Stat 5200: Analysis of Designed Experiments

Jesse Wheeler

Contents

1 Overview

Objectives and overview

- This material is based Chapter 2 on the textbook "A First Course in Design and Analysis of Experiments" (Oehlert, 2000).
- Chapter outcomes include:
  - Understanding the importance of randomization to mitigate confounding and bias.
  - Understanding the important role randomization plays in statistical inference.
  - Reviewing basic inference concepts, such as hypothesis testing and basic t-procedures.

2 Randomization

Randomization and Experiments

- An experiment is called *randonmized* when the assignment method to units involves explicit use of an *understood* probability mechanism.
- Proper randomization is essential to ensure that uncontrolled factors do not introduce systematic bias in your results.
- *Randomization is not "haphazard" assignment.*
- E.g.: Just assigning treatments sequentially to the next experimental unit does *not* count as random, even if it appears to you there is no pattern among the units. **Consider the following assignment methods**:
- In all cases, suppose that we have four treatments that need to be assigned to 16 EUs.

(i) Use 16 slips of paper marked with letters, 4 A's, 4 B's, 4 C's, and 4 D's. You mix them thoroughly in a box or basket. For each EU, you pick a slip and assign it the treatment indicated.

(ii) You assign A to the first 4 units you have at hand, B to the next 4, C to the next 4, and D to the last 4.

(iii) If, when you get to an EU, the second hand is between 1 and 15, the unit gets A; if between 16 and 30, it gets B; if between 31 and 45, it gets C; and if between 46 and 60, it gets D.

Why Randomize?

- "Introducing more randomness into an experiment may seem like a perverse thing to do", however it has two useful consequences (Oehlert, 2000):

  1. Randomization protects against *confounding*.
  2. Randomization can form the basis for inference.

  The course focuses on the impact on confounding.

Example: Protecting against confounding.
Confounding occurs when the effect of one factor or treatment cannot be distinguished from that of another factor or treatment.
EX: In a medical study, we wish to see if treatment A (costly and invasive) or treatment B (standard drug treatment) results in improved health outcomes. We have 100 volunteers, each needs to be assigned a treatment (A or B).
Q: What may "go wrong" if we don't randomize assignment (e.g., let the physician pick).

- If the assignment is truly randomized, then any factors other than the ones included in the treatments (i.e., lurking variables) will tend to average out so they affect all treatments equally and have little net effect.

  For instance,

  - If there are both male and female patients, and assignment is random, there will be approximately equal proportion of men and women getting each treatment.
  - Average ages for each treatment will be close.
  - Other relevant health measures (e.g., blood type, blood pressure, BMI, etc.) will have similar averages across treatment groups.
  - *All other factors* that are not observed, measured, or controlled will tend to affect all treatment groups *equally*.

- Randomization generally costs little in time and effort, but can greatly enhance your analysis and save us from disaster.

Randmizing "Other Things"

- We have taken a simple view so far of experiments: "assign treatments to units, and measure responses".

- Though naturally we do assignments randomly, additional things can and should be randomized:

  - If the EUs are not used simultaneously, you can randomize the order in which they are used.
  - If the EUs are not used at the same location, you can randomize the locations at which they are used.
  - If you use more than one measuring instrument (e.g., multiple machines or undergraduates), you can randomize which units are measured by which instrument.
- Whenever we anticipate that some way the experiment is conducted or measured may potentially impact the response, we can *design* that into the experiment (more to come later).
- *Advice:* Design for known problems, randomize everything else.

3 Randomization Methods

Randomization methods and examples

- Randomization methods largely fall into two categories: physical and numerical.
- For modern designs and experiments, we generally use numeric approaches like a *psuedorandom* number generator.
- A few examples (2 treatments, A and B). What are some potential advantages or disadvantes of each?

  (i) Flip a coin for each patient, heads give treatment A, tails give treatment B.

  (ii) Take patients in pairs, perhaps approximately matched by demographic or physical variables, flipping a coin to choose which gets A, then giving B to the other patient.

  (iii) If you have 30 patients, say select a random sample (without replacement) of size 15 to get A and give B to the remaining 15 patients.

- The analysis we perform depends on how we randomize.
  - Approaches (i) and (iii): two-sample t-procedures.
  - Approach (ii): paired t-test / one-sample t-test.

Computer Randomization

- In 2026, our primary means of randomization will be using computers and *pseudorandom* number generators.
- One of the most important functions we will seed is `set.seed`, which allows our randomization to be *reproducible*.

```
set.seed(123)
```

- Pick any positive integer. Avoid "Seed hacking".

- Another key randomization function is `sample`. This is a flexible method that can be used in a variety of ways.

```
# 4 arguments:
sample(x, size, replace, prob)
```

  - `x`: Any 'R' object, typically a vector (character or numeric). The "things" we want to randomly draw.
  - `size`: How many "things" in `x` that we want to sample. Default is the same size as `x`.
  - `replace`: Do we want sampling with (`TRUE`) or without (`FALSE`, default) replacement.
  - `prob`: Do we want to have un-even sampling probabilities (e.g., biased coin?)

- **EX:** Suppose you have $N = 31$ EU's that you want to assign to 4 treatments, with $n_A = n_B = n_C = 7$ and $n_D = 10$.
- One way to do this physically: create slips of paper labelled $1, 2, \ldots, 31$, put them in a box, shuffle well, and draw numbers sequentially. The first 7 will be assigned treatment A, the next 7 treatment B, etc.
- We can mimic this approach in R:

```
set.seed(1111111)
sample(1:31)  # permute the numbers 1 -- 31.

 [1] 28  4  9 23  1 30 11 31 12 15 22 14 19 16  7 25 21
[18] 27 29  2 17  8  5  3 13  6 20 26 18 24 10
```

- Above, we're permuting the numbers (EUs), and then "labeling" them sequentially with treatment.
- An alternative approach can also be used to permute the treatments, and then sequentially assign EUs.

```
set.seed(1111111)
treatments <- c(
  rep("A", 7), rep("B", 7),
  rep("C", 7), rep("D", 10)
  )
sample(treatments)

 [1] "D" "A" "B" "D" "A" "D" "B" "D" "B" "C" "D" "B"
[13] "C" "C" "A" "D" "C" "D" "D" "A" "C" "B" "A" "A"
[25] "B" "A" "C" "D" "C" "D" "B"
```

- **EX:** Randomizing matched pairs. Assume that we have 30 workers, and two treatments (A and B). We want to give each treatment to each worker, but in random order.
  - Physically, we can use 30 coin flips. For the nth flip, if heads, assign treatment A first, B otherwise.
  - In R:

```r
set.seed(222222)
# "prob" not set, each outcome equally likely
sample(c("A", "B"), size = 30, replace = TRUE)
```

```
 [1] "A" "A" "B" "A" "B" "B" "B" "A" "A" "A" "A" "B"
[13] "A" "B" "A" "A" "A" "B" "B" "A" "A" "B" "A" "B"
[25] "B" "A" "A" "A" "A" "B"
```

- If we instead want a "biased" coin-flip:

```r
set.seed(222222)
# "probs" not set, each outcome equally likely
sample(
  c("H", "T"), size = 30,
  replace = TRUE, prob = c(0.7, 0.3)
)
```

```
 [1] "H" "H" "T" "H" "H" "H" "H" "H" "H" "H" "H" "H"
[13] "H" "H" "H" "H" "T" "H" "H" "H" "T" "H" "T" "H"
[25] "H" "H" "H" "H" "T" "H"
```

4 Randomization as basis for inference

Frequentist statistics

- Randomization also forms the basis of one form of statistical inference.

Review: Hypothesis testing and inference
Suppose we want to compare two competing ideas (hypotheses), and come up with data-driven conclusions.
We will call the prevailing standard idea the "Null Hypothesis" (H_0), and the alternative the "Alternative Hypothesis" (H_1). **Framework:** Assume that H_0 is true; if so, how "frequent" is the observed outcome relative to all possible outcomes? If it is a rare outcome under H_0, we assume H_0 is incorrect and favor H_1.

- **EX:** Paired t-test for testing the effect of workplace conditions.
  - There are 30 individuals in the dataset, each with measurements under the standard (S) and ergonomic (E) conditions.
  - We are interested in comparing the run-stitching times under each condition.
  - Because each individual has measurements under both treatments (order randomly assigned), we can do a matched pairs test for the difference in observation times.
  - Let $d_i = X_{i,E} - X_{i,S}$ denote the difference in run-stitching times between the ergonimic and standard working conditions, respectively.
  - Our null-hypothesis is that the mean $\mu = E[d_i] = 0$, and the alternative is $H_1 : \mu > 0$, because we expect the workers to be faster in the ergonomic workplace.
  - For a parametric test like the t-test, we will need to make a parametric assumption like normality. **Qs:** Why normality? How important is this assumption? What about if the don't look like they come from a normal distribution?

- Differences in runstitching times are given in Table 1.
- Summary Statistics: $\bar{d}_i = 0.175$, $s = 0.645$
- Under the assumption of normality, $H_0 : \mu = 0$, $T = \bar{d}_i/(s/\sqrt{n}) \sim t_{29}$. The value we observed was $T = 1.49$, which is larger than 92.6% of the possible T values we would expect under the null. **Q:** What now?

1.03	-0.04	0.26	0.30	-0.97	0.04	-0.57	1.75	0.01	0.42
0.45	-0.80	0.39	0.25	0.18	0.95	-0.18	0.71	0.42	0.43
-0.48	-1.08	-0.57	1.10	0.27	-0.45	0.62	0.21	-0.21	0.82

Table 1: Differences in runstitching times, Table 2.2 from Oehlert (2000).

Randomization as a form of inference

- What was done: Create a null-hypothesis (means for both groups are the same), compare what we saw to the range of all possibilities. **Q:** Why the normal assumption?
- The key to the procedure above is assuming the observations are random. We'll use this logic to create a *randomization test*.
  - This is an *exact* statistical test, which doesn't require normality assumption!

Randomization Test

- Hypothesis: treatment has no effect. That is, the treatment can be thought of as a label, not something that would change outcomes.
- Pick a statistic of interest (i.e., the mean, median, etc.). Compute this for the observed data.
- Since treatments are just "labels", and assuming we randomized these "labels", compute the same statistic for *all* possible assignments.
- Compare statistics to get P-value: how abnormal is the actual data relative to all random assignment possibilities?

- *Advantage:* No normality assumption. No assumptions at all!
- *Disadvantage:* Can be extremely computationally intensive.
  - For only two assignments, "relabeling" is the same as picking the differences to be positive, or negative.
  - With only $n = 30$ differences, this requires checking all $2^{30} \approx 1.07$ billion possible combinations!

Last 10 observations
Here are differences for the last 10 observations:

```
 [1]  0.62  1.75  0.71  0.21  0.01  0.42 -0.21  0.42
 [9]  0.43  0.82
```

- Even with $n = 10$, we require computing $2^{10} = 1024$.

- Using software, we found that only $4/1024 \approx 0.004$ of the means were larger than what we observed.
- This approach becomes computationally burdensome as n grows.
- Compare this to the P-value of a t-test: 0.007.
- Is the an easier way?
- Sometimes... We don't actually need to know the value of all possible permutations, only the percentage of them that are larger (or smaller) than or equal to what we saw.
- Our data: 9 positive values, 1 negative value (-.21). 7 of the 9 positive values has magnitude larger than 0.21. The remaining numbers are: +0.21, +0.01 Ways to not decrease the mean:
  - Nothing changes.
  - -0.21 changes sign, everything else stays the same.
  - -0.21 changes sign, and +0.01 also changes sign OR +0.21 changes, but not both.
- (Note that if any of the 7 positive values with magnitude > 0.21 changes, then already the total mean must have decreased)
- Thus, without actually computing anything, we can see the p-value is $4/1024$.

Approximate tests

- Even with clever strategies, it's likely computationally unfeasible for even small n and two treatments.
- We generally therefore resort to approximate methods. For instance, it can be argued that the t-test is an approximate test.

Approximate test: random subsamples
Instead of computing 2^{30} possible combinations: randomly sample a subset $(10,000)$ assignments, and compare to the observed data as you would normally.

For our $n = 30$ data set, here's example R code (from scratch) that could compute an approximate randomization test:

```
set.seed(123)
true_mean <- mean(data)
null_means <- replicate(10000, {
  random_signs <- sample(c(1, -1), size = 30, replace = TRUE)
  mean(data * random_signs)
})

# P-value
mean(null_means > true_mean)

[1] 0.0738
```

- Some final comments about these results:
  - The final (approximate) P-value is *random*; you will get slightly different answers with different seeds.
  - Following a binomial approximation, the standard deviation of this P-value is approximately $\sqrt{\hat{p}(1-\hat{p})/10000} = \sqrt{0.074(1-0.074)/10000} = 0.0026$
  - Compare this result to that of the t-test: 0.074. Recall this is only *exact* if the data come from an normal distribution (unlikely).

Acknowledgments

- Compiled on August 12, 2026 using R version 4.5.0.

- We acknowledge students and instructors for previous versions of this course / slides.

References

Oehlert GW (2000). *A First Course in Design and Analysis of Experiments*. W. H. Freeman. ISBN 0-7167-3510-5. 1, 3