5 YEARS OF TEACHING C++: A RETROSPECTIVE

Martin Hořeňovský

PEX

5 YEARS OF TEACHING C++: A RETROSPECTIVE

Martin Hořeňovský

DISCLAIMERS

This talk is about teaching C++ to university students.

But very little of this talk will be C++ specific.

But very little of this talk will be C++ specific.

I will talk about the course's structure and how that worked for us, not about the syllabus itself.

There are no correct answers here …

There are no correct answers here …

… only hindsight and experiences.

CONTENTS

CONTENTS

- Setting the stage

CONTENTS

- Setting the stage
- Organization of the course

CONTENTS

- Setting the stage
- Organization of the course
- Retrospective on our design decisions

CONTENTS

- Setting the stage
- Organization of the course
- Retrospective on our design decisions
- More general advices

SETTING THE STAGE

As undergraduate, I took the C++ course at my faculty.

As undergraduate, I took the C++ course at my faculty.

It was **BAD**.

As undergraduate, I took the C++ course at my faculty.

It was **BAD**.

It taught C with some C++98.

With a friend, we decided to fix it.

With a friend, we decided to fix it.

We wanted to teach modern C++.

With a friend, we decided to fix it.

We wanted to teach modern C++.

We wanted to teach good programming practices.

We actually managed to take over the course.

We actually managed to take over the course.

And rewrite all the materials.

We actually managed to take over the course.

And rewrite all the materials.

And then teach it for a couple of years.

OUR PLANS

High level ideas

High level ideas

- Teach modern C++

High level ideas

- Teach modern C++
- Showcase good practices

High level ideas

- Teach modern C++
- Showcase good practices
  - Unit tests

High level ideas

- Teach modern C++
- Showcase good practices
  - Unit tests
  - Automated code checking

High level ideas

- Teach modern C++
- Showcase good practices
  - Unit tests
  - Automated code checking
  - Valgrind & Sanitizers

High level ideas

- Teach modern C++
- Showcase good practices
  - Unit tests
  - Automated code checking
  - Valgrind & Sanitizers
- Continuously improve the course

Practicalities

Practicalities

- All attendance is optional

Practicalities

- All attendance is optional
- Have regular homeworks

Practicalities

- All attendance is optional
- Have regular homeworks
- Students can run all tests locally

Practicalities

- All attendance is optional
- Have regular homeworks
- Students can run all tests locally
- Collect data about homeworks

Practicalities

- All attendance is optional
- Have regular homeworks
- Students can run all tests locally
- Collect data about homeworks
- Errors must be resolved in students' favour

REQUIREMENTS FROM UNIVERSITY

REQUIREMENTS FROM UNIVERSITY

- Teach C++ up to basics of templates

REQUIREMENTS FROM UNIVERSITY

- Teach C++ up to basics of templates
- 6 ECTS credits (150-180 hours of work)

REQUIREMENTS FROM UNIVERSITY

- Teach C++ up to basics of templates
- 6 ECTS credits (150-180 hours of work)
- 13 weeks, 4 contact hours per week (52 h)

REQUIREMENTS FROM UNIVERSITY

- Teach C++ up to basics of templates
- 6 ECTS credits (150-180 hours of work)
- 13 weeks, 4 contact hours per week (52 h)
- Students have prior programming experience

REQUIREMENTS FROM UNIVERSITY

- Teach C++ up to basics of templates
- 6 ECTS credits (150-180 hours of work)
- 13 weeks, 4 contact hours per week (52 h)
- Students have prior programming experience
- Elective*, ~30 students per semester

REQUIREMENTS FROM UNIVERSITY

- Teach C++ up to basics of templates
- 6 ECTS credits (150-180 hours of work)
- 13 weeks, 4 contact hours per week (52 h)
- Students have prior programming experience
- Elective*, ~30 students per semester
- Course ends with a graded exam

COURSE DESIGN & STRUCTURE

- Attendance **is not** mandatory, coursework **is**

- Attendance **is not** mandatory, coursework **is**
- Grading is on 0-100 points scale

- Attendance **is not** mandatory, coursework **is**
- Grading is on 0-100 points scale
  - There is more than 100 points in the course

- Attendance **is not** mandatory, coursework **is**
- Grading is on 0-100 points scale
  - There is more than 100 points in the course
- BYO machine, OS, compiler

- Attendance **is not** mandatory, coursework **is**
- Grading is on 0-100 points scale
  - There is more than 100 points in the course
- BYO machine, OS, compiler
  - Support CMake/Make/VS sln for builds

- Attendance **is not** mandatory, coursework **is**
- Grading is on 0-100 points scale
  - There is more than 100 points in the course
- BYO machine, OS, compiler
  - Support CMake/Make/VS sln for builds
  - But automated checker runs Linux

- Weekly small homework

- Weekly small homework
  - Target 1-4 hours of work

- Weekly small homework
  - Target 1-4 hours of work
  - Automatic grading

- Weekly small homework
  - Target 1-4 hours of work
  - Automatic grading
  - Closely tied to that week's material

- Weekly small homework
  - Target 1-4 hours of work
  - Automatic grading
  - Closely tied to that week's material
  - Deadline until next week

- Weekly small homework
  - Target 1-4 hours of work
  - Automatic grading
  - Closely tied to that week's material
  - Deadline until next week
  - Weeks would often follow up on the previous one

- One larger, threading-focused homework

- One larger, threading-focused homework
  - Manual grading

- One larger, threading-focused homework
  - Manual grading
  - 6-8 hours of work

- One larger, threading-focused homework
  - Manual grading
  - 6-8 hours of work

- One semestral work

- One larger, threading-focused homework
  - Manual grading
  - 6-8 hours of work

- One semestral work
  - Manual grading

- One larger, threading-focused homework
  - Manual grading
  - 6-8 hours of work

- One semestral work
  - Manual grading
  - Target 30 hours of work

- One larger, threading-focused homework
  - Manual grading
  - 6-8 hours of work

- One semestral work
  - Manual grading
  - Target 30 hours of work
  - Opportunity to write code from scratch

- Three exams

- Three exams
  - Two smaller ones during semester

- Three exams
  - Two smaller ones during semester
  - Larger one as the actual exam

- Three exams
  - Two smaller ones during semester
  - Larger one as the actual exam
  - Pen & paper

- Three exams
  - Two smaller ones during semester
  - Larger one as the actual exam
  - Pen & paper
  - *Minimal* code writing

- Three exams
  - Two smaller ones during semester
  - Larger one as the actual exam
  - Pen & paper
  - *Minimal* code writing
  - Focused on reading code & lifetime puzzlers

So how did it go?

It went almost surprisingly well.

It went almost surprisingly well.

We had to make some adjustments on the fly.

It went almost surprisingly well.

We had to make some adjustments on the fly.

But …

We hit the time targets.

We hit the time targets.

Grades matched expected curve.

We hit the time targets.

Grades matched expected curve.

End semester feedback was good.

REACCREDITATION

2 years in, reaccreditation happened.

2 years in, reaccreditation happened.

This meant new requirements for the course.

- 4 ECTS credits (100-120 hours of work)

- 4 ECTS credits (100-120 hours of work)
  - This means half of non-contact hours gone.

- 4 ECTS credits (100-120 hours of work)
  - This means half of non-contact hours gone.
  - Graded "zápočet" instead of exam

- 4 ECTS credits (100-120 hours of work)
  - This means half of non-contact hours gone.
  - Graded "zápočet" instead of exam
- Mandatory course in 3rd semester

- 4 ECTS credits (100-120 hours of work)
  - This means half of non-contact hours gone.
  - Graded "zápočet" instead of exam
- Mandatory course in 3rd semester
  - Students have less prior programming experience

- 4 ECTS credits (100-120 hours of work)
  - This means half of non-contact hours gone.
  - Graded "zápočet" instead of exam
- Mandatory course in 3rd semester
  - Students have less prior programming experience
  - ~120 students per semester

We had to cut out *a lot* of stuff to fit within semester.

- Threading homework
- Final exam
- In-semester exam

- ~~Threading homework~~
- Final exam
- In-semester exam

- ~~Threading homework~~
- ~~Final exam~~
- In-semester exam

- ~~Threading homework~~
- ~~Final exam~~
- ~~In-semester exam~~

- ~~Threading homework~~
- ~~Final exam~~
- ~~In-semester exam~~
- Reduced weekly homeworks

- ~~Threading homework~~
- ~~Final exam~~
- ~~In-semester exam~~
- Reduced weekly homeworks
  - Removed some weeks completely

- ~~Threading homework~~
- ~~Final exam~~
- ~~In-semester exam~~
- Reduced weekly homeworks
  - Removed some weeks completely
  - Added hints to make things simpler

Overall the new course didn't work well.

Overall the new course didn't work well.

We blew past time targets.

Overall the new course didn't work well.

We blew past time targets.

Feedback at the end of the semester got worse.

We iterated a few times over different semesters,

We iterated a few times over different semesters,

but we never were happy with the results.

Let's talk about our decisions and what I think about them in hindsight.

PROS? CONS? VERDICT?

PROS? CONS? VERDICT?

- +
  - Things that were good about it

PROS? CONS? VERDICT?

- +
  - Things that were good about it
- -
  - Things that were bad about it

PROS? CONS? VERDICT?

- +
  - Things that were good about it
- -
  - Things that were bad about it
- ?
  - What I'd do now, or what we did about it

SUPPORT DIFFERENT PLATFORMS

SUPPORT DIFFERENT PLATFORMS

- +
  - Students prefer working on their machines

SUPPORT DIFFERENT PLATFORMS

- +
  - Students prefer working on their machines
- -
  - Students had to deal with platform-specific issues
  - Students tried using *ancient* compilers

SUPPORT DIFFERENT PLATFORMS

- +
  - Students prefer working on their machines
- -
  - Students had to deal with platform-specific issues
  - Students tried using *ancient* compilers
- ?
  - Package the authoritative VM for students use

ALLOW MAKE/CMAKE/VS SLN

ALLOW MAKE/CMAKE/VS SLN

- +
  - Students could use what they were familiar with

ALLOW MAKE/CMAKE/VS SLN

- +
  - Students could use what they were familiar with
- -
  - Novices were lost as to what they should use
  - Supporting all meant we supported none

ALLOW MAKE/CMAKE/VS SLN

- +
  - Students could use what they were familiar with
- -
  - Novices were lost as to what they should use
  - Supporting all meant we supported none
- ?
  - We ended up making the course CMake only

NON-MANDATORY ATTENDANCE

NON-MANDATORY ATTENDANCE

- +
  - Students proficient with C++ can skip classes

NON-MANDATORY ATTENDANCE

- +
  - Students proficient with C++ can skip classes
- -
  - Students are bad at estimating whether attendance is helpful

NON-MANDATORY ATTENDANCE

- +
  - Students proficient with C++ can skip classes
- -
  - Students are bad at estimating whether attendance is helpful
- ?
  - No change
  - Use positive motivation to get students to attend

WEEKLY HOMEWORK

WEEKLY HOMEWORK

- +
  - Provide reason for regular practice
  - Tightly connected to current material

WEEKLY HOMEWORK

- +
  - Provide reason for regular practice
  - Tightly connected to current material
- -
  - Cause time pressure due to short deadlines
  - Followup tasks mean that overruns are correlated

WEEKLY HOMEWORK

- +
  - Provide reason for regular practice
  - Tightly connected to current material
- -
  - Cause time pressure due to short deadlines
  - Followup tasks mean that overruns are correlated
- ?
  - We combined them into 3 larger homeworks

LARGER HOMEWORKS

LARGER HOMEWORKS

- +
  - Students have more scheduling freedom
  - Avoids filler homeworks

LARGER HOMEWORKS

- +
  - Students have more scheduling freedom
  - Avoids filler homeworks
- -
  - Large chunks of work can be intimidating
  - Student syndrome

LARGER HOMEWORKS

- +
  - Students have more scheduling freedom
  - Avoids filler homeworks
- -
  - Large chunks of work can be intimidating
  - Student syndrome
- ?
  - Keep, direct students towards regular work

WEEKLY SMALL TASKS (<30 MINUTES)

WEEKLY SMALL TASKS (<30 MINUTES)

- +
  - Encourages regular work
  - Good place to put bonus points

WEEKLY SMALL TASKS (<30 MINUTES)

- +
  - Encourages regular work
  - Good place to put bonus points
- -
  - Bad implementation can ruin the goals
  - Hard to make up new ones every semester

WEEKLY SMALL TASKS (<30 MINUTES)

- +
  - Encourages regular work
  - Good place to put bonus points
- -
  - Bad implementation can ruin the goals
  - Hard to make up new ones every semester
- ?
  - Make sure they are optional
  - Make it clear that they should be short

PEN & PAPER EXAM

PEN & PAPER EXAM

- +
  - Checks different skills than rest of course work
  - The only check that students can read code
  - Defers some time investment into the exam weeks

PEN & PAPER EXAM

- +
  - Checks different skills than rest of course work
  - The only check that students can read code
  - Defers some time investment into the exam weeks
- -
  - Takes a lot of investment to avoid duplicates
  - Pass/Fail check in limited time
  - Dealing with student's handwriting

PEN & PAPER EXAM

PEN & PAPER EXAM

- ?

PEN & PAPER EXAM

- **?**
  - Keep, *if the course can fit it*.

PEN & PAPER EXAM

- **?**
  - Keep, *if the course can fit it*.
  - Invest into making sets of exam questions

PEN & PAPER EXAM

- **?**
  - Keep, *if the course can fit it*.
  - Invest into making sets of exam questions
  - Focus on reading code over puzzlers

PEN & PAPER EXAM

- **?**
  - Keep, *if the course can fit it*.
  - Invest into making sets of exam questions
  - Focus on reading code over puzzlers
  - Keep pass/fail element?

SEMESTRAL WORK

SEMESTRAL WORK

- +
  - Gets students to write body of code from nothing
  - Exposes students to using CI for larger project
  - Students can pick their own topic - motivation

SEMESTRAL WORK

- +
  - Gets students to write body of code from nothing
  - Exposes students to using CI for larger project
  - Students can pick their own topic - motivation
- -
  - *Very* hard to grade consistently
  - Lot of work to grade (2+ hours per piece)
  - Few students actually end up using CI

SEMESTRAL WORK

SEMESTRAL WORK

- ?

SEMESTRAL WORK

- ?
  - Keep

SEMESTRAL WORK

- ?
  - Keep
  - Invest effort in consistent grading

SEMESTRAL WORK

- ?
  - Keep
  - Invest effort in consistent grading
  - Make sure to give *formative* feedback

SEMESTRAL WORK

- ?
  - Keep
  - Invest effort in consistent grading
  - Make sure to give *formative* feedback
  - Make test quality part of grading

THREADING HOMEWORK

THREADING HOMEWORK

- +
  - Allows students to focus just on multithreading
  - Introduces students to benchmarking
  - Allows targetted formative feedback just for MT

THREADING HOMEWORK

- +
  - Allows students to focus just on multithreading
  - Introduces students to benchmarking
  - Allows targetted formative feedback just for MT
- -
  - Overlaps with semestral work

THREADING HOMEWORK

- +
  - Allows students to focus just on multithreading
  - Introduces students to benchmarking
  - Allows targetted formative feedback just for MT
- -
  - Overlaps with semestral work
- ?
  - Keep, if there is enough time in the semester

That's all for notes on specific parts of the course

GENERAL ADVICE

BEWARE TRADITION

BEWARE TRADITION

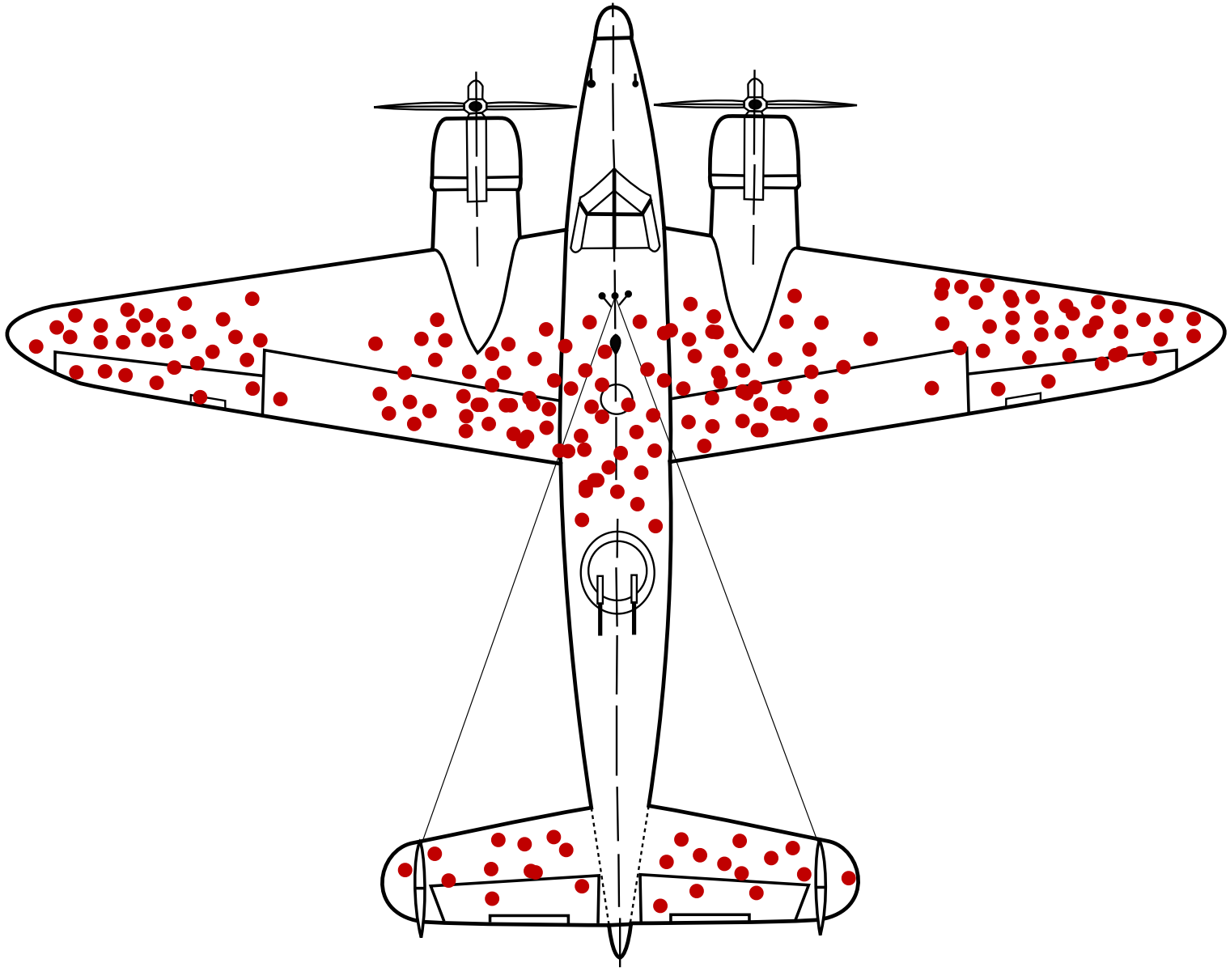
Do not add things to course just because that is how it was done before.

COLLECTING DATA

Some ways to collect data are obvious.

Some ways to collect data are obvious.

You can ask students how long they needed for a task when they submit it for grading.



Collecting data at submission leaves out information from students who did not submit their work.

Collecting data at submission leaves out information from students who did not submit their work.

How can you learn about students that stopped?

Compare how many students submitted each work

Compare how many students submitted each work

Compare weekly student attendance at labs

Track the success rate of your exam questions

Track the success rate of your exam questions

Investigate questions with excessively low success rate

Your university likely collects student feedback

You can also just ask the students

BEWARE

BEWARE

Data is (mostly) objective

BEWARE

Data is (mostly) objective

Your interpretation might not be

BEWARE

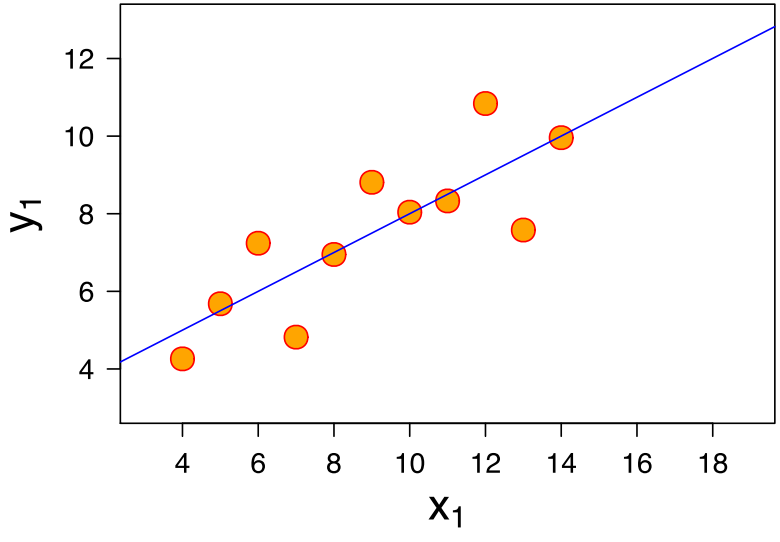
BEWARE

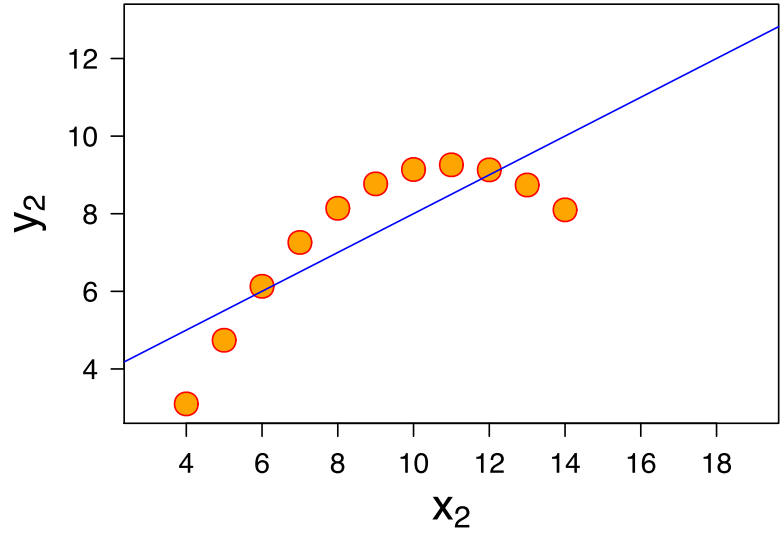
Statistical blindness is real and not your friend

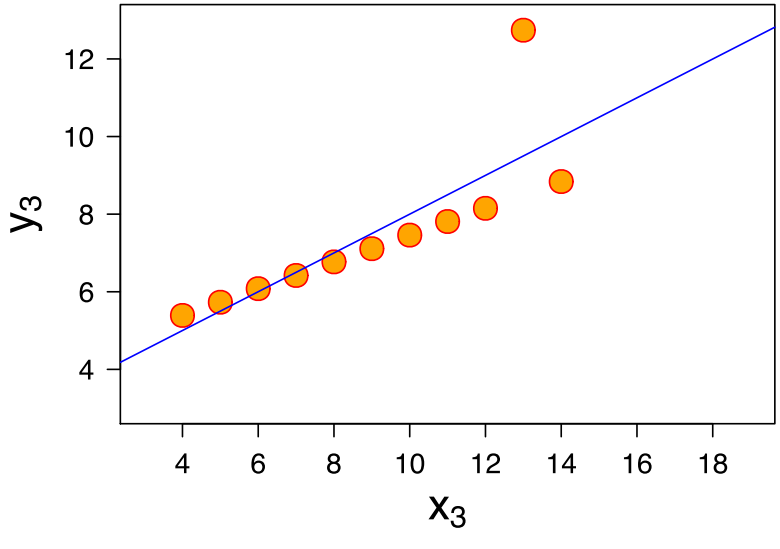
BEWARE

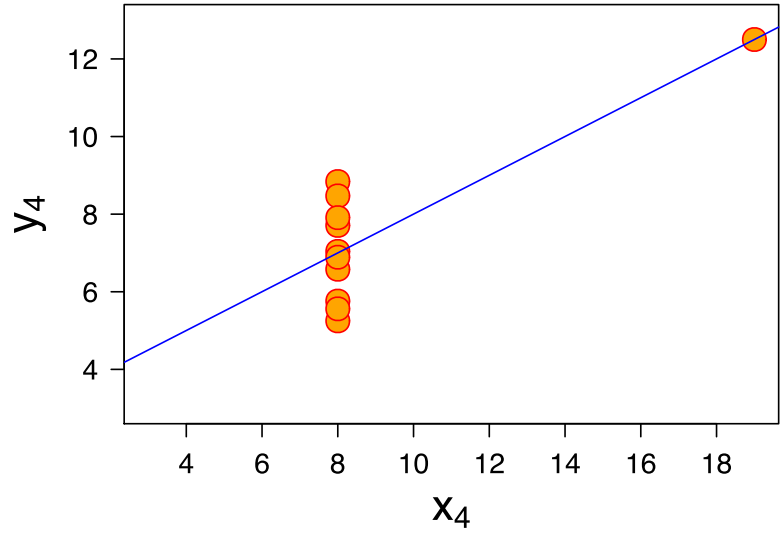
Statistical blindness is real and not your friend

You cannot blindly run statistical analysis on a dataset and assume that you understand it.


y1
x1
4
6
8
10
12
14
16
18


y2
x2
4
6
8
10
12
14
16
18


y3
x3
4
6
8
10
12
14
16
18


y4
x4
4
6
8
10
12
14
16
18

DEALING WITH PLAGIARISM

Check your university's rules first

Check your university's rules first

We had good experience with MOSS tool for checking

Check your university's rules first

We had good experience with MOSS tool for checking

You can learn interesting things in submission history

DESIGNING HOMEWORK

Make sure that what you teach and what is in the homework(s) is aligned and not contrary.

Good way to estimate time students will need for some work is to time yourself and multiply it by 5-7.

EXPECT TO SPEND A LOT OF TIME

Preparing good course *materials* takes a lot of time.

Preparing good course *materials* takes a lot of time.

We were at 60+ person-hours per week of classes before I quit.

Actually teaching the course still takes a ton of time.

Actually teaching the course still takes a ton of time.

I averaged out about 5 hours per week per lab class.

PRACTICE YOUR POKER FACE

Is it still plagiarism if I submit the same work next semester?

We worked together but didn't share code, is it still plagiarism?

What is a leap year?

```
std::vector<
    std::pair<
        std::shared_ptr<
            std::map<std::shared_ptr<Node>, int>
        >,
        std::shared_ptr<Node>
    >
>
```

Requiring homeworks to not have memory leaks is too strict.

I've had students play LoL during the lecture.

I've had students play LoL during the lecture.

Sadly never Dota2.

I have many more stories like these

I have many more stories like these

(you can ask me about them afterwards)

THE END

QUESTIONS

- survivorship-bias.svg, Martin Grandjean (vector), McGeddon (picture), Cameron Moll (concept), CC-BY-SA 4.0
- Anscombe.svg, Schutz, Avenue, CC-BY-SA 3.0