# Study Quality: Credibility and Relevance

Lutz Prechelt, Freie Universität Berlin
Course "Empirical Methods in Software Engineering"

- Credibility, Relevance
  - Need for generalization
- Finer-grained (specialized) criteria differ between quantitative and qualitative studies:
  - Quantitative: internal/external/construct/ conclusion validity
  - Qualitative: Tracy's 8 "Big Tent" criteria
- Frequent ways how quality is ruined in studies

# Studienqualität: Glaubwürdigkeit und Relevanz

Lutz Prechelt, Freie Universität Berlin
V+Ü "Empirische Methoden im Software Engineering"

- Glaubwürdigkeit, Relevanz
  - Relevanz braucht meist Verallgemeinerung
- Speziellere Kriterien sind verschieden je nach Methoden:
  - Quantitative: innere/äußere Gültigkeit, Konstruktgültigkeit, Schlussfolgerungs-Gültigkeit
  - Qualitative: Tracys "Big Tent"-Kriterien
- Häufig vorkommende Arten, wie die Qualität beschädigt wird.

Overall quality applies to the conclusions from a study:

- **Credibility**
  - How trustworthy are the conclusions?
- **Relevance**
  - How interested are we in these conclusions?
    How beneficial is it to have them?

# Fictional example

Assume the following experiment:

- **Task T** consists of subsequent subtasks T1, T2, …
  - task types: build, extend, modify, analyze
- **Group C** has 18 people.
  Each of them solves the tasks using C++
- **Group J** has 18 different people.
  Each of them solves the tasks using Java.

- The experiment observes several **outcomes** (for each task):
  - work time
  - #defects in the product
  - properties of the source code
    - #LOC, #functions, identifiers, …
  - #defects removed underway
  - reliability (via a standard test suite)
  - execution speed, memory consumption

- Each non-random difference of one outcome between C and J is an ***effect*** found by the experiment

# Results vs. conclusions: The need to generalize



- Some studies are existence proofs
  - explain a phenomenon for the first time
- All other studies need to generalize to be useful
  - e.g. our experiment found effect E by investigating one particular setting S
    - e.g. exactly these 36 people doing exactly this task sequence T
  - The community may be interested in the effect E, but nobody will ever have the same setting S again
  - so we need to generalize from the results seen in order to get useful conclusions.

# Results vs. conclusions: The need to generalize (2)

Freie Universität Berlin

- Bad studies do not conclude anything, they just report
  - *"Java had effect E"* past tense: talks about given data only
- or they conclude without generalizing
  - *"Java can have effect E"*

Generalization always increases relevance at the cost of lower credibility

- Good studies must provide reasonable guesses for generalization:
  - *"For isolated, simple, module-level programming tasks of algorithmic types that center on string handling, done by junior SW engineers, Java appears to produce effect E."*
- Not unreasonable ones:
  - *"Java has effect E"* present tense: talks about generalizations

# Finer-grained notions of quality

- Credibility and relevance apply to **all kinds** of empirical studies
  - They are holistic and somewhat vague
- Finer-grained quality criteria differ between quantitative vs. qualitative studies.

- The literature on **quantitative** studies usually talks about a variety of types of "validity"
  - Their specific definitions vary somewhat
  - Positivist culture: There is a reality that is knowable and understandable objectively
- The literature on **qualitative** studies has no uniform top quality construct
  - We will use one source that suggests 8 criteria which most qualitative camps can presumably agree with
  - Interpretivist culture: Relevant empirical work unavoidably involves subjectivity

# Quantitative studies: "Validity"

- Many different definitions exist
  - with modest or larger differences
- Validity has many aspects
  - and many aspect lists exist,
    with modest or larger differences
- Definitions and aspect lists are often geared towards a particular method
  - e.g. controlled experiments or questionnaire surveys
- We use a definition and aspect list geared towards software engineering:

- ***Validity*** *is the degree to which a study makes adequate statements about software engineering reality.*

# **Excursion**: The scientific method

- Since Galilei, physics and other sciences work according to this model:
  - Formulate a theory T about how (some aspect of) the real world behaves
  - Design and conduct experiments X for testing this theory
- Is <u>accepted</u> in all subjects where experimentation is possible
  - Natural sciences: Physics, chemistry, biology, medicine etc.
  - Engineering
  - Parts of many social sciences (economics, sociology, etc.)

- The scientific method is <u>unsuitable</u> where experiments cannot be performed
  - for technical or ethical reasons
  - ersatz: observation + induction/abduction (qualitative & quantitative arguments)
- It is <u>difficult</u> when there are too many factors involved to formulate a precise theory, e.g.
  - social sciences, socio-technical systems
  - **software engineering**
  - ersatz: observation + induction/abduction (qualitative research methods)

# Excursion: The scientific method (2)
# Scientific vs. unscientific theories, pre-theoretical empiricism

Freie Universität Berlin

- T is called a *scientific theory* only if it predicts something specifically and hence can be tested by experiment X
  - Even if T is wrong, it may happen that the results of X are as expected
  - But if X *contradicts* predictions of T, then T must be false
- called "Critical Rationalism", suggested ~1934 by Karl Popper (1904–1994)
    - The prevalent scientific paradigm today
  - Theories T cannot be directly confirmed, only refuted
  - If T cannot be refuted for a long time, it will gradually be accepted as **valid**
    - e.g.: special theory of relativity

- In software engineering, often too little is known for formulating a non-trivial, plausible, testable, general theory
- A different, qualitative style of empiricism is useful then:
  - Observe things that lead to hypotheses from which one could build theories
  - Often these observations have to be qualitative rather than quantitative in order to be useful
  - Eventual quantitative theories will have to be probabilistic

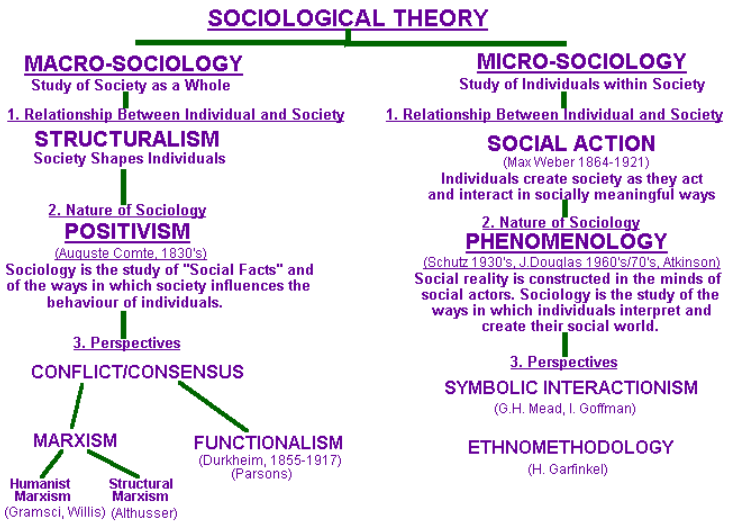
https://www.youtube.com/watch?v=-X8Xfl0JdTQ "Popper, science, and pseudoscience" 9 min.
https://www.timetoast.com/timelines/sr-karl-raimund-popper

# Excursion: The scientific method (3)
# Hard science vs. soft science, physics envy

- People with positivist stance often claim a subject is science only if it produces *quantitative* and *reliable* theories
  - "hard science", such as physics formulas

- they hence claim that subjects involving human behavior are not scientific
  - "soft science"
    - This attitude amounts to "physics envy".

- This is not true: The scientific method can be applied
  - but the theories will be more complex, make weaker predictions, and **require different research methods**.


- Hard science is simpler than soft science
  - That is why it is farther advanced

**End of excursion.**

# Quantitative studies: Some important aspects of **validity**

Freie Universität Berlin

- **Internal** validity:
  - talks about the data observed
  - The degree to which the observations are undistorted by factors not captured by the study
  - e.g. experimenter effect: Experimenter is such a glowing C++ fan that group C has higher motivation
- **Construct** validity:
  - talks about the interpretation of the data within the study
  - The degree to which the study properly characterizes what it claims (or implies) to characterize
  - e.g. "defects" are only those detected by the rather simplistic test suite
- **External** validity:
  - talks about the representativity of the data beyond the study
  - Degree to which the observation setting is typical for large relevant groups of settings
  - e.g. tasks are arbitrary and unnatural
- **Conclusion** validity:
  - talks about the conclusions drawn from the data, given the internal/construct/ external validity
  - Degree to which the generalizations made appear warranted

These four aspects do not overlap.

# Construct validity:
# Bad operationalization (ambiguous proxies)



- All observations are made by some observation procedure
  - the "**operationalization**"
  - e.g. work time by a clock
  - e.g. reliability by a test suite
  - e.g. #defects by test suite + debugging
- This procedure is almost always imperfect
  - e.g. work time: What if a subject goes to the loo during the experiment? (etc. pp.)
  - e.g. reliability: Is the test suite reflecting practical use well? (Unlikely!)
  - e.g. #defects: Is the test suite thorough enough? Is the debugging canonical? Reliable?
- A given operationalization may *often* measure what we want
  - but will sometimes measure something else
    - in some cases more often than not
- It will therefore be ambiguous as a proxy of the concept our study wants to observe

(We will revisit operationalization below.)

proxy: Stellvertreter

# Conclusion validity: Correlation vs. causation

Freie Universität Berlin

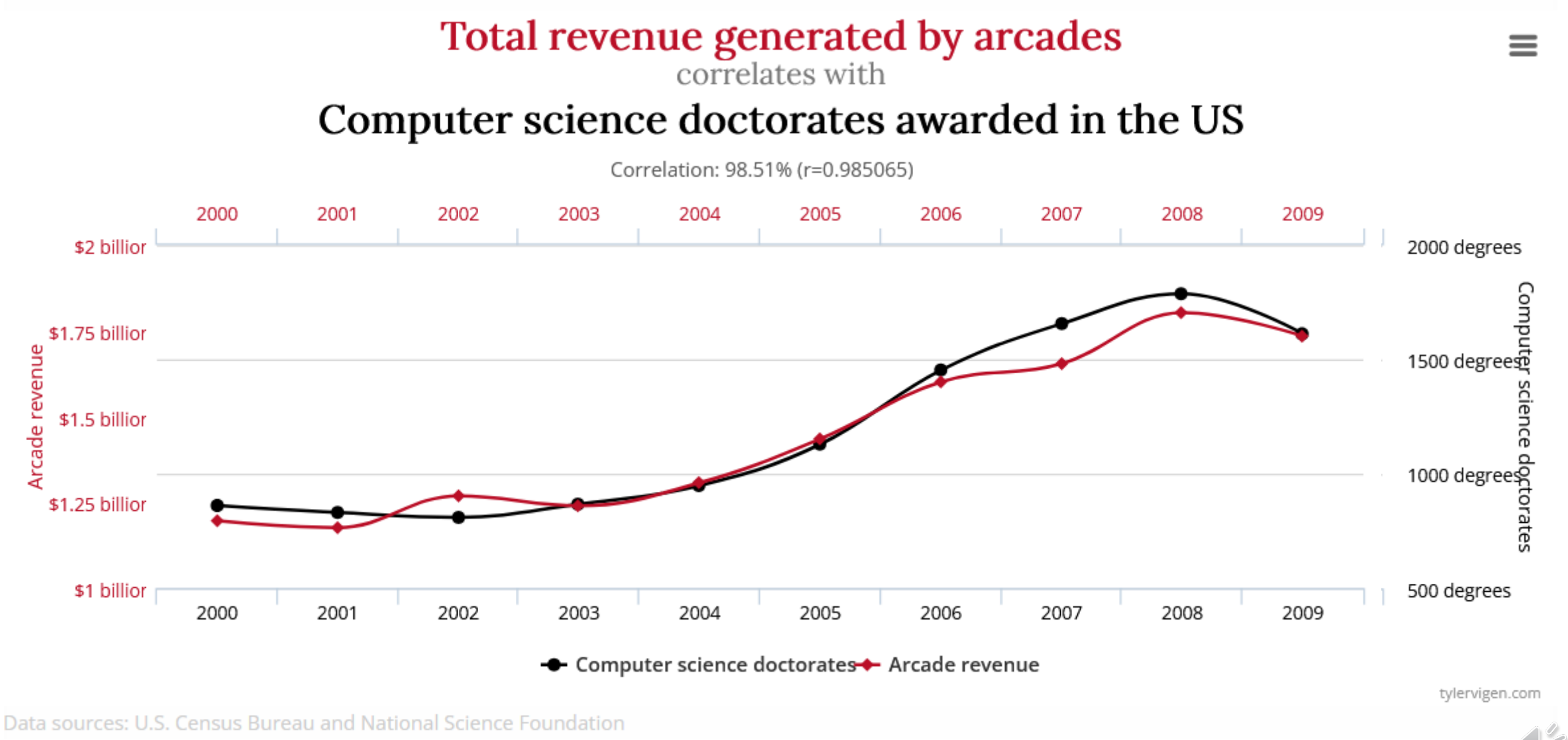
- Causational conclusions have high relevance:
  - *"Factor A causes outcome B"*
- So most studies will be interested in causation.
- But no method can demonstrate causation if human beings are involved
  - and only controlled experiments even come close:
  - *"In our setting, C++ caused more defects, compared to Java"*
- Most studies only demonstrate correlations:
  - When A was higher, B tended to be higher, but we don't know why

Beware of formulations for correlations!:

- OK:
  - *"team size correlates positively with the size of the codebase"*
  - *"the larger teams tended to have or produce the larger codebases"*
- Highly problematic:
  - *"larger teams come with larger code bases"*
  - *"a larger code base goes along with a larger team"*
- Not OK:
  - *"larger team leads to larger codebase"*

There are more ways how conclusions can go wrong, e.g. overgeneralization (see below)

# Correlations are worth reporting only if some causation appears likely




http://www.tylervigen.com/spurious-correlations

arcade: Spielhalle

# Criteria for qualitative studies

- Sarah Tracy: "Qualitative Quality: Eight "Big-Tent" Criteria for Excellent Qualitative Research",
  Qualitative Inquiry 16(10), 2010

- Several schools of qualitative methods formulate their own sets of quality criteria
  - with many overlaps,
    but also many differences in details and even different sets of principles
- This article attempts an impartial unification of those views
- (The result is also applicable to quantitative methods
  - and it is an interesting exercise to do so).

# Criteria for qualitative studies: Tracy's 8 criteria

Freie Universität Berlin

1. **Worthy topic**
   - aims for Relevance
2. **Rich rigor**
   - uses sufficient theoretical constructs, data, and analysis methods
3. **Sincerity** (transparency)
   - about biases, methods, challenges
4. **Credibility**
   - thick description, concrete detail, explication of tacit (non-textual) knowledge, showing rather than telling, triangulation, multivocality, member reflections
5. **Resonance**
   - moves readers through evocative representation, naturalistic generalization, and transferable findings
6. **Significant contribution**
   - achieves Relevance
7. **Ethical**
8. **Meaningful coherence**
   - achieves (parts of) stated goals, uses appropriate methods (~construct validity), connects all parts (including literature) properly

# Frequent patterns for low study quality in Software Engineering, or: How to ruin your study

# (1) Modest credibility problem: Violating an ordinal scale

Freie Universität Berlin

Data come on different scales that support different operations:

- Nominal scale [weakest]
  - e.g. C, C++, Java, Python, Ruby
  - supports only counting
- Ordinal scale
  - e.g. strongly disagree, disagree, neutral, agree, strongly agree
  - supports comparison: <, =, >
- Difference scale
  - e.g. Celsius temperature, calendar years
  - fixed unit, but arbitrary zero-point
  - supports differences
  - "10° C is twice as warm as 5° C" is nonsense
- Ratio scale
  - e.g. size, work time
  - fixed unit, canonical zero-point
  - supports addition and division
  - "2h is twice as long as 1h"
- Absolute scale [strongest]
  - e.g. number of classes
  - canonical unit, canonical zero-point
  - supports addition and division without regard to measurement unit

**Frequent mistake:**

- Computing means (and even ratios) on ordinal scales:
  - "Mean satisfaction for A is 3.6, 20% higher than for B"

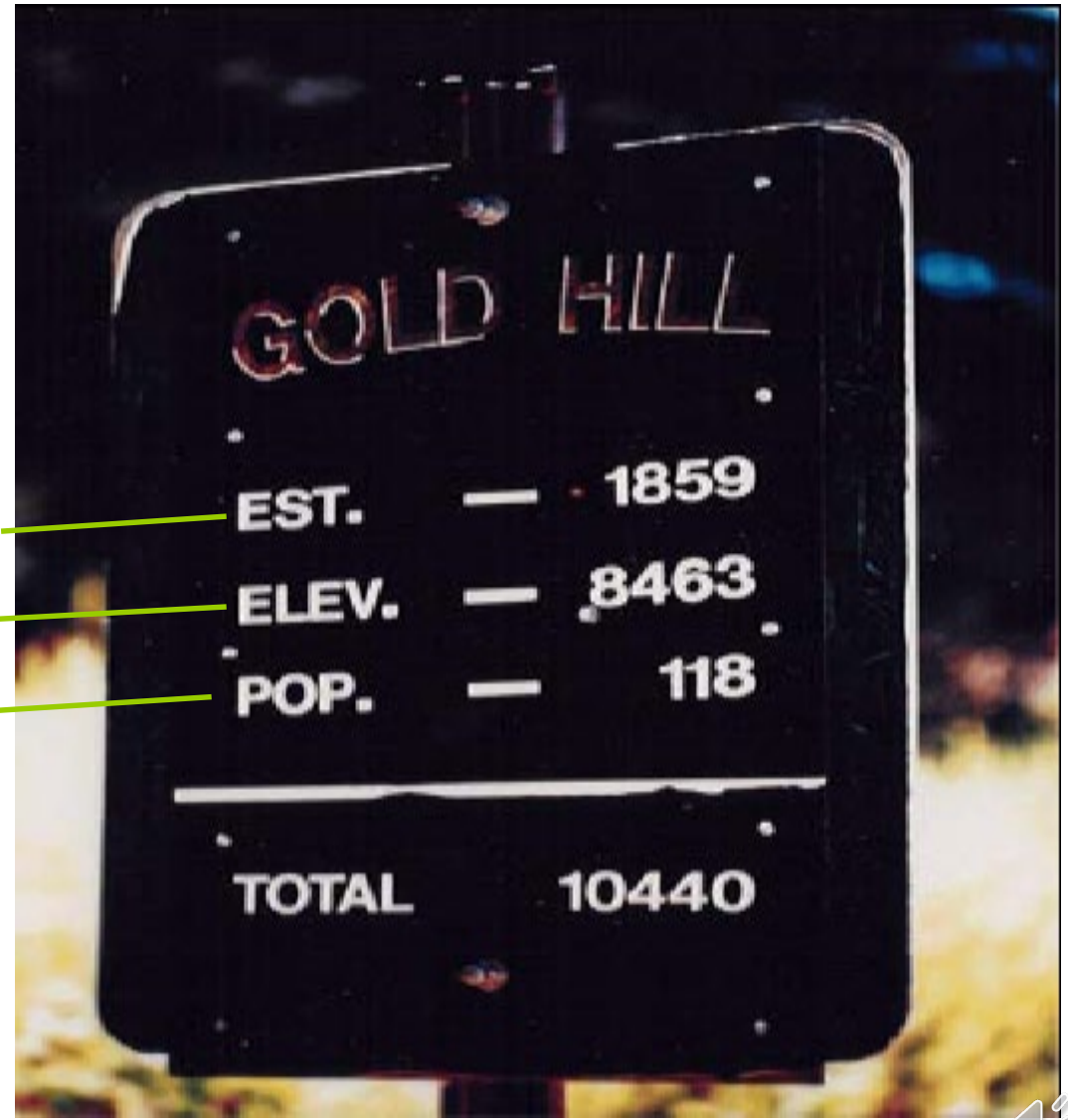
a little bad

bad!

# (1)
# Exercise: Which scale is this?

Freie Universität Berlin

| | |
|---|---|
| Difference scale | Established |
| Difference or ratio scale | Elevation |
| Absolute scale | Population |



# (2) Modest credibility problem: Failing to visualize



- If key results data is presented in large tables, the results become less "anschaulich" and hence less credible

- This article gets it almost right, but could have gone badly wrong:
  - should have used color and column order 1, 2, 4, 3, 5
  - N. Tsantalis et al.: "Accurate and Efficient Refactoring Detection in Commit History", ICSE 2018
    - The table compares RMiner (this article) to RefDiff (older work) separately on 15 types of refactoring.

| Refactoring Type | RMiner | | RefDiff | |
|---|---|---|---|---|
| | Precision | Recall | Precision | Recall |
| Inline Method | 98.96 | 86.36 | 84.35 | 88.18 |
| Extract Method | 98.63 | 84.72 | 93.03 | 90.95 |
| Move Field | 88.42 | 95.45 | 30.19 | 45.45 |
| Move Class | 100 | 96.24 | 99.90 | 93.53 |
| Extract Interface | 100 | 100 | 76.92 | 55.56 |
| Push Down Method | 100 | 100 | 95.00 | 61.29 |
| Push Down Field | 100 | 86.21 | 100 | 100 |
| Change Package | 85.00 | 100 | N/A | N/A |
| Pull Up Method | 100 | 90.48 | 80.60 | 85.71 |
| Pull Up Field | 100 | 96.30 | 64.00 | 59.26 |
| Move Method | 95.17 | 76.36 | 32.25 | 92.25 |
| Rename Method | 97.78 | 83.28 | 85.54 | 89.59 |
| Extract Superclass | 95.08 | 100 | 100 | 18.97 |
| Rename Class | 98.33 | 71.08 | 89.71 | 73.49 |
| Extract & Move Method | 95.92 | 41.23 | 73.02 | 80.70 |
| Overall | 97.96 | 87.20 | 75.71 | 85.76 |

# (3) Credibility problem:
# Broken assumption "developers will do The Right Thing™"



Assuming developers will never use a tool's output in a harmful manner.

- Fictious example:
  - A tool analyzes source code to point out various classes of potential defects. Precision is shown to be 50%

  Typical assumption:
  - User will not break correct code by "fixing" a defect that is in fact no defect.

# (4) Medium or big credibility problem: Broken operationalization



- If we want to observe attribute X, this can rarely be done perfectly.
  - We usually need to approximate.
- **Operationalization** is the step from the abstract idea X to the actual measurement
- Three approximation steps:
  - AM: idea to measure
  - AP: measure to measurement procedure
  - AD: measurement procedure to data

A given operationalization may be too bad to be credible

- Theoretical example:
  - Attribute: Debugging effort for a given defect D
    - made and found by programmer P
  - Measure: Work time invested from recognizing D until D is resolved
    - AM: What about breaks?
  - Procedure: P keeps a time log
    - AP: This will make debugging a much more conscious work step. Impact?
    - AP: Deciding "D is resolved" is difficult
  - Data: Real-life time logs
    - AD: Many episodes will be missing
    - AD: Many will be incomplete (no end)
    - AD: Many will include intrusions from other activities etc. etc.

# (4) Big credibility problem: Broken operationalization: Real example of a fundamental problem

Freie Universität Berlin

- AlDAbd18: "Empirical Evaluation of the Impact of OO Code Refactoring on Quality Attributes: A SLR"
  - based on **76 studies**, many using multiple datasets

| | Quality attribute | ++ | + | Insign. 0 | 0 | - | – |
|---|---|---|---|---|---|---|---|
| Estimated External | Maintainability | 0 | 39 | 0 | 13 | 41 | 0 |
| | Understandability | 0 | 30 | 0 | 13 | 40 | 0 |
| | Reusability | 0 | 35 | 0 | 13 | 40 | 0 |
| | Adaptability | 0 | 25 | 0 | 13 | 40 | 0 |
| | Flexibility | 0 | 9 | 0 | 0 | 1 | 0 |
| | Testability | 0 | 31 | 0 | 12 | 35 | 0 |
| | Extensibility | 0 | 7 | 0 | 2 | 0 | 0 |
| | Effectiveness | 0 | 7 | 0 | 0 | 0 | 0 |
| Measured External | Reliability | 1 | 8 | 4 | 0 | 0 | 1 |
| | Maintainability | 6 | 0 | 4 | 0 | 0 | 24 |
| Internal | Coupling | 3 | 108 | 18 | 129 | 146 | 21 |
| | Cohesion | 52 | 201 | 54 | 48 | 72 | 0 |
| | Complexity | 3 | 89 | 1 | 69 | 112 | 13 |
| | Size | 1 | 42 | 7 | 17 | 42 | 4 |
| | Inheritance | 0 | 30 | 3 | 100 | 13 | 0 |
| | Composition | 0 | 2 | 0 | 3 | 2 | 0 |
| | Data encapsulation | 0 | 2 | 0 | 3 | 3 | 0 |
| | Polymorphism | 0 | 1 | 0 | 1 | 5 | 0 |

- These studies consider the refactoring *steps* in isolation
  - although refactoring's purpose is usually to simplify other *changes*
    - not the program as such
  - Proper operationalization needs to evaluate the alternative reality in which other changes are made without the prior refactorings.
    - Super difficult!
    - A big fat AM problem.

worse??

# (5) Big credibility problem:
# Broken tacit assumption "ROI will easily be achieved"



Tool evaluations will often show benefits while ignoring the cost to get them.

- Fictious example (continued):
  - A tool analyzes source code to point out various classes of potential defects.
    Precision is shown to be 50%

  Typical assumptions:
  - Each of these defects is worth analyzing and understanding
  - The effort for recognizing the false positives to be false is not a problem

  (Automated repair has an even more complex cost/benefit situation.)

ROI: return-on-invest (amortization)

# (6) Big credibility problem: Overgeneralizing

Freie Universität Berlin

Broken tradeoff between credibility and relevance.

Semi-real example:

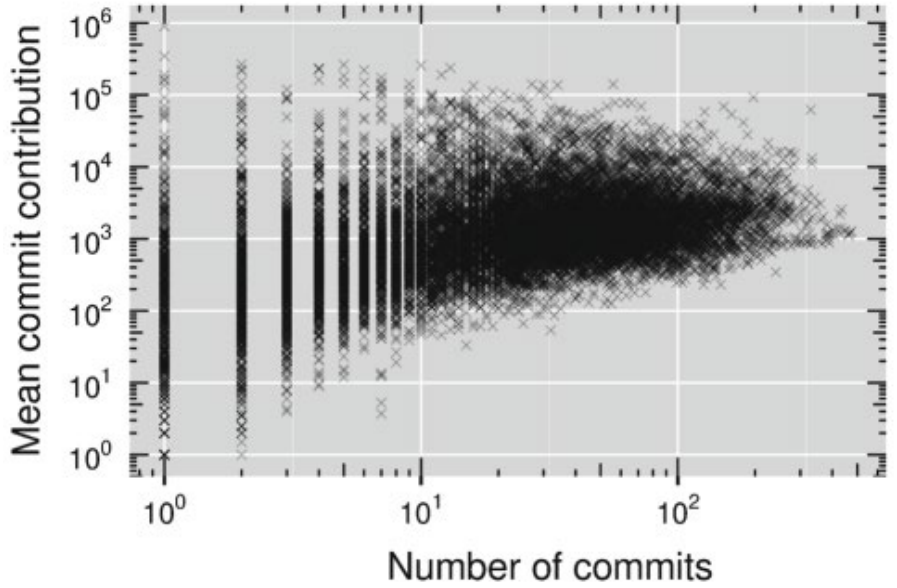
- Facts (taken from a real example):
  - 42 student subjects from University U; 2 pairs of toy programs of ~300 LOC;
    compare program variants with/without design pattern;
    measure time to finish an extension task correctly.
    Finished 16% faster (p = 0.03) with (vs. without) Observer pattern.
    Finished 29% faster (p = 0.005) with (vs. without) Decorator pattern

- Acceptable conclusion:
  - For subjects with similar background as ours, using the Observer or Decorator patterns can help finish program extension tasks faster – at least for small and clean programs.
- Botched conclusion (fictious example):
  - Programs using design patterns are 16% to 29% faster to maintain than equivalent programs that do not use design patterns.

# (7) Big relevance problem:
# No explanation of usefulness [confuses engineering with science]

Freie Universität Berlin

[SchMavSch16]: "From Aristotle to Ringelmann: a large-scale analysis of team productivity and coordination in Open Source Software projects", EMSE 2016




- Looks at absolute productivity of 49 Open Source projects in relation to their changing team size over time.
  - Very complicated analysis, lots of good approaches in many respects
- Finds that absolute productivity increases sublinearly.

The evidence is broken:
Uses commit size as productivity measure and **assumes** that the expected commit size does not depend on developer, which is ridiculous for OSS.

- Does not explain how this knowledge is relevant
  - Usefulness of the result is not obvious, whether directly or indirectly.
  - It is probably not useful. Why not?

# Summary

- *Generic* quality criteria for an empirical study: credibility and relevance
  - Relevance usually requires generalization -- which is often missing
- Finer-grained (specialized) criteria differ between quantitative and qualitative studies:
- Quantitative:
  - Validity: internal, external, construct, conclusion
- Qualitative:
  - worthy topic, rich rigor, sincerity, resonance etc.
- Some common patterns of damaging quality are:
  - Violating scales
  - Lack of visualization
  - Broken assumptions regarding effort, tool usefulness, or operationalizations
  - Overgeneralization
  - Not explaining usefulness
- Very often the problem is a broken assumption

# Thank you!