CS 520/620
Advanced Software Engineering
Fall 2016

October 13, 2016
Recap

● Overview of the scientific method
● Basics of statistical analysis
  ○ Why do we need statistical analysis?
  ○ Why is this particularly important for randomized algorithms?
  ○ What’s the difference between statistically significant and practically significant?
  ○ What’s the difference between parametric and non-parametric statistics?
  ○ What is internal, external, and construct validity?
Recap: the scientific method

- Question
- Observations
- Hypothesis
- Experiment
- Predictions
Recap: the scientific method

Question → Observations → Hypothesis

Experiment

Predictions

Data collection and analysis!

Repeatable

Falsifiable
Recap: basics of statistical analyses

Example: Number of faults detected by two techniques A and B
A: 5 3 6 3 3    avg = 4.0
B: 0 0 0 0 22    avg = 4.4

- Why do we need statistical analysis?
- Why is this particularly important for randomized algorithms?
- What’s the difference between statistically significant and practically significant?
- What’s the difference between parametric and non-parametric statistics?
- What is internal, external, and construct validity?
Recap: basics of statistical analyses

Example: Number of faults detected by two techniques A and B
A: 5 3 6 3 3 avg = 4.0
B: 0 0 0 0 22 avg = 4.4

- Why do we need statistical analysis?
- Why is this particularly important for randomized algorithms?
- What’s the difference between statistically significant and practically significant?
- What’s the difference between parametric and non-parametric statistics?
- What is internal, external, and construct validity?
Recap: basics of statistical analyses

Example: Number of faults detected by two techniques A and B
A:  5  3  6  3  3  avg = 4.0  
B:  0  0  0  0  22 avg = 4.4

- Why do we need statistical analysis?
- Why is this particularly important for randomized algorithms?
- What’s the difference between statistically significant and practically significant?
- What’s the difference between parametric and non-parametric statistics?
- What is internal, external, and construct validity?
Recap: basics of statistical analyses

Example: Number of faults detected by two techniques A and B

<table>
<thead>
<tr>
<th>A</th>
<th>5</th>
<th>3</th>
<th>6</th>
<th>3</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>22</td>
</tr>
</tbody>
</table>

- Why do we need statistical analysis?
- Why is this particularly important for randomized algorithms?
- What’s the difference between statistically significant and practically significant?
- What’s the difference between parametric and non-parametric statistics?
- What is internal, external, and construct validity?
External and construct validity

External validity
How well does the experimental design generalize to the real world (i.e., other populations, situations, etc.)?

Construct validity
How well does the experimental design measure what it is supposed to measure? Does it use the right metrics and collect the right measurements?
Internal validity

How well does the experimental design isolate the effect/variables that it studies (i.e., control for confounds)?

Classic example
Throughout the year, murder rates and ice cream sales are highly positively correlated. Possible explanations?
Internal validity

How well does the experimental design isolate the effect/variables that it studies (i.e., control for confounds)?

Classic example
Throughout the year, murder rates and ice cream sales are highly positively correlated. Possible explanations?
- Possibilities:
  - Resurrected zombies primarily feed off ice cream
Internal validity

How well does the experimental design isolate the effect/variables that it studies (i.e., control for confounds)?

Classic example
Throughout the year, murder rates and ice cream sales are highly positively correlated. Possible explanations?

● Possibilities:
  ● Resurrected zombies primarily feed off ice cream
  ● Excessive ice cream consumption makes others jealous
Internal validity

How well does the experimental design isolate the effect/variables that it studies (i.e., control for confounds)?

Classic example
Throughout the year, murder rates and ice cream sales are highly positively correlated. Possible explanations?

- Possibilities:
  - Resurrected zombies primarily feed off ice cream
  - Excessive ice cream consumption makes others jealous

Actually, the weather is a non-controlled confound!
Threats to validity: example experiment

Research question:
Does coffee consumption improve code quality?

Methodology
- I program on project 1: Mondays with coffee.
- I program on project 2: Fridays without coffee.
- Measure code quality in number of defects I encounter.
- Measure coffee consumption in dollars spent on coffee beans, as listed on my grocery-shopping receipt.
Threats to validity: example experiment

Research question:
Does coffee consumption improve code quality?

Methodology
● I program on project 1: Mondays with coffee.
● I program on project 2: Fridays without coffee.
● Measure code quality in number of defects I encounter.
● Measure coffee consumption in dollars spent on coffee beans, as listed on my grocery-shopping receipt.

What are threats to construct, internal, and external validity?
Today

Introduction to software testing

- Structural code coverage
  - Statement coverage
  - Decision coverage
  - Condition coverage

- Mutation testing (if time permits)
Software testing

Software testing can show the **presence of defects**, but never show their absence! (Edsger W. Dijkstra)

- A good test is one that fails because of a defect.
- When should we stop testing if no (new) test fails?
Software testing

Software testing can show the **presence of defects**, but never show their absence! (Edsger W. Dijkstra)

- A good test is one that fails because of a defect.
- When should we stop testing if no (new) test fails?

Test adequacy metric

- The best test-adequacy metric is: *ratio of detected defects*
- Problem: the set of defects is unknown
- Solution: use a proxy metric
Structural code coverage: live example

```java
Entry point

a == null || a.length == 0

sum = 0
i = 0

i < a.length

num = a[i]

num < 0

sum += num

++i

return sum / a.length

throw new IllegalArgumentException("Array a must not be null or empty!")

Exceptional exit

Normal exit
```
Structural code coverage: live example

See code examples (online)
Statement coverage

- **Every statement** in the program must be **executed at least once**
- Given the CFG, this is equivalent to node coverage
Structural code coverage: statement coverage

Entry point

- `a==null || a.length==0` → `throw new IllegalArgumentException("Array a must not be null or empty!")` (Exceptional exit)

- `sum = 0` → `i = 0`

- `i<a.length` → `return sum/a.length` (Normal exit)

- `true` → `num = a[i]`

- `false` → `sum += num`

- `true` → `++i`

- `false` → `num < 0` → `sum -= num`
Condition coverage vs. decision coverage

Terminology

- **Condition**: a boolean expression that cannot be decomposed into simpler boolean expressions.

- **Decision**: a boolean expression that is composed of conditions, using 0 or more logical connectors (a decision with 0 logical connectors is a condition).

- **Example**: if (a && b) { … }
  - $a$ and $b$ are *conditions*.
  - The boolean expression $a \&\& b$ is a *decision*. 
Decision coverage

- **Every decision** in the program must take on **all possible outcomes** (*true/false*) **at least once**
- Given the CFG, this is equivalent to **edge coverage**
- **Example:** \((a > 0 \& b > 0)\)
  - a=1, b=1
  - a=0, b=0
Structural code coverage: decision coverage

```java
Entry point

a==null || a.length==0

sum = 0
i = 0

i < a.length

num = a[i]

num < 0

num += num
sum -= num
++i

true
false
false
true
false
true

return sum/a.length

throw new IllegalArgumentException("Array a must not be null or empty!")

Normal exit

Exceptional exit
```
Condition coverage

- **Every condition** in the program must take on all possible outcomes (true/false) **at least once**
- **Example:** $(a>0 \ & \ b>0)$
  - $a=1, b=0$
  - $a=0, b=1$
Structural code coverage: condition coverage

Entry point

a==null || a.length==0
false

sum = 0
i = 0

i<a.length
false

i = a.length
true

num = a[i]

num < 0
false
sum += num
true
sum -= num
++i

return sum/a.length

throw new IllegalArgumentException("Array a must not be null or empty!")

Exceptional exit

Normal exit
Structural code coverage: subsumption

Given two coverage criteria A and B, A subsumes B iff satisfying A implies satisfying B

- Subsumption relationships:
  - Decision coverage subsumes statement coverage
  - Decision coverage does not subsume condition coverage
  - Condition coverage does not subsume decision coverage