Michael Barber
Data scientist working in "data science for social good" industries
Jan 28 · 20 min read

Data science you need to know!
A/B testing

This is part 2 of a 5-part series of posts aiming to quickly introduce some core concepts in data science and data analysis, with a specific focus on areas that I feel are overlooked or treated briefly in other materials.

This post outlines A/B testing, and the steps necessary to plan and build your own robust A/B test.

This post will suit data scientists working in product development, and product managers hoping to communicate better with their data scientists.

. . .

The humble A/B test (also known as a randomised controlled trial, or RCT, in the other sciences) is a powerful tool for product development.

As well as being perhaps the most accurate tool for estimating effect size (and therefore ROI), it is also able to provide us with **causality**, a very elusive thing in data science! With causality we can finally lay to rest the “correlation vs causation” argument, and prove that our new product actually works.

. . .

Imagine that you are CEO of Amazon, and trying to work out whether rearranging your website into a new format affects conversion rate (i.e. the proportion of visitors to Amazon who become customers):

One approach would be to run both versions to selected customers and and make a judgement based on these numbers alone:

	Layout A	Layout B
Visitors	122	118
Customers	22	25
Conversion %	18.0%	21.2%

In this case we would conclude that Layout B is superior to Layout A.

However, such a simple approach suffers from two possible errors that statistics students will be very familiar with:

- Type I error—or falsely concluding that your intervention was successful (which here might be falsely concluding that layout B is better than Layout A). Also known as a false positive result.
- Type II error—falsely concluding that your intervention was not successful. Also known as a false negative result.

> *These errors derive from a problem we touched on in a previous post ; namely, trying to draw conclusions about a population (in this case, all Amazon customers) from a sample (in this case, the visitors who participated in our layout trial).*

An A/B test will enable us to *accurately* quantify our effect size and errors, and so calculate the probability that we have made a type I or

type II error.

I would argue that only once we understand the true effect size and robustness of our results, can we proceed to making business-impact decisions. To phrase this another way, *we should only estimate the ROI (return on investment) of a new product once we understand our effect size and errors.*

Enter statistics:

This post will outline the design principles of A/B tests and how to ensure that a trial is effective and cost-efficient. We will be relying on the concepts introduced in our last post (statistical power and p-values), so feel free to jump back a post if these need refreshing.

Once you understand A/B testing in data science you will also understand randomised trials, which are commonly used in:

- Medicine, to understand if a drug works or not
- Economics, to understand human behaviour
- Foreign aid and charitable work (the reputable ones at least), to understand which interventions are most effective at alleviating problems (health, poverty, etc)

Principles of A/B tests

We might start thinking about an A/B test based on a question or idea from a colleague. For example, we might have a hunch that *SMS reminders for loan repayments will reduce loan defaults*. With a little bit of work we can take this question and turn it into a hypothesis and then an A/B test that will evaluate the exact gain (or lack of gain) that results from the new SMS system

To do this, we first need to form our question as a **hypothesis**, we then need to work out our **randomization strategy**, **sample size** and finally our **method of measurement**.

The hypothesis

A hypothesis is a formal statement describing the relationship you want to test. A hypothesis must be a simple, clear and testable statement (more on test-ability below) that contrasts a control sample (e.g. Layout A) with a treatment sample (e.g. Layout B).

To form a hypothesis, we re-phrase “does an SMS system improve repayment” into two statements, a null hypothesis and an alternative hypothesis:

- Null hypothesis (H0) : The null hypothesis usually states that there is **no difference** between treatment and control groups. (To put this another way, we’re saying our treatment outcome will be statistically similar to our control outcome)
- Alternative hypothesis (H1): The alternative hypothesis states that **there is a difference** between treatment and control groups. (In other words, the treatment outcome will be statistically different to the control outcome)

Notably, a hypothesis should include reference to the ***population*** under study (Amazon.com US visitors, London bank customers etc), the ***intervention*** (website layout A and B, targeted loan repayment SMS), the ***comparison*** group (what are comparing to), the ***outcome*** (what will you measure) and the ***time*** (at what point will you measure it).

Population, **I**ntervention, **C**omparison, **O**utcome, **T**ime = **PICOT**. Remember PICOT when defining your hypotheses.

To give an example of a well formed hypothesis from our Amazon example:

- Null hypothesis (H0): Amazon.com visitors that receive Layout B will not have higher end-of-visit conversion rates compares to visitors that receive Layout A
- Alternative hypothesis (H1): Amazon.com visitors that receive Layout B will have higher end-of-visit conversion rates compared to visitors that receive layout A

Note that the above clearly states our PICOT:

- Population: individuals who have visited the Amazon.com site
- Intervention: new website layout (Layout B)
- Comparison: visitors receiving Layout A
- Outcome: Conversion rate
- Time: End of visit to Amazon.com

We can contrast this with a weak hypothesis from an agricultural context, such as:

- **H0:** Banks with nicer colours will not effect loan repayment
- **H1:** Banks with nicer colours will effect loan repayment

Why is this so bad? Take a moment to think before reading on.

- There is no clear definition of “nicer colours”, my nice and your nice might not match. This is an example of a poor intervention definition from PICOT
- What banks? Where, and what level? Do we mean bank branches, if so, are we studying all branches around the world or just those in Manchester city centre? This is a poor population specification
- How are we measuring this? Loan default rates, days past due, total branch losses? This is an example of outcome specification from PICOT.

A strong hypothesis will hold the A/B test together and provide guidance on the design and analysis. You can see that the above hypothesis is useless for these tasks.

. . .

Randomisation

Returning to our Amazon example, once we have a well formed hypothesis we can think about randomisation strategies. To extend our example from above, we could randomise our visitors in two ways:

- Randomly assign visitors to Layout A or B
- Allow visitors to opt-in to new layout betas

What would be the difference between these two setups? Before we answer this question, let us examine the reasons *why* we randomize in an A/B test.

Randomization in an A/B test serves two related purposes:

1. Distributing co-variates evenly
2. Eliminating statistical bias

3. **Co-variates** are factors that might influence your outcome variable, for example, visitor geolocation, gender and risk-appetite. Note that some of these are *observable* and measurable (such as location and gender) and some of these are *unobservable* (such as risk appetite, which is difficult to measure).

4. **Statistical bias** can occur *when your sample is substantially different from your target population*. In an A/B test we are assuming our sample is representative of our population, deviations from this assumption can lead us to an incorrect understanding of our population and generating conclusions that look robust but are actually invalid!

Common forms of bias at this stage of A/B test design (which also effect our co-variate distributions) are:

- **Randomization bias**—bias due to poor randomization resulting in *unbalanced Treatment/Control groups* (e.g. Texan visitors are over-represented in our treatment visitors vs. our control visitors). This allows some co-variates (e.g. being Texan) to exert more influence in one group than another.

- **Selection bias**—bias would also result if we were to allow visitors to assign *themselves* to Treatment/Control groups. This is because there may be *unobservable* co-variates that are associated with a Treatment/Control choice. For example, visitors with more risk appetite and might select themselves into the beta testing group, and so our Treatment visitors might have both a treatment effect *and* a risk appetite effect

Both of these will lead to what is called *confounding bias*, this means it will be difficult to untangle effects that are due to poor randomisation vs. effects that are due to the actual intervention.

However, if each participant has an equal chance of being randomly assigned to a Treatment/Control group then randomization will be free of bias. This will result in both observable (e.g. location) and unobservable co-variates (such as risk appetite) being spread equally to Treatment and Control groups. It is this spreading of co-variates that allows us to understand causality.

To revisit our example at the top of this section, our choice between two strategies:

1. Randomly assign visitors to Layout A or B

2. Allow visitors to opt-in to new layout tests

Which of these now seems like a better strategy from a statistical point of view?

Whilst there might be logistical or organizational reasons why we can't do the former strategy, it is certainly a more statistically robust trial. This is because we have complete control to assign visitors (and therefore co-variates) to each group. If we used strategy 2, allowing visitors to opt-in to new layouts, then there would likely be unobservable factors at play that might weaken our A/B test and confound effects from unobservable factors with our outcome variable.

It is therefore important to select treatment and control groups *totally randomly*, the best way to achieve this is by letting R do the work for you. I have included a function below that will do this for you. Please note the use of the set.seed function—we will be using random numbers to assign our participants to Treatment/Control status and so set.seed will make this randomness reproducible.

. . .

Let's now look at the function to assign Treatment/Control status, which I have here called *RCT_random*:

```
#lets first make up a fake list of  IDS from 1 to 1000 and
1000 random variables drawn from a normal distribution

df = data.frame("ID"=seq(1,1000),
"randomvariable"=rnorm(1000))

#lets now make a function to do the work - you can copy
paste this function into your own scripts
# it needs to be given a dataframe and a list of naming
options
# Options might be "treatment" and "control", or if there
are more than 2 options then it might be "Control",
"treatment1", "treatment2", or just "LayoutA", "LayoutB"

RCT_random = function(dataframey, values_to_add){

  set.seed(111)
  dataframey$values_to_add[sample(1:nrow(dataframey),
nrow(dataframey), FALSE)] <- rep(values_to_add)
  colnames(dataframey)
[which(colnames(dataframey)=="values_to_add")] = "Status"
  return(dataframey) }
```

```
# so this will take the dataframe called "df" and randomly
assign each ROW to "Treatment" or "control"
df_new = RCT_random(df, c("Treatment","Control"))
```

We have now taken our original dataframe

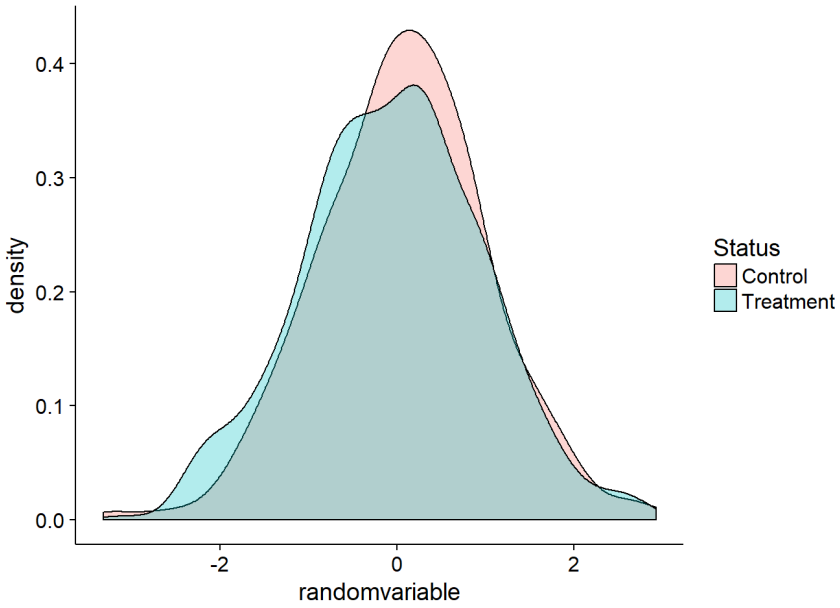
```
  ID randomvariable
1  1     -0.3307359
2  2     -0.3116238
3  3     -2.3023457
4  4     -0.1708760
5  5      0.1402782
6  6     -1.4974267
```

and randomly assigned Treatment and Control status to them with my function “RCT_random”:

```
  ID randomvariable    Status
1  1     -0.3307359 Treatment
2  2     -0.3116238 Treatment
3  3     -2.3023457   Control
4  4     -0.1708760 Treatment
5  5      0.1402782   Control
6  6     -1.4974267   Control
```

We should now double-check our randomization to ensure it has proceeded as expected, to do this we can look at the distributions of the most important key variables. It is also possible to run an appropriate hypothesis test to assess whether the distributions are different. For these hypothesis tests we will set our P-value threshold to 0.01 (and not the 0.05 that is commonly used), this is due to the *multiple comparison problem* that we will cover in another post.

Let’s now look at the summary statistics for Treatment and Control groups and make sure that “randomvariable” is similar:



They look pretty similar! So far so good, randomization has been successful.

Pop quiz

Let's look at some examples of randomization strategies for below and try to decide whether they are proper or improper randomization:

1. Participants are allowed to decide whether to be in a treatment or control group
2. Any participant with a national ID number ending in an odd number is assigned to treatment, any participant with an ID ending in an even number is assigned to control.
3. Participants east of my office are assigned to control, participants west of my office are assigned to treatment
4. We flip a coin to decide whether a participant is control or treatment

Which of the above are truly randomized? Take a moment to think about this before continuing on to the suggested answers below.

1. Participants are allowed to decide whether to be in a treatment or control group. *This is not random, as participants may have a reason for choosing a group which would lead to self-selection bias*
2. Any participant with a national ID number ending in an odd number is assigned to treatment, any participant with an ID ending in an even number is assigned to control. *Almost random,*

but not quite! The issue here is that there might be some unknown factor associated with national ID that influences our results. If we 100% knew that national IDs were made randomly then this would be fine, however as we cannot know that we should use an alternative method

3. Participants east of my office are assigned to control, participants west of my office are assigned to treatment. *This is a terrible idea, this will likely lead to confounding bias as there are likely to be geographical differences between participants east and west*

4. We flip a coin to decide whether a participant is control or treatment, Heads means treatment, Tails means Control.*This is the only truly random system, as it is based only on chance*

. . .

Cluster randomisation

Sometimes, we won't be able to randomise at the individual level but we will still want to measure at that level. For example, if we were to randomly assess the impact of incentives for bank staff on customer loan repayment rates then it would not be possible to randomise at the customer levels (as these *share* banks and therefore bank staff) and we would instead need to randomise at the bank level whilst still measuring the customer level outcome (repayment rate).

The process of randomizing at one level but measuring at another causes complications in our A/B test design. Specifically, the inherent similarity of customers sharing a bank (in the above example) will lead us to have narrower distributions and under-estimate our errors. This, in turn, will have knock on effects for our study power and our study results (i.e. we'll have a higher rate of false positives due to false confidence in our data). The concept that individuals treated in groups will behave similarly is known as **clustering** and our solution is to use a **cluster randomized trial** or **cluster A/B test.**

A cluster randomized trial is similar to an A/B test but our unit of randomization becomes the cluster rather than the individual (so in the above example, the bank is the cluster). We can measure clustering with the **intra-cluster correlation** or **ICC** which will tell us how correlated the responses of individuals within a cluster are. ICC runs from 0 (no correlation between members of the same cluster) to 1.0

(complete correlation). A higher correlation causes more analysis headaches, so we want this to be 0!

ICC can cause our distributions to seem narrower than they really are, this, in turn, will have knock-on effects on our statistical power, coefficients, confidence intervals and p-values. It essentially leads to a false confidence in our results. You should be able to rationalize why under-estimating our errors would lead to these effects using the concepts outlined in a previous post.

It is possible to calculate the ICC before a trial using historical data. This estimated ICC will enable you to adjust your methods as necessary to produce a robust trial.

Let's first look at how we can calculate ICCs:

```
# make some fake data

#this data will have an ID number, variableA (our variable
of interest), the bank name and the district name
df = data.frame(ID=seq(1,100), variableA=rnorm(100,500,50),
bank_name=c("first_bank","second_bank","third_bank","last_ba
nk"), District=c("A","B"))

library(knitr)
kable(df[1:5,], format="markdown", align="c")
```

We can calculate the ICC using the snippet of code below:

```
ICC_CI <- function(cluster_level,outcomevar, dataf){

  #load library
require(ICC)
  set.seed(123)
  si = round(dim(dataf)[1]*0.66)
  values_f <- c()
  for(i in seq(1:50)){
  samp_f = dataf[sample(nrow(dataf), si), ]
  x_f = ICCbare(cluster_level,outcomevar,samp_f)
  values_f <- c(values_f, x_f)
  }
  # note that 1.96StDevs = 95% confidence interval bounds in
a normal dist.
   ret = data.frame("Mean ICC" = round(mean(values_f,
na.rm=TRUE),3), "CI" = round(1.96*sd(values_f,
na.rm=TRUE),3))
   ret$Significant = ifelse(ret$Mean.ICC > ret$CI, "Y", "N")
  return( ret)

  }
```

```
stored_ICC <- ICC_CI("bank_name", "variableA", df)
```

```
  Mean.ICC    CI Significant
1    0.022 0.081           N
```

We can see from this calculation that our ICC between customers in the same bank is (0.022 +/- 0.081). As the confidence intervals (CI) cross zero we can see that this is not significant (hence "Significant = N"). In other words, it appears that banks account for a negligible amount of similarity between customers.

. . .

However, if we have a significant ICC then we will need to adjust our trial design and analysis plans to mitigate ICC effects. We generally have two options here:

ICC solution 1

Option 1: Calculate a summary metric for each cluster (e.g. cluster mean). Each cluster then provides only one data point and allows us to continue with the assumption that *our data is independent*, we can then proceed with standard statistical tools (e.g. T-test) as normal.

So if we have 500 individuals in 45 groups, we end up with 45 data points. This means that our power, sample size and analysis calculations **also need to be carried out at the cluster level**. It also means that we can simply analyse our data at the cluster level and ignore ICC from here on out (as we essentially set ICC = 1.0 and proceed with this assumption).

ICC solution 2

Use the ICC with the number of individuals to work out our new sample size.

For option 2, we will continue to calculate power, sample size and analysis metrics at the individual level but with some corrections to account for the falsely narrow distributions.

For our sample size, we will inflate it with the `ICC_correction` function below:

```
ICC_correction <- function(samplesize, num_clusters,
ICC_estimate){

  average_cluster_size = samplesize/num_clusters

  factor_inflate = 1 + (average_cluster_size - 1) *
ICC_estimate

  return(data.frame("New sample
size"=round(samplesize*factor_inflate), "Old sample
size"=samplesize   ))
}

ICC_correction(200, 50, 0.2)
```

```
##   New.sample.size Old.sample.size
## 1             320             200
```

So an initial sample size of `200` customers, with 30 clusters (banks) and an ICC of `0.2` would lead to a new sample size of `320`. Note how a relatively small ICC increases our sample size by more than 50%.

We can also explore the difference between adding new clusters vs new individuals in existing clusters. For example:

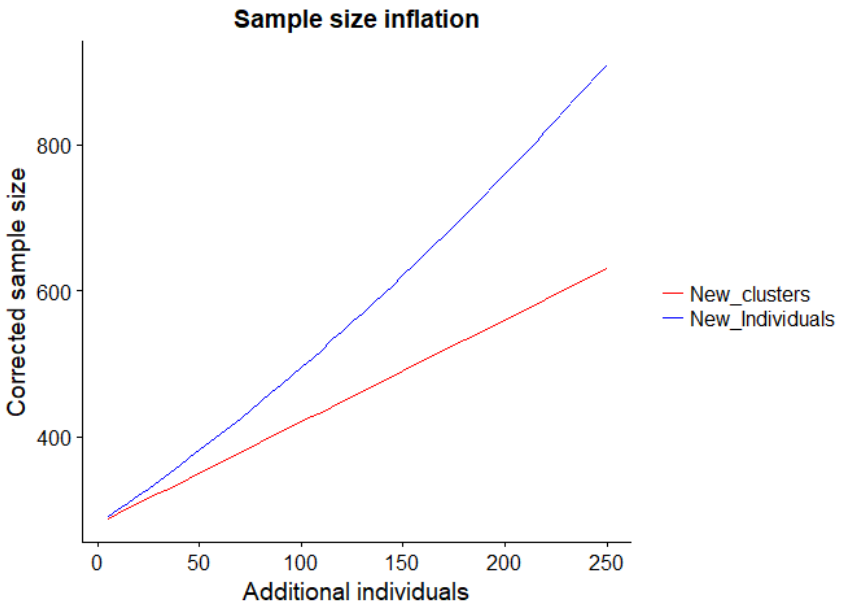
```
scenario1 <- ICC_correction(200,20,0.2)$New.sample.size
#average 10 farmers per cluster, and  20 clusters

scenario2 <- ICC_correction(200,40,0.2)$New.sample.size #
average 5 farmers per cluster, and 40 clusters
```

`scenario1= 560 `

`scenario2= 360`

We can see that doubling the number of clusters (but keeping the number of participants the same, so that customers are just spread out to more clusters) leads to a sizable reduction in corrected sample size.

I've run a simulation below that will plot the relationship between adding new individuals (to existing clusters) vs adding new clusters and how this effects sample size (holding power constant at 0.8):



In general, we can see that adding **new clusters** rather than **new individuals** to existing clusters, inflates our corrected sample size to a lesser extent (there is a link to this simulation code at the bottom of this post).

By adding new clusters, we are minimising the amount of *total* variance explained by within-cluster variance, thus gaining information. Another intuitive explanation is that if I already have 10 individuals in a single cluster, and we have non-zero ICC, then every additional individual in that cluster provides less and less information and her characteristics are more and more predictable based on cluster averages.

NB. In the next post, we will look at the actual analysis of ICC data with fixed and random effect modelling.

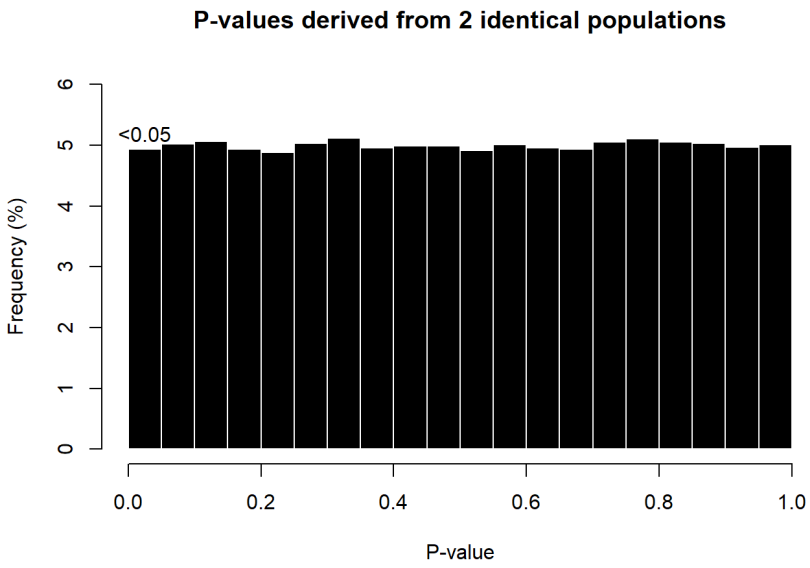
P-value threshold, aka alpha

Once we have a testable hypothesis and a randomization strategy, we will want to think about data analysis. Generally, we will test the effectiveness of an intervention with a hypothesis test. This hypothesis test will yield a p-value, which is the *probability that our data could generated purely by chance*—in other words, the probability that we wrongly reject H0 (a **false positive result**).

When using a hypothesis test we must set an acceptable rate of false positives, aka a *p-value threshold* or *alpha level*. The most common p-value threshold is 0.05 (a pretty arbitrary number). This means that we are willing to accept a 5% risk of generating a false positive and

wrongly concluding that there is a difference between our treatments when in fact there is not.

We can demonstrate this below, if we randomly test two **samples** from identical **populations** then 5% of the time we will mistakenly identify the samples as being from different populations:



In some cases we might want to set a threshold of 0.01 (1%) or 0.1 (10%). Generally speaking, the more unwilling we are to be incorrect, the lower the threshold. So for an intervention that might have adverse effects on customers, we would want to be very sure of the positive effects (0.01 threshold) and unwilling to accept negative effects (0.1 threshold). Note that the lower our threshold the larger the sample size needed to detect any effect.

As always, full R code for these figures is available at the bottom of this post.

Sample size calculations in practice

Combining the materials from this post and post #1 gives us a good theoretical foundation for sample size calculations. However a common question I want to address here is: *"where do I get the data from for a sample size calculation?"*. We can use the Amazon layout trial example from above to understand how to answer this question.

The main piece of information needed before a sample size calculation is an estimate of intervention effect size. There are generally two ways

we can derive an effect size estimate for our calculations:

- Analyse historical data or pilot studies
- set a minimum detectable effect (MDE)

Estimates from historical data or pilot studies

The first of these involves literature review and/or a small pilot study to estimate the differences between treatment and control groups. Note that if you are using pilot data, then the estimate of effect size is likely to be very rough, so I recommend halving the effect size to be conservative and using that for calculations. If you are using literature values then again, treat them cautiously and consider any literature value to be an upper estimate of effect size (this is due to the Winner's curse phenomenon which will be covered in a later post).

Minimum detectable effects (MDEs)

Perhaps the most business-savvy approach to this question utilises MDEs.

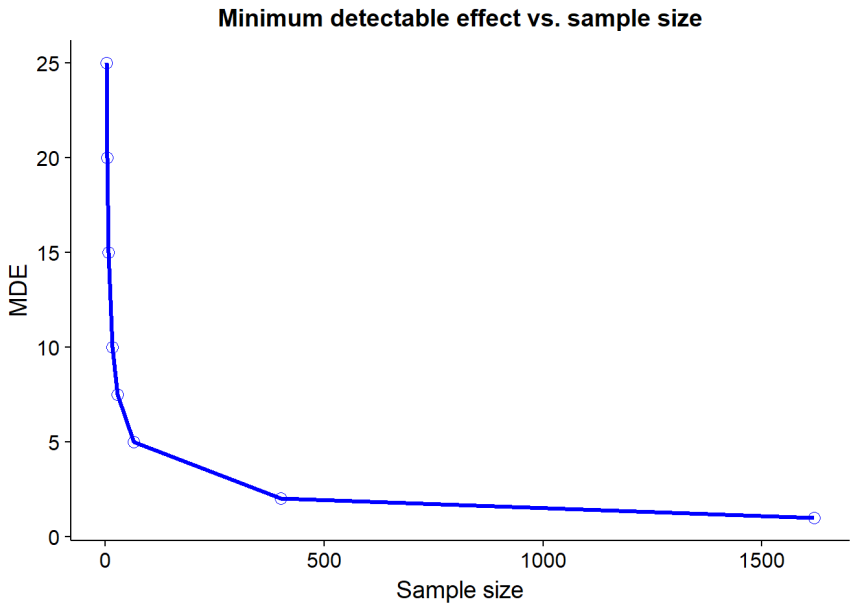
The MDE approach will ask *"what is the minimum effect that I would need to see for the intervention to be worthwhile?"*, we would then set the effect size to that value. The reasoning here is that if the return on investment (ROI) of a proposed intervention is negative (or very small) then we don't need to be able to precisely measure such a small value to make a decision.

For example, implementing Layout B across Amazon.com might be quite expensive. We could calculate that the implementation would cost approximately $20,000 of staff time, in which case we would only care about being able to detect effects greater than $20,000. For arguments sake, let's say that $20,000 corresponds to a 2% point bump in conversion rate in the US market, we then set our MDE to 2 points and our effect size to 2 points for our calculations.

The MDE approach can be *very* powerful. Use it wisely during both pre-analysis (to estimate sample size) and post-analysis to say *what size effect we would have been able to detect with a power of 0.8*. You can see a worked example below for pre-analysis MDE, I have also included a function `plot_MDE` which you can use on your own data.

```
#make some fake data, let's pretend its baseline data
```

```
dat <- data.frame("mean.client.expenditure" =
rnorm(1000,100,10))



#this function will plot MDE for various differences
# differs is a list of intervention effects that you want to
consider
plot_MDE <- function(historical_data, differs){

  #initialise empty vec
p <- c()


#remember our effect size function from post 1?
cohen_d <- function(d1,d2) {
  m1 <- mean(d1, na.rm=TRUE)
  m2 <- mean(d2, na.rm=TRUE)
  s1 <- sd(d1, na.rm=TRUE)
  s2 <- sd(d2, na.rm=TRUE)
  spo <- sqrt((s1**2 + s2**2)/2)
  d <- (m1 - m2)/spo
  rpb <- d / sqrt((d**2)+4)
  ret <- list("rpb" = rpb, "effectsi" = d)
  return(ret)  }

#load libs
require(pwr)


for(i in seq(1:length(differs) ) ) {
  samp1 <- historical_data
  xnu = differs[[i]]
  #this is a better version if you can understand it:
    samp2 <- samp1 + rnorm(length(samp1), xnu, xnu/10) #add
some noise
  inp <- cohen_d(samp1, samp2)


  p[i] <- pwr.2p.test(h=inp$effectsi , sig.level=0.05,
power=0.8, n=NULL)$n

}

require(ggplot2)
print(ggplot() + geom_point(aes(x=p, y= differs), size=3,
color="blue", shape=1) + geom_line(aes(x=p, y=differs),
size=1.2, color="blue") + xlab("Sample size")+ ylab("MDE") +
ggtitle("Minimum detectable effect vs. sample size"))

library(knitr)
mde_tab = data.frame("MDE"=differs, "Sample size"=p)
kable(mde_tab, digits=2)


}

#set some differences for the loop, here, 1,2,5 (etc) are
dollar increases
diffs <- c(1,2,5,7.5,10,15,20,25)


#plot
plot_MDE(dat$mean.client.expenditure, diffs)
```



Note that this function will give you the Y-axis in whatever units you gave to the function (in this case, US dollars). You can use the MDE to calculate an initial sample size and then use the ICC correction to obtain the ICC-corrected sample size for that MDE.

. . .

Measurement

The final part of the A/B test is the measurement itself. This is often a neglected part of test design and therefore a major source of later analysis issues. Any hypothesis needs to have a good measurement strategy to be useful.

I want to take a tangent to describe some analysis I did on an A/B test looking at airtime usage (aka phone credit) in Malawi for a telecommunication company to illustrate the pitfalls here.

This A/B test examined the effect of a new product on airtime usage. During the A/B test, we asked people to recall how much money they had spent in the last month on airtime, and unbeknownst to them, we also had data from the telecommunications company on the *actual* amount spent by the same customers. When we compared the two metrics, we found there was very little correlation! Furthermore, when we looked at who was most likely to over-estimate their airtime spend, we found it was young, urban males.

Now consider—had we only had the self-reported data, we would have thought that young urban men were big spenders on airtime, drawn many conclusions from this, and found many "statistically significant" (in terms of P-values of < 0.05) relationships! We would have made many recommendations to the telecommunications company about the new product, and our results would have held up to thorough statistical interrogation. In short—there would have been very little way to know we were wrong, but we would have been wrong nevertheless!

The moral of this story is that the best statistics in the world will not save a trial from poor measurement. The other moral is that self-reported data is often terrible (beware survey-heavy NGOs!), as it is influenced by how people want to be perceived (perhaps by the interviewer, by the community, or by their own embarrassment about "low" airtime spends).

It is vital to spend some time thinking about the caveats and weaknesses of measurement strategies, and then to try to mitigate those weaknesses as much as possible (e.g. through obtaining corroborating data as above).

Finally, once the data is collected and analysed, it's important to make it accessible and reproducible. This means writing clean code (ideally in R-markdown or Jupyter notebook) and saving the raw data on company servers.

Every data scientist in a company new to big data has witnessed the litany of random CSV files and analysis scripts strewn across dozens of employee hard-drives. A system for storing such files and scripts for possible future use is necessary, and deserves its own blog post entirely!

. . .

If you liked this post, or found is useful, then please clap so I know!

Summary

We have now seen the key parts of an A/B test:

- **Hypothesis**: A good hypothesis is *testable* and *measurable*. It must have a clearly defined evaluation criteria (e.g. are we measuring the average or the median? At the individual or group level?).

- **Randomization**: We must randomize using R and ensuring our Treatment and Control groups are balanced. We can use RCT_random function to achieve this. We should also account for cluster effects and use the correct *unit of randomization*. It is possible to calculate the ICC and adjust sample size accordingly, it is also fair to assume an ICC of 1.0 and have the sample size calculated from the number of clusters
- **Power**: Know your statistical power, sample size and MDE before and after a trial (see previous post)
- **Measurement**: Measurements must be robust and reliable, think about the ways that we might be inaccurate and try to minimize the importance of self-reported metrics in any study you design!

You can view a fuller version of this post (complete with all R code) on my GitHub

Next time

Linear models, mixed-effect models, and more on hypothesis testing!

. . .

Originally published at michael-bar.github.io.