

# Metrics and Measurement

17-313 Fall 2025

Foundations of Software Engineering

<https://cmu-17313q.github.io>

Eduardo Feo Flushing

# Administrivia

- Project 1B due Sunday, Sep. 7th, 11:59PM
- Reading for Wednesday
  - Posted on the course website
  - Reading Quiz due before the lecture (check Gradescope)
  - Optional: “Flight/Risk” on Amazon Prime, “Downfall: the case against Boeing” on Netflix

# Today's Learning Goals

- Explain the importance of measurement and metrics in Software Engineering
- Provide examples of metrics for software qualities and process
- Apply goal-based frameworks for decision making using metrics
- Identify the limitations and dangers of decisions and incentives based on measurements

# Measurement in everyday life

- Economics
  - price, inflation rate, stock price, volume
- Medicine
  - heart rate, blood pressure, body temperature, ECG
- Engineering
  - Force, torque, heat transfer coefficient, thermal efficiency
- Nature
  -

NS New Scientist

## Ants use pedometers to find home

An experiment that involves attaching stilts to ants' legs reveals that the insects somehow keep a record of how many steps they take.

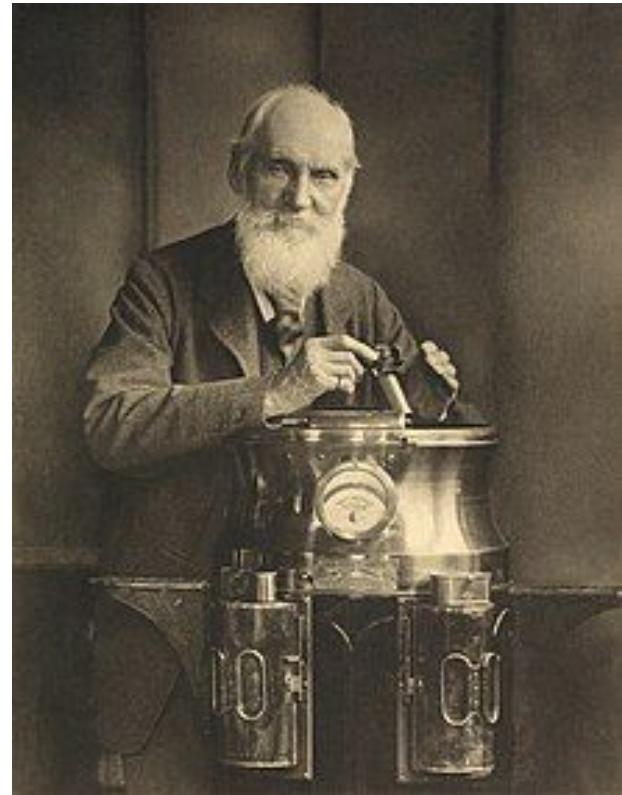
Jun 29, 2006



*“To measure is to know;  
if you can not measure it,  
you can not improve it”*

William Thomson, Lord Kelvin

$$K = \left( \frac{5}{9} (F - 32) \right) + 273.15$$



# Software Development... before Software Engineering



# Software Engineering

# Software Engineering: Principles, practices (technical and non-technical) for confidently building high-quality software.

What does this mean?

How do we know?

*Measurement* and metrics are **key** concerns.

# Outline

- Measurements and Metrics
- How to use measurements and metrics?
- Case study: Autonomous Vehicle Software
- Risks and challenges
- Metrics and incentives

# Outline

- **Measurements and Quality Attributes**
  - How to use measurements and metrics?
  - Case study: Autonomous Vehicle Software
  - Risks and challenges
  - Metrics and incentives

# What is Measurement?

- **Measurement is the empirical, objective assignment of numbers, according to a rule derived from a model or theory, to attributes of objects or events with the intent of describing them.** – Craner, Bond, “Software Engineering Metrics: What Do They Measure and How Do We Know?”

# What is Measurement?

A more realistic definition

- A quantitatively expressed reduction of uncertainty based on one or more observations. – Hubbard, "How to Measure Anything ..."

## Entity

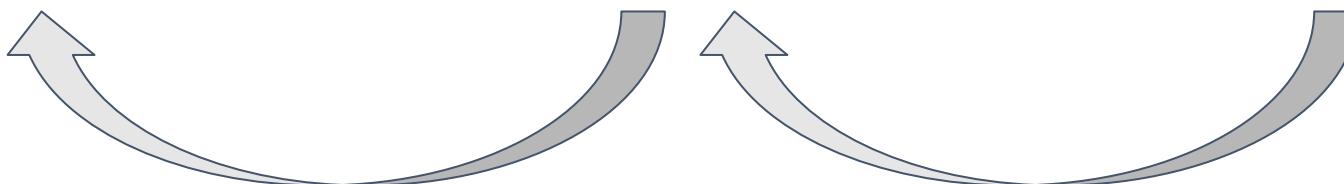
Object or Process

## Quality Attribute

Quality of Interest

## Measurement

Method to obtain a number or a symbol



# What entities do we care about? (examples)

- Software product
- Modules
- Software development process
- People

# What **software** qualities do we care about? (examples)

- Functionality (e.g., data integrity)
- Scalability
- Security
- Extensibility
- Bugginess
- Documentation
- Performance
- Installability
- Availability
- Consistency
- Portability
- Regulatory compliance

# What **process** qualities do we care about? (examples)

- Development efficiency
- Meeting efficiency
- Conformance to processes
- Reliability of predictions
- Fairness in decision making
- Regulatory compliance
- On-time release

# What **people** qualities do we care about? (examples)

- Developers
  - Maintainability
  - Performance
  - Employee satisfaction
  - Communication and collaboration
  - Efficiency and flow
  - Satisfaction with engineering system
  - Regulatory compliance
- Customers
  - Satisfaction
  - Ease of use
  - Feature usage
  - Regulatory compliance

# Non-trivial qualities

- Software
  - Code elegance
  - Code maintainability
- Process
  - Fairness in decision making
- Team
  - Team collaboration
  - Creativity

# Everything is measurable

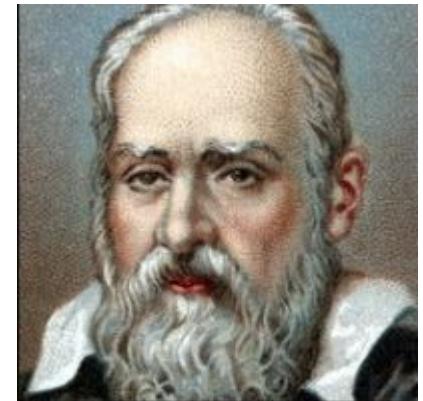
- If X is something we care about, then X, by definition, must be detectable.
- If X is detectable, then it must be detectable in some amount.
- If we can observe it in some amount, then it must be measurable.

Douglas Hubbard, How to Measure Anything, 2010

# Make it measurable

*“Measure what is measurable, and make measurable what is not so.”*

Galileo Galilei



# Examples: Code Complexity

# Lines of Code

- Easy to measure

```
> wc -l file1  
file2...
```

LOC	projects
450	Expression Evaluator
2,000	Sudoku
100,000	Apache Maven
500,000	Git
3,000,000	MySQL
15,000,000	gcc
50,000,000	Windows 10
2,000,000,000	Google (MonoRepo)

# Normalizing Lines of Code

- Ignore comments and empty lines
- Ignore lines < 2 characters
- Pretty print source code first
- Count statements (logical lines of code)
- See also: cloc

```
for (i = 0; i < 100; i += 1) printf("hello"); /* How many lines of code is this? */
```

```
/* How many lines of code is this? */
```

```
for (
    i = 0;
    i < 100;
    i += 1
) {
    printf("hello");
}
```

# Normalization by Language

Language	Statement factor (productivity)	Line factor
C	1	1
C++	2.5	1
Fortran	2	0.8
Java	2.5	1.5
Perl	6	6
Smalltalk	6	6.25
Python	6	6.5

Source: "Code Complete: A Practical Handbook of Software Construction", S. McConnell, Microsoft Press (2004) and <http://www.codinghorror.com/blog/2005/08/are-all-programming-languages-the-same.html> u.a.

# Halstead Volume

- Introduced by Maurice Howard Halstead in 1977
- Halstead Volume =  
number of operators/operands \*  
 $\log_2(\text{number of distinct operators/operands})$
- Approximates size of elements and vocabulary

# Cyclomatic Complexity

- Proposed by McCabe 1976
- Based on control flow graph, measures linearly independent paths through a program
  - $\approx$  number of decisions
  - Number of test cases needed to achieve branch coverage

$M = \text{edges of CFG} - \text{nodes of CFG} + 2 * \text{connected components}$

*“For each module, either limit cyclomatic complexity to [X] or provide a written explanation of why the limit was exceeded.”*  
– NIST Structured Testing methodology

```
if (c1) {  
    f1();  
} else {  
    f2();  
}  
if (c2) {  
    f3();  
} else {  
    f4();  
}
```

# Object-Oriented Metrics

- Number of Methods per Class
- Depth of Inheritance Tree
- Number of Child Classes
- Coupling between Object Classes
- Calls to Methods in Unrelated Classes
- ...

# Outline

- Measurements and Metrics
- **How to use measurements and metrics?**
- Case study: Autonomous Vehicle Software
- Risks and challenges
- Metrics and incentives

# Goal-based frameworks

*"Every measurement action must be motivated by a particular goal or need that is clearly defined and easily understandable."*

Measurement must be defined in a top-down fashion.



Measure  
What  
Matters

2.4572 in

2.4572

How Google, Bono, and the Gates Foundation Rock the World with OKRs

John Doerr

WITH A FOREWORD BY LARRY PAGE

# The GQM framework

**Goal:** What do you want to achieve?

**Questions:** What do you need to answer to know whether your goal is met?

**Metrics:** What measurements do you need in order to answer each question?

## THE GOAL QUESTION METRIC APPROACH

Victor R. Basili<sup>1</sup> Gianluigi Caldiera<sup>1</sup> H. Dieter Rombach<sup>2</sup>

(<sup>1</sup>) Institute for Advanced Computer Studies  
Department of Computer Science  
University Of Maryland  
College Park, Maryland

(<sup>2</sup>) FB Informatik  
Universität Kaiserslautern  
Kaiserslautern, Germany

# GQM: Defining Goals

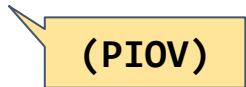
P: **Purpose** (improve, evaluate, monitor, ...)

I: **Issue** (reliability, usability, effectiveness, ...)

O: **Object** (final product, component, process, activity)

V: **Viewpoint** (any stakeholder)

**Goal:**



**Evaluate the effectiveness of the organization's coding standard from the team's perspective**

**Questions:**

How comprehensible are the coding standards?

What is the impact of coding standards on the efficiency and productivity of the team?

**Metrics:**

Survey results measuring team members' understanding

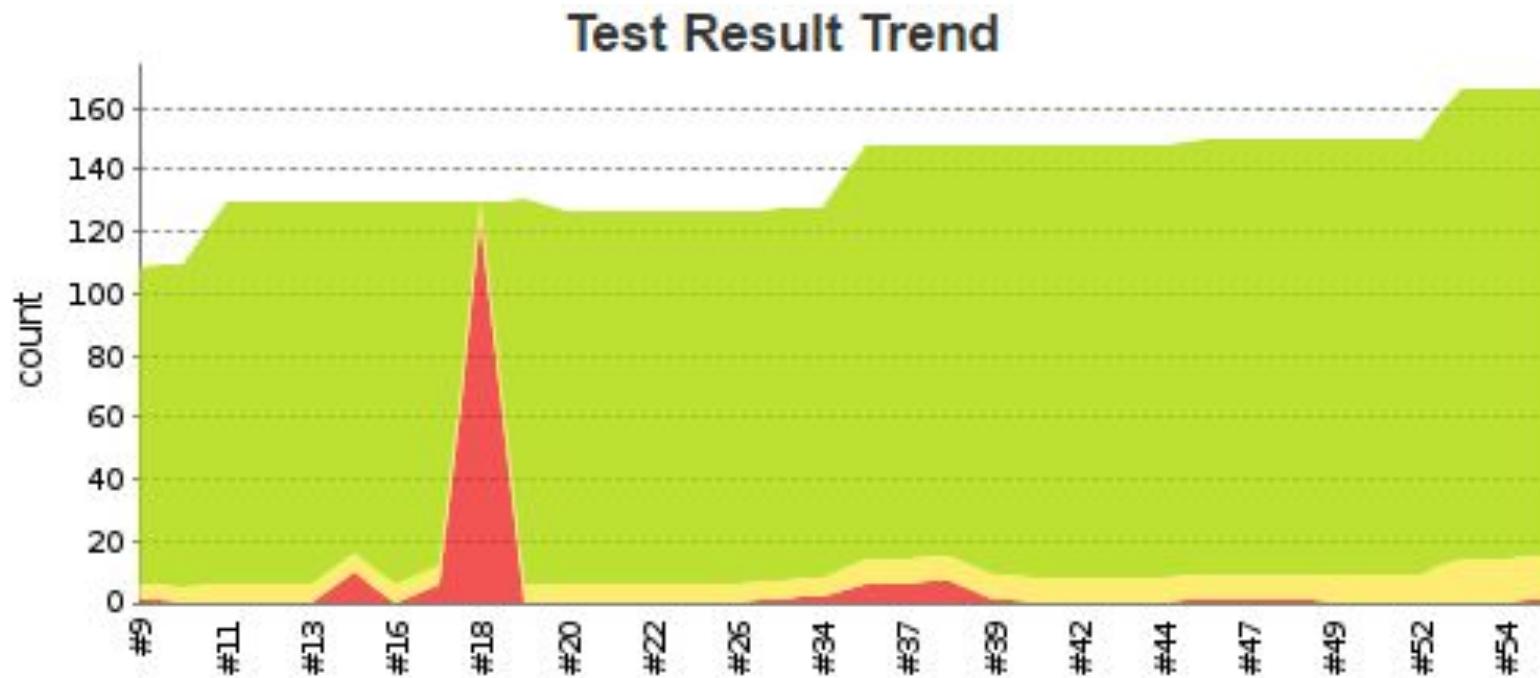
Number of revisions required to achieve standard compliance

Code size: LOC, number of classes, number of functions

# Measurement for Decision Making

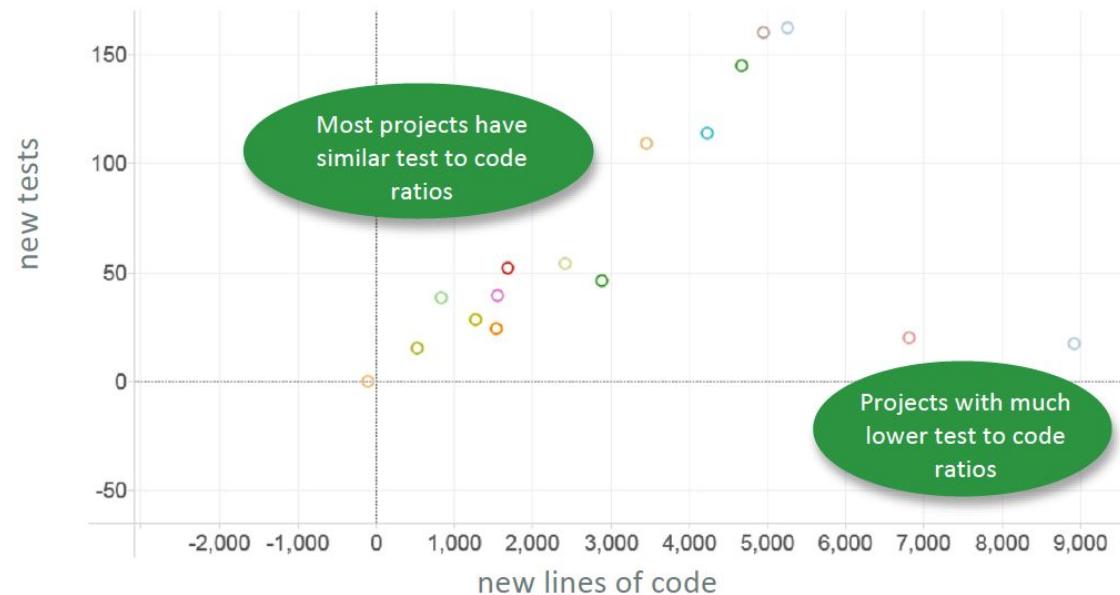
- Fund project?
- More testing?
- Fast enough? Secure enough?
- Code quality sufficient?
- Which feature to focus on?
- Developer bonus?
- Time and cost estimation? Predictions reliable?

# Trend analyses



# Benchmarking against standards

- Monitor many projects or many modules, get typical values for metrics
- Report deviations



# Outline

- Measurements and Metrics
- How to use measurements and metrics?
- **Case study: Autonomous Vehicle Software**
- Risks and challenges
- Metrics and incentives

# Case study: Autonomous Vehicles

# AV Software is

---



# Let's apply the Goal-Question-Metric framework to explore various aspects of AV software



# AV Software is safe



**Goal: Evaluate and enhance the **safety** of the autonomous vehicle system from the perspectives of **safety engineers** and **passengers**.**



Note that  
we followed  
PIOV to  
define our  
goal

# Goal: Evaluate and enhance the **safety** of the **autonomous vehicle system** from the perspectives of **safety engineers**.

- How extensively are the critical safety functions of the vehicle's software covered by automated tests?
- How accurate are the vehicle's machine learning models in detecting and classifying road objects and obstacles?
- How consistently does the vehicle maintain safe and effective operation across different driving scenarios?



what we need to answer to know  
whether our goal is met

## (1) Code coverage

- Amount of code executed during testing.
  - Statement coverage, line coverage, branch coverage, etc.
  - E.g., 75% branch coverage   
3/4 if-else outcomes have been executed

## Possible measurement

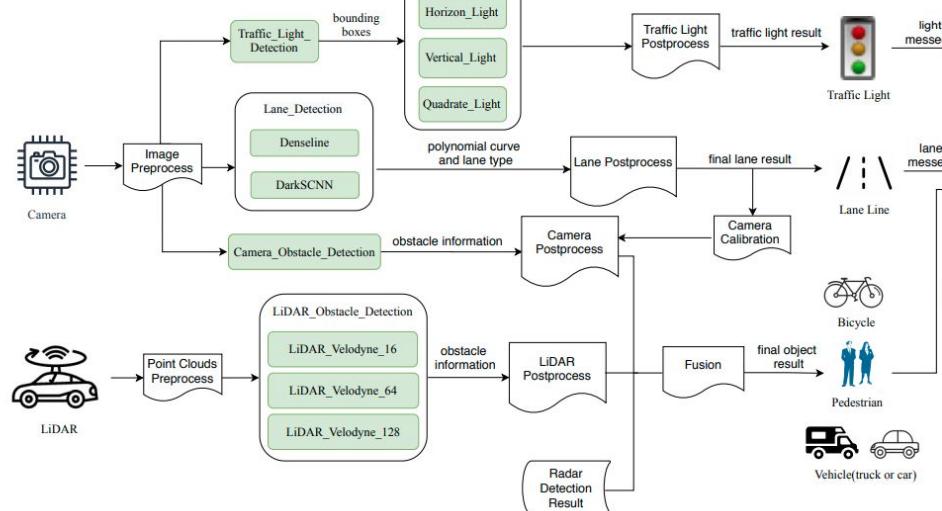
*How extensively are the critical safety functions of the vehicle's software covered by automated tests?*

```
1698 : const TrajectoryPoints& StGraphData::init_point() const { return init_point_; }
:
2264 : const SpeedLimit& StGraphData::speed_limit() const { return speed_limit_; }
:
212736 : double StGraphData::cruise_speed() const {
212736 :     return cruise_speed_ > 0.0 ? cruise_speed_ : FLAGS_default_cruise_speed;
:
1698 : double StGraphData::path_length() const { return path_data_length_; }
:
1698 : double StGraphData::total_time_by_conf() const { return total_time_by_conf_; }
:
1698 : planning_internal::STGraphDebug* StGraphData::mutable_st_graph_debug() {
1698 :     return st_graph_debug_;
:
566 : bool StGraphData::SetSTDivableBoundary(
:
566 :     const std::vector<std::tuple<double, double, double>>& s_boundary,
566 :     const std::vector<std::tuple<double, double, double>>& v_obs_info) {
:
566 :     if (s_boundary.size() != v_obs_info.size()) {
566 :         return false;
:
40752 :     for (size_t i = 0; i < s_boundary.size(); ++i) {
40732 :         auto st_bound_instance = st_drivable_boundary_.add_st_boundary();
40732 :         st_bound_instance->set_t(std::get<0>(s_boundary[i]));
40732 :         st_bound_instance->set_s_lower(std::get<1>(s_boundary[i]));
40732 :         st_bound_instance->set_s_upper(std::get<2>(s_boundary[i]));
40732 :         if (std::get<1>(v_obs_info[i]) > -kObsSpeedIgnoreThreshold) {
40732 :             st_bound_instance->set_v_obs_lower(std::get<1>(v_obs_info[i]));
:
40186 :         if (std::get<2>(v_obs_info[i]) < kObsSpeedIgnoreThreshold) {
40186 :             st_bound_instance->set_v_obs_upper(std::get<2>(v_obs_info[i]));
:
50254 :         }
:
40186 :     }
:
40186 : }
```

## (2) Model Accuracy

- Train machine-learning models on labelled data (sensor data + ground truth).
- Compute accuracy on a separate labelled test set.
- E.g., 90% accuracy implies that object recognition is right for 90% of the test inputs.

Possible measurement  
*How accurate are the vehicle's machine Learning models in detecting and classifying road objects and obstacles?*



Source: Peng et al. ESEC/FSE'20

# (3) Failure Rate

- Frequency of crashes / fatalities
- Per 1,000 rides, per million miles, per month (in the news)

## Possible measurement

*How consistently does the vehicle maintain safe and effective operation across different driving scenarios?*

TRANSPO / WAYMO / TECH

**Waymo's driverless cars were involved in two crashes and 18 'minor contact events' over 1 million miles**



Image: Allen J. Schaben / Los Angeles Times

/ The Alphabet-owned company pulls back the curtain on more stats from its public road testing. Of the 20 incidents, only two met the federal government's reporting criteria, and no one was injured.

By Andrew J. Hawkins, transportation editor with 10+ years of experience who covers EVs, public transportation, and aviation. His work has appeared in The New York Daily News and City & State.

Feb 28, 2023, 8:00 PM GMT-3 | □ 1 Comment / 1 New



**'Complete meltdown': Driverless cars in San Francisco stall causing a traffic jam**

By Jordan Valinsky, CNN Business

Updated 3:45 PM EDT, Mon August 14, 2023

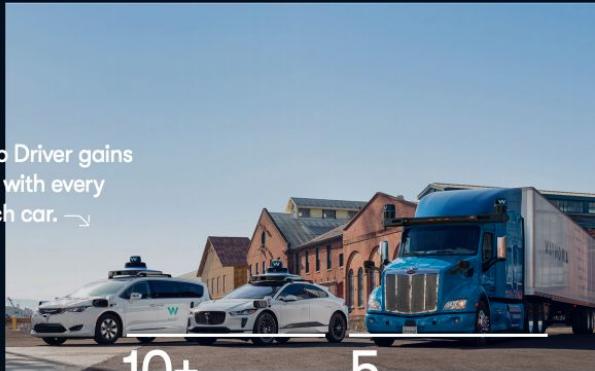


A Cruise autonomous taxi in San Francisco.

David Paul Morris/Bloomberg/Getty Images

# (4) Mileage

## Building the World's Most Experienced Driver™



### Possible measurement

*How consistently does the vehicle maintain safe and effective operation across different driving scenarios?*

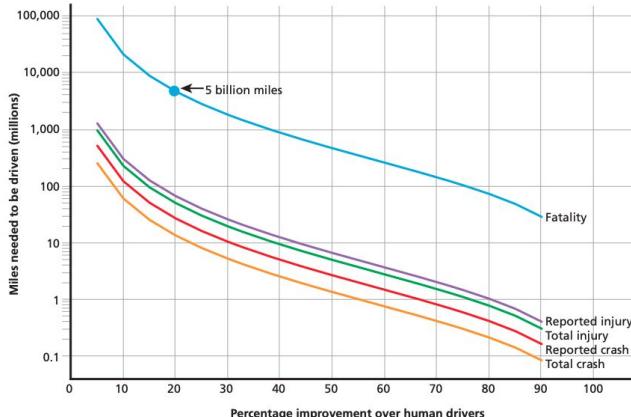


### Driving to Safety

#### How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability?

Nidhi Kalra, Susan M. Paddock

Figure 3. Miles Needed to Demonstrate with 95% Confidence that the Autonomous Vehicle Failure Rate Is Lower than the Human Driver Failure Rate



# Participation Activity

- Apply the **Goal-Question-Metric** framework to **explore one aspect of AV software**
- Define **one goal, two questions**, and at **least one metric** per question
- Write it down on a piece of paper with your Andrew ID(s) on it.
- You can work in groups of 2-3.
- Share with the class!

- Software
  - Test coverage
  - Model accuracy
  - Size of codebase
  - Age of codebase
- Software Process
  - Time since the most recent change
  - Frequency of code releases
  - Number of emails sent during development
- Contributors
  - Number of contributors
  - Age of contributors
  - Employee satisfaction of contributors
- Documentation
  - Amount of code documentation
- Application
  - Customer satisfaction
  - Mileage
  - Crash/kill rate

Example Metrics

# Example

**Goal:** Ensure energy efficiency and sustainability from the point of view of the organization and environmental analysts

**Question 1:** What is the vehicle's energy consumption under different driving conditions?

**Metrics:** Kilowatt-hours per 100 kilometers under city, highway, and mixed driving conditions.

**Question 2:** How efficient is the battery management system?

**Metrics:** Battery life, number of charge cycles

# Outline

- Measurements and Metrics
- How to use measurements and metrics?
- Case study: Autonomous Vehicle Software
- **Risks and challenges**
- Metrics and incentives

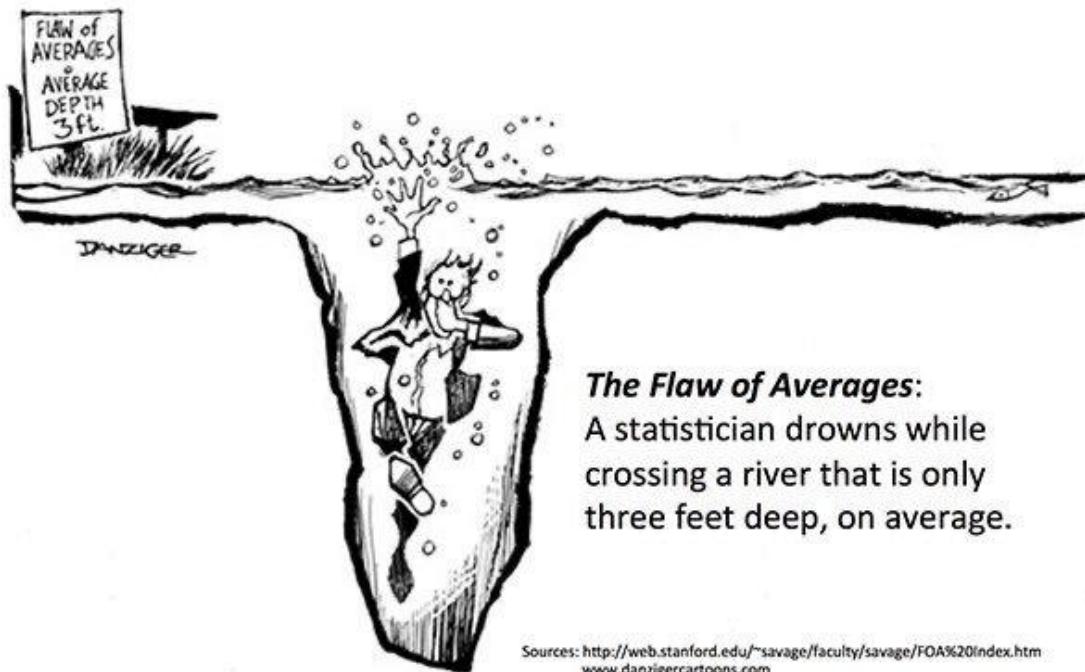


# The streetlight effect



- A known observational bias.
- People tend to look for something only where it's easiest to do so.
  - If you drop your keys at night, you'll tend to look for it under streetlights.

# Bad statistics: What could possibly go wrong?

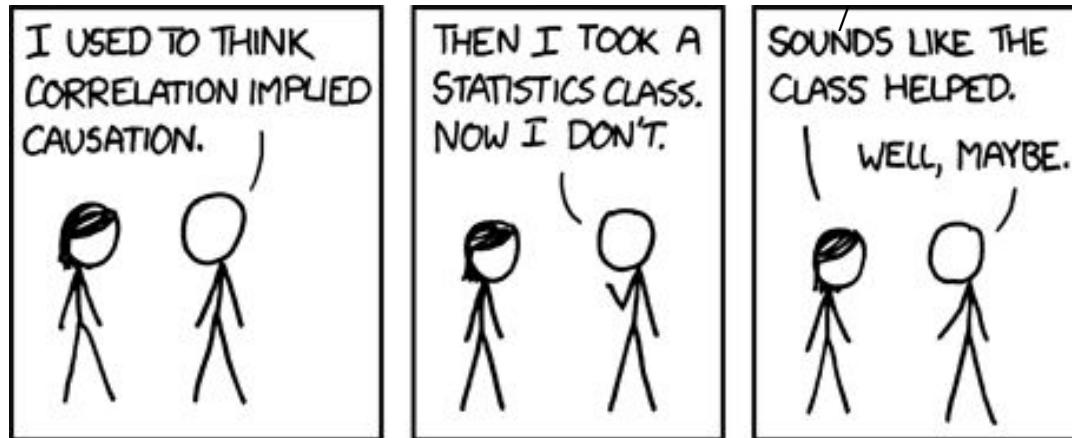


***The Flaw of Averages:***  
A statistician drowns while  
crossing a river that is only  
three feet deep, on average.

Sources: <http://web.stanford.edu/~savage/faculty/savage/FOA%20Index.htm>  
[www.danzigercartoons.com](http://www.danzigercartoons.com)

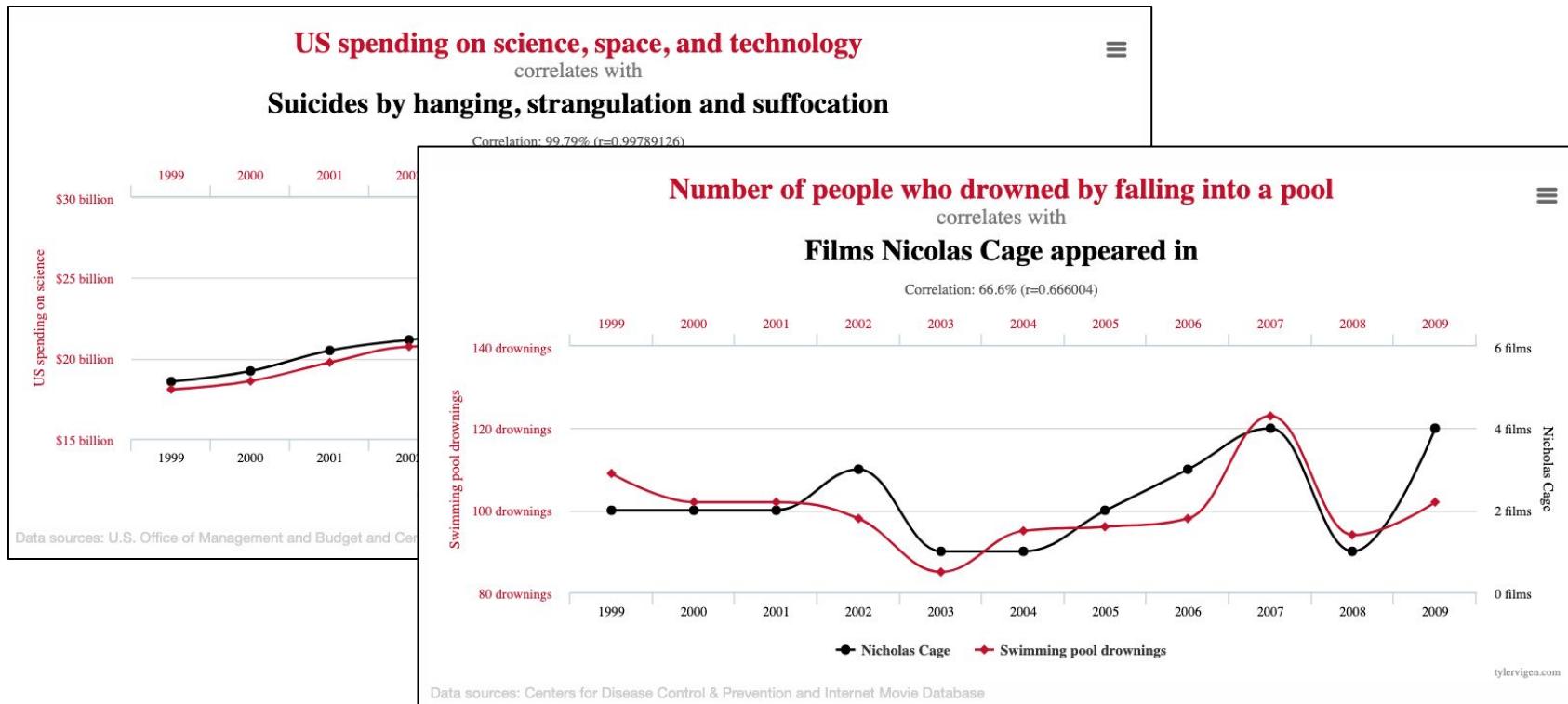
# Making inferences

<http://xkcd.com/552>

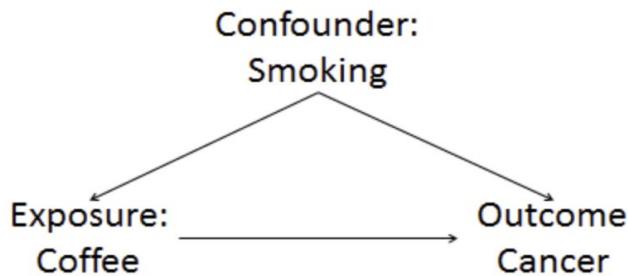
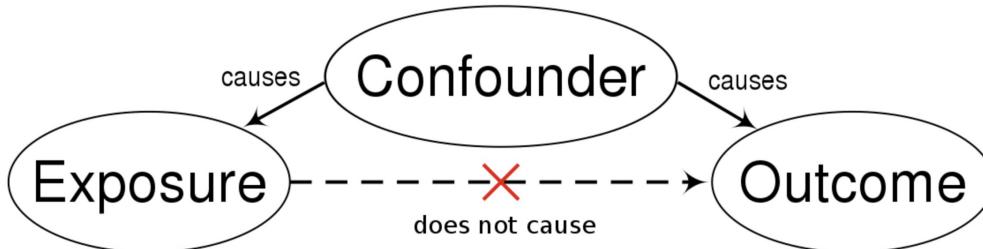


- To infer causation:
  - Provide a theory (from domain knowledge, independent of data)
  - Show correlation
  - Demonstrate ability to predict new cases (replicate/validate)

# Spurious Correlations



# Confounding variables



- If you look only at the coffee consumption → cancer relationship, you can get very misleading results
- Smoking is a confounder

# Coverage is not strongly correlated with test suite effectiveness

Authors:  [Laura Inozemtseva](#),  [Reid Holmes](#) [Authors Info & Affiliations](#)

ICSE 2014: Proceedings of the 36th International Conference on Software Engineering • May 2014 • Pages 435–445 • <https://doi.org/10.1145/2568225.2568271>

“We found that there is a low to moderate correlation between coverage and effectiveness when the number of test cases in the suite is controlled for.”

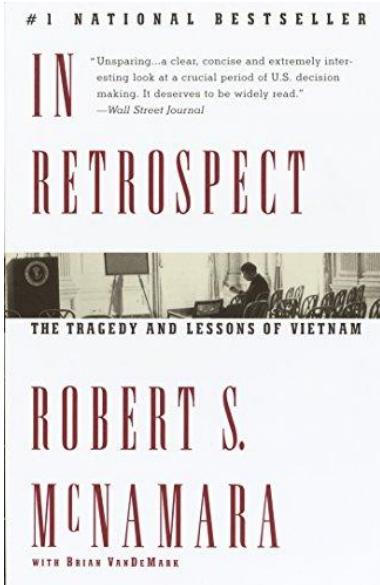
63

Most studies did not account for the confounding influence of test suite size

# Measurement reliability

- Extent to which a measurement yields similar results when applied multiple times
- Goal is to reduce uncertainty, increase consistency
- Example: Performance
  - Time, memory usage
  - Cache misses, I/O operations, instruction execution count, etc.
- Law of large numbers
  - Taking multiple measurements to reduce error
  - Trade-off with cost

# McNamara fallacy



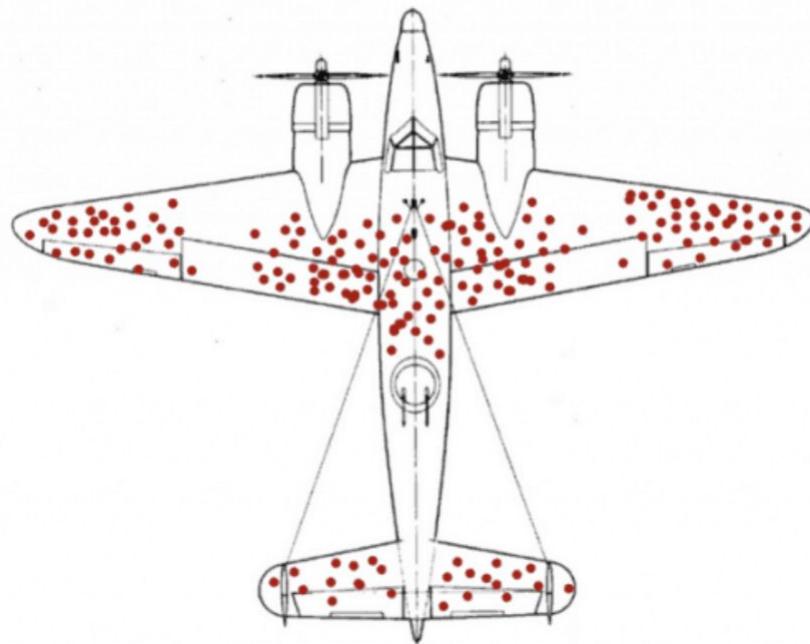
# McNamara fallacy

- Measure whatever can be easily measured.
- Disregard that which cannot be measured easily.
- Presume that which cannot be measured easily is not important.
- Presume that which cannot be measured easily does not exist.



<https://chronotopeblog.com/2015/04/04/the-mcnamara-fallacy-and-the-problem-with-numbers-in-education/>

# Survivorship bias



# Outline

- Measurements and Metrics
- How to use measurements and metrics?
- Case study: Autonomous Vehicle Software
- Risks and challenges
- **Metrics and incentives**

# Goodhart's law: "When a measure becomes a target, it ceases to be a good measure."



<http://dilbert.com/strips/comic/1995-11-13/>

## *The Price of Wells Fargo's Fake Account Scandal Grows by \$3 Billion*

The bank reached a settlement with federal prosecutors and the Securities and Exchange Commission after abusing customers.

 Share full article



 199



Wells Fargo used fraud to open up fake accounts and force customers into services

# Incentivizing Productivity

- What happens when developer bonuses are based on
  - Lines of code per day?
  - Amount of documentation written?
  - Low number of reported bugs in their code?
  - Low number of open bugs in their code?
  - High number of fixed bugs?
  - Accuracy of time estimates?

# What you need to know



Metrics are important in Software Engineering



Apply goal-oriented approaches to software metrics



Provide examples of metrics for software qualities and process



Understand limitations and dangers of decisions and incentives based on measurements