My demo notebook

Andrew D. Nguyen

2025-04-19

Table of contents

Preface

Hello, welcome to my demo notebook. The purpose of this notebook is to log ideas I've explored. It'll be useful for me, and hopefully, also for you!

The themes I've explored include causal inference, bayesian stats, frequentist stats, decision making under uncertainty, and function-valued traits.

Part I

Causal inference section

1 Title: Causal diagram simulations

Start Date: 2025-03-23
Last modified: 2025-04-19

2 Load libraries

```
library(tidyverse) # for data wrangling
library(gtsummary) # for formatting model outputs into a nice html table
library(DiagrammeR) # for drawing dags
library(webshot2) #to help render doc
```

3 What to condition on and what not to condition on

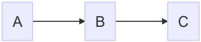
In causal inference, the building of statistical models depends on how the variables from a collected dataset relate to one another. Here, I show the arrangement of variables with a Directed Acyclic Graphs (DAG)s. The different Dags show cases where it is not appropriate to condition on certain variables.

3.1 Chain in a DAG

Here is a chain, where B is a mediator. Mediators should not be conditioned on because it will limit the association between A and C.

```
mermaid("graph LR
        A-->B
        B-->C")
```

```
file:///C:\Users\anbe6\AppData\Local\Temp\RtmpsL19U3\file86b41f3370cd\widget86b4109c60c9.html
```

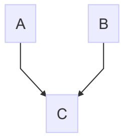

A
B
C

3.2 Colliders in a DAG

Here is a collider, where C is a collider. Colliders should not be conditioned on because there will be a spurious association between a and b.

```r
mermaid("graph TD
        A-->C
        B-->C")
```

```
file:///C:\Users\anbe6\AppData\Local\Temp\RtmpsL19U3\file86b462d376ec\widget86b473891f68.html
```

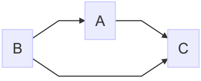

A
B
C

3.3 Confounders in a DAG

Here is a confounder, where B is a confound. Confounders SHOULD be conditioned on.

```
mermaid("graph LR
        A-->C
        B-->A
        B-->C")
```

```
file:///C:\Users\anbe6\AppData\Local\Temp\RtmpsL19U3\file86b47f753feb\widget86b4497661ff.html
```


A
B
C

4 Why we should not condition on a mediator

To determine why we should not condition on a mediator, we can simulate a chain DAG and compare a model where we condition on B vs a model without B (marginal model).

$$a \sim Normal(50, 5)$$

$$b \sim a + \epsilon$$

$$c \sim b + \epsilon$$

Random error: $\epsilon \sim Normal(0, 1)$

Note: It is expected for c and a to have 1:1 when b transmits effect.

```
# simulate mediator
#A->B->C
a<-rnorm(n=100,mean=50,sd=5)
b<-a+rnorm(n=100,mean=0,sd=1)#b is a function of a with random error centered around 0 c
c<-b+rnorm(n=100,mean=0,sd=1)

#now fit a model of c~a
mod1<-lm(c~a)
mod1|>
  tbl_regression()
```

```
ggplot(data=tibble(a,c),aes(x=a,y=c))+geom_point()+stat_smooth(method="lm")
```

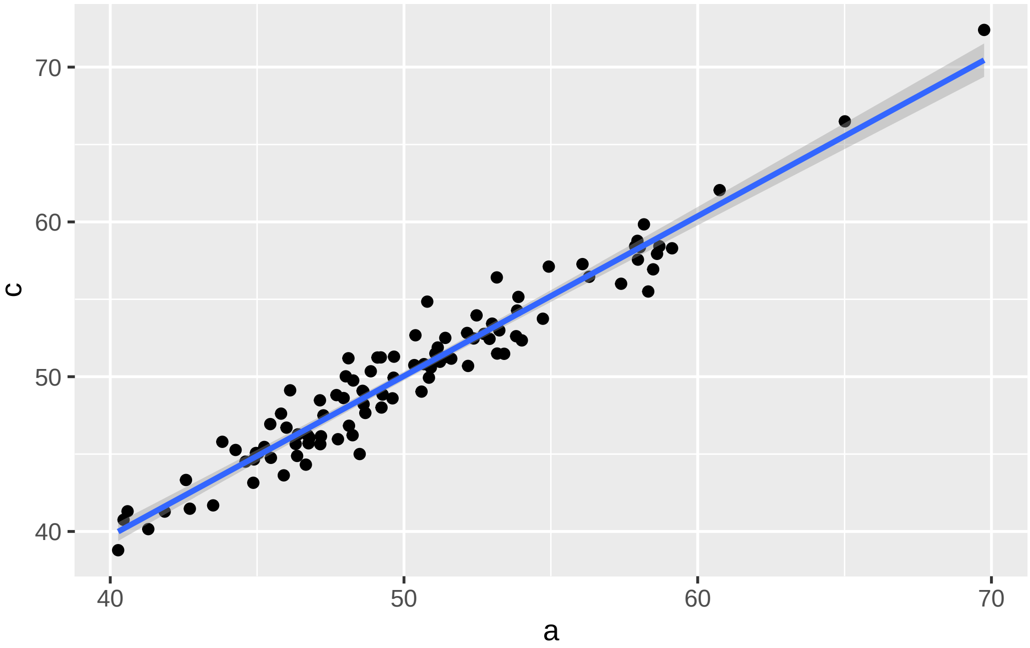
```
`geom_smooth()` using formula = 'y ~ x'
```

Characteristic	Beta	95% CI	p-value
a	1.0	0.98, 1.1	<0.001

Abbreviation: CI = Confidence Interval

Characteristic	Beta	95% CI	p-value
a	0.03	-0.19, 0.25	0.8
b	0.99	0.78, 1.2	<0.001

Abbreviation: CI = Confidence Interval



```
#let's see how the estimate changes when we condition on a mediator
mod2<-lm(c~a+b)
mod2|>
  tbl_regression()
```

Conditioning on the mediator (b) reduces the causal effect of a–>c

4.1 Changing the mediator to a categorical variable to visualize

```
# try to set b as a categorical variable
a<-rnorm(n=100,mean=50,sd=5)
b<-if_else(a<50,0,1)
c<-b+rnorm(n=100,mean=0,sd=1)
```

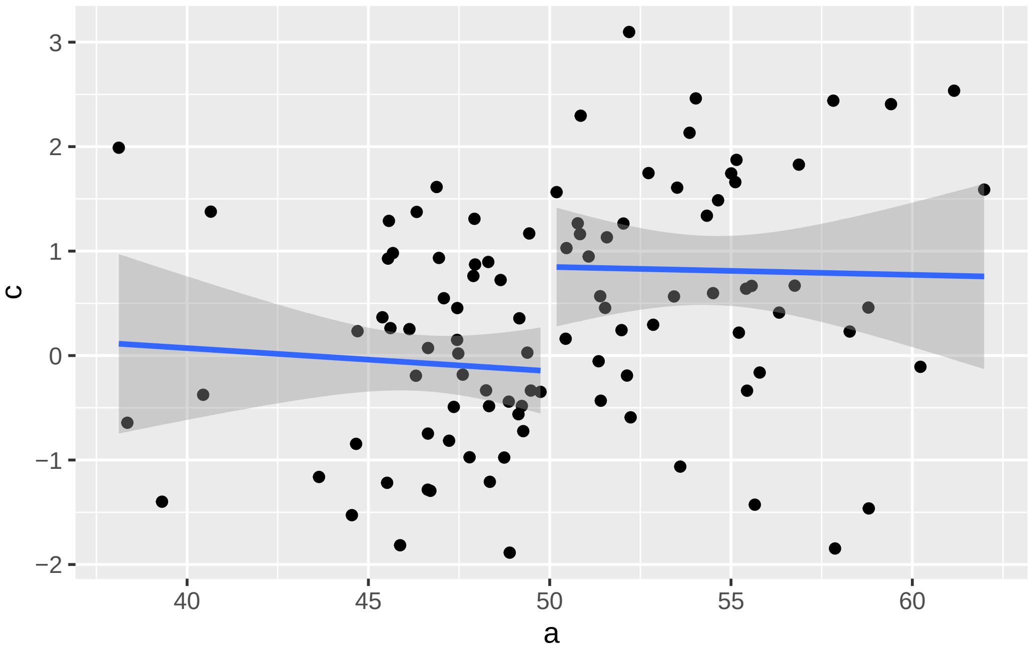
Characteristic	Beta	95% CI	p-value
a	-0.01	-0.09, 0.06	0.7
b	1.0	0.30, 1.7	0.006

Abbreviation: CI = Confidence Interval

```
mod2.11<-lm(c~a+b)
mod2.11|>
  tbl_regression()
```

```
#grouping by b (mediator) disrupts the correlation by a nd c
ggplot(data=tibble(a,c),aes(x=a,y=c,group=factor(b)))+geom_point()+stat_smooth(method="lm")
```

```
`geom_smooth()` using formula = 'y ~ x'
```

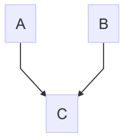


5 Why we should not condition on a collider

Let’s start with a dag that illustrates a collider. The dag is A–>C and B–>C.To determine why we should not condition on a collider, we can compare a model where we do condition on C vs a model without conditioning on C. A and B are independent and should not correlate.

```
mermaid("graph TD
        A-->C
        B-->C")
```

```
file:///C:\Users\anbe6\AppData\Local\Temp\RtmpsL19U3\file86b4451e106b\widget86b4fe311cf.html
```


A
B
C

Characteristic	**Beta**	**95% CI**	**p-value**
a	0.22	0.02, 0.42	0.031

Abbreviation: CI = Confidence Interval

Characteristic	**Beta**	**95% CI**	**p-value**
a	-0.42	-0.60, -0.23	<0.001
c	0.51	0.41, 0.61	<0.001

Abbreviation: CI = Confidence Interval

$$a \sim Normal(50, 5)$$

$$b \sim Normal(0, 5)$$

$$c \sim a + b + \epsilon$$

Random error: $\epsilon \sim Normal(0, 1)$

Note: It is expected that a and b are un-related

```r
a<-rnorm(n=100,mean=50,sd=5)
b<-rnorm(n=100,mean=0,sd=5) #b is a function of a + random error
c<-a+b+rnorm(n=100,mean=0,sd=5) # c is a fucntion of a and b with random error

#fit a model between a-> b
mod3<-lm(b~a)
mod3|>
  tbl_regression()
```

```r
#There is no association between A and B.

#fit a model with a collider c
mod4<-lm(b~a+c)
mod4|>
  tbl_regression()
```

```r
#There is an association when conditioning on C.
```

Conditioning on a collider (c) introduces the causal effect of a–>b that is negative.

Characteristic	Beta	95% CI	p-value
a	1.0	0.82, 1.2	<0.001
b	1.0	0.81, 1.2	<0.001

Abbreviation: CI = Confidence Interval

Characteristic	Beta	95% CI	p-value
a	1.2	0.97, 1.5	<0.001

Abbreviation: CI = Confidence Interval

5.0.1 Side note: recovering effects of a and b on c

```
mod4.1<-lm(c~a+b)
mod4.1|>
  tbl_regression()
```

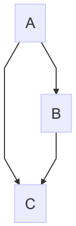
```
#now let's fit model with just one predictor alone

mod4.2<-lm(c~a)
mod4.2|>
  tbl_regression()
```

5.1 More complicated collider case

where:

```
mermaid("graph TD
        A-->B
        A-->C
        B-->C")
```

```
file:///C:\Users\anbe6\AppData\Local\Temp\RtmpsL19U3\file86b43b2fb01\widget86b45dee71eb.html
```


A
B
C

Characteristic	Beta	95% CI	p-value
a	0.94	0.75, 1.1	<0.001

Abbreviation: CI = Confidence Interval

$$a \sim Normal(50, 5)$$

$$b \sim a + \epsilon$$

$$c \sim a + b + \epsilon$$

Random error: $\epsilon \sim Normal(0, 5)$

Note: It is expected for a and b to have 1:1 relationship

```r
a<-rnorm(n=100,mean=50,sd=5)
b<-a+rnorm(n=100,mean=0,sd=5) #b is a function of a + random error
c<-a+b+rnorm(n=100,mean=0,sd=5) # c is a fucntion of a and b with random error

#fit a model between a-> b
#there is a 1:1 relationship
mod3.1<-lm(b~a)
mod3.1|>
  tbl_regression()
```

```r
#There is no association between A and B.

#fit a model with a collider c
mod4.1<-lm(b~a+c)
mod4.1|>
  tbl_regression()
```

Note that the association between a and b is negative when conditioning on c, the collider (above) and when we do it here when a and b are correlated, the correlation breaks.

Characteristic	**Beta**	**95% CI**	**p-value**
a	0.00	-0.26, 0.25	>0.9
c	0.45	0.35, 0.55	<0.001

Abbreviation: CI = Confidence Interval

6 Why we should condition on a confounder

Simulate data and compare conditioning vs not on a confound.

$$b \sim Normal(5, 5)$$

$$a \sim Normal(50, 5) + b + \epsilon$$

$$c \sim a + b + \epsilon$$

Random error: $\epsilon \sim Normal(0, 1)$

Note: It is expected for C to have a 1:1 effect from a.

```r
b<-rnorm(n=100,mean=5,sd=5)
a<-rnorm(n=100,mean=50,sd=5)+b+rnorm(n=100,mean=0,sd=1)
c<-a+b+rnorm(n=100,mean=0,sd=1)

#without conditioning
mod5<-lm(c~a)
mod5|>
  tbl_regression()
```

```r
#with conditioning
mod6<-lm(c~a+b)
mod6|>
  tbl_regression()
```

Conditioning on b recovers the 1:1 relationship from a on c.

Characteristic	Beta	95% CI	p-value
a	1.5	1.4, 1.6	<0.001

Abbreviation: CI = Confidence Interval

Characteristic	**Beta**	**95% CI**	**p-value**
a	1.0	0.95, 1.0	<0.001
b	0.99	0.93, 1.1	<0.001

Abbreviation: CI = Confidence Interval

7 Summary

Drawing out a DAG for a specific case is important to determine how to analyze the data. For many observational studies, conditioning on confounders is critical and selecting the whole set of confounders is also important. Generally, baseline covariates are the confounders that is typically conditioned upon. It takes clinical expertise to decide which covariates to include.

8 DAGs and adjustment sets

9 Intro

R can determine the adjustment set of confounders for you if you specify the DAG. Description of function adjustmentSets {dagitty}:

> Enumerates sets of covariates that (asymptotically) allow unbiased estimation of causal effects from observational data, assuming that the input causal graph is correct

10 Load libraries

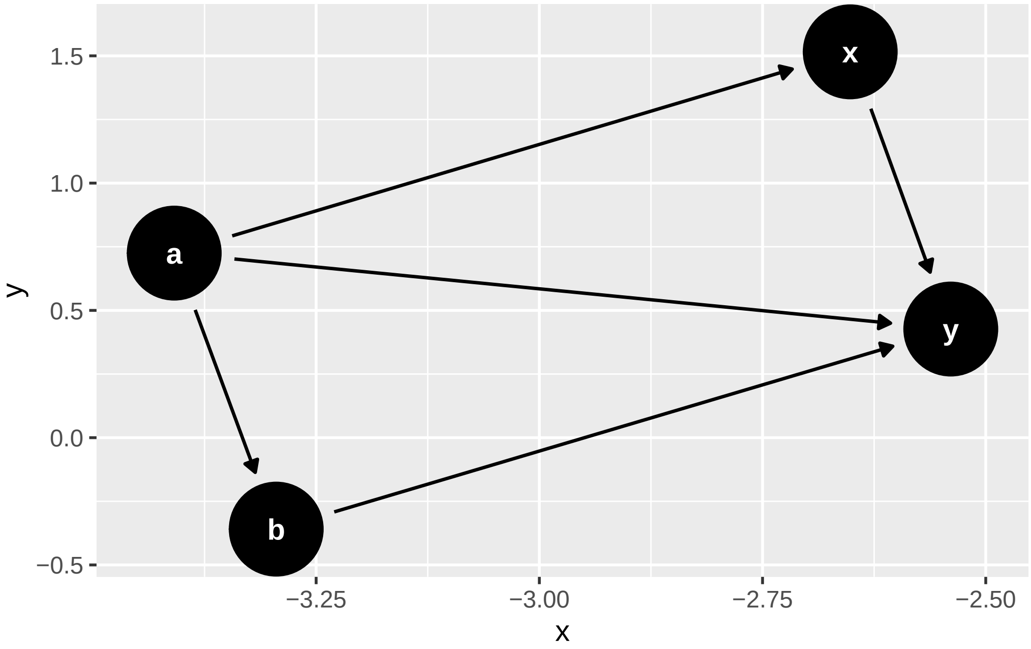
```r
library(tidyverse)
library(ggdag)
```

11 set up dag

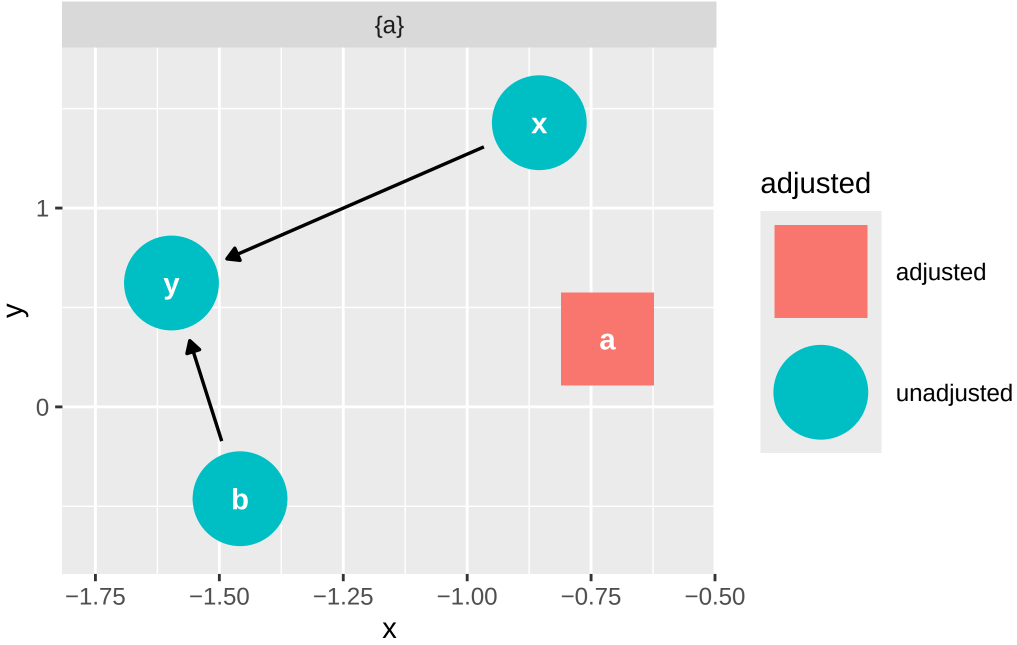
```
dag<-dagify(
  y~ x + a + b,
  x~ a,
  b~ a,
  exposure="x",
  outcome="y"
)

tidy_dagitty(dag)|>
  dag_adjustment_sets()
```

```
# A DAG with 4 nodes and 5 edges
#
# Exposure: x
# Outcome: y
#
# A tibble: 6 x 10
  name        x      y direction to       xend   yend circular adjusted   set
  <chr>   <dbl>  <dbl> <fct>     <chr>   <dbl>  <dbl> <lgl>    <chr>      <chr>
1 a     -0.693 -0.594 ->        b      0.358  -0.292 FALSE    adjusted   {a}
2 a     -0.693 -0.594 ->        x     -1.14   -1.60  FALSE    adjusted   {a}
3 a     -0.693 -0.594 ->        y     -0.0863 -1.29  FALSE    adjusted   {a}
4 b      0.358 -0.292 ->        y     -0.0863 -1.29  FALSE    unadjusted {a}
5 x     -1.14   -1.60  ->        y     -0.0863 -1.29  FALSE    unadjusted {a}
6 y     -0.0863 -1.29  <NA>      <NA>  NA      NA     FALSE    unadjusted {a}
```

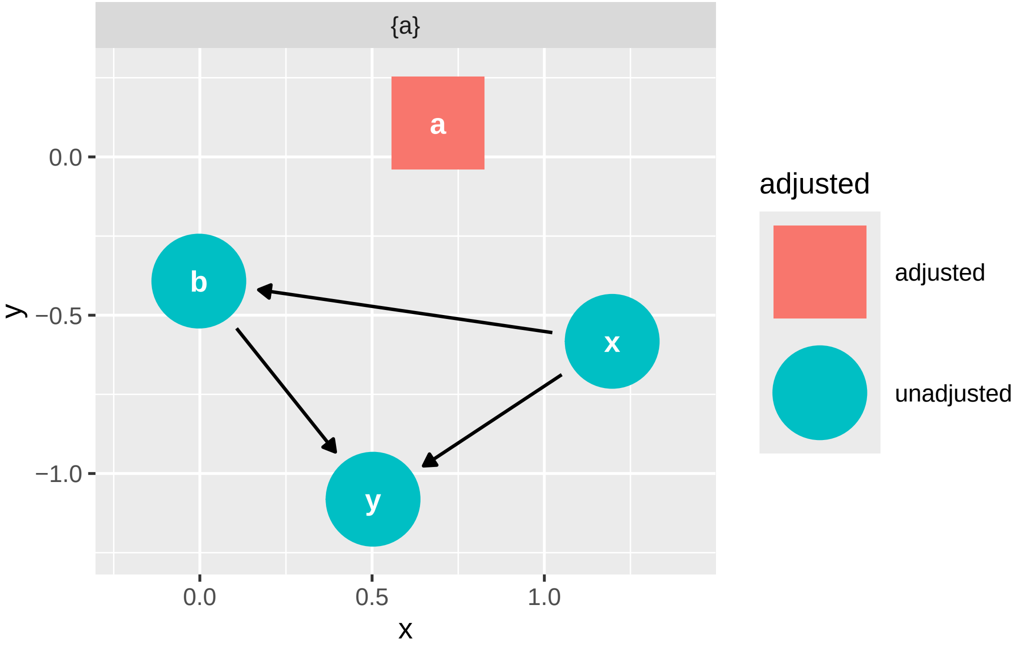
```
ggdag(dag)
```



```
ggdag_adjustment_set(dag)
```



```
# What if x causes b?
dag2<-dagify(
  y~ x + a + b,
  x~ a,
  b~ a+x,
  exposure="x",
  outcome="y"
)
ggdag_adjustment_set(dag2)
```



12 Session info

```
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] ggdag_0.2.13    lubridate_1.9.4 forcats_1.0.0   stringr_1.5.1
 [5] dplyr_1.1.4     purrr_1.0.4     readr_2.1.5     tidyr_1.3.1
 [9] tibble_3.2.1    ggplot2_3.5.2   tidyverse_2.0.0

loaded via a namespace (and not attached):
 [1] viridis_0.6.5      utf8_1.2.4         generics_0.1.3     stringi_1.8.7
 [5] hms_1.1.3          digest_0.6.37      magrittr_2.0.3     evaluate_1.0.3
 [9] grid_4.5.0         timechange_0.3.0   fastmap_1.2.0      jsonlite_2.0.0
[13] ggrepel_0.9.6      tinytex_0.57       gridExtra_2.3      dagitty_0.3-4
[17] viridisLite_0.4.2  scales_1.3.0       tweenr_2.0.3       cli_3.6.4
```

```
[21] graphlayouts_1.2.2 rlang_1.1.6        polyclip_1.10-7    tidygraph_1.3.1
[25] munsell_0.5.1      cachem_1.1.0       withr_3.0.2        yaml_2.3.10
[29] tools_4.5.0        tzdb_0.5.0         memoise_2.0.1      colorspace_2.1-1
[33] boot_1.3-31        curl_6.2.2         vctrs_0.6.5        R6_2.6.1
[37] lifecycle_1.0.4    V8_6.0.3           MASS_7.3-65        ggraph_2.2.1
[41] pkgconfig_2.0.3    pillar_1.10.2      gtable_0.3.6       glue_1.8.0
[45] Rcpp_1.0.14        ggforce_0.4.2      xfun_0.52          tidyselect_1.2.1
[49] knitr_1.50         farver_2.1.2       htmltools_0.5.8.1  igraph_2.1.4
[53] labeling_0.4.3     rmarkdown_2.29     compiler_4.5.0
```

13 Time series causal impact with CausalImpact

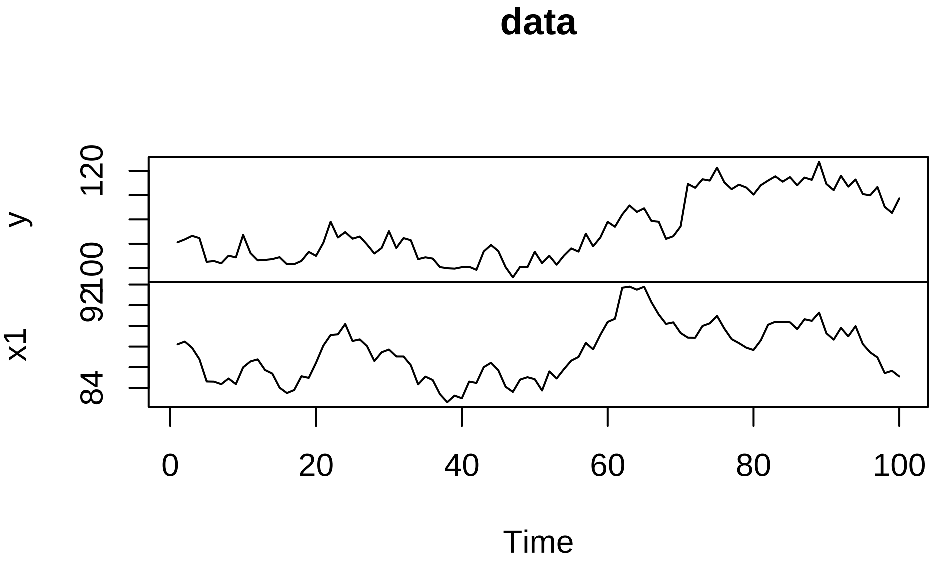
14 Load libraries

```r
library(CausalImpact) # R package for determining
library(dplyr) # R package for data wrangling
library(ggplot2) # R package for plotting
library(gt) # R package for constructing tables

#.libPaths()
```

15 Simulate synthetic control data and focal time series data

Following this tutorial: https://google.github.io/CausalImpact/CausalImpact.html

```r
set.seed(1)
x1 <- 100 + arima.sim(model = list(ar = 0.999), n = 100) # createa  control variable
y <- 1.2 * x1 + rnorm(100) # create the quality metric variable that is dependent on x1 (con
y[71:100] <- y[71:100] + 10
data <- cbind(y, x1) # combine the datasets
plot(data) # plot the datasets, roughly
```



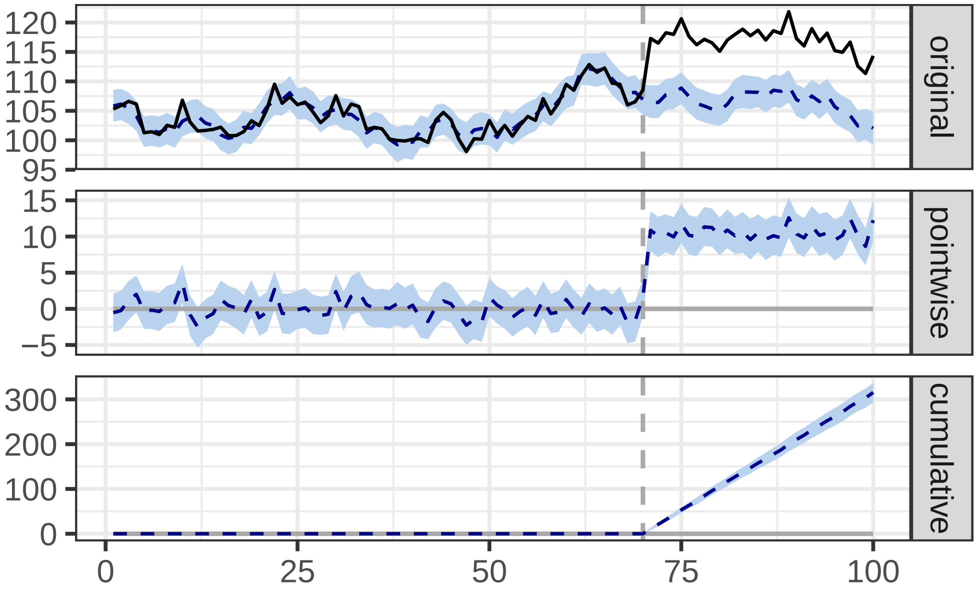
16 What the simulated data look like:

```r
#let's see what the data look like
head(round(tibble(y1=data[,1],x1=data[,2]),1),6)|>
  gt()
```

y1	x1
105.3	88.2
105.9	88.5
106.6	87.9
106.2	86.8
101.3	84.6
101.4	84.6

17 Run analysis

```
pre<-c(1,70) # set the pre period with no intervention
post<-c(71,100) # set the post period, after the intervention

impact<-CausalImpact(data,pre,post) # Conduct the analysis

plot(impact) # plot the results
```



```
#summary(impact,"report")
```

18 Analysis report from the R package

Analysis report {CausalImpact}

During the post-intervention period, the response variable had an average value of approx. 117.05. By contrast, in the absence of an intervention, we would have expected an average response of 106.54. The 95% interval of this counterfactual prediction is [105.84, 107.29]. Subtracting this prediction from the observed response yields an estimate of the causal effect the intervention had on the response variable. This effect is 10.51 with a 95% interval of [9.76, 11.21]. For a discussion of the significance of this effect, see below.

Summing up the individual data points during the post-intervention period (which can only sometimes be meaningfully interpreted), the response variable had an overall value of 3.51K. By contrast, had the intervention not taken place, we would have expected a sum of 3.20K. The 95% interval of this prediction is [3.18K, 3.22K].

The above results are given in terms of absolute numbers. In relative terms, the response variable showed an increase of +10%. The 95% interval of this percentage is [+9%, +11%].

This means that the positive effect observed during the intervention period is statistically significant and unlikely to be due to random fluctuations. It should be noted, however, that the question of whether this increase also bears substantive significance can only be answered by comparing the absolute effect (10.51) to the original goal of the underlying intervention.

The probability of obtaining this effect by chance is very small (Bayesian one-sided tail-area probability p = 0.001). This means the causal effect can be considered statistically significant.

18.1 Output table

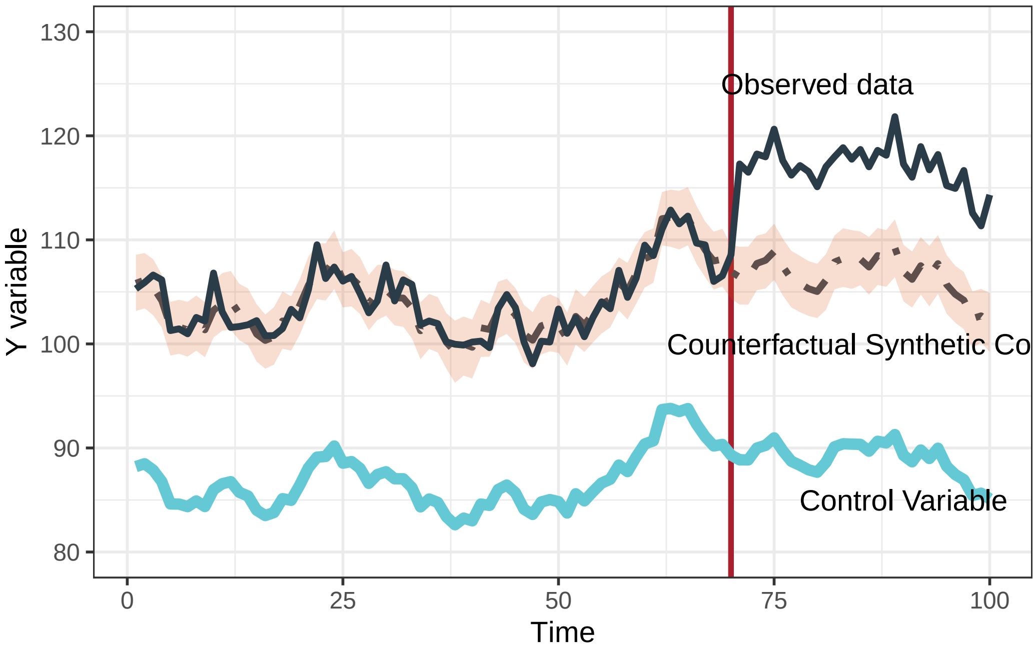
```r
knitr::kable(t(round(impact$summary,2)))
```

	Average	Cumulative
Actual	117.05	3511.46
Pred	106.54	3196.12

	Average	Cumulative
Pred.lower	105.84	3175.10
Pred.upper	107.29	3218.60
Pred.sd	0.37	11.17
AbsEffect	10.51	315.34
AbsEffect.lower	9.76	292.85
AbsEffect.upper	11.21	336.36
AbsEffect.sd	0.37	11.17
RelEffect	0.10	0.10
RelEffect.lower	0.09	0.09
RelEffect.upper	0.11	0.11
RelEffect.sd	0.00	0.00
alpha	0.05	0.05
p	0.00	0.00

19 stylized figure

```
#splitting out datasets
#names(impact$series)


orig<-impact$series|>
  data.frame()|>
  tibble()|>
  dplyr::select(response,point.pred,point.pred.lower,point.pred.upper)

cf<-ggplot(orig,aes(x=seq(1,100,1),y=point.pred))+geom_vline(xintercept=70,colour='#AA1E2D',
cf
```



```
#ggsave(cf,filename="Timeseriesbayesian.png",width=8,height=5,dpi=600,unit="in")
#geom_text(aes(x=c(10,75),y=c(90,125),label=c("Synthetic Control","Observed data")))+


#cf1<-ggplot(orig,aes(x=seq(1,100,1),y=point.pred))+geom_vline(xintercept=70,colour='#AA1E2D
#cf1
#ggsave(cf1,filename="observed_data_timeseries.png",width=10,height=5,unit="in",dpi=600)


#cf2<-ggplot(orig,aes(x=seq(1,100,1),y=point.pred))+geom_vline(xintercept=70,colour='#AA1E2D
#cf2
#ggsave(cf2,filename="observed_data_timeseries_with_counterfactual.png",width=10,height=5,uni
```

20 Session Info

```r
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] gt_1.0.0            ggplot2_3.5.2       dplyr_1.1.4
[4] CausalImpact_1.3.0  bsts_0.9.10         xts_0.14.1
[7] zoo_1.8-14          BoomSpikeSlab_1.2.6 Boom_0.9.15

loaded via a namespace (and not attached):
 [1] gtable_0.3.6      jsonlite_2.0.0    compiler_4.5.0    tidyselect_1.2.1
 [5] tinytex_0.57      xml2_1.3.8        assertthat_0.2.1  scales_1.3.0
 [9] yaml_2.3.10       fastmap_1.2.0     lattice_0.22-6    R6_2.6.1
[13] labeling_0.4.3    generics_0.1.3    knitr_1.50        MASS_7.3-65
[17] tibble_3.2.1      munsell_0.5.1     pillar_1.10.2     rlang_1.1.6
```

```
[21] xfun_0.52         cli_3.6.4         withr_3.0.2       magrittr_2.0.3
[25] digest_0.6.37     grid_4.5.0        lifecycle_1.0.4   vctrs_0.6.5
[29] evaluate_1.0.3    glue_1.8.0        farver_2.1.2      colorspace_2.1-1
[33] rmarkdown_2.29    tools_4.5.0       pkgconfig_2.0.3   htmltools_0.5.8.1
```

Part II

Notes on statistical models

21 Accelerated Failure time models

22 Load libraries

```
library(tidyr)
library(ggplot2)
library(dplyr)
library(survival)
library(survminer)
library(hrbrthemes)

#colour range
oh_cols<- c('#65C9D5',  '#EDE668',  '#AA1E2D',  '#F26828',  '#FDCEB0',  '#C3C3C8',  '#74308C
cc<-colorRampPalette(c('#65C9D5', '#AA1E2D'))

my_colors <- cc(500)

cc1<-colorRampPalette(c('#C3C3C8','#74308C'))
my_colors1 <- cc1(10)
```

23 Accelerated failture time (AFT) model

Notes based on https://univ-pau.hal.science/hal-02953269/document

$$log(T) = \beta_0 + \beta' X + \sigma\epsilon$$

- T is the given survival time
- β_0 is the intercept

- β' is the set of slope parameters
- X is the vector of covariates
- ϵ is the error term
- σ is the additional parameter that scales the error

The AFT model assumes that covariates have a multiplicative effect on the survival time and an additive effect on $log(T)$. And we can isolate T as:

$$T = exp(\beta_0) \times exp(\beta' X) \times exp(\sigma\epsilon)$$

23.1 The Exponential distribution

The exponential distribution has the survival function, $S_T(t) = e^{-\lambda t}$ for all $t \geq 0, \lambda > 0$ and the hazard function is constant, $h_T(t) = \lambda$. If the lifetime of T is exponential, then ϵ follows a Gumbel distribution witht he survival function $S_\epsilon(y) = exp(-e^y)$ and obtain:

$$S_{T|X}(t|x) = exp(-\lambda t); \frac{1}{\lambda} = exp(\beta_0 + \beta' x)$$

23.2 The Weibull distribution

The survival function:

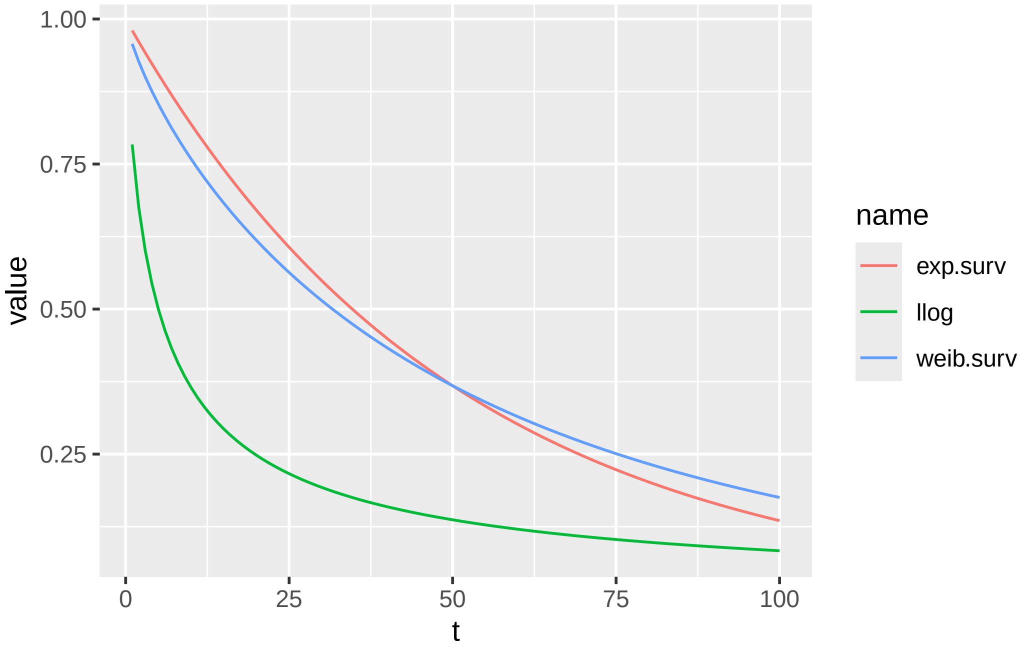
$$S_T(t) = e^{(-\lambda t)^\alpha}$$

where…

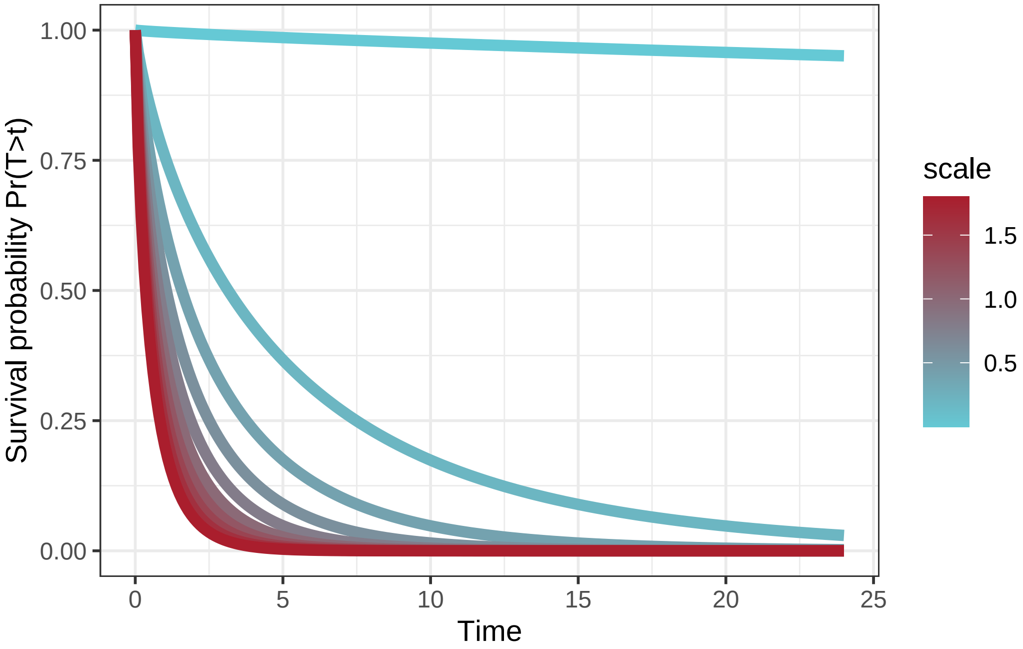
- α is the shape parameter
- λ is the scale parameter

In an AFT regression, $\frac{1}{\lambda} = exp(\beta_0 + \beta' x)$ and $\alpha = \frac{1}{\sigma}$. The exponential model is a special case of the weibull model with a shape parameter equal to 1.

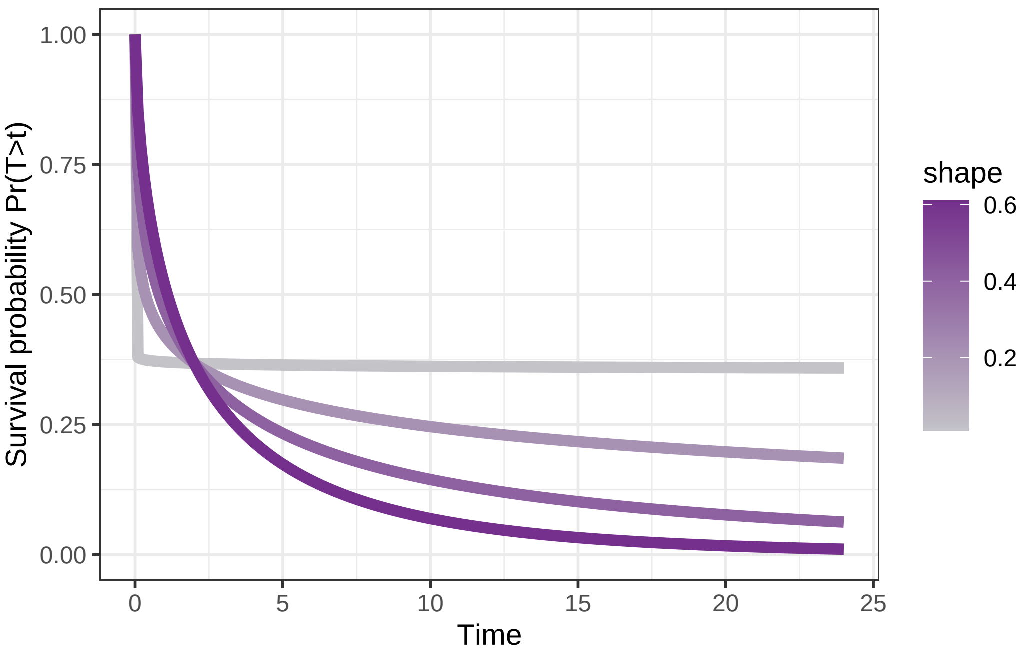
```
expo<-tibble(t=1:100,exp.surv=exp(-.02*t),llog=1/(1+(.2*t)^.8),weib.surv=exp(-(.02*t)^.8))
expo1<-expo|>
  pivot_longer(cols=exp.surv:weib.surv)
ggplot(expo1,aes(x=t,y=value,color=name))+geom_line()
```



```
#let's simulate different weibull
#simulate different scale parameters
dat<-expand.grid(scale=seq(0.001,2,.2),time=seq(0,24,.1))|>
  group_by(scale)|>
  mutate(weib=exp(-(scale*time)^.8))

ggplot(dat,aes(x=time,y=weib,colour=scale,group=scale,fill=scale))+geom_line(linewidth=2)+sca
```



```
#Higher scale values = quicker drop in survival probability.


#now simulate different shape parameters
dat2<-expand.grid(shape=seq(0.01,.8,.2),time=seq(0,24,.1))|>
  group_by(shape)|>
  mutate(weib=exp(-(.5*time)^shape))



ggplot(dat2,aes(x=time,y=weib,colour=shape,group=shape,fill=shape))+geom_line(linewidth=2)+s
```



```
#higher shape values lead to steeper drop
```

24 Simulating cox coefficients

```
#simulating beta

#exp(.16)
loghaz1<-.16 #continuous regression coefficient
x<-1:10 #continuous variable

d<-tibble(haz=exp(loghaz1*x)# coefficient associated with hazard rate
          ,age=x)|>
  mutate(loghaz=log(haz),
         loghazcoeff=loghaz1#coefficient of hazard rate with one unit increase of continuous
         )
d

ggplot(d,aes(x=age,y=loghaz))+geom_line()+geom_point()
ggplot(d,aes(x=age,y=haz))+geom_line()+geom_point()

#simulate curves
d2<-d|>
  group_by(haz)|>
  do(hazfun=exp(.$haz*x))

d2|>
```

25 analyzing high risk patient data for length of stay

```
d<-read.csv("07_30daysurvival_dataset-parsed-foranalysis.csv")

#fit aft model
d$losstatus<-1
aft1<-survreg(Surv(los,losstatus)~1,data=d,dist="weibull")
summary(aft1)

#exp(-.1882)
#ws<-exp(summary(aft1)$coef)
ws<-1/exp(summary(aft1)$coef)
wsc<-1/aft1$scale
#plot out model
modfit<-tibble(surv=dweibull(x=seq(1,30,1),shape=1/exp(summary(aft1)$coef),scale=wsc),time=se
ggplot(modfit,aes(x=time,y=surv))+geom_line()+scale_x_continuous(labels=seq(0,30,2),breaks=se

#make predictions from model and compare to raw data
#predict(aft1,type="quantile",p=c(0.25,.5,.75))
#prediction equation
#exp(-(.02*t)^.8)
time<-seq(0,30,.01)
pred<-tibble(time,surv=exp(-(ws*time)^wsc))

#compare with km plots
km<-survfit(Surv(los,losstatus)~1,data=d)
kmd<-tibble(time=summary(km)$time,surv=summary(km)$surv)


ggplot(pred,aes(x=time,y=surv))+geom_line()+geom_line(data=kmd,aes(x=time,y=surv),color="blu
```

26 Session info

```
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] hrbrthemes_0.8.7 survminer_0.5.0  ggpubr_0.6.0     survival_3.8-3
[5] dplyr_1.1.4      ggplot2_3.5.2    tidyr_1.3.1

loaded via a namespace (and not attached):
 [1] generics_0.1.3         fontLiberation_0.1.0   rstatix_0.7.2
 [4] lattice_0.22-6         extrafontdb_1.0        digest_0.6.37
 [7] magrittr_2.0.3         evaluate_1.0.3         grid_4.5.0
[10] fastmap_1.2.0          jsonlite_2.0.0         Matrix_1.7-3
[13] backports_1.5.0        Formula_1.2-5          tinytex_0.57
[16] gridExtra_2.3          purrr_1.0.4            scales_1.3.0
```

```
[19] fontBitstreamVera_0.1.1 abind_1.4-8             cli_3.6.4
[22] fontquiver_0.2.1        KMsurv_0.1-5            rlang_1.1.6
[25] munsell_0.5.1           splines_4.5.0           withr_3.0.2
[28] yaml_2.3.10             gdtools_0.4.2           tools_4.5.0
[31] ggsignif_0.6.4          colorspace_2.1-1        km.ci_0.5-6
[34] broom_1.0.8             vctrs_0.6.5             R6_2.6.1
[37] zoo_1.8-14              lifecycle_1.0.4         car_3.1-3
[40] pkgconfig_2.0.3         pillar_1.10.2           gtable_0.3.6
[43] Rcpp_1.0.14             glue_1.8.0              data.table_1.17.0
[46] systemfonts_1.2.2       xfun_0.52               tibble_3.2.1
[49] tidyselect_1.2.1        knitr_1.50              farver_2.1.2
[52] extrafont_0.19          xtable_1.8-4            survMisc_0.5.6
[55] htmltools_0.5.8.1       labeling_0.4.3          rmarkdown_2.29
[58] carData_3.0-5           Rttf2pt1_1.3.12         compiler_4.5.0
```

27 Longitudinal ordinal regression model simulations

28 Load libraries

```r
library(tidyverse)
library(brms)
library(ggbeeswarm)
library(MASS) # fit ordinal logistic regression
library(brant) # check proportional odds assumption
library(marginaleffects) # contrats, g-computation
library(patchwork) # visualization
library(ordinal) # fitting longitudinal ordinal model
library(ggeffects)# plotting long ordinal model
library(tidybayes)
oh_cols<- c('#50BECB',  '#46A6B2',  '#65C9D5',  '#97D3DC',  '#CDEBF0',  '#EDE668',  '#AA1E2D
oh_cols<- c('#65C9D5',  '#EDE668',  '#AA1E2D',  '#F26828',  '#FDCEB0',  '#C3C3C8',  '#74308C
```

29 Cross-sectional ordinal design

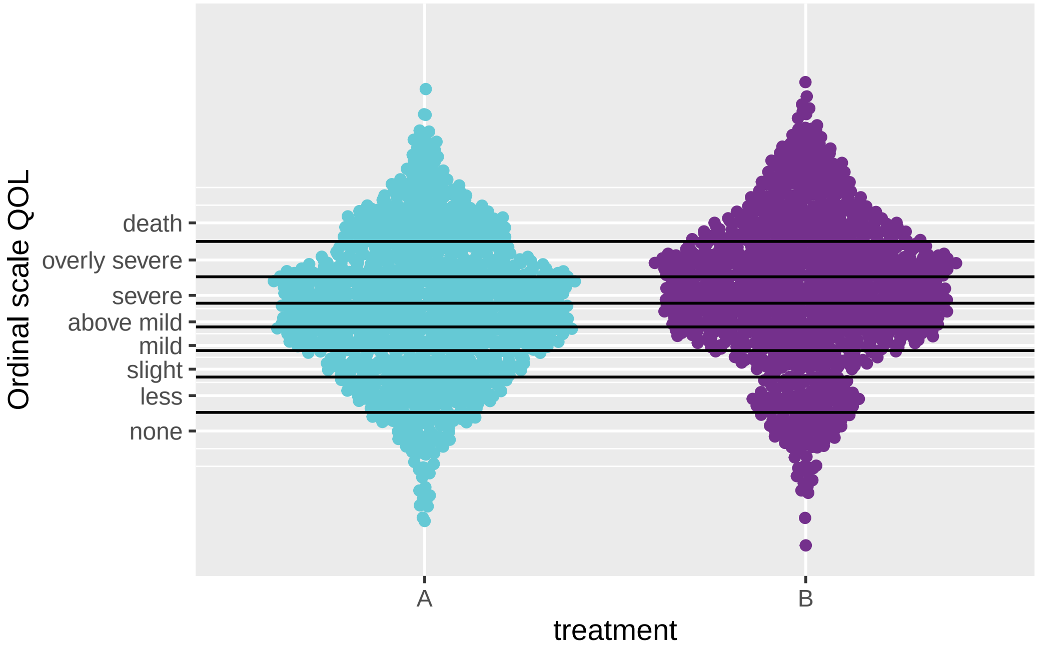
29.1 Simulating ordinal data

Simulating a RCT, treatment A vs treatment B, and their impact on quality of life. How to simulate?

- For treatment B, Draw from a normal distribution, normal (100, std=20).
- Split normal distribution based on 7 cut offs ; so there are 8 ordinal categories
- Sample treatment A from normal (110,std=20), and categorize based on treatment B splits/categories.

```
#sample 1000 patients
n<-1000

od<-tibble(A=rnorm(n=n,mean=105,sd=20),B=rnorm(n=n,mean=110,sd=20))|>
  pivot_longer(names_to = "treatment",values_to = "num",A:B)
#get  cutoffs
probs=seq(0,1,1/8)[2:8]
#get quantiles
quantiles <- qnorm(probs, mean = 100, sd = 20)#

od<-od|>
  mutate(QOL=if_else(num<quantiles[1],1,if_else(num<quantiles[2],2,if_else(num<quantiles[3],3
  mutate(QOL=factor(as.character(QOL)))

#compare ordinal values between groups
#ggplot(od,aes(x=treatment,y=QOL,colour=treatment))+geom_quasirandom()
#visualize the data
fig5<-ggplot(od,aes(x=treatment,y=num,colour=treatment))+geom_quasirandom()+theme(legend.posi
fig5
```

```
Warning: Removed 3 rows containing missing values or values outside the scale range
(`position_quasirandom()`).
```



```
ggsave(fig5,filename="01_QOL-ordinal_vs_treatmentA-B_crosssectional.png",unit="in",dpi=600,w
```

```
Warning: Removed 3 rows containing missing values or values outside the scale range
(`position_quasirandom()`).
```

29.2 Fit ordinal logistic mode

```
# ordinal model
mod1 <- polr(QOL~ treatment, data = od, Hess=TRUE)
summary(mod1) # model output
```

```
Call:
polr(formula = QOL ~ treatment, data = od, Hess = TRUE)

Coefficients:
            Value Std. Error t value
treatmentB 0.3824    0.07875   4.856

Intercepts:
    Value    Std. Error t value
1|2  -2.3899   0.0935   -25.5513
2|3  -1.5277   0.0714   -21.4012
3|4  -1.0181   0.0642   -15.8554
4|5  -0.4327   0.0601    -7.1975
5|6   0.1110   0.0594     1.8702
6|7   0.6737   0.0613    10.9911
7|8   1.5148   0.0692    21.8855

Residual Deviance: 8033.123
AIC: 8049.123
```

```r
exp(coef(mod1)) # treatment B has a 2.2 increased odds ratio in QOL than treatment A
```

```
treatmentB
  1.465828
```

```r
#check proportional odds
brant(mod1)
```

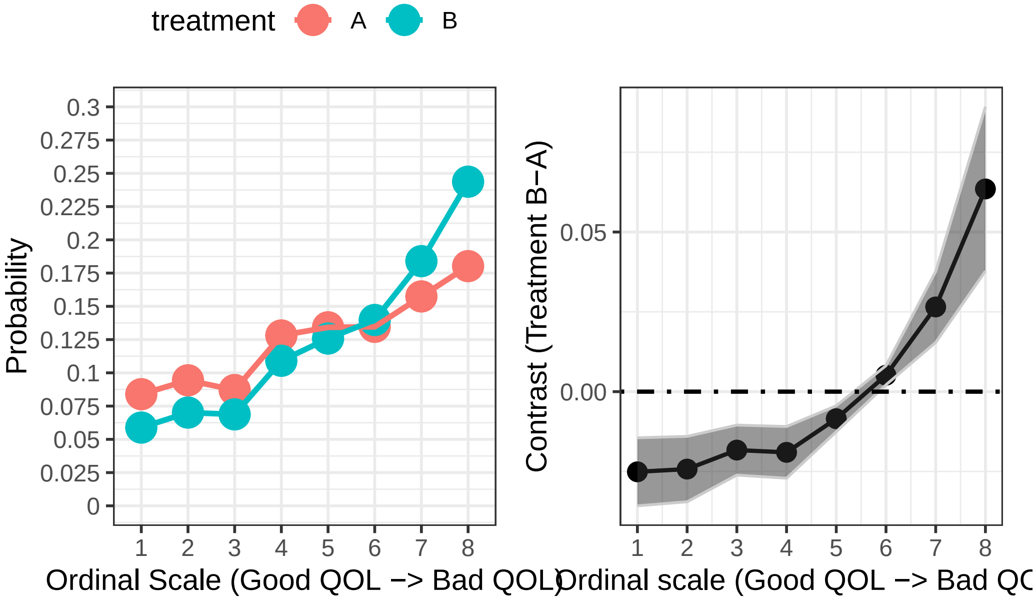
```
--------------------------------------------
Test for    X2  df  probability
--------------------------------------------
Omnibus     6.47    6   0.37
treatmentB  6.47    6   0.37
--------------------------------------------

H0: Parallel Regression Assumption holds
```

```r
##predict values
new.dat<-data.frame(treatment=c("A","B"))

#get predictions
npred<-predict(mod1,new.dat,type="probs")|>
  data.frame()|>
  mutate(treatment=c("A","B"))|>
  pivot_longer(X1:X8,names_to = "Ordinal",values_to="Probability")|>
  mutate(ord.num=substr(Ordinal,2,2))

#ggplot(npred,aes(x=treatment,y=Probability,colour=treatment))+geom_point(size=5)+facet_wrap
```

```
#plot on more continuous scale
fig1<-ggplot(npred,aes(x=ord.num,y=Probability,colour=treatment,group=treatment))+geom_point
```

29.3 Let's see if we can conduct g-computation

```
#try marginaleffects
nd<-expand.grid(treatment=c("A","B"),ind=1:1000)
#gcomp<-avg_comparisons(mod1,variables = "treatment",newdata=nd)
gcomp<-avg_comparisons(mod1,variables = "treatment",newdata=datagrid(newdata = od,grid_type="
gdat<-gcomp|>
  broom::tidy()

fig2<-ggplot(gdat,aes(x=1:8,y=estimate))+geom_point(size=3)+geom_line(linewidth=.75)+geom_ri

fig12<-fig1+fig2
fig12
```



```
ggsave(fig12,filename="01_two_panel_ordinal_scale_contrast_treatmentA_treatmentB.png",width=
```

30 Longitudinal ordinal design

Simulating RCT, treatment A vs B, and their impact on quality of life. QOL is tracked over time. Treatment A reduces QOL at a faster rate than treatment B. How to simulate?

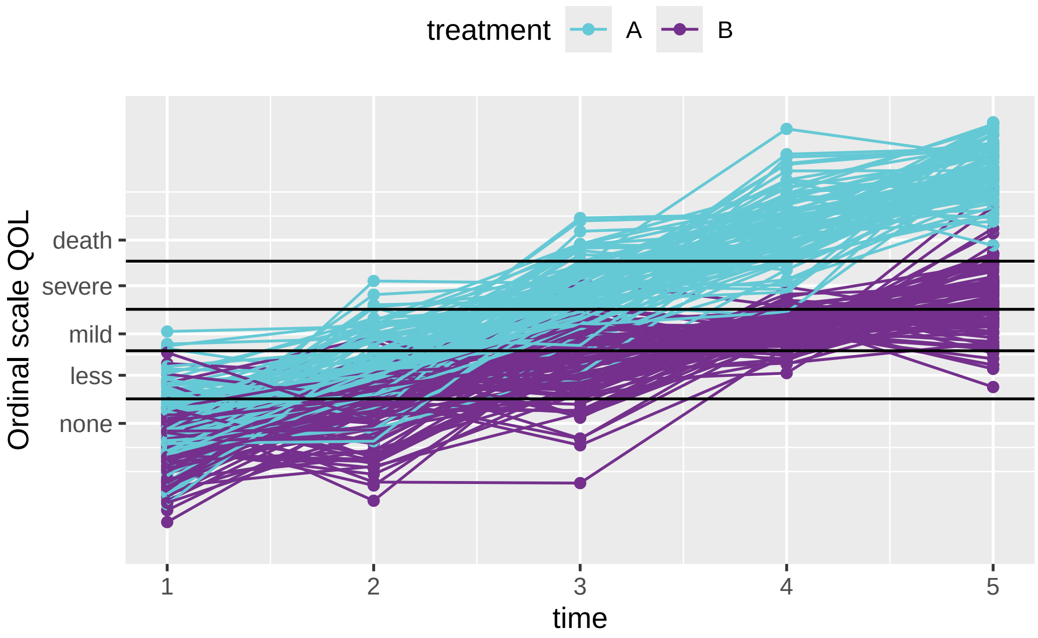
- Get a global normal distribution, normal(200,25) and split data
- Have 4 ordinal levels (none, low, mild, severe, death)
- Have 5 time points -> increase mean from 120, every 20 over each time point for treatment B, for treatment A increase from 120, every 40 over each time point.

```r
#number of patients
n<-100
#get  cutoffs
probs=seq(0,1,1/5)[2:5]
#get quantiles
quantiles <- qnorm(probs, mean = 180, sd = 50)#

tp1<-tibble(A=rnorm(n=n,mean=120,sd=20),B=rnorm(n=n,mean=110,sd=20),time=1)|>
  pivot_longer(names_to = "treatment",values_to = "num",A:B)|>
  mutate(id=1:length(num))

tp2<-tibble(A=rnorm(n=n,mean=160,sd=20),B=rnorm(n=n,mean=130,sd=20),time=2)|>
  pivot_longer(names_to = "treatment",values_to = "num",A:B)|>
  mutate(id=1:length(num))

tp3<-tibble(A=rnorm(n=n,mean=200,sd=20),B=rnorm(n=n,mean=160,sd=20),time=3)|>
  pivot_longer(names_to = "treatment",values_to = "num",A:B)|>
  mutate(id=1:length(num))

tp4<-tibble(A=rnorm(n=n,mean=240,sd=20),B=rnorm(n=n,mean=180,sd=10),time=4)|>
  pivot_longer(names_to = "treatment",values_to = "num",A:B)|>
  mutate(id=1:length(num))

tp5<-tibble(A=rnorm(n=n,mean=280,sd=20),B=rnorm(n=n,mean=200,sd=20),time=5)|>
  pivot_longer(names_to = "treatment",values_to = "num",A:B)|>
  mutate(id=1:length(num))
```

```
#combine longitudinal data
tpd<-rbind(tp1,tp2,tp3,tp4,tp5)|>
  arrange(id,time,treatment)|>
  mutate(QOL=if_else(num<quantiles[1],1,if_else(num<quantiles[2],2,if_else(num<quantiles[3],


##spagehtti plots

fig4<-ggplot(tpd,aes(x=time,y=num,colour=treatment,group=factor(id)))+geom_point()+geom_line
fig4
```

```
Warning: Removed 8 rows containing missing values or values outside the scale range
(`geom_point()`).
```

```
Warning: Removed 8 rows containing missing values or values outside the scale range
(`geom_line()`).
```



```
#ggsave(fig4,filename="01_ordinal_scale_fig_ztreatmentA_B_vstime_longitudinal.png",width=5,he
```

30.1 Fitting longitudinal ordinal regression model

frequentist doesnt work well for estimating contrasts, but I'm going to try with the brms package (bayesian)

(didn't run this code because it takes too long)

```
#set up the model
#ordinal longitudinal random effects model
#random intercept and random slope
#mod2<-clmm(QOL~treatment+time+(1+time|id),data=tpd)
tpd$QOL <- as.ordered(tpd$QOL)
mod2<-brm(QOL~treatment+time+(1+time|id),data=tpd,family=cumulative(),iter = 1000)
#prior=set_prior("normal(0,5)",class="b")
summary(mod2)
# I should save the model, bc it takes forever to run
saveRDS(mod2,"Output_datasets/longitudinal_ordinal_bayesian_brmsmodel_simulateddata_time_and_



####model checks
##check the model:
pp_check(mod2)
#trace plots
mcmc_plot(mod2, type = "trace")
mcmc_plot(mod2, type = "dens_overlay")

#
# Generate posterior predictions
predictions <- add_predicted_draws(mod2, newdata = tpd)

# Plot predictions
ggplot(predictions, aes(x = time, y = .prediction, color = treatment)) +
  geom_line() +
  labs(title = "Posterior Predictive Distribution")




##########



#################3g-comp
```

```
##g-computation
nd1<-expand.grid(treatment=c("A","B"),ind=1:100)

gcomp2<-avg_comparisons(mod2,variables = "treatment",newdata=datagrid(newdata = tpd,grid_type
#predict(mod2,nd1,type="probs")

gcomp2
```

31 Session Info

```r
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] tidybayes_3.0.7        ggeffects_2.2.1        ordinal_2023.12-4.1
 [4] patchwork_1.3.0        marginaleffects_0.25.1 brant_0.3-0
 [7] MASS_7.3-65            ggbeeswarm_0.7.2       brms_2.22.0
[10] Rcpp_1.0.14            lubridate_1.9.4        forcats_1.0.0
[13] stringr_1.5.1          dplyr_1.1.4            purrr_1.0.4
[16] readr_2.1.5            tidyr_1.3.1            tibble_3.2.1
[19] ggplot2_3.5.2          tidyverse_2.0.0

loaded via a namespace (and not attached):
 [1] gtable_0.3.6         beeswarm_0.4.0       tensorA_0.36.2.1
 [4] xfun_0.52             insight_1.1.0        lattice_0.22-6
 [7] numDeriv_2016.8-1.1   tzdb_0.5.0           vctrs_0.6.5
[10] tools_4.5.0           generics_0.1.3       parallel_4.5.0
[13] ucminf_1.2.2          pkgconfig_2.0.3      Matrix_1.7-3
[16] data.table_1.17.0     checkmate_2.3.2      distributional_0.5.0
[19] RcppParallel_5.1.10   lifecycle_1.0.4      farver_2.1.2
[22] compiler_4.5.0        textshaping_1.0.0    Brobdingnag_1.2-9
[25] munsell_0.5.1         tinytex_0.57         vipor_0.4.7
[28] htmltools_0.5.8.1     bayesplot_1.12.0     yaml_2.3.10
[31] pillar_1.10.2         arrayhelpers_1.1-0   bridgesampling_1.1-2
[34] abind_1.4-8           nlme_3.1-168         posterior_1.6.1
[37] svUnit_1.0.6          tidyselect_1.2.1     digest_0.6.37
[40] mvtnorm_1.3-3         stringi_1.8.7        labeling_0.4.3
[43] fastmap_1.2.0         grid_4.5.0           colorspace_2.1-1
[46] cli_3.6.4             magrittr_2.0.3       loo_2.8.0
[49] broom_1.0.8           withr_3.0.2          scales_1.3.0
[52] backports_1.5.0       timechange_0.3.0     rmarkdown_2.29
[55] matrixStats_1.5.0     ragg_1.4.0           hms_1.1.3
[58] coda_0.19-4.1         evaluate_1.0.3       knitr_1.50
[61] ggdist_3.3.2          rstantools_2.4.0     rlang_1.1.6
[64] glue_1.8.0            jsonlite_2.0.0       R6_2.6.1
[67] systemfonts_1.2.2
```

Part III

Misc Bayesian statistics

32 Bayesian inference in discrete case

Date: 2025-04-19

33 Load libraries

```r
library(tidyverse)
```

34 Bayesian inference in discrete case: simple example

The scenario (taken from Dr. Jingchen (Monika) Hu from youtube, S22 Math 347 course): A chinese food restaurant owner wants to increase the business profits and wants to know when people prefer to come into her restaurant. Specifically, she is interested in how often people chose Friday. So, Friday = "success", and all other days are considered "failures". She wants to use a Bayesian approach to estimate the probability that patrons will believe Friday is their favorite day to eat.

34.1 Steps

She wants to use a Bayesian approach and involves the following steps:

1. **Set up the prior expectations of success** $\pi(p)$ or $\pi(success)$
2. **Collect data and estimate the likelihood** -> use binomial distribution

Likelihood of p and Binomial probability mass function (pmf):

$$\pi(y|p_i) = L(p_i) = P(Y = y) = \binom{n}{y} p^y (1-p)^{n-y}$$

Assumptions of binomial experiment:

1. repeating same task/trial many times

2. on each trial, 2 possible outcomes: "success" or "failure"

3. Prob of success, p, same for each trial

4. Results of outcomes from different trials are independent

3. **Apply Baye's rule**

Bayes rule:

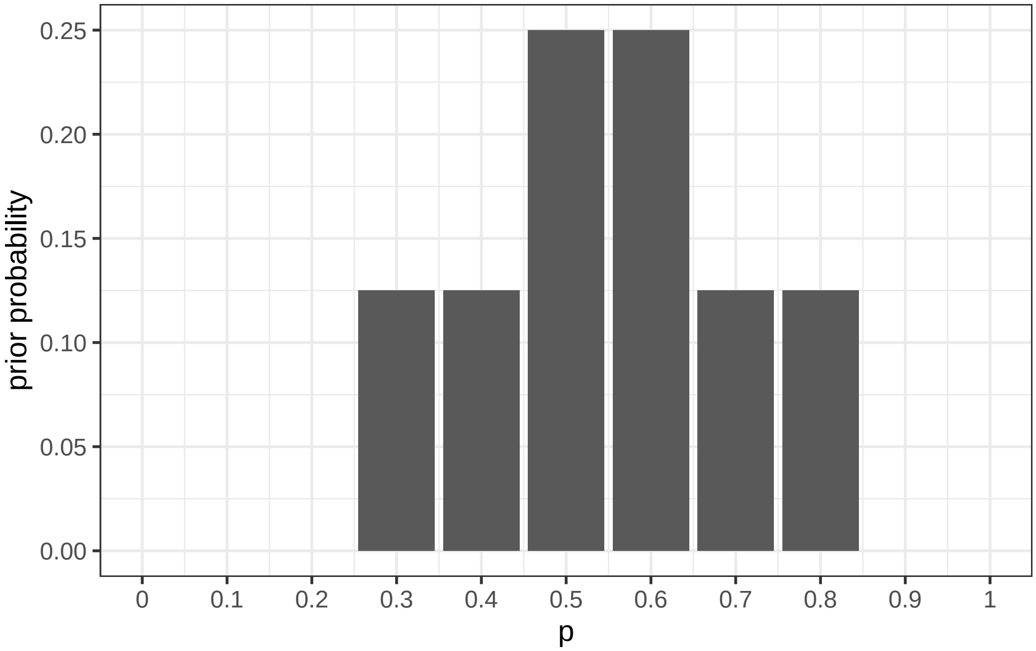
$$\pi(p_i|y) = \frac{\pi(y|p_i) \times \pi(p_i)}{\pi(y)}$$

$$\pi(y) = \sum_j \pi(p_j \times L(p_j))$$

The denominator gives the marginal distribution of the observation y by the law of total probability.

34.2 Set up prior $\pi(p)$

```
#probabilities of success to consider
p<-seq(.3,.8,.1)
#p

#probabilities for each of p
prior<-c(.125,.125,.25,.25,.125,.125)

d<-tibble(prior,p)

ggplot(d,aes(x=p,y=prior))+geom_bar(stat="identity")+theme_bw()+scale_x_continuous(limits=c(
```



34.3 Calculate likelihood -binomial

She surveyed 20 patrons and 12 chose Friday. So this looks like

$$L(p_i) = \binom{20}{12} p^{12} \times (1-p)^{20-12}$$

```
#use the density binomial function , dbinom()
d$likelihood<-dbinom(x=12,size=20,prob=d$p)
knitr::kable(d)
```

prior	p	likelihood
0.125	0.3	0.0038593
0.125	0.4	0.0354974
0.250	0.5	0.1201344
0.250	0.6	0.1797058
0.125	0.7	0.1143967
0.125	0.8	0.0221609

34.4 Apply Baye's rule and calculate the posterior probability ($\pi(p_i|y)$)

$\pi(p_i|y)$ is the posterior probability of $p = p_i$ given the number of successes y.

```
d$marg<-sum(d$prior*d$likelihood)

d$posterior<-(d$prior*d$likelihood)/d$marg

#plot table
knitr::kable(d)
```

prior	p	likelihood	marg	posterior
0.125	0.3	0.0038593	0.0969493	0.0049759
0.125	0.4	0.0354974	0.0969493	0.0457680
0.250	0.5	0.1201344	0.0969493	0.3097865
0.250	0.6	0.1797058	0.0969493	0.4634013
0.125	0.7	0.1143967	0.0969493	0.1474955
0.125	0.8	0.0221609	0.0969493	0.0285728

```
#let's plot everything out
#ggplot(d,aes(x=p,y=posterior))+geom_point()
```

34.4.1 inferential question: What is the posterior prob that over half of the customers prefer to eat out on friday for dinner?

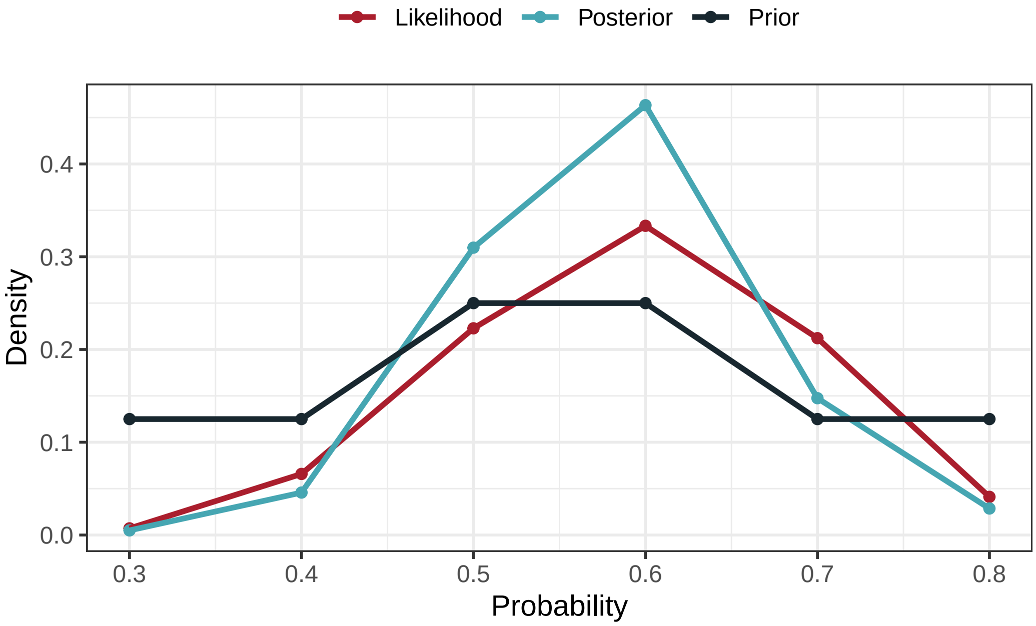
```
an<-d|>
  filter(p>.5)|>
  dplyr::summarise(oh=sum(posterior))
```

$Prob(p > 0.5) = 0.639469613008657$

34.4.2 Let's plot out the prior, likelihood, and posterior

I'm going to normalize the likelihood function with 3x the max for plottig purposes.

```
d$sl<-d$likelihood/(max(d$likelihood)*3)

d2<-d|>
  select(p,prior,sl,posterior)|>
  pivot_longer(prior:posterior)|>
  mutate(parameter=if_else(name=="prior","Prior",if_else(name=="sl","Likelihood","Posterior"


ggplot(d2,aes(x=p,y=value,colour=parameter))+geom_point()+geom_line(linewidth=1)+xlab("Proba
```



35 Sessioninfo

```r
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] lubridate_1.9.4 forcats_1.0.0   stringr_1.5.1   dplyr_1.1.4
 [5] purrr_1.0.4     readr_2.1.5     tidyr_1.3.1     tibble_3.2.1
 [9] ggplot2_3.5.2   tidyverse_2.0.0

loaded via a namespace (and not attached):
 [1] gtable_0.3.6      jsonlite_2.0.0    compiler_4.5.0    tidyselect_1.2.1
 [5] tinytex_0.57      scales_1.3.0      yaml_2.3.10       fastmap_1.2.0
 [9] R6_2.6.1          labeling_0.4.3    generics_0.1.3    knitr_1.50
[13] munsell_0.5.1     pillar_1.10.2     tzdb_0.5.0        rlang_1.1.6
[17] stringi_1.8.7     xfun_0.52         timechange_0.3.0  cli_3.6.4
```

```
[21] withr_3.0.2       magrittr_2.0.3    digest_0.6.37     grid_4.5.0
[25] hms_1.1.3         lifecycle_1.0.4   vctrs_0.6.5       evaluate_1.0.3
[29] glue_1.8.0        farver_2.1.2      colorspace_2.1-1  rmarkdown_2.29
[33] tools_4.5.0       pkgconfig_2.0.3   htmltools_0.5.8.1
```

36 Lab1: Bayesian inference with beta priors, Jingchen Hu

37 Intro

This is a lab by a professor, Jingchen Hu, which goes over Bayesian inference with beta priors.

38 Load libraries

```
library(tidyverse)
library(ProbBayes)
```

39 Posterior predictive checking

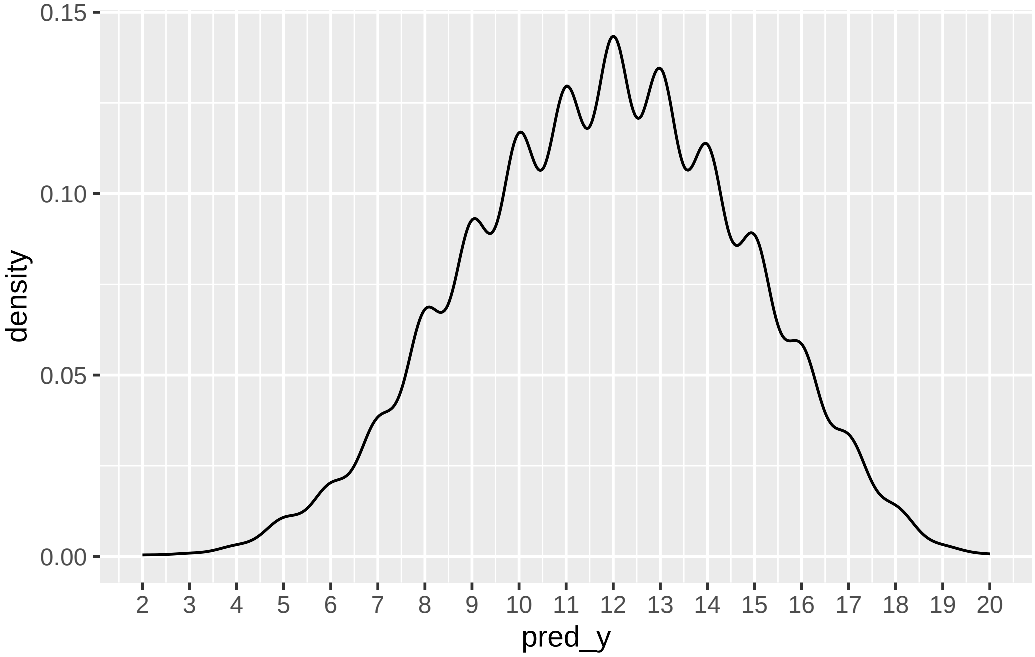
```
S<-10000 # number of simulations
a<-3.06 # a in beta(a,b)
b<-2.56 # b in beta(a,b)
n<-20 # number of trials
y<-12 # number of successes

newy=as.data.frame(rep(NA,S))
names(newy)=c("y")

set.seed(123)
for (s in 1:S){
  pred_p_sim<-rbeta(1, a+y, b+n-y) # step 1 ; get posterior param
  pred_y_sim<-rbinom(1,n,pred_p_sim) # step 2; based on param, predict outcome-> # of success
  newy[s,]=pred_y_sim
}
knitr::kable(head(newy))
```

y
14
13
8
12
14
5

```
#how i would write the simluation

dat<-tibble(pred_p=rbeta(S,a+y,b+n-y))|>
  rowwise()|>
  mutate(pred_y=rbinom(1,n,pred_p))

sum(dat$pred_y>=5&dat$pred_y<=15)/S
```

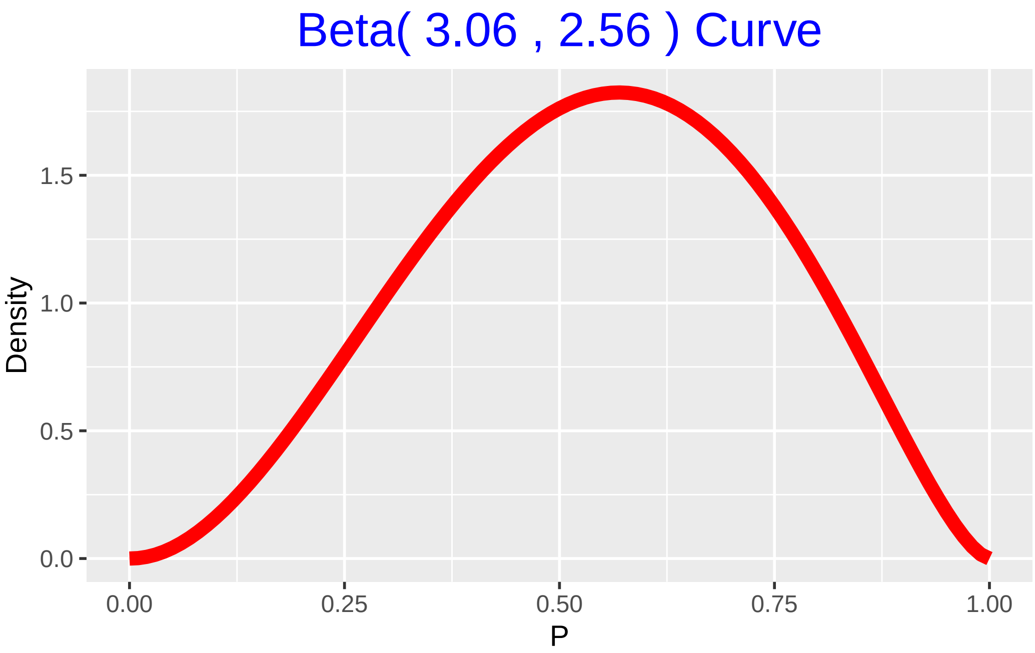
```
[1] 0.8943
```

```
#dat$pred_y<-rbinom(1000,n,dat$pred_p)
ggplot(data=dat,aes(pred_y))+geom_density()+scale_x_continuous(breaks=seq(0,20,1),labels=seq
```

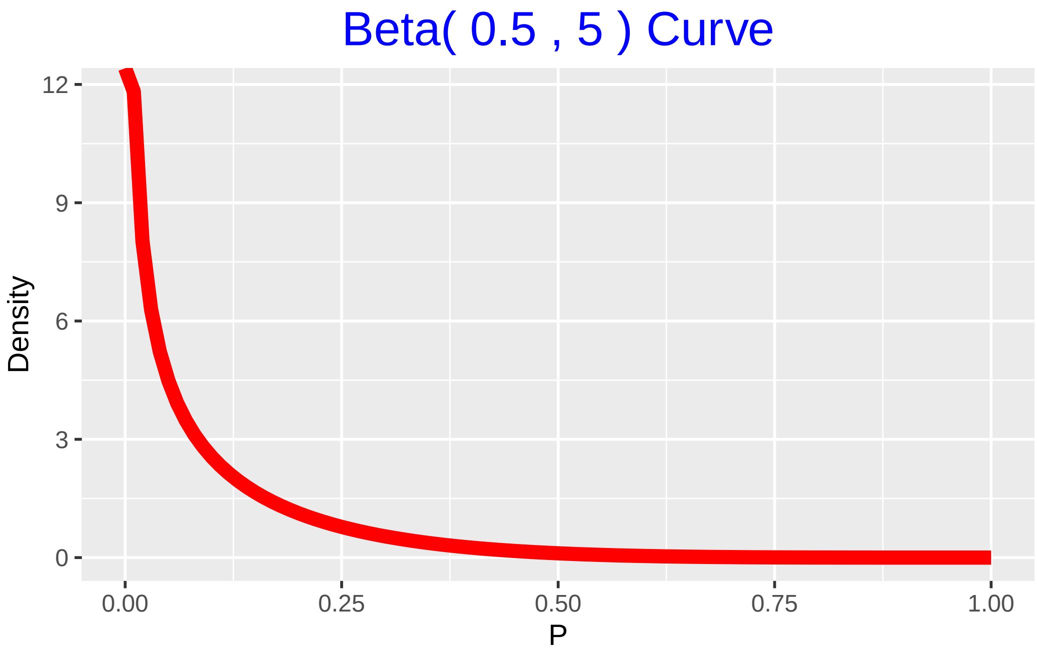


40 Let's try to simulate a situation with mismatched prior with the data

```
beta_draw(c(3.06,2.56)) #prior fromp revious section
```



```
beta_draw(c(0.5,5)) #this looks liek a good prior to mess up the data
```



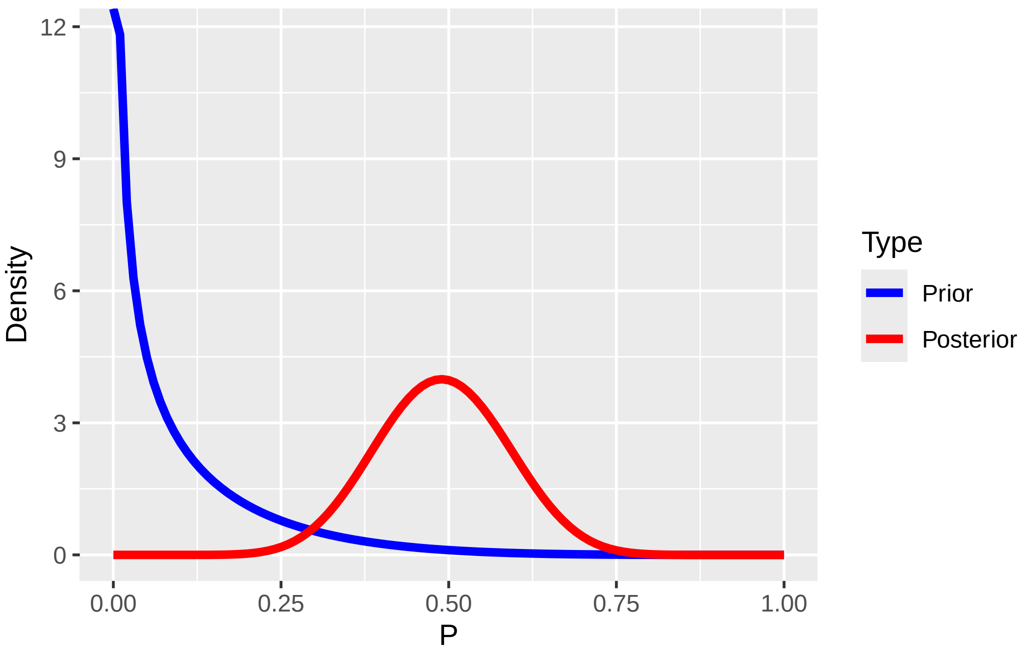
```
s<-10000
n<-20 # trials
y<-12 #successes
a<-.5
b<-5

dat2<-tibble(pred_p=rbeta(S,a+y,b+n-y))|>
  rowwise()|>
  mutate(pred_y=rbinom(1,n,pred_p))

# model check : how often pr(y > ypred|y)
sum(y>dat2$pred_y)/S # how often collected data above posterior prediction
```

```
[1] 0.7158
```

```
1-sum(y>dat2$pred_y)/S #how often collected data below posterior prediction
```

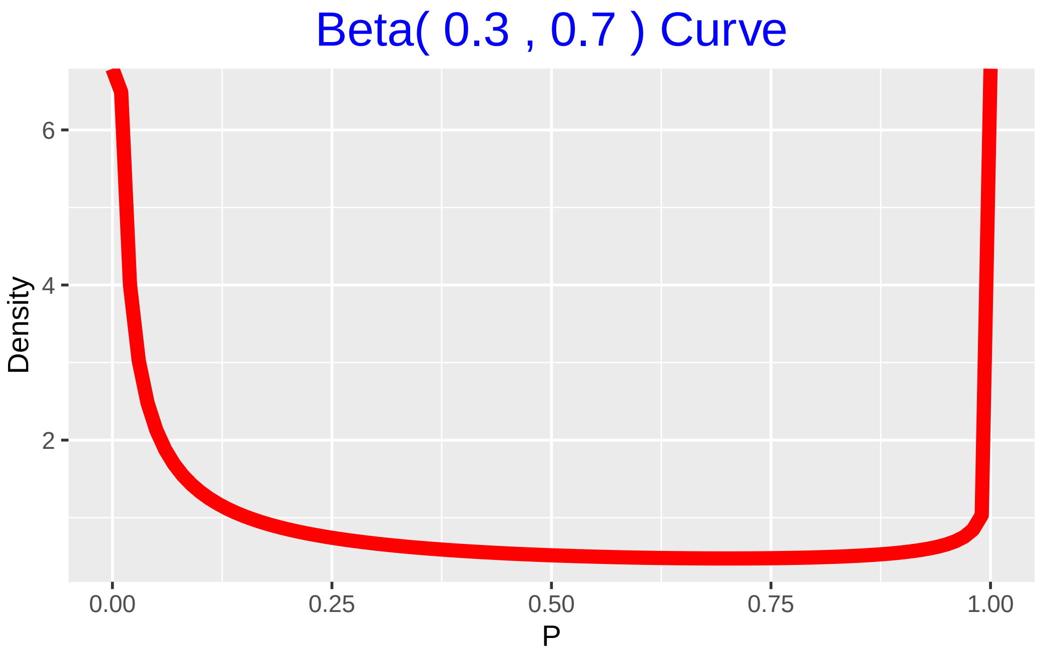
```
[1] 0.2842
```

```
#draw posterior
beta_prior_post(c(.5,5),c(a+y,b+n-y))
```


12
9
6
3
0
Density
0.00
0.25
0.50
0.75
1.00
P
Type
Prior
Posterior

41 Session info

```r
beta_draw(c(.3,.7))
```



```r
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8
```

```
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] ProbBayes_1.1     shiny_1.10.0      gridExtra_2.3     LearnBayes_2.15.1
 [5] lubridate_1.9.4   forcats_1.0.0     stringr_1.5.1     dplyr_1.1.4
 [9] purrr_1.0.4       readr_2.1.5       tidyr_1.3.1       tibble_3.2.1
[13] ggplot2_3.5.2     tidyverse_2.0.0

loaded via a namespace (and not attached):
 [1] generics_0.1.3    stringi_1.8.7     hms_1.1.3         digest_0.6.37
 [5] magrittr_2.0.3    evaluate_1.0.3    grid_4.5.0        timechange_0.3.0
 [9] fastmap_1.2.0     jsonlite_2.0.0    tinytex_0.57      promises_1.3.2
[13] scales_1.3.0      cli_3.6.4         rlang_1.1.6       munsell_0.5.1
[17] withr_3.0.2       yaml_2.3.10       tools_4.5.0       tzdb_0.5.0
[21] colorspace_2.1-1  httpuv_1.6.16     vctrs_0.6.5       R6_2.6.1
[25] mime_0.13         lifecycle_1.0.4   pkgconfig_2.0.3   pillar_1.10.2
[29] later_1.4.2       gtable_0.3.6      glue_1.8.0        Rcpp_1.0.14
[33] xfun_0.52         tidyselect_1.2.1  knitr_1.50        farver_2.1.2
[37] xtable_1.8-4      htmltools_0.5.8.1 rmarkdown_2.29    labeling_0.4.3
[41] compiler_4.5.0
```

Part IV

Misc Frequentist statistics

42 Randomization and sample size estimation

43 Introduction:

The aim of this demo is to showcase how to estimate sample sizes and conduct randomization in clinical trial designs. R has a suite of packages geared towards clinical trial design, monitoring, and analyses (CRAN R Projects- Clinical Trials Zhang, Zhang, and Zhang (2021)). I'm also modeling my demo off of Peter Higgin's *Reproducible Medical Research with R* book, chapter 20 (Higgins (2023)).

44 Load Libraries

```
library(tidyverse) # for ggplot2, data visualization and data filtering with dplyr
library(ggbeeswarm) # for quasirandom plotting
library(pwr) # power analysis
library(gsDesign) # power analysis for survival
library(blockrand) # randomization package
library(randomizeR)# another randomization package

#ggplot2 settings I like:
T<-theme_bw()+theme(,text=element_text(size=18),
                    axis.text=element_text(size=18),
                    panel.grid.major=element_blank(),
                    panel.grid.minor.x = element_blank(),
                    panel.grid = element_blank(),
                    legend.key = element_blank(),
                    axis.title.y=element_text(margin=margin(t=0,r=15,b=0,l=0)),
                    axis.title.x=element_text(margin=margin(t=15,r=,b=0,l=0)))
#+ theme(legend.position="none")
```

45 Sample size calculations

It is important to find the appropriate number of participants in a clinical trial because too few participants may lead to an inability to detect differences (studies may be underpowered) and too many participants lead to excessive use of resources.

The critical information to obtain are:

- alpha (α) level - probability of committing a type I error (false positive)
- beta (β) level - probability of committing a type II error (false negative)
  - sometimes the program will ask for power, which is 1-β and is the probability of detecting differences if they truly exist
- effect size between groups (*cohen's d*)
  - Note that cohen's d is expressed as: $\frac{(\bar{\mu}_1 - \bar{\mu}_2)}{S_{pooled}}$, where $\bar{\mu}_1$ is the mean of one group and $\bar{\mu}_2$ is the mean of the other group, and S_{pooled} is the pooled standard deviation. Therefore, cohen's d is interpreted in units of standard deviation.
- drop out rates

Other information to consider for survival analyses:

- rate of enrollment
- length of study
- hazard rate of each group

45.1 T-test designs and sample size calculations

We will be using the *pwr* package. To start, let's estimate a sample size with:

- alpha = 0.05

- power = 0.80 (1-β)
- effect size, cohen's d = 0.5, which is considered a moderate effect size

```
pwr::pwr.t.test(sig.level=0.05,type="two.sample",power=.8,d=.5)
```

```

     Two-sample t test power calculation 

              n = 63.76561
              d = 0.5
      sig.level = 0.05
          power = 0.8
    alternative = two.sided

NOTE: n is number in *each* group
```

Key result: 64 volunteers per arm. We round up because there can be no fractional individuals.

For a different study design, let's assume there is a before and after measurement of a continuous variable and this would produce paired results with the same assumptions about α,β, and cohen's d.

```
pwr::pwr.t.test(sig.level=0.05,type="paired",power=.8,d=.5)
```

```

     Paired t test power calculation 

              n = 33.36713
              d = 0.5
      sig.level = 0.05
          power = 0.8
    alternative = two.sided

NOTE: n is number of *pairs*
```

Key result: 34 volunteers per arm.

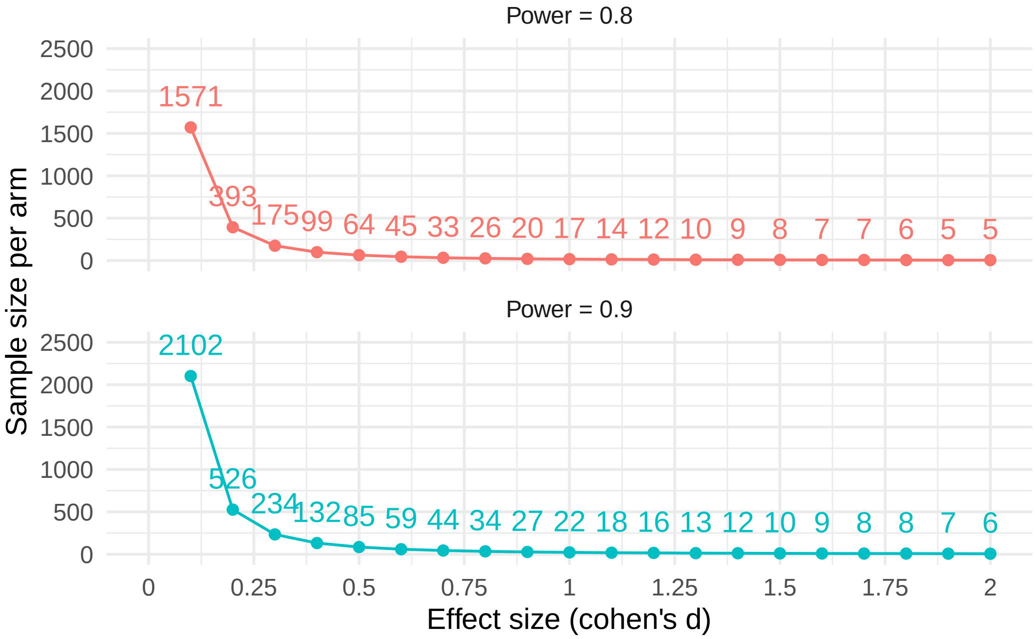
We also may need to consider drop out rates. For example, if there is a 20% drop out rate, then add 20% to the sample sizes per arm. 34 x 20% = ~7, so we would need 41 volunteers.

45.1.1 Simulating effect sizes and power

What if we don't know the effect size and want to find out sample sizes based on different inputs of effect size and power?

- simulating effect sizes from 0 to 2 in .1 increments
- over two levels of power (0.8 and .9)

```
#code to get sample size from pwr.t.test
#round(pwr::pwr.t.test(sig.level=0.05,type="two.sample",power=.8,d=c(.5),n=NULL)$n,0)

d<-data.frame(efsize=rep(seq(0.1,2,.1),2),
              power=c(rep(.8,length(seq(0.1,2,.1))),
                      rep(.9,length(seq(0.1,2,.1)))))
d%>%
  group_by(efsize,power)%>%
  mutate(n=round(pwr::pwr.t.test(sig.level=0.05,
                                 type="two.sample",power=power,
                                 d=efsize,n=NULL)$n,0),
         power2=paste("Power = ",power,sep=""))%>%
  #power2 is for plotting
  ggplot(.,aes(x=efsize,y=n,colour=factor(power)))+
  geom_point()+geom_line()+
  theme_minimal()+
  geom_text(aes(label=n),vjust=-1)+
  facet_wrap(~power2,ncol=1)+
  xlab("Effect size (cohen's d)")+
  ylab("Sample size per arm")+
  theme(legend.position = "none")+
  scale_x_continuous(,limits=c(0,2)
                     ,breaks=seq(0,2,.25),
                     labels=seq(0,2,.25))+
  scale_y_continuous(limits=c(0,2500))
```



45.2 Chi-square 2x2 contingency table design

In this design, let's say there are counts of diseased and not-diseased individuals that were exposed and not exposed to some chemical. We want to find the association between the two variables and we need to specify the expected proportions of the 2x2 under the alternative hypothesis.

```
#ES.w2() # chi-square for test of association
#pwr.chisq.test()

d2<-data.frame(exposure=c("exposed","not exposed"),non_diseased=c(.25,.3),diseased=c(0.25,.2
knitr::kable(d2)
```

exposure	non_diseased	diseased
exposed	0.25	0.25
not exposed	0.30	0.20

Now that we have the expected 2x2 matrix under the alternative hypothesis (not independent), then we need to identify the effect size with *ES.w2()* and then plug and chug with *pwr.chisq.test()* with $\alpha = 0.05$ and power $= 0.8$.

```
ef.sim.dat<-ES.w2(d2[,-1])
pwr.chisq.test(w=ef.sim.dat,df=1,power=.8,sig.level=.05)
```

```
     Chi squared power calculation

              w = 0.1005038
              N = 777.0372
             df = 1
      sig.level = 0.05
          power = 0.8

NOTE: N is the number of observations
```

Key result: We need 778 observations. Note that the degrees of freedom on 1 in this case for a 2x2 contingency table, which is calculated as (# of columns - 1) x (# of rows -1).

45.3 Time to event types of designs (survival analyses)

I will be using the *gsDesign* package and referencing an online resource **here**.

```
hr=.7 # hazard ratio
controlMedian<-8 # 8 months
lambda1 <- log(2) / controlMedian #estimated hazard rate of control

nSurvival(
  lambda1 = lambda1,
  lambda2 = lambda1 * hr, #hazard rate for experimental
  Ts = 24, #24 months
  Tr = 6, # 6 months
  eta = .1, # value per month dropout rate
  ratio = 1, # equal sampling
  alpha = .05,
  beta = .2
)
```

```
Fixed design, two-arm trial with time-to-event
outcome (Lachin and Foulkes, 1986).
Study duration (fixed):          Ts=24
```

```
Accrual duration (fixed):        Tr=6
Uniform accrual:              entry="unif"
Control median:      log(2)/lambda1=8
Experimental median: log(2)/lambda2=11.4
Censoring median:        log(2)/eta=6.9
Control failure rate:       lambda1=0.087
Experimental failure rate:  lambda2=0.061
Censoring rate:                 eta=0.1
Power:                 100*(1-beta)=80%
Type I error (1-sided):   100*alpha=5%
Equal randomization:          ratio=1
Sample size based on hazard ratio=0.7 (type="rr")
Sample size (computed):           n=476
Events required (computed): nEvents=195
```

46 Randomization (stratified permuted block randomization)

Using the t-test design (2 trial arms), we'd like to implement block randomization across a strata. Often times, we can't sample participants all at once and so we need to apply treatments in groupings as they enroll in the study, or blocks. To ensure equal sampling of treatments across different sub-populations, randomization can be conducted at the level of different sub-populations (strata). For example, randomization can be conducted for males and females separately. With a sample size of 200 per arm, in a two-arm trial, I'm randomization across a gender strata (100 each so they're equally sampled).

46.1 ...with *blockrand* package

```
mrand<-blockrand(n = 100,
                     num.levels = 2, # three treatments
                     levels = c("Con.Arm", "Treat.Arm"), # arm names
                     stratum = "Strat.male", # stratum name
                     id.prefix = "SM", # stratum abbrev
                     block.sizes = c(3,4), # times arms = 6,8
                     block.prefix = "blksm") # stratum abbrev

frand<-blockrand(n = 100,
                     num.levels = 2, # three treatments
                     levels = c("Con.Arm", "Treat.Arm"), # arm names
                     stratum = "Strat.female", # stratum name
                     id.prefix = "SF", # stratum abbrev
                     block.sizes = c(3,4), # times arms = 6,8
                     block.prefix = "blkfm") # stratum abbrev
totrand<-rbind(mrand,frand)
knitr::kable(head(totrand,25))
```

id	stratum	block.id	block.size	treatment
SM001	Strat.male	blksm01	6	Con.Arm
SM002	Strat.male	blksm01	6	Treat.Arm
SM003	Strat.male	blksm01	6	Treat.Arm
SM004	Strat.male	blksm01	6	Con.Arm
SM005	Strat.male	blksm01	6	Treat.Arm
SM006	Strat.male	blksm01	6	Con.Arm
SM007	Strat.male	blksm02	8	Con.Arm
SM008	Strat.male	blksm02	8	Con.Arm
SM009	Strat.male	blksm02	8	Treat.Arm
SM010	Strat.male	blksm02	8	Con.Arm
SM011	Strat.male	blksm02	8	Treat.Arm
SM012	Strat.male	blksm02	8	Treat.Arm
SM013	Strat.male	blksm02	8	Treat.Arm
SM014	Strat.male	blksm02	8	Con.Arm
SM015	Strat.male	blksm03	6	Treat.Arm
SM016	Strat.male	blksm03	6	Treat.Arm
SM017	Strat.male	blksm03	6	Con.Arm
SM018	Strat.male	blksm03	6	Con.Arm
SM019	Strat.male	blksm03	6	Treat.Arm
SM020	Strat.male	blksm03	6	Con.Arm
SM021	Strat.male	blksm04	6	Con.Arm
SM022	Strat.male	blksm04	6	Treat.Arm
SM023	Strat.male	blksm04	6	Con.Arm
SM024	Strat.male	blksm04	6	Con.Arm
SM025	Strat.male	blksm04	6	Treat.Arm

We can then create patient randomization "cards" based on the *blockrand()* output.

```r
plotblockrand(totrand,'mystudy.pdf',
              top=list(text=c('MyStudy','Patient:%ID%','Treatment:%TREAT%'),
                       col=c('black','black','red'),font=c(1,1,4)),
              middle=list(text=c("MyStudy","Sex:%STRAT%","Patient:%ID%"),
                          col=c('black','blue','green'),font=c(1,2,3)),
              bottom="Call123-4567toreportpatiententry", cut.marks=TRUE)
```

46.2 ...with *randomizeR* package

Using the randomized permuted block randomization function, *rpbrPar()*. The details:

Fix the possible random block lengths rb, the number of treatment groups K, the sample size N and the vector of the ratio. Afterwards, one block length is randomly selected of the random block lengths. The patients are assigned according to the ratio to the corresponding treatment groups. This procedure is repeated until N patients are assigned. Within each block all possible randomization sequences are equiprobable.

```r
#randomization parameters
males<-rpbrPar(N=100, #total sample size
               rb=6, # block length parameter
               K=2) # number of groups
rr<-genSeq(males) # saving randomization procedure
rr.out<-as.vector(getRandList(rr))# grab randomizations
#put into dataframe and make it look better
male.r.dat<-data.frame(sex="M",
                       subject=paste("SM",
                                     seq(1:length(rr.out)),sep=""),
                       treatment=rr.out,
                       treatmentname=ifelse(rr.out=="A","Control","Treatment"))
#male.r.dat
knitr::kable(head(male.r.dat,10))
```

sex	subject	treatment	treatmentname
M	SM1	A	Control
M	SM2	B	Treatment
M	SM3	A	Control
M	SM4	A	Control
M	SM5	B	Treatment
M	SM6	B	Treatment
M	SM7	B	Treatment
M	SM8	A	Control
M	SM9	A	Control
M	SM10	A	Control

47 Session Info

```r
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8 
[2] LC_CTYPE=English_United States.utf8   
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C                          
[5] LC_TIME=English_United States.utf8    

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base     

other attached packages:
 [1] randomizeR_3.0.2 mvtnorm_1.3-3    survival_3.8-3   plotrix_3.8-4   
 [5] blockrand_1.5    gsDesign_3.6.7   pwr_1.3-0        ggbeeswarm_0.7.2
 [9] lubridate_1.9.4  forcats_1.0.0    stringr_1.5.1    dplyr_1.1.4     
[13] purrr_1.0.4      readr_2.1.5      tidyr_1.3.1      tibble_3.2.1    
[17] ggplot2_3.5.2    tidyverse_2.0.0 

loaded via a namespace (and not attached):
 [1] gtable_0.3.6      beeswarm_0.4.0    xfun_0.52         coin_1.4-3       
 [5] insight_1.1.0     lattice_0.22-6    tzdb_0.5.0        vctrs_0.6.5      
 [9] tools_4.5.0       generics_0.1.3    sandwich_3.1-1    stats4_4.5.0     
```

```
[13] parallel_4.5.0    pkgconfig_2.0.3   Matrix_1.7-3      gt_1.0.0
[17] lifecycle_1.0.4   farver_2.1.2      compiler_4.5.0    munsell_0.5.1
[21] tinytex_0.57      codetools_0.2-20  PwrGSD_2.3.8      vipor_0.4.7
[25] htmltools_0.5.8.1 yaml_2.3.10       pillar_1.10.2     MASS_7.3-65
[29] multcomp_1.4-28   r2rtf_1.1.4       tidyselect_1.2.1  digest_0.6.37
[33] stringi_1.8.7     reshape2_1.4.4    labeling_0.4.3    splines_4.5.0
[37] fastmap_1.2.0     grid_4.5.0        colorspace_2.1-1  cli_3.6.4
[41] magrittr_2.0.3    TH.data_1.1-3     libcoin_1.0-10    withr_3.0.2
[45] scales_1.3.0      timechange_0.3.0  rmarkdown_2.29    matrixStats_1.5.0
[49] zoo_1.8-14        modeltools_0.2-23 hms_1.1.3         evaluate_1.0.3
[53] knitr_1.50        rlang_1.1.6       Rcpp_1.0.14       xtable_1.8-4
[57] glue_1.8.0        xml2_1.3.8        jsonlite_2.0.0    R6_2.6.1
[61] plyr_1.8.9
```

48 One-Way ANOVA vignette

49 Libraries

```r
library(plyr)
library(tidyr)
```

50 Reviewing Analysis of Variance (ANOVA)

I'll be following Chapter 10 of *A Primer of Ecological Statistics* by Gotelli and Ellison

Key Concepts:

ANOVA aims to determine differences in a continuous variable between 2 or more groups.

ANOVA bulit on partitioning on the concept of paritioning of the sum of squares (SS_{total}). How do we calculate this?

The total sum of squares of the data is the sum of squared deviations of each observation (Y_i) from the grand mean ($\bar{Y}$). There are $i = 1$ to a treatment levels and $j = 1$ to n replicates per treatment.

recap:

- i refers to the treatment levels
  - it looks like a represents the number of different treatments
- j refers to each observations, with n being the number of replicates

$$SS_{total} = \sum_{i=1}^{a}\sum_{j=1}^{n}(Y_{ij} - \bar{Y})^2$$

The total sum of squares is the deviation of each observation from the grand mean.

SS_{total} can then be partitioned into different components, mainly **among and within groups**.

$$SS_{total} = SS_{among} + SS_{within}$$

So for the *among* group SS:

$$SS_{among} = \sum_{i=1}^{a}\sum_{j=1}^{n}(\bar{Y}_i - \bar{Y})^2$$

So for the *within* group SS:

$$SS_{within} = \sum_{i=1}^{a}\sum_{j=1}^{n}(Y_{ij} - \bar{Y}_i)^2$$

Bringing it all together:

$$\sum_{i=1}^{a}\sum_{j=1}^{n}(Y_{ij} - \bar{Y})^2 = \sum_{i=1}^{a}\sum_{j=1}^{n}(\bar{Y}_i - \bar{Y})^2 + \sum_{i=1}^{a}\sum_{j=1}^{n}(Y_{ij} - \bar{Y}_i)^2$$

50.1 Enter data

```
#data,
# a = 3, 3 treatments
# n = 4, 4 reps
n=4
unman<-c(10,12,12,13)
control<-c(9,11,11,12)
treat<-c(12,13,15,16)

wide.dat<-data.frame(unman,control,treat);wide.dat
```

```
  unman control treat
1    10       9    12
2    12      11    13
3    12      11    15
4    13      12    16
```

```
long.dat<-gather(wide.dat,treatment,measure,unman:treat);long.dat
```

```
   treatment measure
1      unman      10
2      unman      12
3      unman      12
4      unman      13
5    control       9
6    control      11
7    control      11
8    control      12
```

```
9       treat       12
10      treat       13
11      treat       15
12      treat       16
```

```r
#global mean
grandmean<-round(mean(c(unman,control,treat)),2);grandmean
```

```
[1] 12.17
```

50.2 SS_{total} calculations

```r
sum((long.dat$measure-grandmean)^2)
```

```
[1] 41.6668
```

50.3 SS_{among} calculations

```r
## calculating 1 case

(mean(unman)-grandmean)^2
```

```
[1] 0.1764
```

```r
### making a whole function
#with ddply
ssa<-function(n=n,vec=c(1,3,3),grandmean=grandmean){
  SSa<-(mean(vec)-grandmean)^2
  SSa
}
ssa(vec=unman,n=n,grandmean=grandmean) # verify function
```

```
[1] 0.1764
```

```
## executing function
ssam<-ddply(long.dat,.(treatment),summarize,ssamong=ssa(vec=measure,n=n,grandmean=grandmean)
```

```
  treatment ssamong
1   control  2.0164
2     treat  3.3489
3     unman  0.1764
```

```
SSAM<-n*sum(ssam$ssamong);SSAM
```

```
[1] 22.1668
```

```
# for a balanced design!
```

50.4 SS_{within} calculations

```
## calculating 1 case
sum((unman-mean(unman))^2)
```

```
[1] 4.75
```

```
### making a whole function
#with ddply
sswi<-function(x){
  SSwithin<-sum((x-mean(x))^2)
  SSwithin
}
sswi(unman) # verify function
```

```
[1] 4.75
```

```
SSwi<-ddply(long.dat,.(treatment),summarize,sswi=sswi(measure));SSwi
```

```
  treatment  sswi
1   control  4.75
2     treat 10.00
3     unman  4.75
```

```
sumwithin<-sum(SSwi$sswi);sumwithin
```

```
[1] 19.5
```

51 Assumptions of ANOVAs

1. Samples are indepednent and identically distributed.
2. Variances are homogeneous among groups
    - variance within each group approx to variance within other groups

3. Residuals are normally distributed

4. Samples are classified correctly

5. Maine effects are additive

52 Hypothesis testing

Definitions:

- Y_{ij}: replicated j associated with treatment level i
- μ is the *true* grand mean or average
- A_i is the additive linear component associated with level i of treatment A.
    - There is a different coefficient A_i associated with each treatment level (i).
    - positive coefficients mean that the treatment level has a higher value than grand mean

Alternative Hypothesis, H_a: $Y_{ij} = \mu + A_i + \epsilon_{ij}$

Null Hypothesis, H_o: $Y_{ij} = \mu + \epsilon_{ij}$

53 Verify with aov() function

```
knitr::kable(round(summary(aov(measure~treatment,data=long.dat))[[1]],2))
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
treatment	2	22.17	11.08	5.12	0.03
Residuals	9	19.50	2.17	NA	NA

54 Simulating 95% confidence intervals

55 Load Libraries

```r
library(tidyverse)
```

56 Simulating 95% frequentist confidence intervals

A good explanation here:

A 95% confidence interval is constructed such that if the model assumptions are correct and if you were to hypothetically repeat the experiment or sampling many many times, 95% of the intervals constructed would contain the true value of the parameter.

My own words: **The 95% confidence interval is when the true parameter is contained within the interval 95% of the time from constructing the 95% confidence interval from repeated experiments under the assumption of a correct model.**

Let's gain intuition by what this means:

1. Simulate data and do it a bunch of times

2. Then calculate 95% confidence interval with say a t-test
3. Determine how many times the true parameter (which we set in step 1) is in between the confidence intervals

```r
#1) simulate data
sim<-10000
#dataset size
n<-100
# sampel data with mean 10, sd =1
x<-rnorm(n,mean=10,sd=1)
#fit t.test ; grab lower and upper confidence interval
#as.vector(c(t.test(x)$conf.int,t.test(x)$estimate))


## now simulate across sim

#for loop is prob best
#prep dataset
#d<-tibble(lower=rep(0,sim),upper=rep(0,sim),mean=rep(0,sim))
```

```
d<-array(0,dim=c(sim,2))

for (i in 1:sim){
  x<-rnorm(n,mean=10,sd=1)
  d[i,]<-as.vector(c(t.test(x)$conf.int))
}


#head(d)
d<-data.frame(d)
names(d)<-c("lower","upper")
knitr::kable(head(d))
```

lower	upper
9.702081	10.11195
9.952184	10.37956
9.761763	10.15359
9.911854	10.27917
9.860354	10.24825
9.935028	10.33487

```
#count how many times the lower and upper confidence interval is below true value of 10
d<-d|>
  mutate(out=1*(lower<10 & upper>10))
mean(d$out)
```

```
[1] 0.9471
```

```
#cases where confidence interval is does not include true parameter
d|>
  filter(out==0)|>
  head()
```

```
     lower    upper out
1 10.03174 10.44910   0
2 10.07089 10.48082   0
3 10.00663 10.36656   0
4 10.00221 10.39309   0
5 10.00641 10.41685   0
6 10.00121 10.43583   0
```

Additional notes:

Cementing interpretation: When you have a single 95% CI on a single sample, it doesn't mean, that the population mean belongs to this particular interval with a particular probability. If you were to repeat the experiment many many times and calculate this interval on each fo the samples, then 95% of the repeated samples would have the true population mean.

57 Session info

```
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] lubridate_1.9.4 forcats_1.0.0   stringr_1.5.1   dplyr_1.1.4
 [5] purrr_1.0.4     readr_2.1.5     tidyr_1.3.1     tibble_3.2.1
 [9] ggplot2_3.5.2   tidyverse_2.0.0

loaded via a namespace (and not attached):
 [1] gtable_0.3.6      jsonlite_2.0.0    compiler_4.5.0    tidyselect_1.2.1
 [5] tinytex_0.57      scales_1.3.0      yaml_2.3.10       fastmap_1.2.0
 [9] R6_2.6.1          generics_0.1.3    knitr_1.50        munsell_0.5.1
[13] pillar_1.10.2     tzdb_0.5.0        rlang_1.1.6       stringi_1.8.7
[17] xfun_0.52         timechange_0.3.0  cli_3.6.4         withr_3.0.2
```

```
[21] magrittr_2.0.3    digest_0.6.37     grid_4.5.0        hms_1.1.3
[25] lifecycle_1.0.4   vctrs_0.6.5       evaluate_1.0.3    glue_1.8.0
[29] colorspace_2.1-1  rmarkdown_2.29    tools_4.5.0       pkgconfig_2.0.3
[33] htmltools_0.5.8.1
```

Part V

Decision making under uncertainty

58 Dempster-Shafer Theory, notes

59 Introduction - plain English understanding

Dempster-Shafer Theory (DST) is a way to account for uncertainty when making a decision. For example, imagine the answer to a question has two mutually exclusive outcomes: yes or no. Then, DST, would describe this as the *frame of discernment*, or θ, where,

$$\theta = \{yes, no\}$$

But, when we try to estimate these answers, there are multiple possibilities, especially if you're unsure. For example, sometimes when you measure something, you're not sure if the answer is yes or no. All possible states are represented as a power set, 2^{θ}, such that

$$2^{\theta} = \{\{\emptyset\}, \{yes\}, \{no\}, \{\theta\}\}$$

and notice that $\theta = \{yes, no\}$, which is the set where you're sure whether the answer is yes or no.

Now, each set can have a numerical value assigned to them and can be expressed as:

$$m : 2^{\theta} -> [0, 1]$$

and the value is referred to as a mass. From the formulation above, the mass of the powerset is between 0 and 1, referred to as the mass function. The sum of the masses of the powerset add up to 1:

$$\sum_{A \in 2^{\theta}} m(A) = 1$$

. In other words, for all members (A) within the powerset 2^{θ}, the sum of all these members is 1. Naturally, values closer to 1 means there is more evidence for that particular set (m(A)). For example,

2^{θ}	Mass
{yes}	0.2
{no}	0.6
{yes, no}	0.2

2^θ	Mass
{∅}	0

the mass assignments all sum to 1. Notice that the ∅ is 0, which is a feature of DST expressed in this way. And notice that typically, in say a logistic regression, the posterior probabilities are a way to fill in these masses for a {yes} or {no} answer.

Most notably, the support for what we care about, θ have overlaps in the sets. For example, {yes,no} overlaps with {yes}. So how can we express the real answer given this framework? DST attempts to do this by forming two levels of support for an answer in θ such that:

- The belief is the lowest level of support
- The plausibility is the highest level of support

The belief in {yes} is the sum of the masses of subset (B) of {yes}, expressed as:

$$bel(\{yes\}) = \sum_{B \subset \{yes\}} m(B).$$

The plausibility of {yes} is the sum of all masses of sets B that intersects with {yes}, expressed as:

$$pl(\{yes\} = \sum_{B \cap \{yes\}} m(B).$$

In our example from the mass assignments, they would induce the following beliefs and plausibilities:

2^θ	Mass	Belief	Plausibility
{yes}	0.2	0.2	0.4
{no}	0.6	0.6	0.8
{yes, no}	0.2	1.0	1.0
{∅}	0.0	0.0	0.0

For example, the belief in {yes} is the mass of {yes} (0.2). The plausibility in {yes} is the mass of {yes} and {yes,no} because they intersect (0.2 + 0.2 = 0.4).

60 Possible ways to decide based on beliefs and plausibility

Given this evidence, how do we decide on whether the answer to the question? There would be 3 outcomes instead of 2:

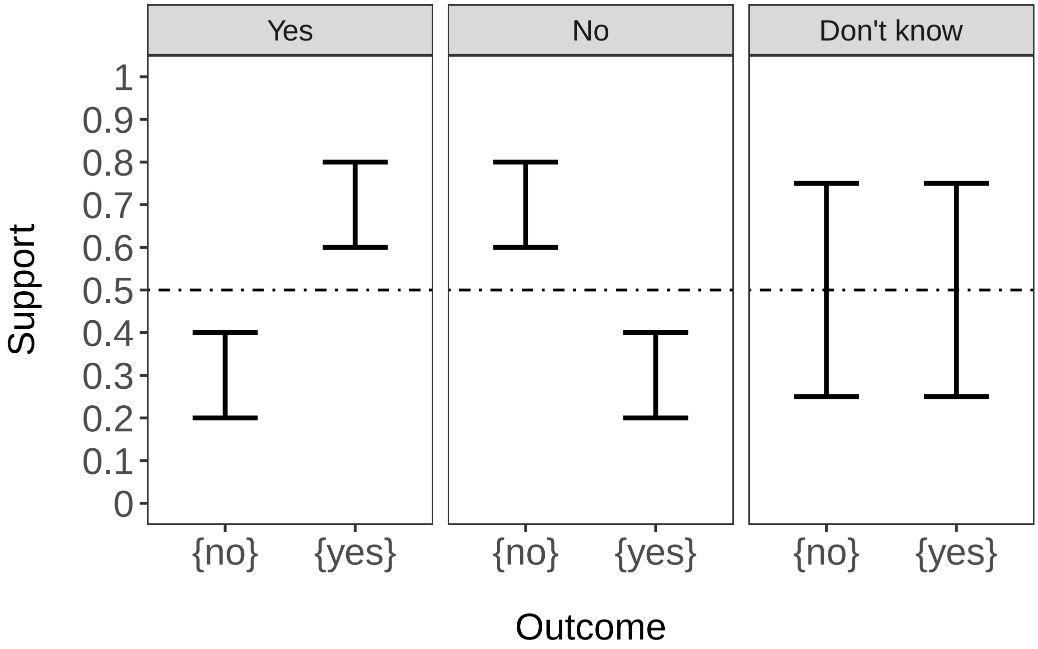
- Yes
- No
- Don't know ({yes,no})

I'll simulate some data and show how. Load packages first.

```r
library(tidyverse) # data wrangling, visualization
library(EvCombR) # package for dempster shafer
##ggplot2 settings
T<-theme_bw()+theme(,text=element_text(size=14),
                    axis.text=element_text(size=14),
                    panel.grid.major=element_blank(),
                    panel.grid.minor.x = element_blank(),
                    panel.grid = element_blank(),
                    legend.key = element_blank(),                    axis.title.y=element_te

axis.title.x=element_text(margin=margin(t=15,r=,b=0,l=0)))+
theme(legend.position="none")
```

```r
# simulate some beliefs
# and plausibilities
# that would lead to each type of outcome
outcome<-rep(c("{yes}","{no}","{yes,no}"),3)
decision<-c(rep("No",3),rep(c("Yes"),3),rep(c("Don't know"),3))
belief<-c(c(.2,.6,.2),c(.6,.2,.2),c(.25,.25,.5))
pl<-c(c(0.4,.8,1),c(0.8,.4,1),c(0.75,.75,1))
d<-data.frame(outcome,decision,belief,pl)
d<-d%>%
  dplyr::filter(outcome!="{yes,no}")
#filter out {yes,no}
```

```
d$decision<-factor(d$decision,levels=c("Yes","No","Don't know"))

#plot it out
ggplot(d,aes(x=outcome))+
  geom_errorbar(aes(ymax=pl,ymin=belief),
                width=.5,linewidth=.85)+
  facet_wrap(~decision)+
  scale_y_continuous(limits = c(0,1),
                     breaks=seq(0,1,.1),
                     labels=seq(0,1,.1))+
  ylab("Support")+xlab("Outcome")+
  geom_hline(yintercept=.5,linetype="dotdash")+T
```



Here, you can see that the lower bound of the range in each outcome is the belief and the upper bound of the range is the plausibility. Each panel would show the decision being made. Notice that in the "Don't know" panel, the levels of support overlap. There are ways to make decisions even if the levels of support overlap but we won't get into that.

DST is very flexible and there are no set rules for constructing mass functions.

Typically, posterior probabilities from say a logistic regression would show a flat level of support (only the belief). However, the posterior probabilities can be restructured into a mass function by accounting for model uncertainty. Model uncertainty can be extracted from a confusion matrix. So the posterior probability for yes would be scaled by the positive predictive value, which is the degree of confidence in the model to make a yes call. This is how we can account for uncertainty when using a predictive model.

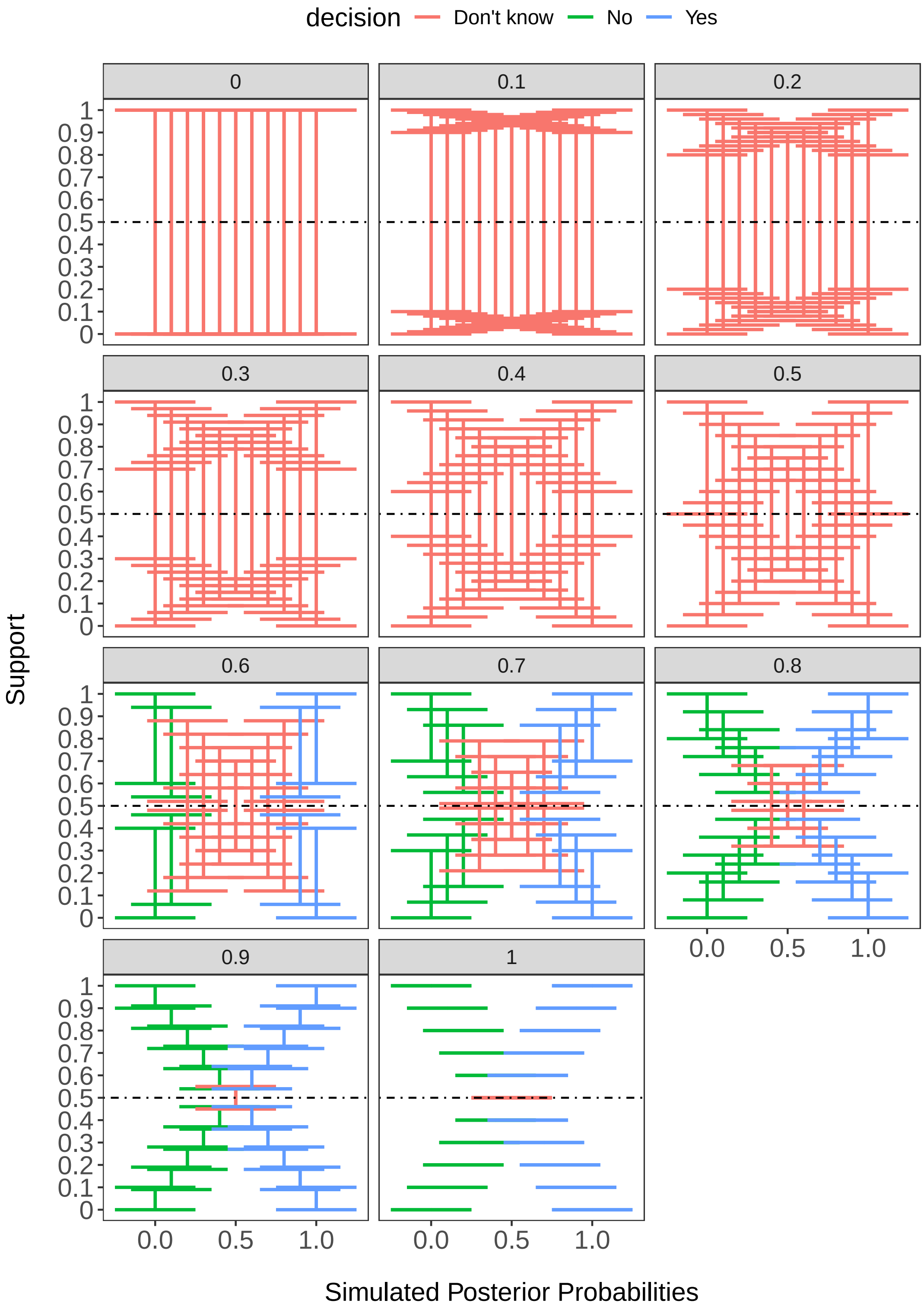
61 Simulations of model uncertainty

Here, I'll simulate posterior probabilities, and model uncertainties to determine how it impacts decision making.

```
#simulate a range of model uncertainty
#both in the positive predictive value
# and negative predictive value

## simulate a 1000 samples with posterior probs
yes<-rep(seq(0,1,.1),11)

#number of different model uncertainties
mu<-sort(rep(seq(0,1,.1),11)) #simulate model
# uncertainty from 0 to 1 in .1 steps
dat<-data.frame(cbind(yes,mu))
dat$no<- (1-dat$yes) # get the no post prob
#assuming model uncertainty is the same
# same PPV and NPV
##now we can construct mass assignments
dat<-dat%>%
  mutate(y.mass=yes*mu,n.mass=no*mu,yn.mass=1-y.mass-n.mass,y.pl=yn.mass+y.mass,n.pl=yn.mass+

##lets set up factors by decision point
#if the uppper bound of yes is less than
#lower bound of no, then decide NO

#if upper bound of no is less than
#lower bound of yes, then decide YES
dat<-dat%>%
  mutate(decision=ifelse(y.pl<n.mass,"No",ifelse(n.pl<y.mass,"Yes","Don't know")))
glimpse(dat)
```

```
Rows: 121
Columns: 9
$ yes      <dbl> 0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 0.0, 0~
$ mu       <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0~
$ no       <dbl> 1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0.0, 1.0, 0~
$ y.mass   <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0~
$ n.mass   <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0~
$ yn.mass  <dbl> 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.9, 0~
$ y.pl     <dbl> 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1~
$ n.pl     <dbl> 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1~
$ decision <chr> "Don't know", "Don't know", "Don't know", "Don't know", "Don'~
```

```
#need to convert to long
d.long<-dat%>%
  pivot_longer(cols=y.mass:n.mass,names_to = "Outcome",values_to = "support")%>%
  mutate(plaus=support+yn.mass)

##let's plot out the simulation
ggplot(d.long,aes(x=yes,colour=decision))+
  geom_errorbar(aes(ymax=plaus,ymin=support),
               width=.5,linewidth=.85)+
  facet_wrap(~mu,nrow=4)+
  scale_y_continuous(limits = c(0,1),
                     breaks=seq(0,1,.1),
                     labels=seq(0,1,.1))+
  ylab("Support")+xlab("Simulated Posterior Probabilities")+
  geom_hline(yintercept=.5,linetype="dotdash")+T+theme(legend.position = "top")
```


decision
Don't know
No
Yes
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Support
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.0
0.5
1.0
Simulated Posterior Probabilities

Thoughts:

What you can see is that decisions can't be made if there is too much uncertainty (0-0.5 model performance). But, at 0.6 model performance, posterior probabilities have to be very high in order to make a confident decision. And then you can see traditionally, if we assume a perfect model, model performance = 1, then the support bands are flat, which is typically how we treat predictive models in general.

62 Summary

DST offers a flexible and creative approach to accounting for uncertainty when making a decision. DST re-frames sources of evidence such that uncertain cases can have some level of evidence assigned to it. The assignment of the degree of evidence, or masses can resemble probabilities across all answers. The approach outlined here involves restructuring poster probabilities from a model into masses. This is done by scaling the posterior probabilities by model performance, which is derived from the confusion matrix. In traditional machine learning approaches, data are split into training and testing. The confusion matrix of the model applied to the testing dataset could be used. Based on the masses, then the degree of support can be calculated to help inform decision-making.

63 RshinyApp

Try plugging in your own posterior probabilities or masses and plug in your own PPV or NPV in this **RShinyApp**.

64 References

- Rathman JF, Yang C, Zhou H. **Dempster-Shafer theory for combining in silico evidence and estimating uncertainty in chemical risk assessment**. Comput Toxicol. 2018;6:16-31. doi:10.1016/j.comtox.2018.03.001

65 Session info

```r
sessionInfo()
```

```
R version 4.5.0 (2025-04-11 ucrt)
Platform: x86_64-w64-mingw32/x64
Running under: Windows 11 x64 (build 26100)

Matrix products: default
  LAPACK version 3.12.1

locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] EvCombR_0.1-4   lubridate_1.9.4 forcats_1.0.0   stringr_1.5.1
 [5] dplyr_1.1.4     purrr_1.0.4     readr_2.1.5     tidyr_1.3.1
 [9] tibble_3.2.1    ggplot2_3.5.2   tidyverse_2.0.0

loaded via a namespace (and not attached):
 [1] gtable_0.3.6      jsonlite_2.0.0    compiler_4.5.0    tidyselect_1.2.1
 [5] tinytex_0.57      scales_1.3.0      yaml_2.3.10       fastmap_1.2.0
 [9] R6_2.6.1          labeling_0.4.3    generics_0.1.3    knitr_1.50
[13] munsell_0.5.1     pillar_1.10.2     tzdb_0.5.0        rlang_1.1.6
[17] stringi_1.8.7     xfun_0.52         timechange_0.3.0  cli_3.6.4
```

```
[21] withr_3.0.2       magrittr_2.0.3    digest_0.6.37     grid_4.5.0
[25] hms_1.1.3         lifecycle_1.0.4   vctrs_0.6.5       evaluate_1.0.3
[29] glue_1.8.0        farver_2.1.2      colorspace_2.1-1  rmarkdown_2.29
[33] tools_4.5.0       pkgconfig_2.0.3   htmltools_0.5.8.1
```

Part VI

Function-valued traits

References

Higgins, Peter D. R. 2023. *Chapter 20 Randomization for Clinical Trials with R | Reproducible Medical Research with R*. https://bookdown.org/pdr_higgins/rmrwr/randomization-for-clinical-trials-with-r.html.

Zhang, Ed, W. G. Zhang, and R. G. Zhang. 2021. "CRAN Task View: Clinical Trial Design, Monitoring, and Analysis." https://CRAN.R-project.org/view=ClinicalTrials.