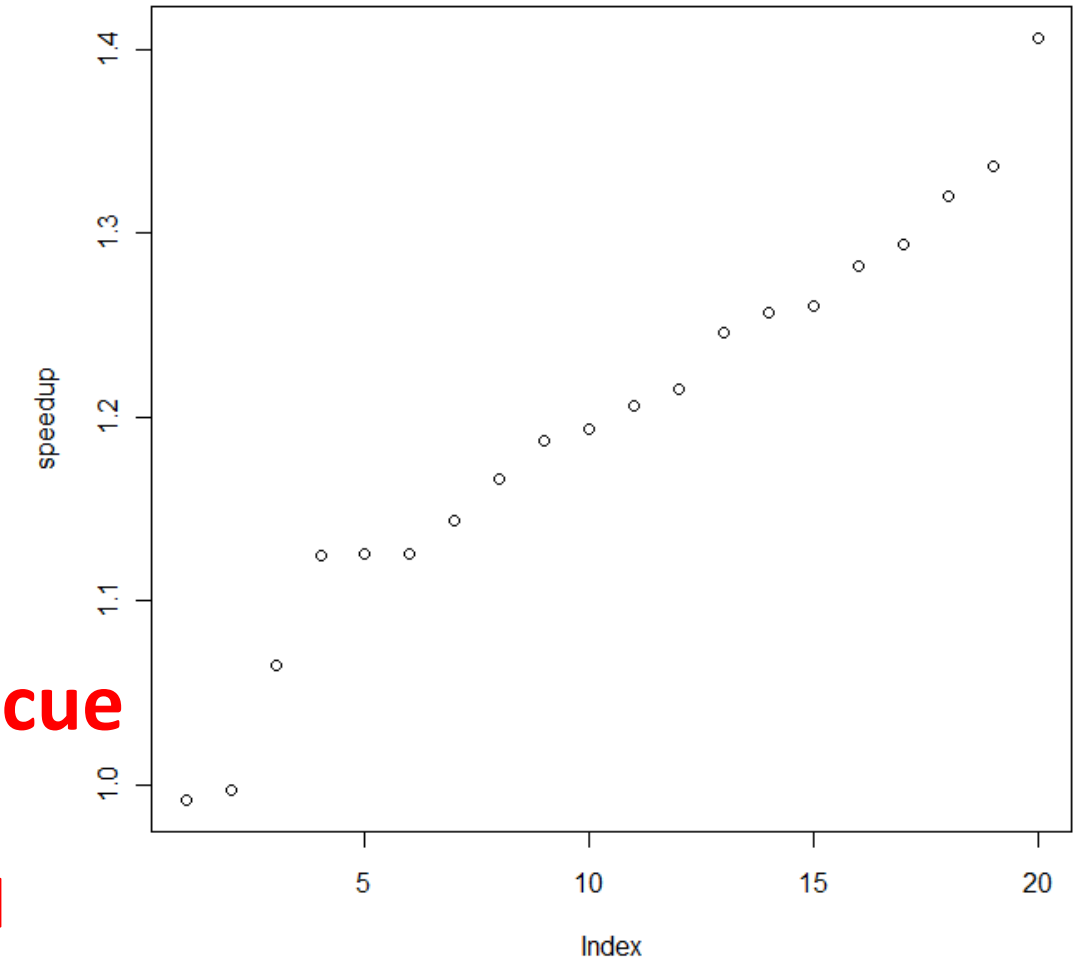
Let's evaluate the speedup of my whizbang idea

What do we do about Variability?



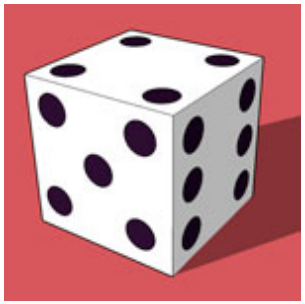
Let's evaluate the speedup of my whizbang idea

What do we do about Variability?



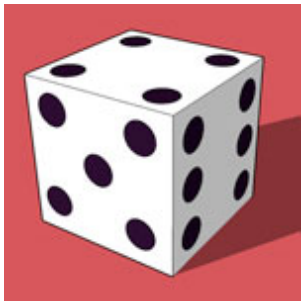
- **Statistics to the rescue**
  - **mean**
  - **confidence interval**

Intuition for T-Test



- 1-6 is uniformly likely (p = 1/6)
- Throw die 10 times: calculate mean

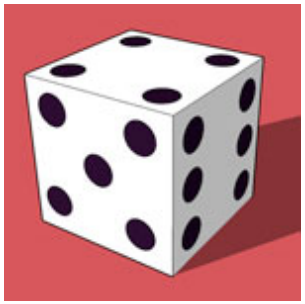
Intuition for T-Test



- 1-6 is uniformly likely (p = 1/6)
- Throw die 10 times: calculate mean

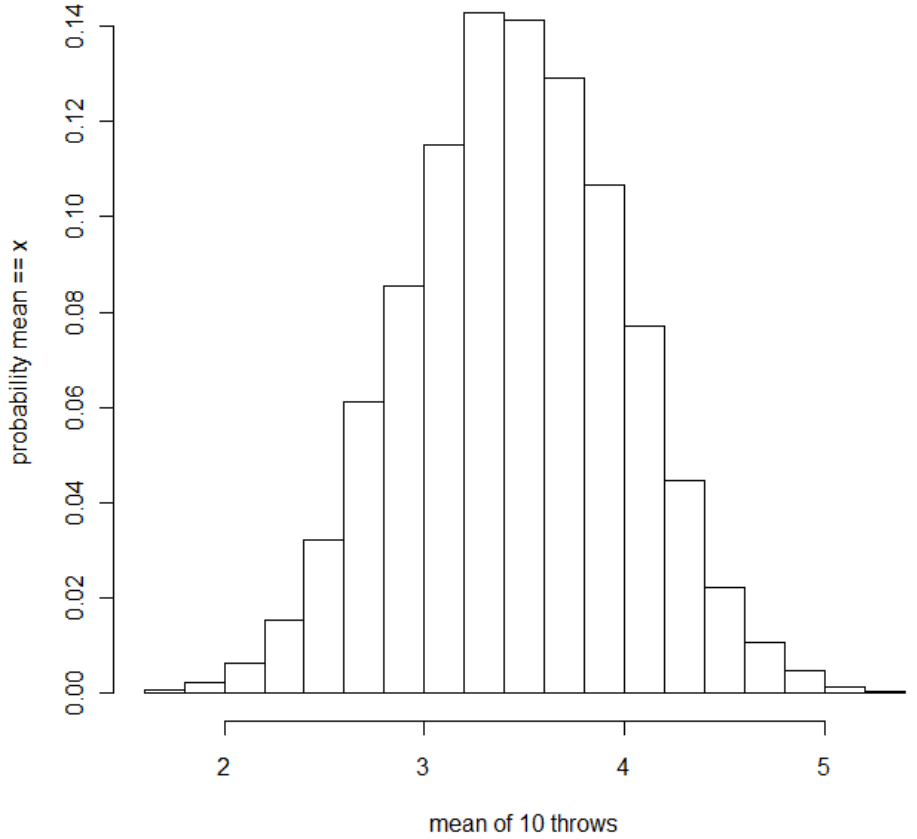
Trial	Mean of 10 throws
1	4.0
2	4.3
3	4.9
4	3.8
5	4.3
6	2.9
…	…

Intuition for T-Test

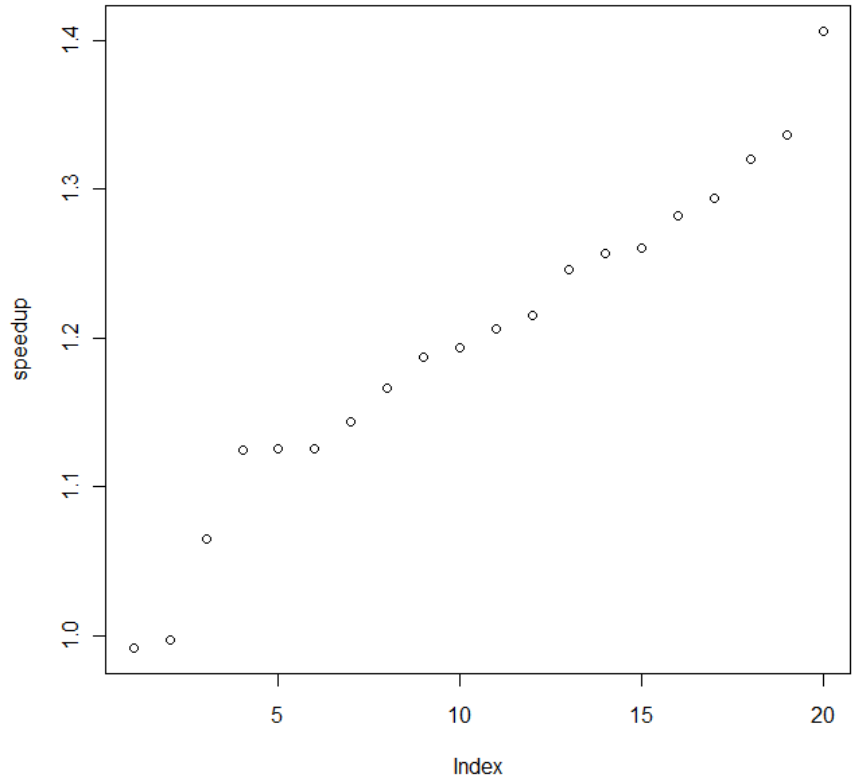


- 1-6 is uniformly likely (p = 1/6)
- Throw die 10 times: calculate mean

Trial	Mean of 10 throws
1	4.0
2	4.3
3	4.9
4	3.8
5	4.3
6	2.9
…	…

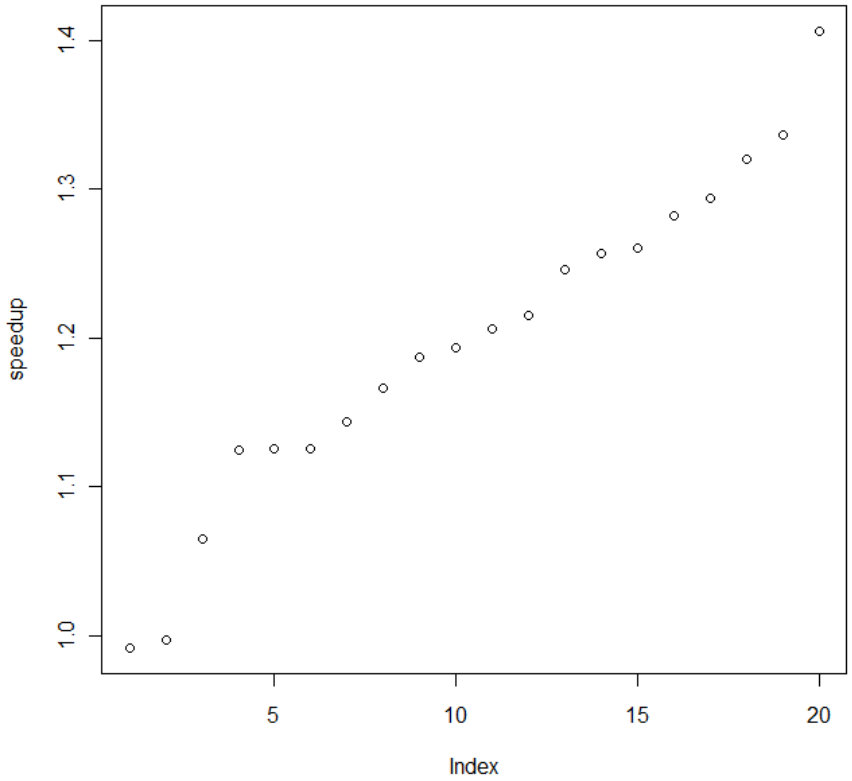


Back to our Benchmark: Managing Variability



Back to our Benchmark: Managing Variability

```
> x=scan('file')
Read 20 items
> t.test(x)

        One Sample t-test

data:  x
t = 49.277, df = 19, p-value < 2.2e-16
95 percent confidence interval:
 1.146525 1.248241
sample estimates:
mean of x
 1.197383
```



So we can handle Variability. What about Bias?

Evaluating compiler optimizations

System = gcc -O2 perlbench
System + Innovation = gcc -O3 perlbench

Evaluating compiler optimizations

System = gcc -O2 perlbench
System + Innovation = gcc -O3 perlbench

Madan:
speedup = 1.18 ± 0.0002

Conclusion: O3 is good

Evaluating compiler optimizations

System = gcc -O2 perlbench
System + Innovation = gcc -O3 perlbench

Madan:
speedup = 1.18 ± 0.0002

Conclusion: O3 is good

Todd:
speedup = 0.84 ± 0.0002

Conclusion: O3 is bad

Evaluating compiler optimizations

System = gcc -O2 perlbench
System + Innovation = gcc -O3 perlbench

Madan:
speedup = 1.18 ± 0.0002

Conclusion: O3 is good

Todd:
speedup = 0.84 ± 0.0002

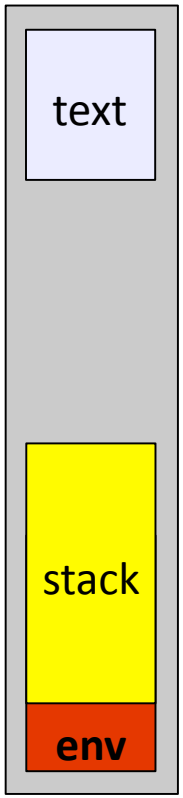
Conclusion: O3 is bad

Why does this happen?

Differences in our experimental setup

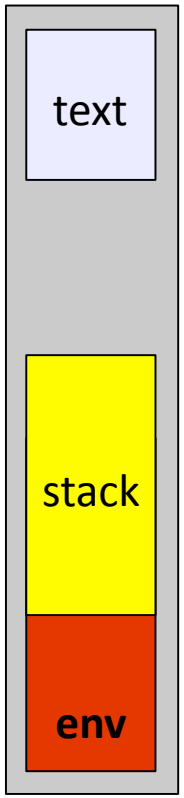
Madan:

HOME=/home/madan

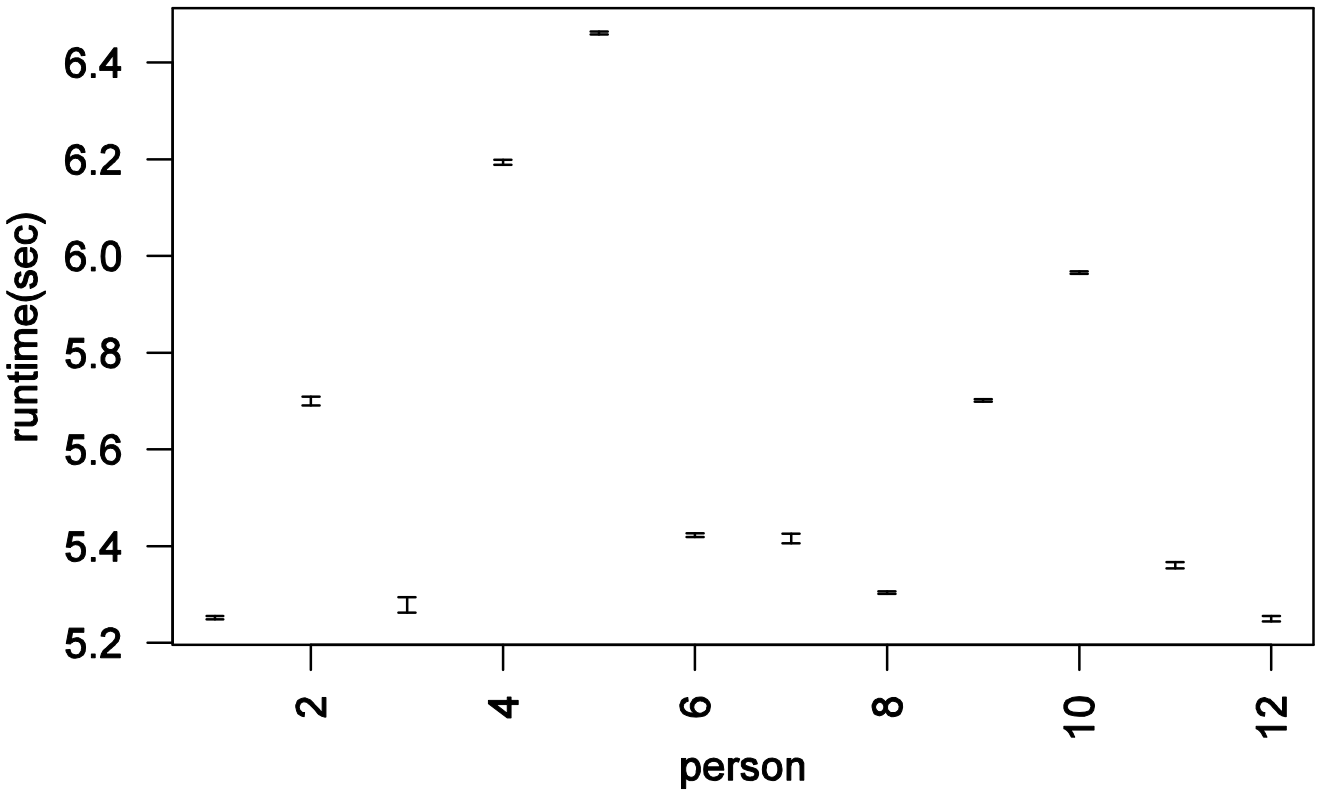


Todd:

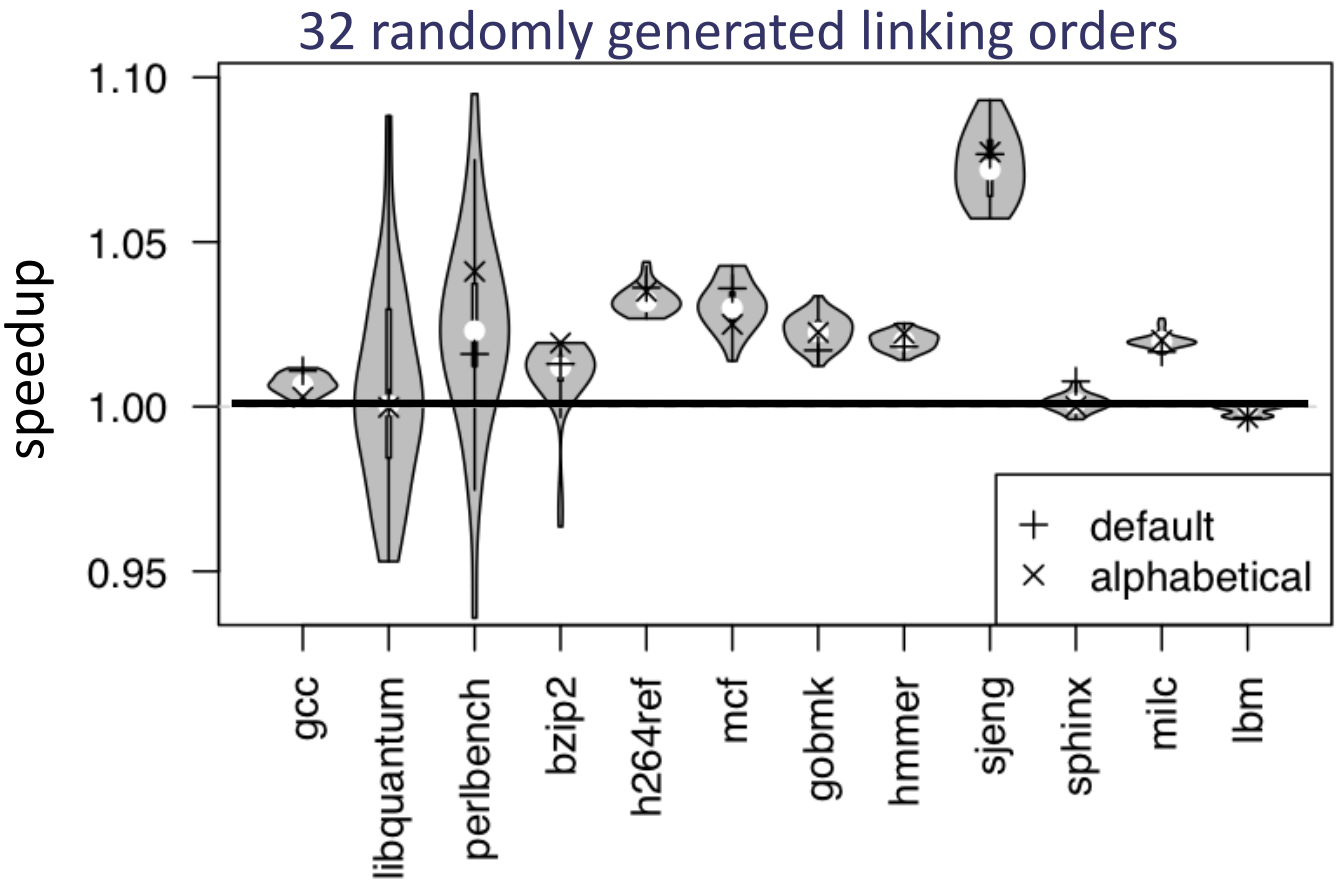
HOME=/home/toddmytkowicz



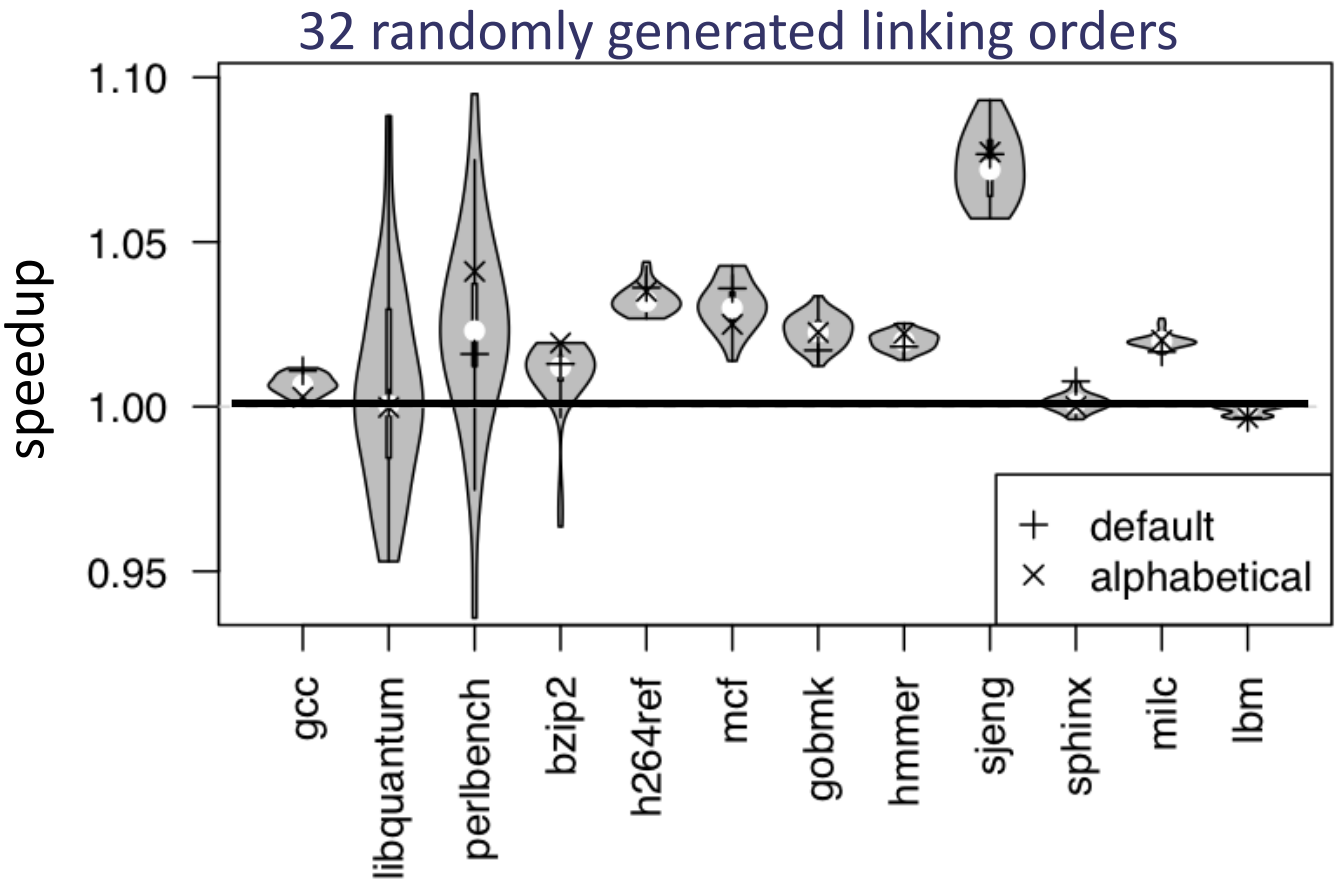
Runtime of SPEC CPU 2006 perlbench depends on who runs it!



Bias from linking order



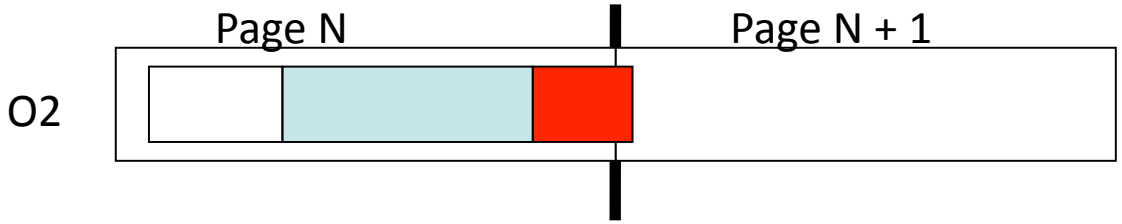
Bias from linking order



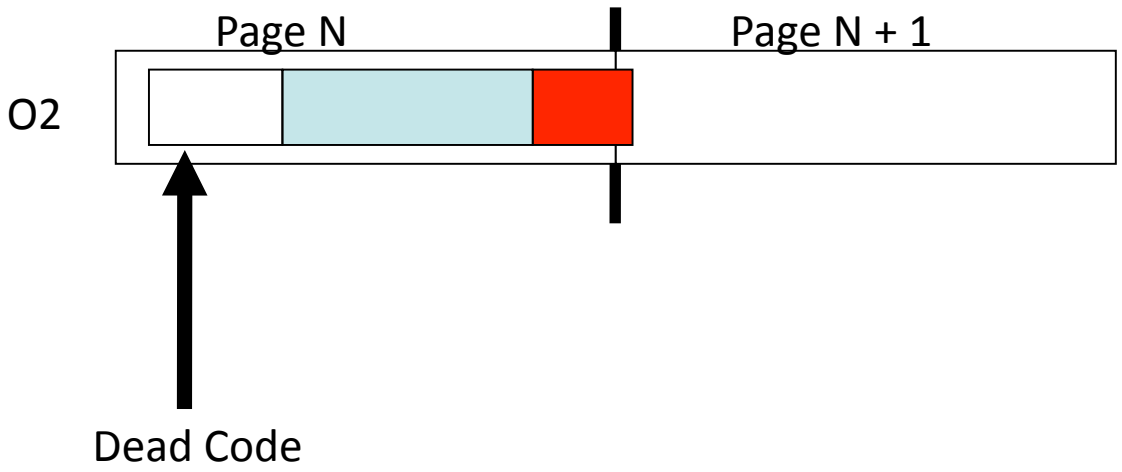
Order of .o files can lead to contradictory conclusions

Where exactly does Bias come from?

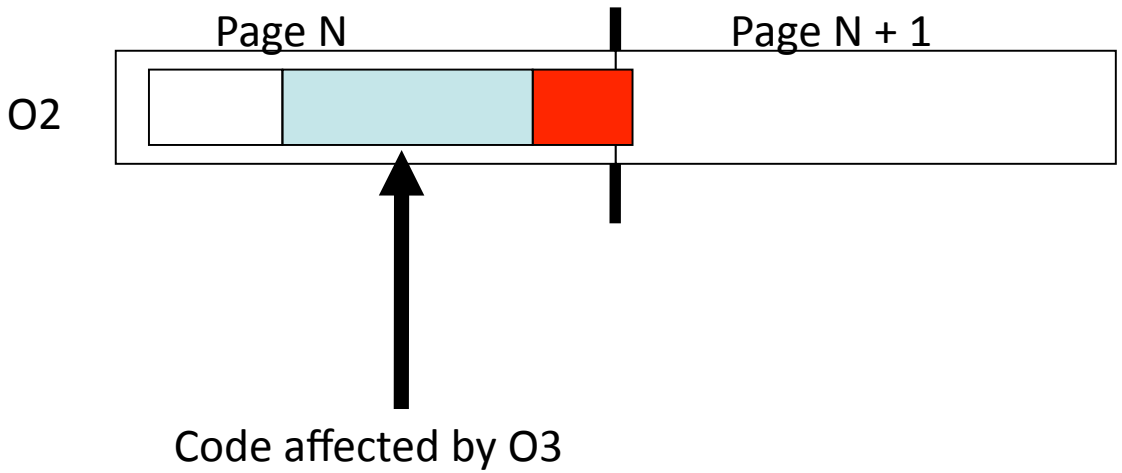
Interactions with hardware buffers



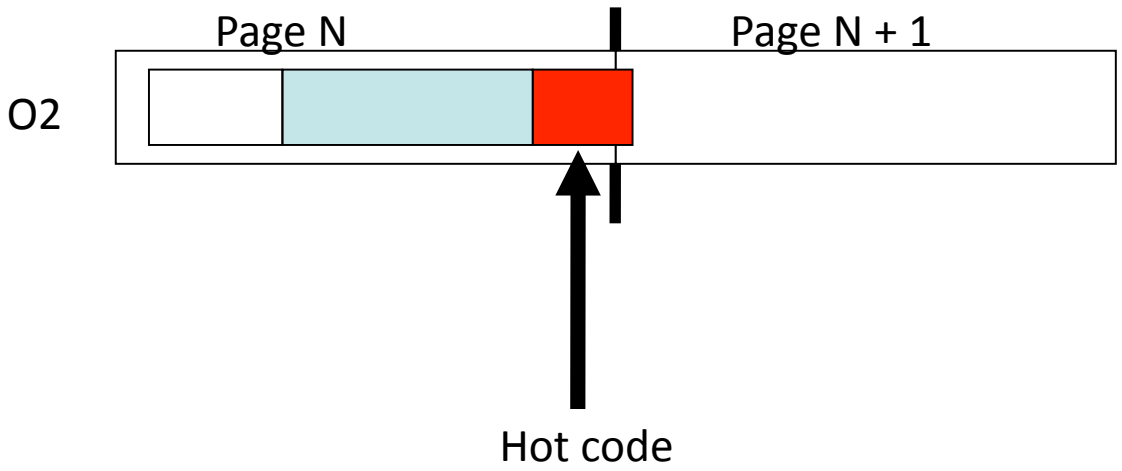
Interactions with hardware buffers



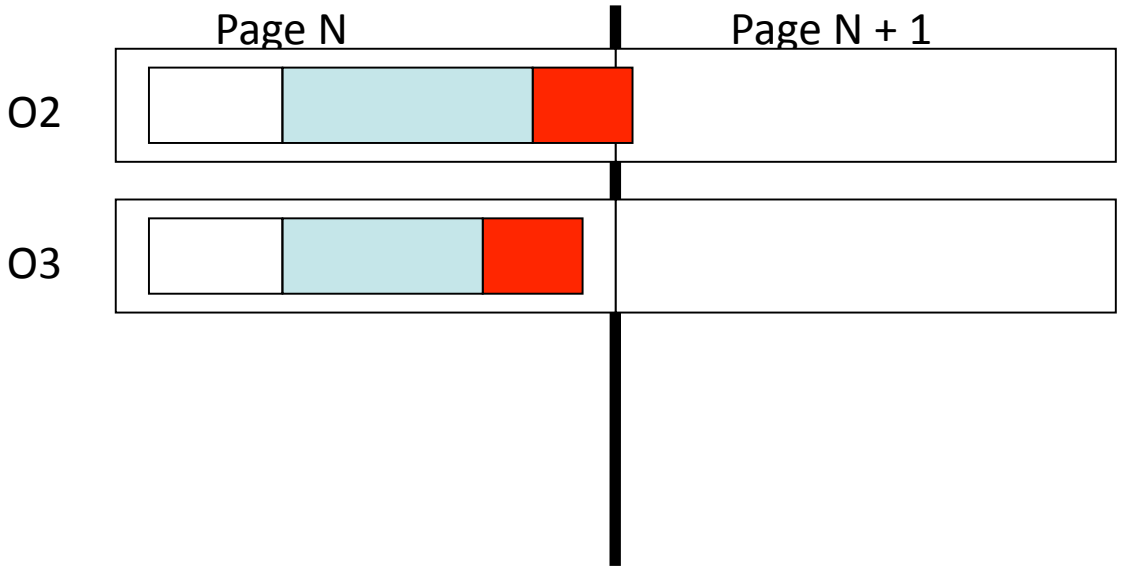
Interactions with hardware buffers



Interactions with hardware buffers

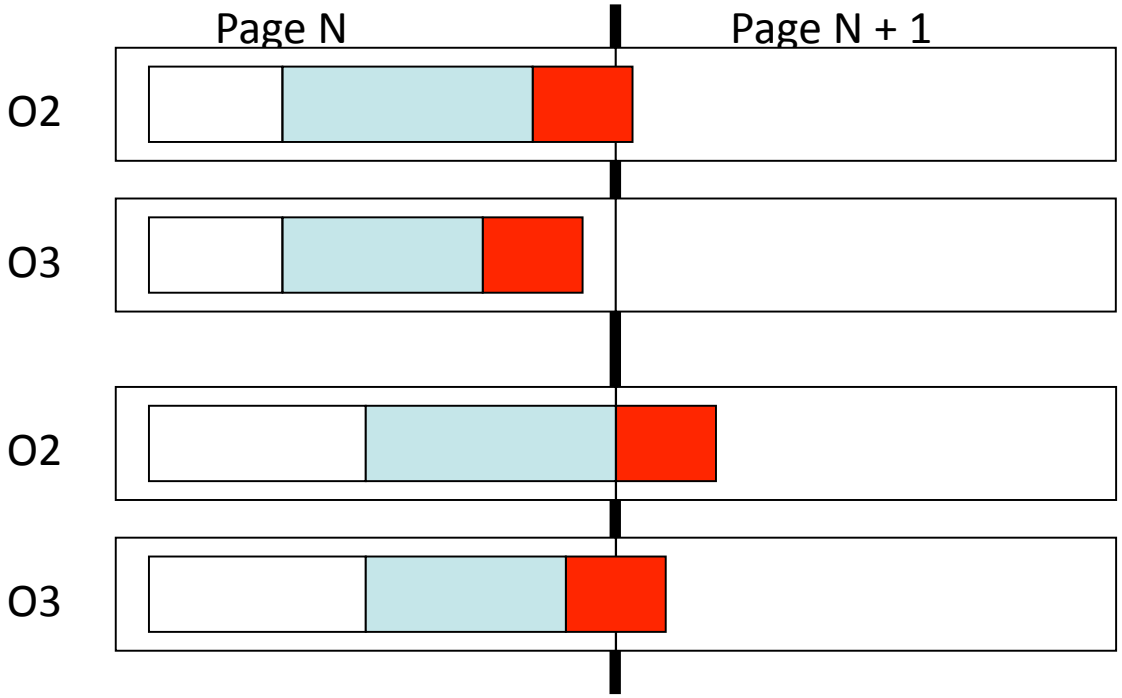


Interactions with hardware buffers



O3 better than O2

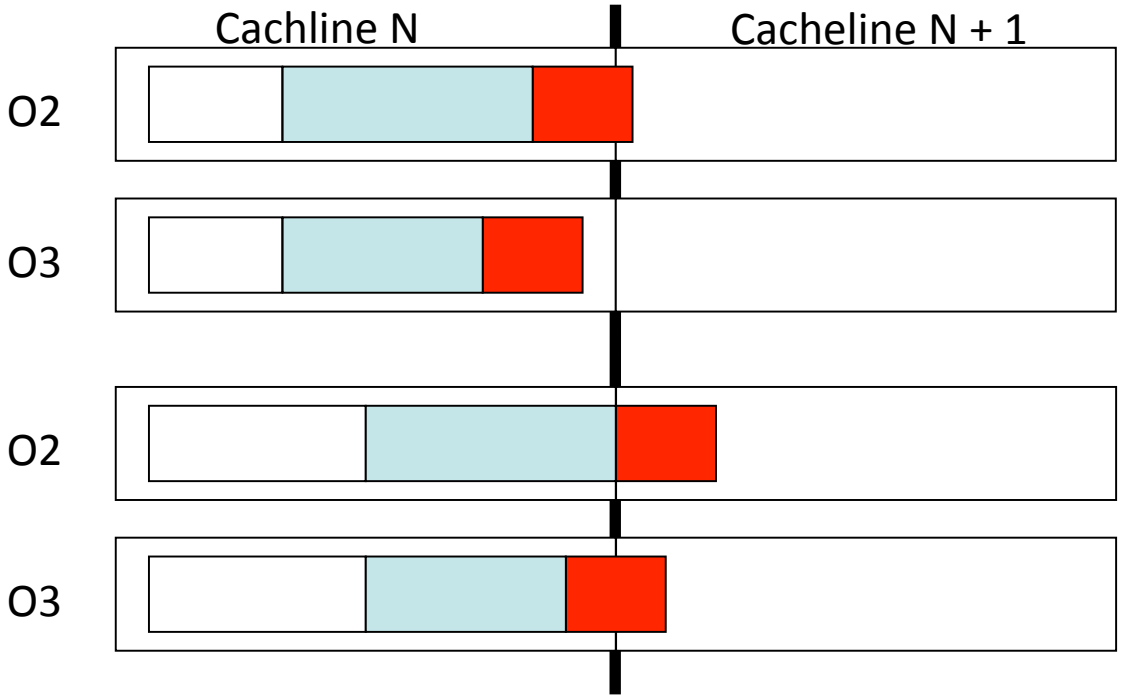
Interactions with hardware buffers



O3 better than O2

O2 better than O3

Interactions with hardware buffers



O3 better than O2

O2 better than O3

Other Sources of Bias

- JIT
- Garbage Collection
- CPU Affinity
- Domain specific (e.g. size of input data)

- How do we manage these?

Other Sources of Bias

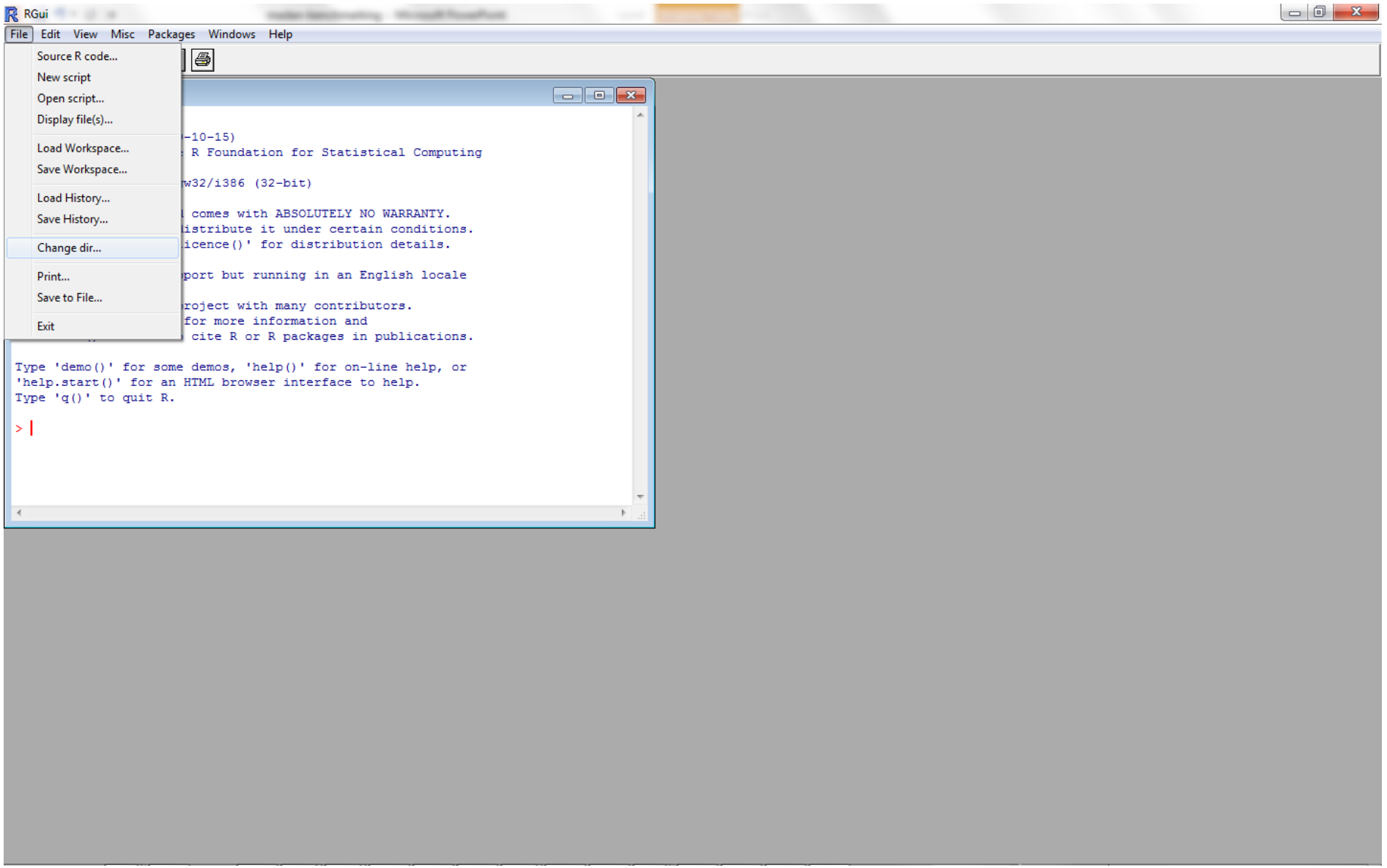
How do we manage these?

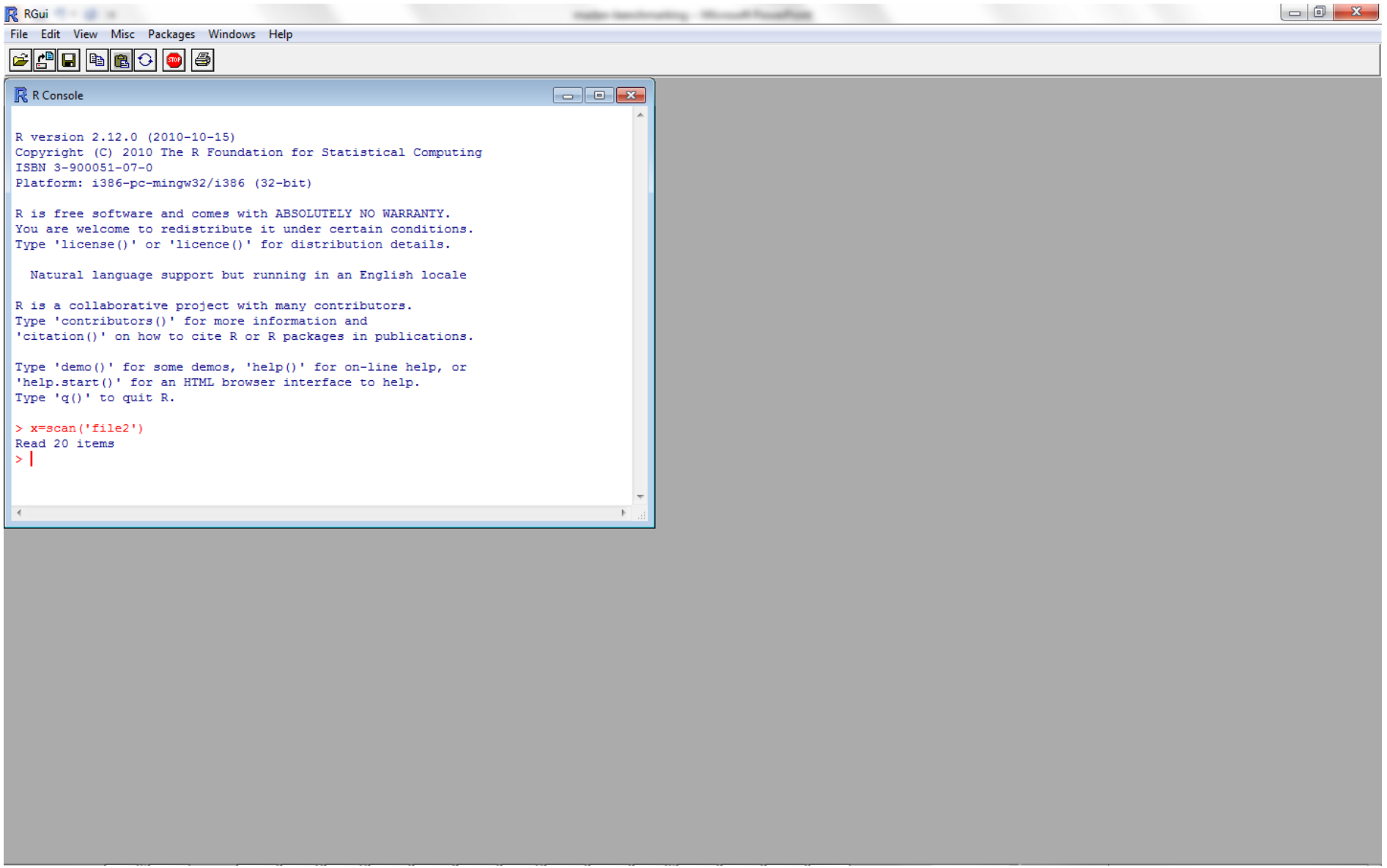
- JIT:
  - ngen to remove impact of JIT
  - “warmup” phase to JIT code before measurement
- Garbage Collection
  - Try different heap sizes (JVM)
  - “warmup” phase to build data structures
  - Ensure program is not “leaking” memory
- CPU Affinity
  - Try to bind threads to CPUs (SetProcessAffinityMask)
- Domain Specific:
  - Up to you!

R for the T-Test

- Where to download
  - http://cran.r-project.org

- Simple intro to get data into R
- Simple intro to do t.test


RGui
File
Edit
View
Misc
Packages
Windows
Help
Source R code...
New script
Open script...
Display file(s)...
Load Workspace...
Save Workspace...
Load History...
Save History...
Change dir...
Print...
Save to File...
Exit
-10-15)
R Foundation for Statistical Computing
w32/i386 (32-bit)
comes with ABSOLUTELY NO WARRANTY.
istribute it under certain conditions.
icence()' for distribution details.
port but running in an English locale
roject with many contributors.
for more information and
cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
>


RGui
File Edit View Misc Packages Windows Help
R Console
R version 2.12.0 (2010-10-15)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
> x=scan('file2')
Read 20 items
>

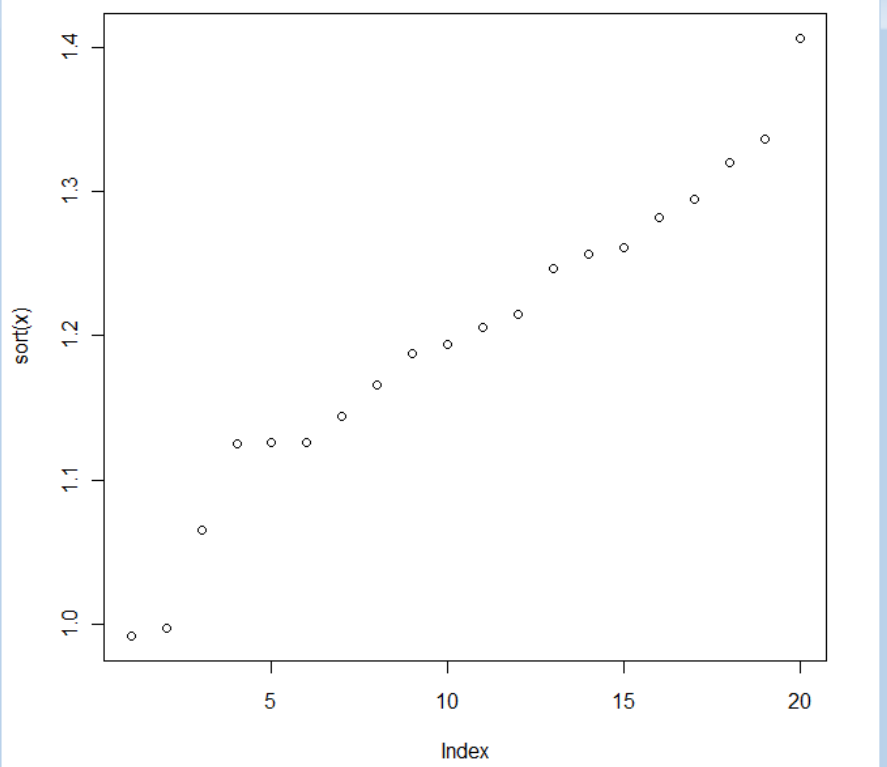
RGui

File History Resize Windows

R Console

```
R version 2.12.0 (2010-10-15)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: i386-pc-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> x=scan('file2')
Read 20 items
> plot(sort(x))
>
```

R Graphics: Device 2 (ACTIVE)



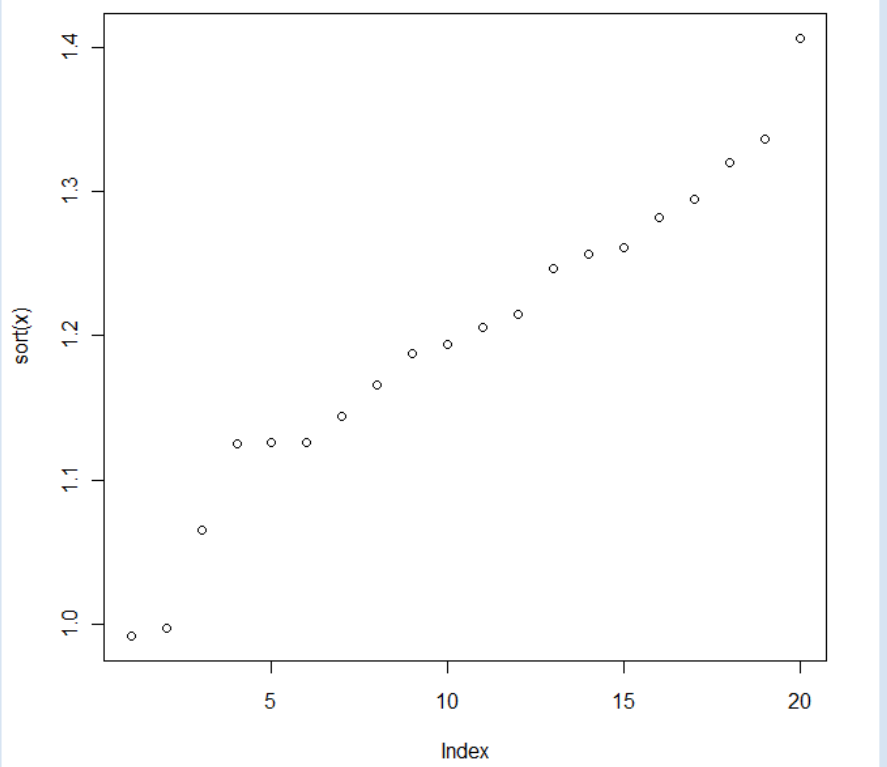
RGui

File Edit View Misc Packages Windows Help

R Console

```
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> x=scan('file2')
Read 20 items
> plot(sort(x))
> t.test(x)

        One Sample t-test

data:  x
t = 49.277, df = 19, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 1.146525 1.248241
sample estimates:
mean of x
 1.197383

>
```

R Graphics: Device 2 (ACTIVE)



Some Conclusions

- Performance Evaluations are hard!
  - Variability and Bias are not easy to deal with

- Other experimental sciences go to great effort to work around variability and bias
  - We should too!