CS 376: Methods Exercises

For each exercise, choose the appropriate analysis approach (e.g., paired or unpaired t-test, ANOVA or repeated measures ANOVA, chi square) and execute it. I recommend using online calculators, though learning R increases your street cred as a data scientist.

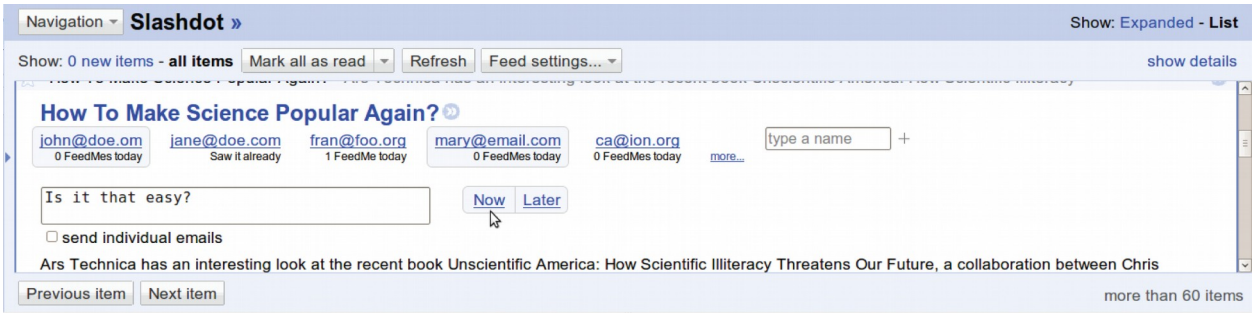
First, download all data at: https://hcicourses.stanford.edu/cs376/2018/slides/methods.zip

Second, if you'd like them, tutorials and online calculators:

- t-test and paired t-test: http://graphpad.com/quickcalcs/ttest1/?Format=SD
- ANOVA: http://vassarstats.net/anova1u.html
- Posthoc tests (Bonferroni and Tukey): http://statistica.mooo.com/OneWay_Anova_with_TukeyHSD
- Repeated measures and two-way ANOVA: http://vassarstats.net/anova202corr.html
- Chi square: http://graphpad.com/quickcalcs/chisquared1.cfm

1. FeedMe

In "Enhancing Directed Content Sharing on the Web", we wanted to encourage people to share more interesting web content with their friends and contacts. We focused on web junkies who consume a huge amount of content through online news readers (e.g., RSS). We created a browser extension called FeedMe that recommended friends who might be interested, based on their previous sharing activity, and made it two clicks to share it with them:



Among other things we measured, we hypothesized that users would prefer the recommendation interface. We ran a within-subjects experiment where users had one week with just a box where they typed in their friends' addresses to send, and the other week where we included the recommendations (above) as well. We called them "Aspen" and "Sierra" so that participants wouldn't think that we were trying to convince them that one was better than the other. We counterbalanced so that half the people saw Aspen the first week, and half saw Sierra, and then we switched after a week and each user shifted to the other condition. At the end of the two weeks, we asked participants to choose their favorite interface: Aspen (recommendations), Sierra (control condition), or neither.

Which of the two interfaces do people prefer most? Consider which of the answer options you really want to be comparing in order to answer this question.
Data is at: https://hcicourses.stanford.edu/cs376/2018/slides/feedme.csv
(Don't forget to create summary statistics and/or graph it first.)

2. Twitch crowdsourcing

In "Twitch Crowdsourcing: Crowd Contributions in Short Bursts of Time", we explored whether it might be possible to get people to participate in crowdsourcing tasks in a couple of seconds each time they unlocked their phone. We replaced the phone unlock screen with something that had them help populate a local census, rank images, or verify NLP fact extractions from Stanford's Wikipedia page:

We wanted to understand whether these unlock screens impacted cognitive load and speed any more than a simple, traditional slide-to-unlock. We had participants do a challenging short-term memory task, and measured their completion times. Between stages of the task, we either showed one of these Twitch Crowdsourcing activities (Census/Dress, Census/Energy, Census/Activity, Census/People, or Structure the Web), or the control (Slide-to-Unlock). Because reaction times are typically not normally distributed, and most statistical tests assume normally distributed outcomes, we did a mathematical transformation on the times, so the final dependent variable is the 'transformedDuration' column. Each participant ('phoneID') did the task many times.

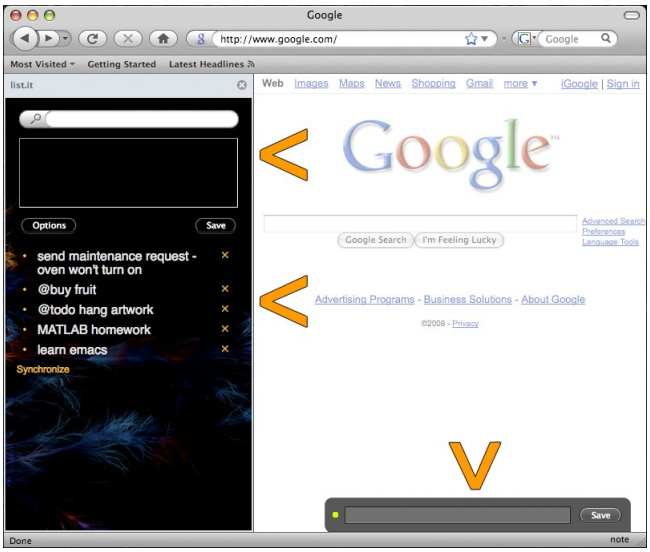
Is there a significant difference in reaction times between unlock approaches?

Bonus: control for participant (phoneID) in your analysis

Data is at: https://hcicourses.stanford.edu/cs376/2018/slides/twitch-unlocks.csv
(Don't forget to create summary statistics and/or graph it first.)

3. Information scraps

In “Note to Self: Examining Personal Information Keeping in a Lightweight Note-Taking Tool”, we wanted to better understand *information scraps*, or the pieces of personal information that fall outside our existing tools. So, we created a browser plugin that was a quick-capture and quick-access note tool, like a plaintext Evernote in the browser. We got lots of people to use it, and many let us look at their note behavior for scientific purposes.



The list.it interface. Top left: note capture and search; Middle left: example note; Bottom right: quick capture bar.

As we looked at the data, we found that only 28% of notes were ever deleted, and nearly a quarter of those notes were deleted within a day of being captured. So, we started wondering whether such notes were mainly serving as memory triggers, and thus might be shorter in length. We split up all notes according to whether they were deleted in less than a day, or not.

Is there a significant difference in character length between notes that were deleted within a day, and those that were not?

Data is at: https://hcicourses.stanford.edu/cs376/2018/slides/listit-notes.csv
(Don’t forget to create summary statistics and/or graph it first.)

4. Twitch crowdsourcing redux

We showed that the ANOVA was significant when considering different unlock screens. However, since the ANOVA just tells you that there is a difference *somewhere* between those means, we didn’t know which unlock screens were different from which other ones: Census/Dress, Census/Energy, Census/Activity, Census/People, and the control Slide-to-Unlock.

Which pairs of conditions are significantly different than each other?

Data is at: https://hcicourses.stanford.edu/cs376/2018/slides/twitch-unlocks.csv

(Don’t forget to create summary statistics and/or graph it first.)

5. Micro- vs. Macro-tasking

In “Break It Down: A Comparison of Macro- and Microtasks”, we sought to understand whether information workers and crowd workers were better off doing large chunks of tasks (macrotasks) or splitting them up into lots of tiny microtasks like is common on Amazon Mechanical Turk. For example, are workers better at adding together a whole receipt at a time? Or would they better off with microtasks that direct them to add pairs of numbers incrementally? We hypothesized that microtasks might be slower, but more robust to interruption. So we did a 2x2 experiment where we manipulated whether the task was a micro- or macro-task and whether the worker got interrupted with another task in the middle. We measured the total time it took to complete the task, minus the time it took to complete the interruption.

Are micro- or macro-tasks faster, and which one is more robust to interruptions? Any interaction effects to report?

Data is at: https://hcicourses.stanford.edu/cs376/2018/slides/cs376_addition_tasktime.csv
(Don’t forget to create summary statistics and/or graph it first.)