Metrics and Measurement

17-313 Fall 2023

Foundations of Software Engineering

https://cmu-313.github.io

Andrew Begel and Rohan Padhye

Administrivia

- Project 1(b) due on Thursday (Sept 7th) at midnight
  - Get started early, ask for help, and check the #technical-questions channel; chances are your questions have been asked by others!
  - Be sure to document any technical SE challenges you're facing (flaky tests, etc.) for the reflection
- Team formation this week! Attend recitation.

Smoking Section

- Last full row

Today's Learning Goals

- Use measurements as a decision tool to reduce uncertainty
- Understand difficulty of measurement; discuss validity of measurements
- Provide examples of metrics for software qualities and process
- Understand limitations and dangers of decisions and incentives based on measurements

Software Engineering

Software Engineering: Principles, t[illegible]chnical and non-[illegible] confidently building [illegible]ality software.

What does this mean?
How do we know?
→ *Measurement* and metrics are **key** concerns.

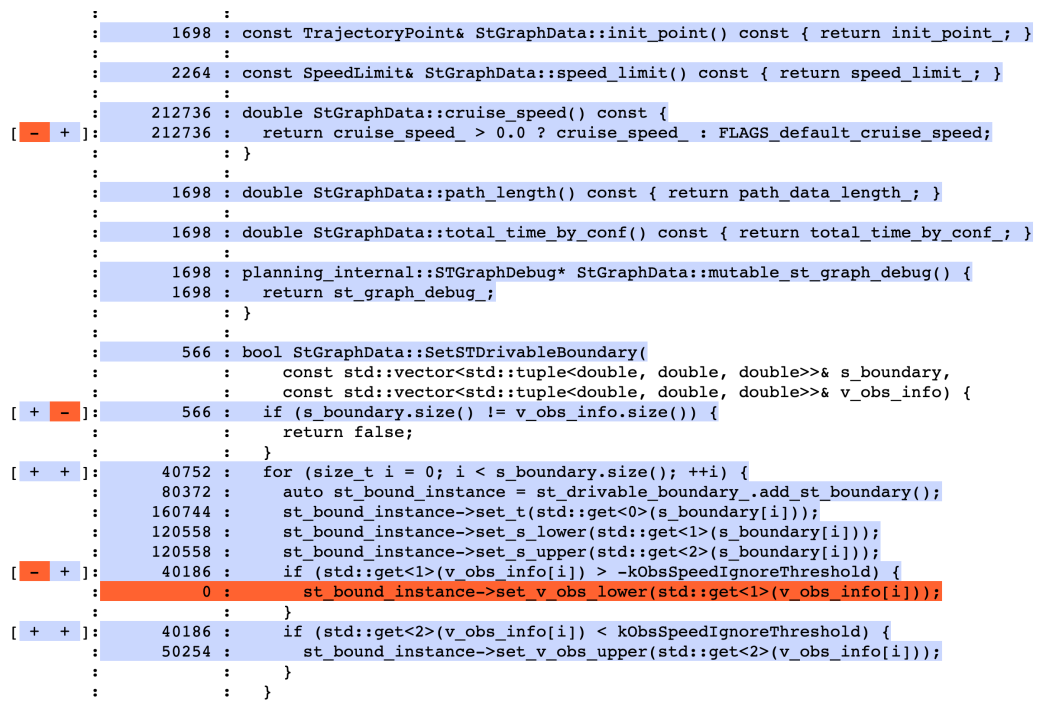
Case study: Autonomous Vehicles

AV Software is _____

By what methods can we judge AV software quality (e.g., safety)?

(1) Test coverage

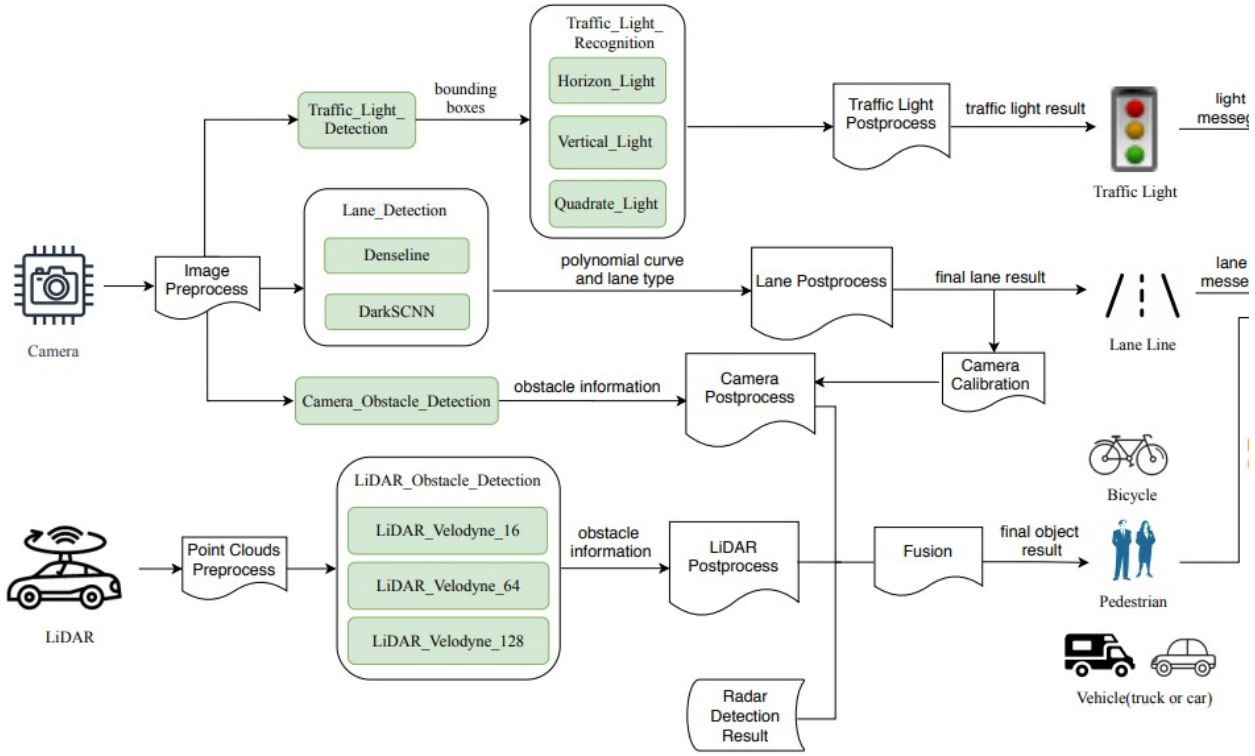
- Amount of code executed during testing.
- Statement coverage, line coverage, branch coverage, etc.
- E.g., 75% branch coverage → 3/4 if-else outcomes have been executed



```
        :          :
        :     1698 : const TrajectoryPoint& 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::SetSTDrivableBoundary(
        :          :     const std::vector<std::tuple<double, double, double>>& s_boundary,
        :          :     const std::vector<std::tuple<double, double, double>>& v_obs_info) {
[ +  - ]:      566 :   if (s_boundary.size() != v_obs_info.size()) {
        :          :     return false;
        :          :   }
[ +  + ]:    40752 :   for (size_t i = 0; i < s_boundary.size(); ++i) {
        :    80372 :     auto st_bound_instance = st_drivable_boundary_.add_st_boundary();
        :   160744 :     st_bound_instance->set_t(std::get<0>(s_boundary[i]));
        :   120558 :     st_bound_instance->set_s_lower(std::get<1>(s_boundary[i]));
        :   120558 :     st_bound_instance->set_s_upper(std::get<2>(s_boundary[i]));
[ -  + ]:    40186 :     if (std::get<1>(v_obs_info[i]) > -kObsSpeedIgnoreThreshold) {
        :        0 :       st_bound_instance->set_v_obs_lower(std::get<1>(v_obs_info[i]));
        :          :     }
[ +  + ]:    40186 :     if (std::get<2>(v_obs_info[i]) < kObsSpeedIgnoreThreshold) {
        :    50254 :       st_bound_instance->set_v_obs_upper(std::get<2>(v_obs_info[i]));
        :          :     }
        :          :   }
```

(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.



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)

(4) Mileage

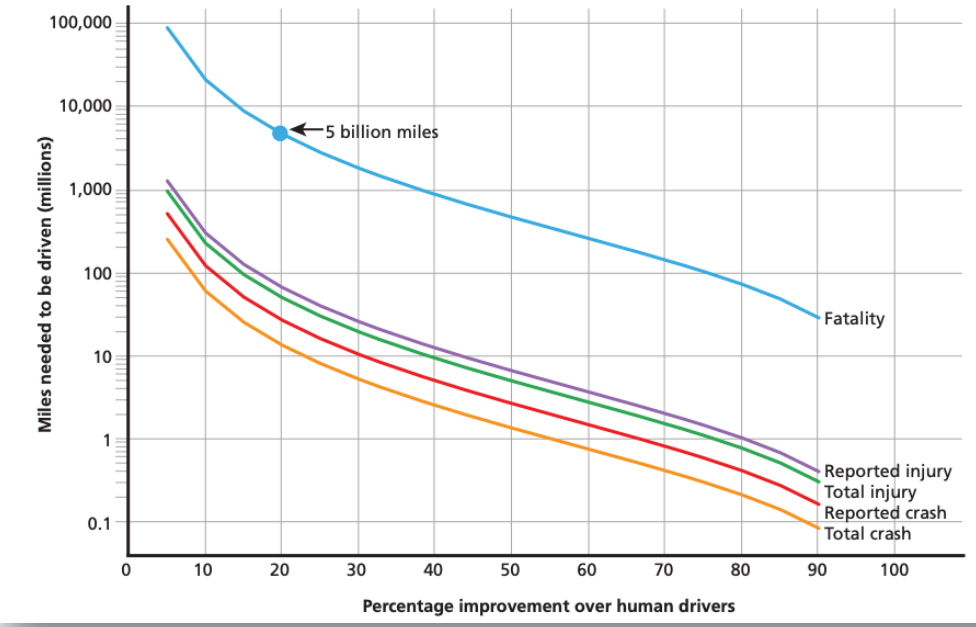
RAND CORPORATION

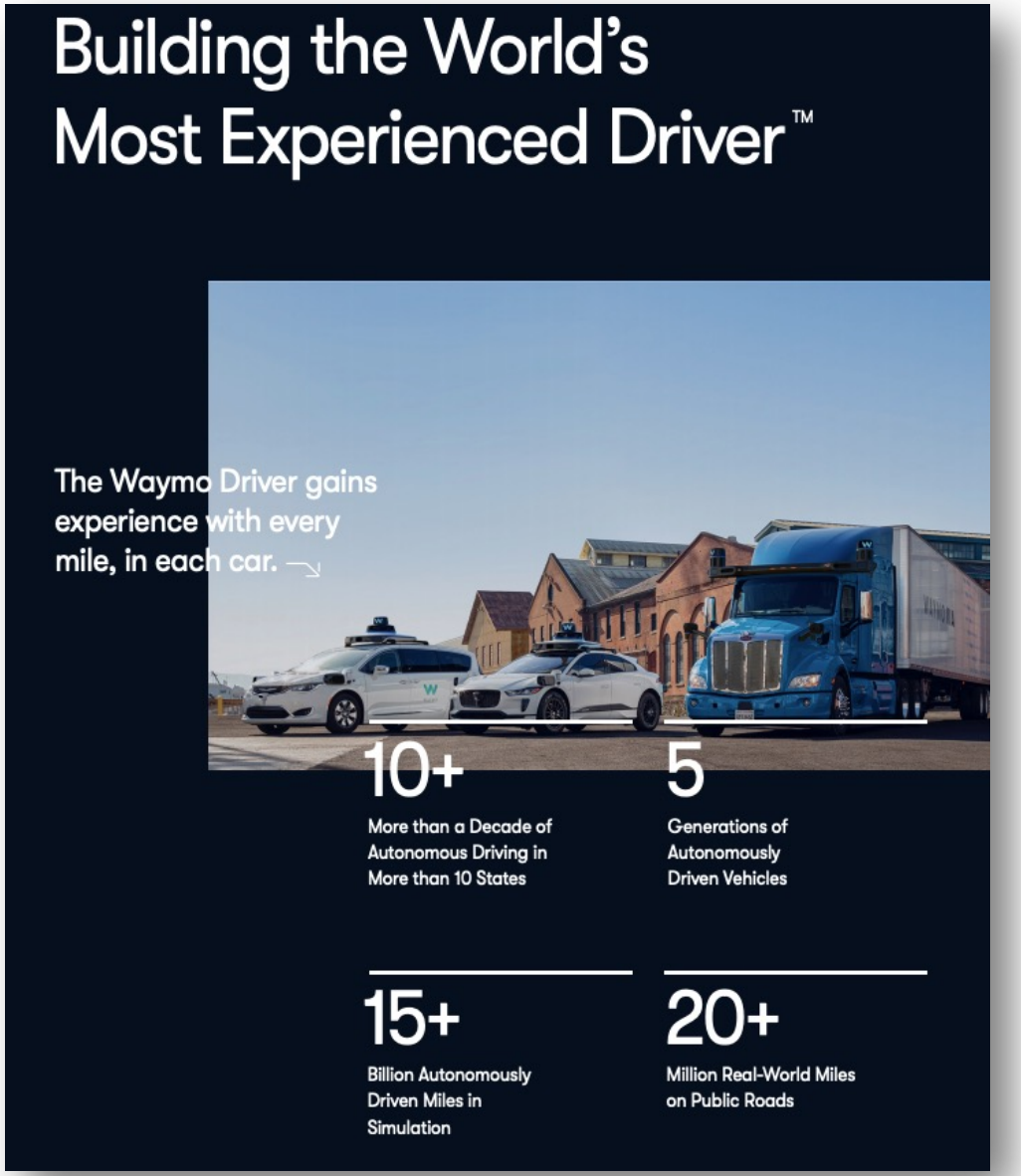
Driving to Safety

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





Source: waymo.com/safety (September 2021)

Participation Activity

- Brainstorm “pros” and “cons” for using various quality metrics to judge AV software.
- Pick 3 metrics and write down 1 pro and 1 con for each. 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!
- Take a pic with your phone and upload to Gradescope now. Everyone needs an individual submission.

- 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

Measurement and Metrics

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

Software Quality Metrics

- IEEE 1061 definition: “A software quality metric is a function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given attribute that affects its quality.”
- Metrics have been proposed for many quality attributes; may define own metrics

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)

- On-time release
- Development speed
- Meeting efficiency
- Conformance to processes
- Time spent on rework
- Reliability of predictions
- Fairness in decision making
- Number of builds
- Code review acceptance rate
- Regulatory compliance

- Measure time, costs, actions, resources, and quality of work packages; compare with predictions
- Use information from issue trackers, communication networks, team structures, etc…

What people qualities do we care about? (examples)

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

Everything is measurable

- If X is something we care about, then X, by definition, must be detectable.
  - How could we care about things like “quality,” “risk,” “security,” or “public image” if these things were totally undetectable, directly or indirectly?
  - If we have reason to care about some unknown quantity, it is because we think it corresponds to desirable or undesirable results in some way.
- If X is detectable, then it must be detectable in some amount.
  - If you can observe a thing at all, you can observe more of it or less of it
- If we can observe it in some amount, then it must be measurable.

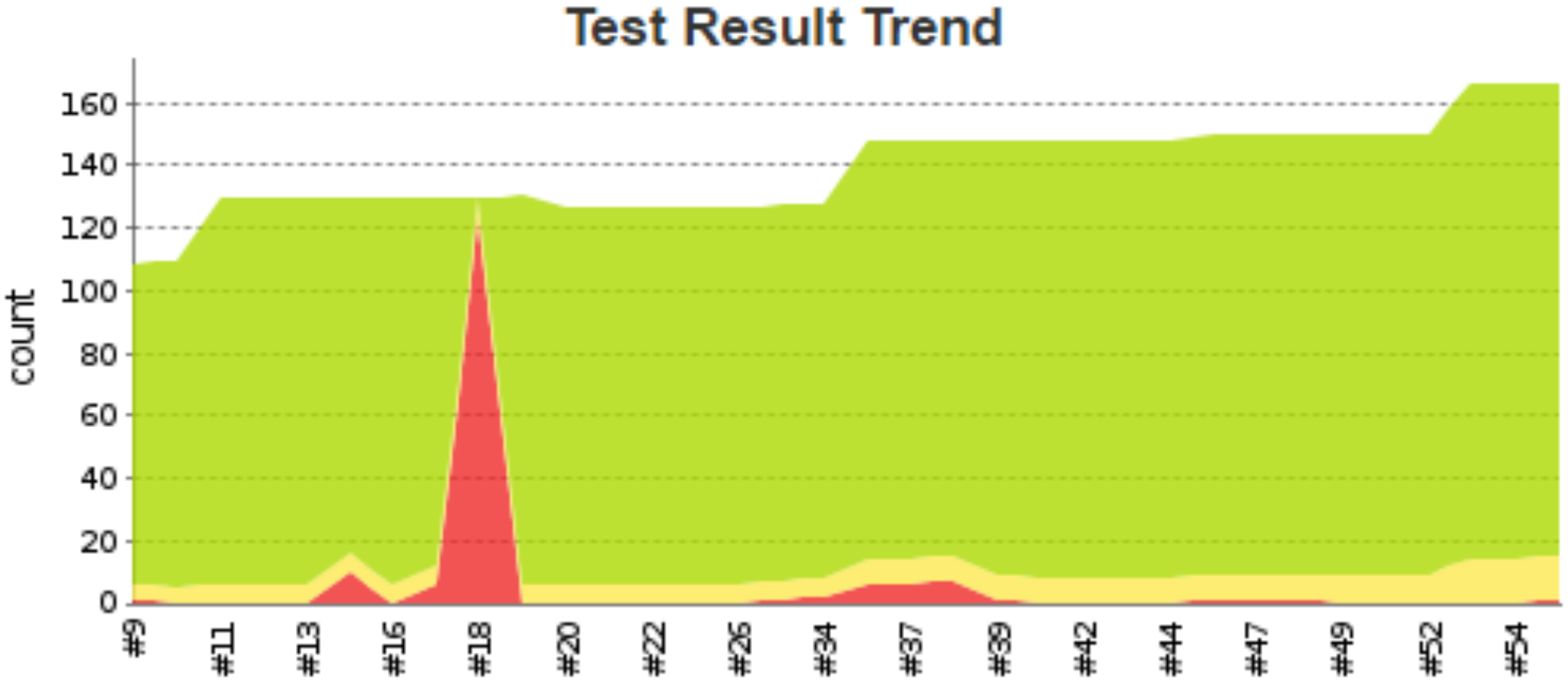
Douglas Hubbard, How to Measure Anything, 2010

Why Measure?

Measurement for Decision Making

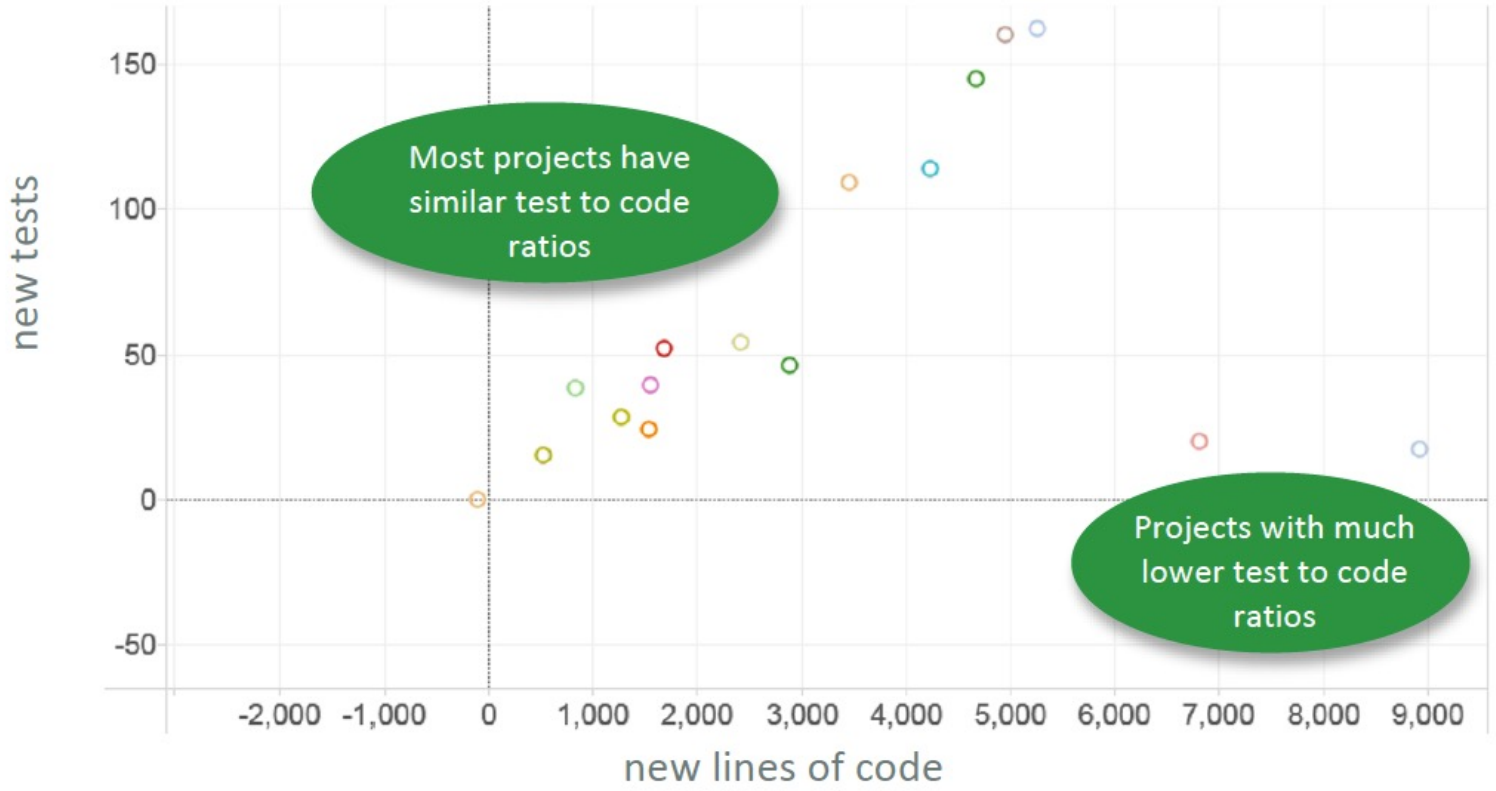
- Fund project?
- More testing?
- Fast enough? Secure enough?
- Code quality sufficient?
- Which feature to focus on?
- Developer bonus?
- Time and cost estimation? Predictions reliable?

Trend analyses



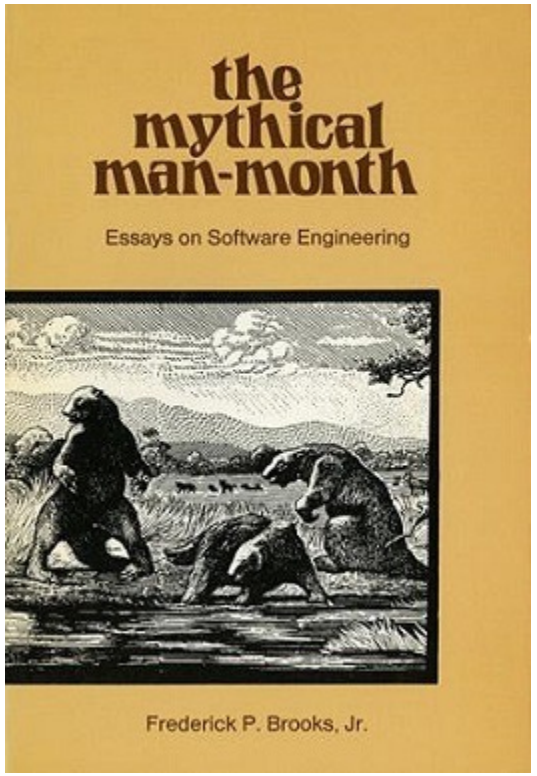
Benchmarking against standards

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

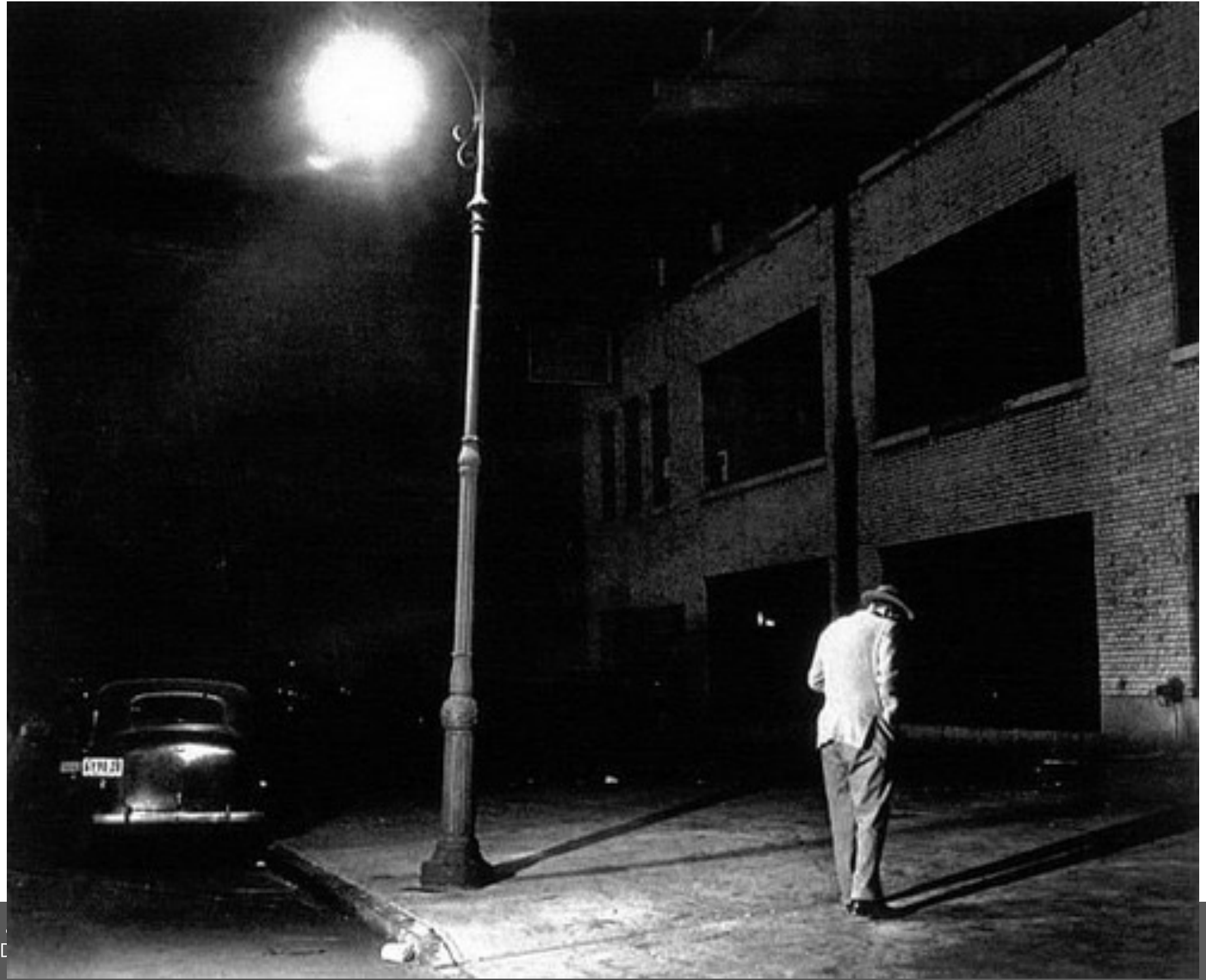


Antipatterns in effort estimation

- IBM in the 60s: Would account in “person-months” e.g. Team of 2 working 3 months = 6 person-months
- LoC ~ Person-months ~ $$$
- Brooks: “Adding manpower to a late software project [just] makes it later.”



Measurement is Difficult

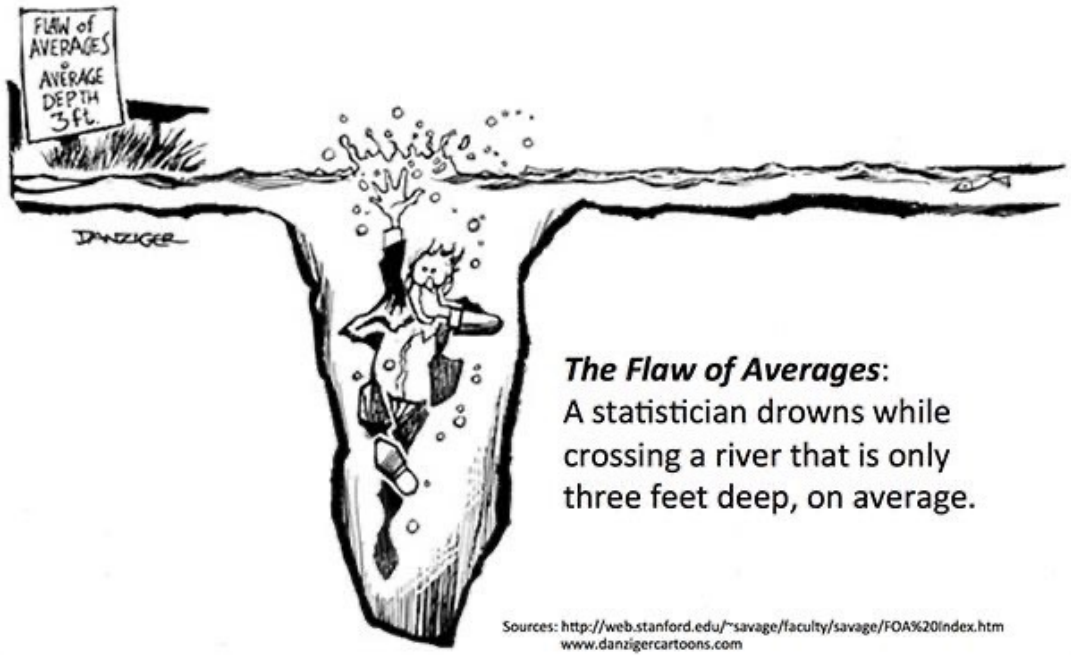


The streetlight effect

- 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.

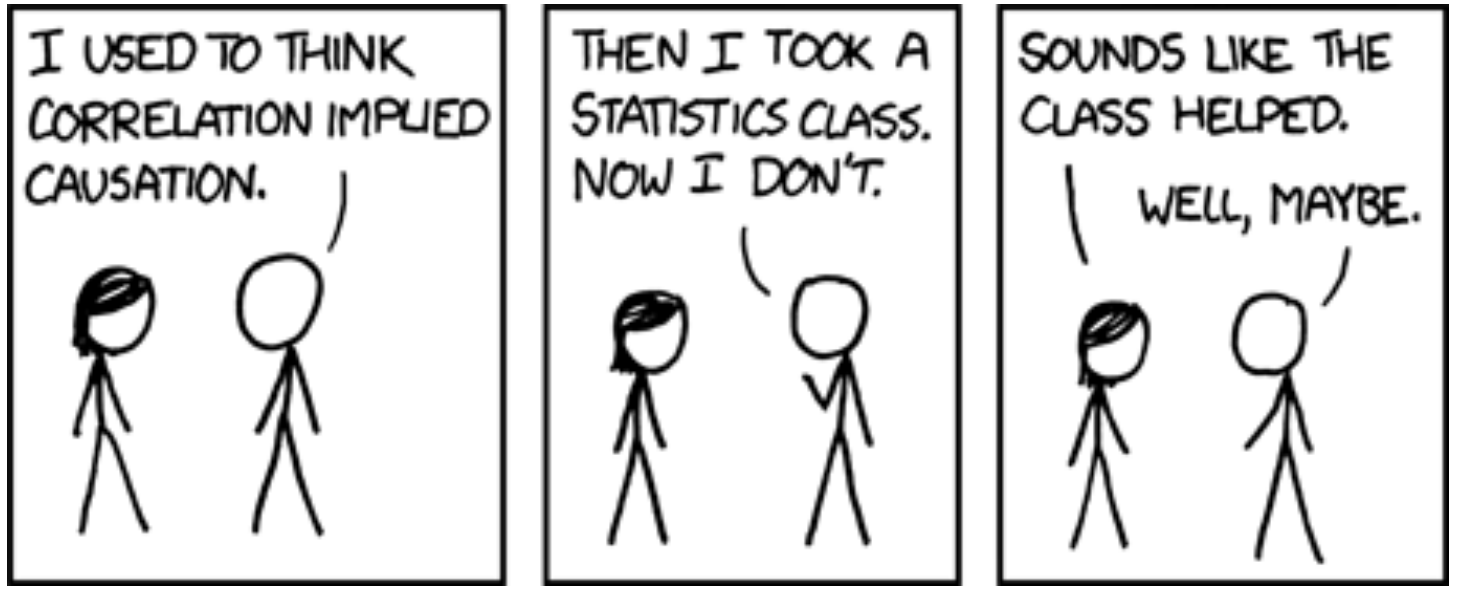
What could possibly go wrong?

- Bad statistics: A basic misunderstanding of measurement theory and what is being measured.
- Bad decisions: The incorrect use of measurement data, leading to unintended side effects.
- Bad incentives: Disregard for the human factors, or how the cultural change of taking measurements will affect people.



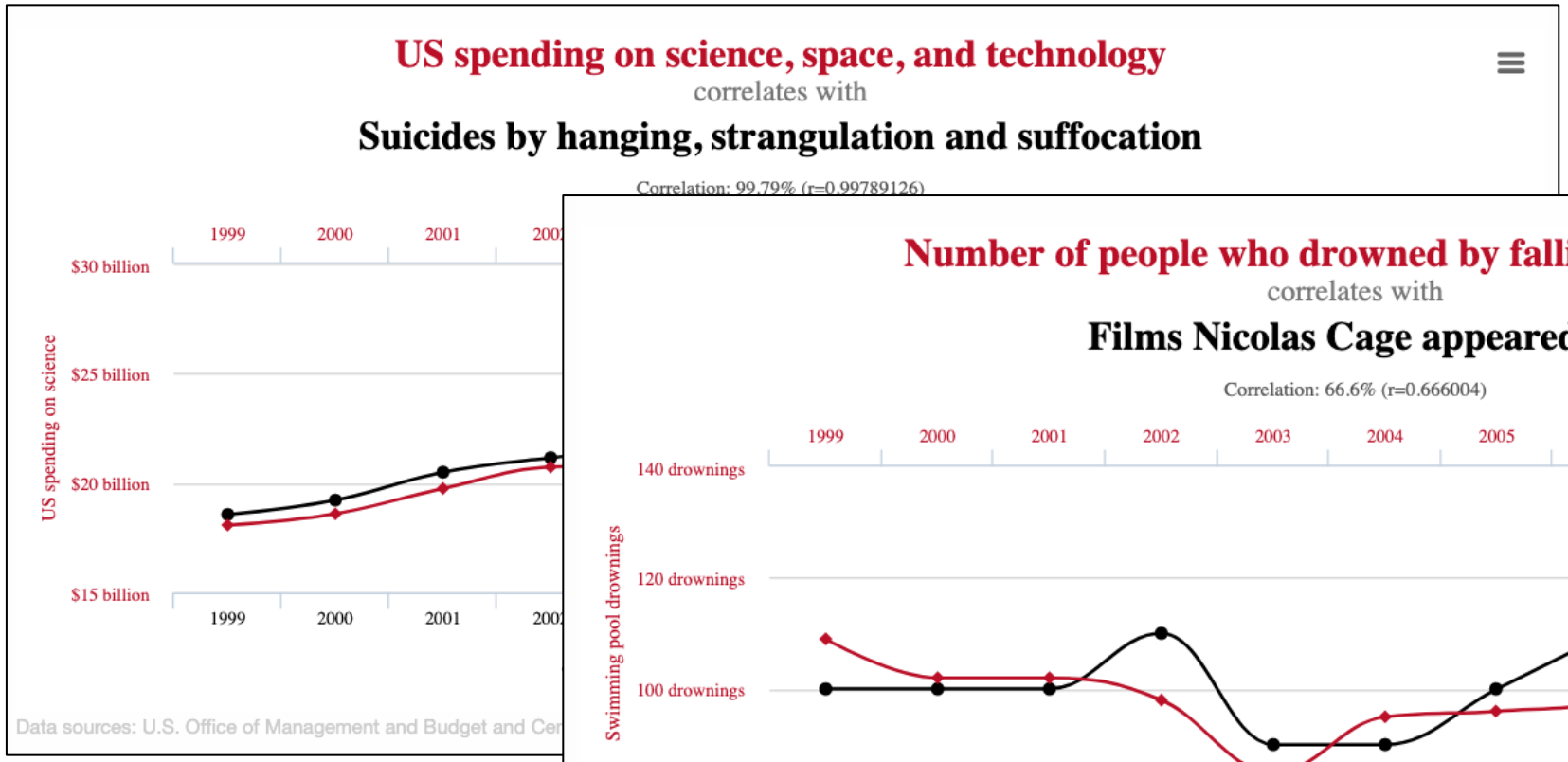
Making inferences

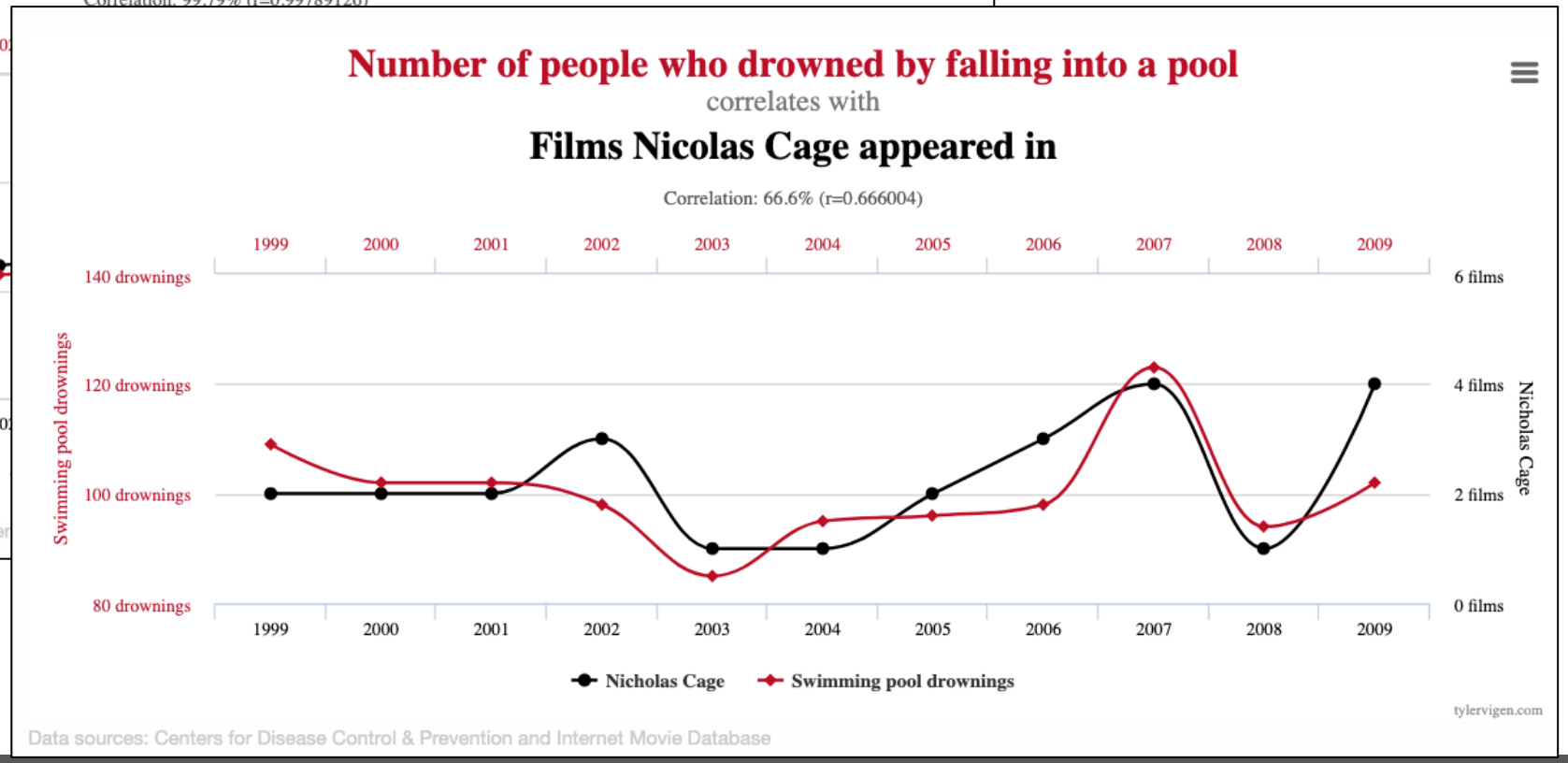
http://xkcd.com/552/



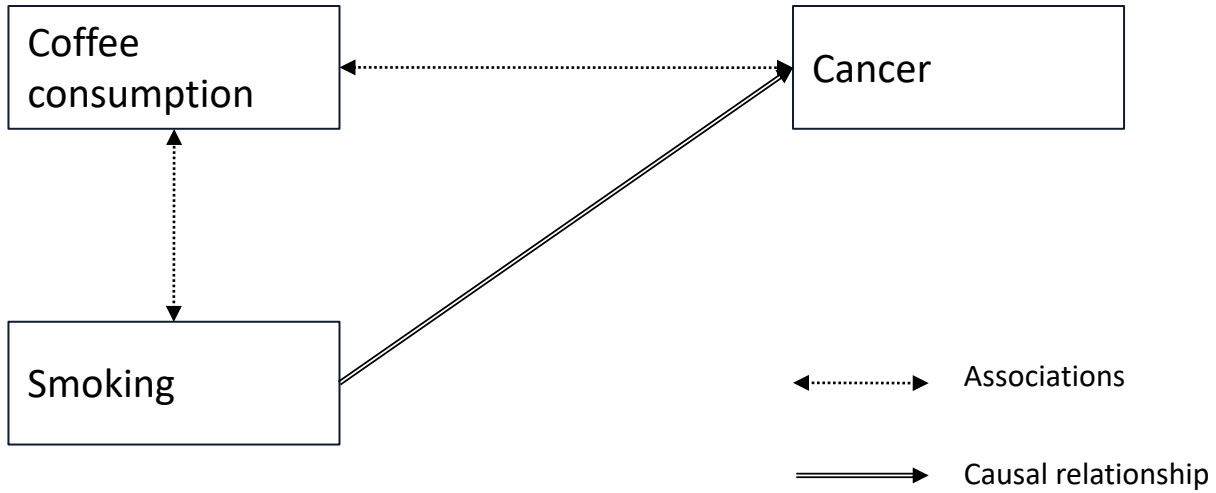
- To infer causation:
  - Provide a theory (from domain knowledge, independent of data)
  - Show correlation
  - Demonstrate ability to predict new cases (replicate/validate)

Spurious Correlations





Confounding variables



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

RESEARCH-ARTICLE

Coverage is not strongly correlated with test suite effectiveness

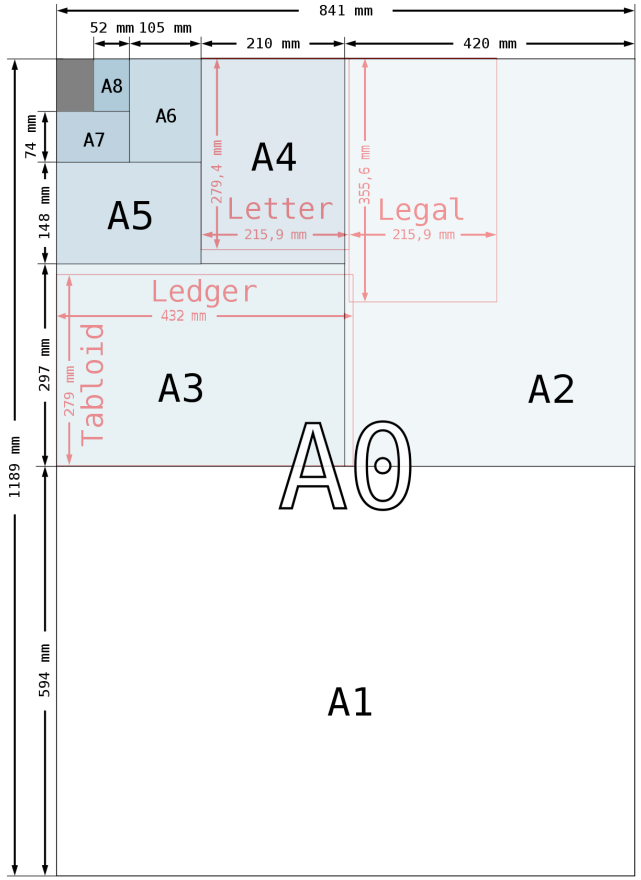
Authors: Laura Inozemtseva, Reid Holmes Authors Info & Affiliations

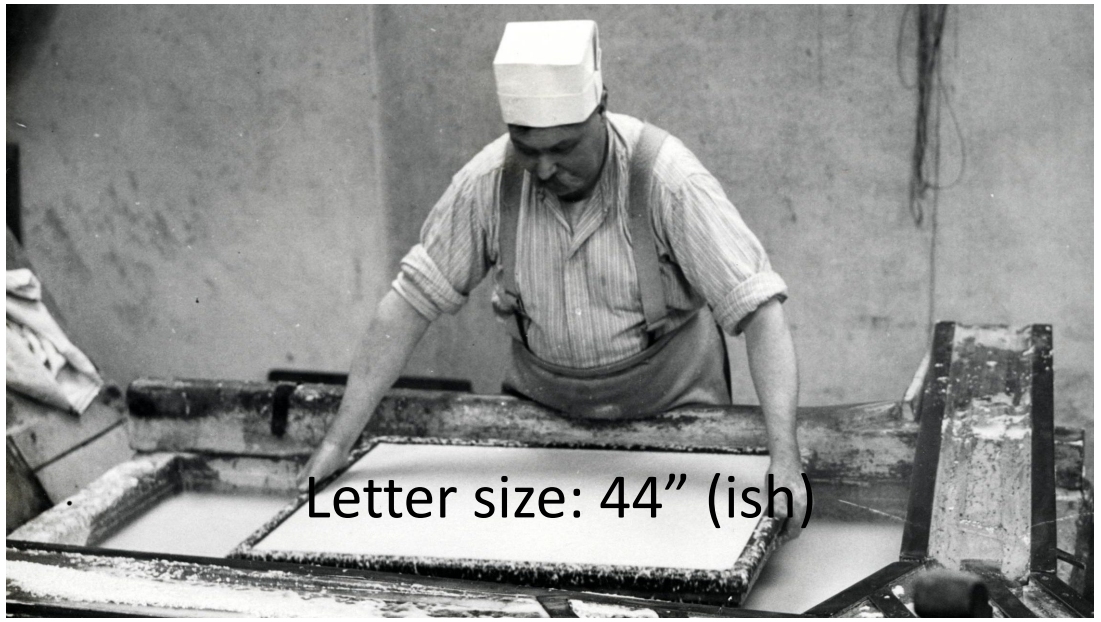
ICSE 2014: Proceedings of the 36th International Conference on Software Engineering • May 2014 • Pages 435–445 • https://doi.org/10.1145/2568225.2568271

"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."

Measurements validity

- Construct validity – Are we measuring what we intended to measure?
- Internal validity – The extent to which the measurement can be used to explain some other characteristic of the entity being measured
- External validity – Concerns the generalization of the findings to contexts and environments, other than the one studied


841 mm
52 mm
105 mm
210 mm
420 mm
A8
A6
A7
A4
A5
74 mm
148 mm
279,4 mm
355,6 mm
Letter
215,9 mm
Legal
215,9 mm
Ledger
432 mm
Tabloid
279 mm
297 mm
A3
A2
A0
1189 mm
594 mm
A1


Letter size: 44” (ish)

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

#1 NATIONAL BESTSELLER

IN RETROSPECT

"Unsparing...a clear, concise and extremely interesting look at a crucial period of U.S. decision making. It deserves to be widely read."
—*Wall Street Journal*

THE TRAGEDY AND LESSONS OF VIETNAM

ROBERT S. McNAMARA

WITH BRIAN VANDEMARK

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.

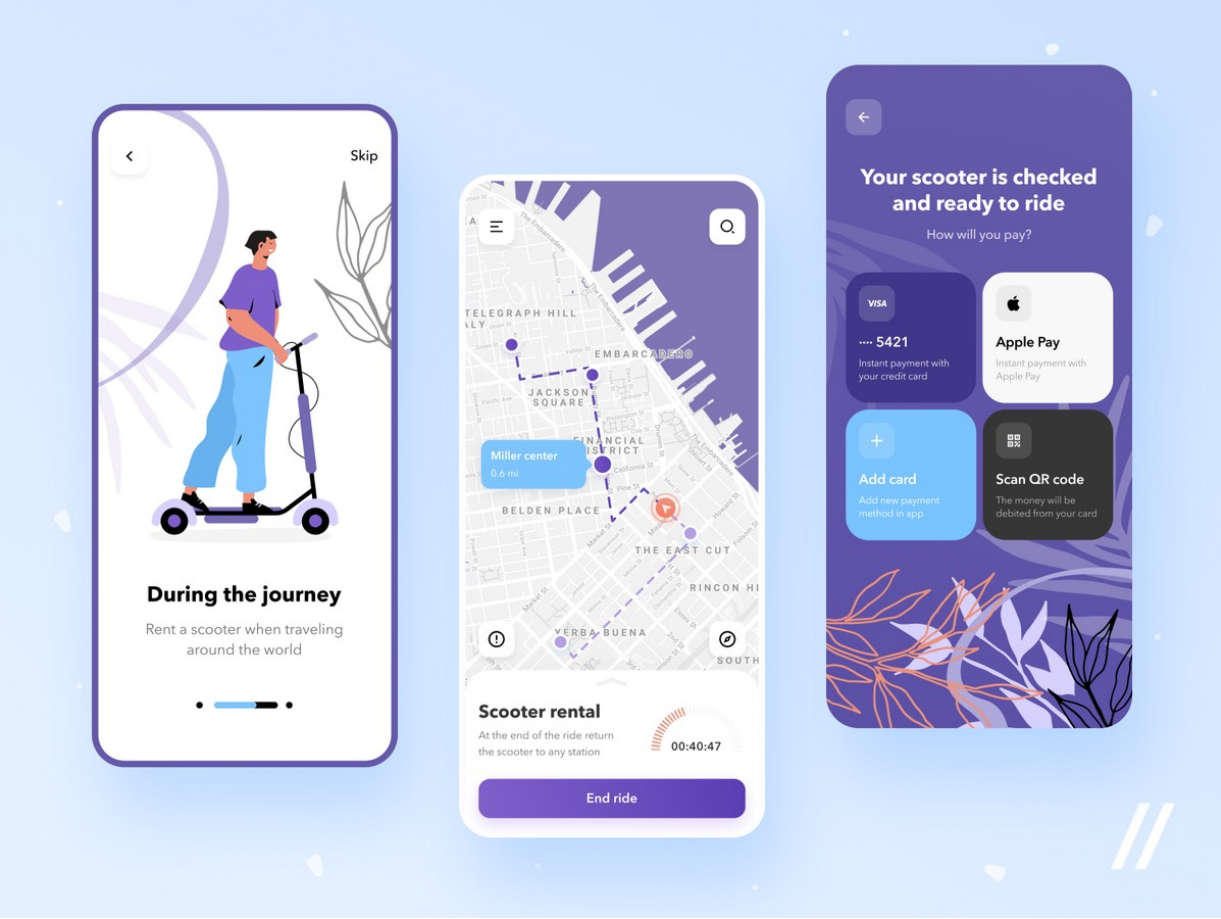
https://chronotopeblog.com/2015/04/04/the-mcnamara-fallacy-and-the-problem-with-numbers-in-education/

The McNamara Fallacy

- There seems to be a general misunderstanding to the effect that a mathematical model cannot be undertaken until every constant and functional relationship is known to high accuracy. This often leads to the omission of admittedly highly significant factors (most of the “intangibles” influences on decisions) because these are unmeasured or unmeasurable. To omit such variables is equivalent to saying that they have zero effect… Probably the only value known to be wrong…
  - J. W. Forrester, Industrial Dynamics, The MIT Press, 1961

Discussion: Measuring usability

- What does it mean?
- How would you measure it?
- Metrics
  - Time to perform task?
  - App load time?
  - Discovering menu options?
- Measurements
  - Amount of documentation
  - Stars on app store
  - Telemetry
  - Surveys, interviews, controlled experiments, expert judgment
  - A/B testing



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/

Simplistic Productivity Metrics

- Lines of code per day?
  - Industry average 10-50 lines/day
  - Debugging + rework ca. 50% of time
- Function/object/application points per month
- Bugs fixed?
- Milestones reached?

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?

Developer Productivity Myths

- Productivity is all about developer activity
- Productivity is only about individual performance
- One productivity metric can tell us everything
- Productivity measures are useful *only* for managers
- Productivity is only about engineering systems and developer tools

Warning

- Most software metrics are controversial
  - Usually only plausibility arguments, rarely rigorously validated
  - Cyclomatic complexity was repeatedly refuted, yet is *still* used
  - “Similar to the attempt of measuring the intelligence of a person in terms of the weight or circumference of the brain”
- Use carefully!
- Code size dominates many metrics
- Avoid claims about human factors (e.g., readability) and quality, unless validated
- Calibrate metrics in project history and other projects
- Metrics can be gamed; you get what you measure

Summary

- Measurement is difficult but important for decision making
- Software metrics are easy to measure but hard to interpret, validity often not established
- Many metrics exist, often composed; pick or design suitable metrics if needed
- Careful in use: monitoring vs incentives
- Strategies beyond metrics

Questions to consider (Projects)

- What properties do we care about and how do we measure them?
- What is being measured? Does it (to what degree) capture the thing you care about? What are its limitations?
- How should it be incorporated into process?
- What are potentially negative side effects or incentives?