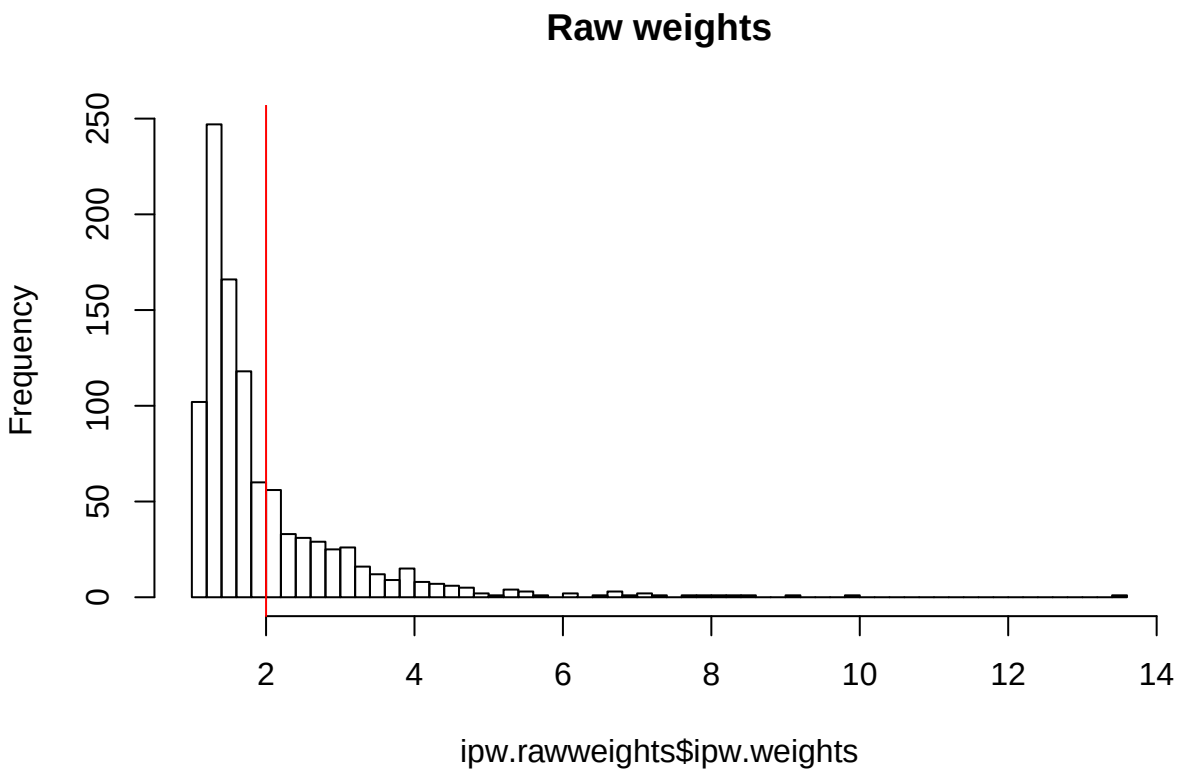
Exercise 5

The fifth example explores the use of inverse of treatment probability weights (IPTWs), on some of the datasets we looked at before.

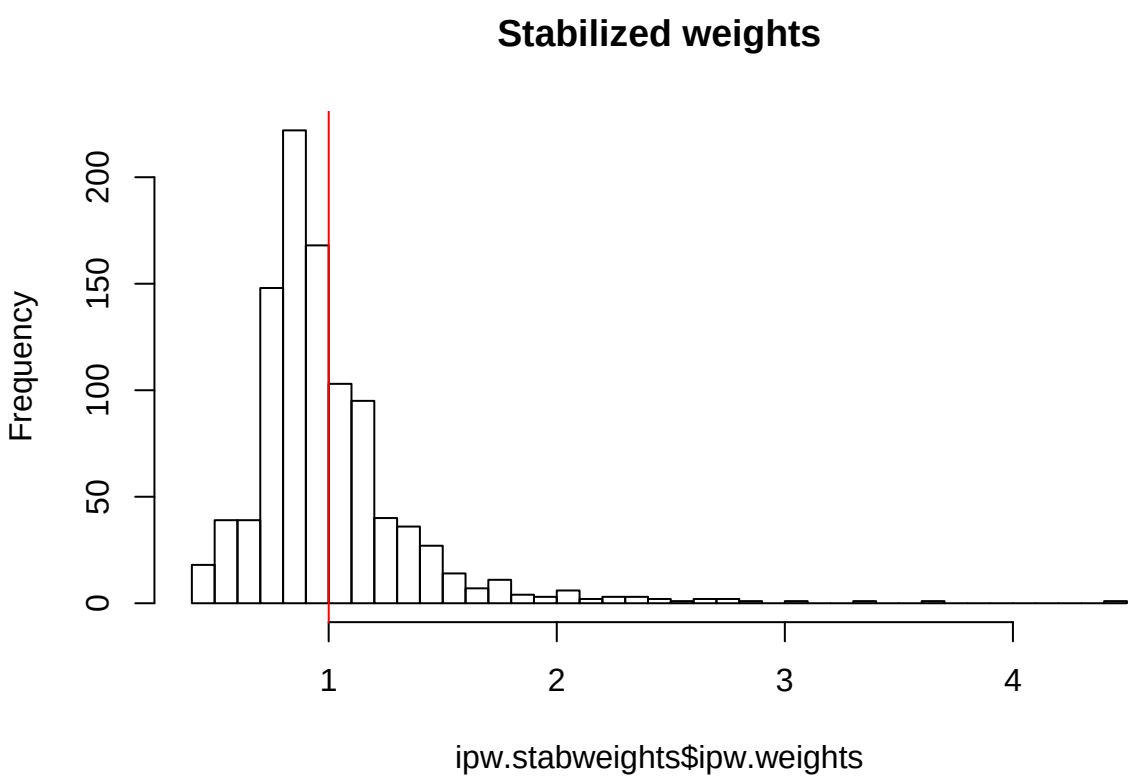
Here is a worked-out, and commented example of using ipw to perform the weighting. Once the weights are estimated, we use a very simple outcome model, but we use robust standard errors, that account for the uncertainty in the weights. This comes with a small drawback, that we cannot feed them back into emmeans. In the process of model-fitting, it occurs that the treatment variable is recoded from a factor to a numeric variable, and emmeans only allows factors.

The data is simulated, and if you like you can ignore the simulation code. It is provided here for the sole reasons that you can reproduce the example quickly without having to download data files.

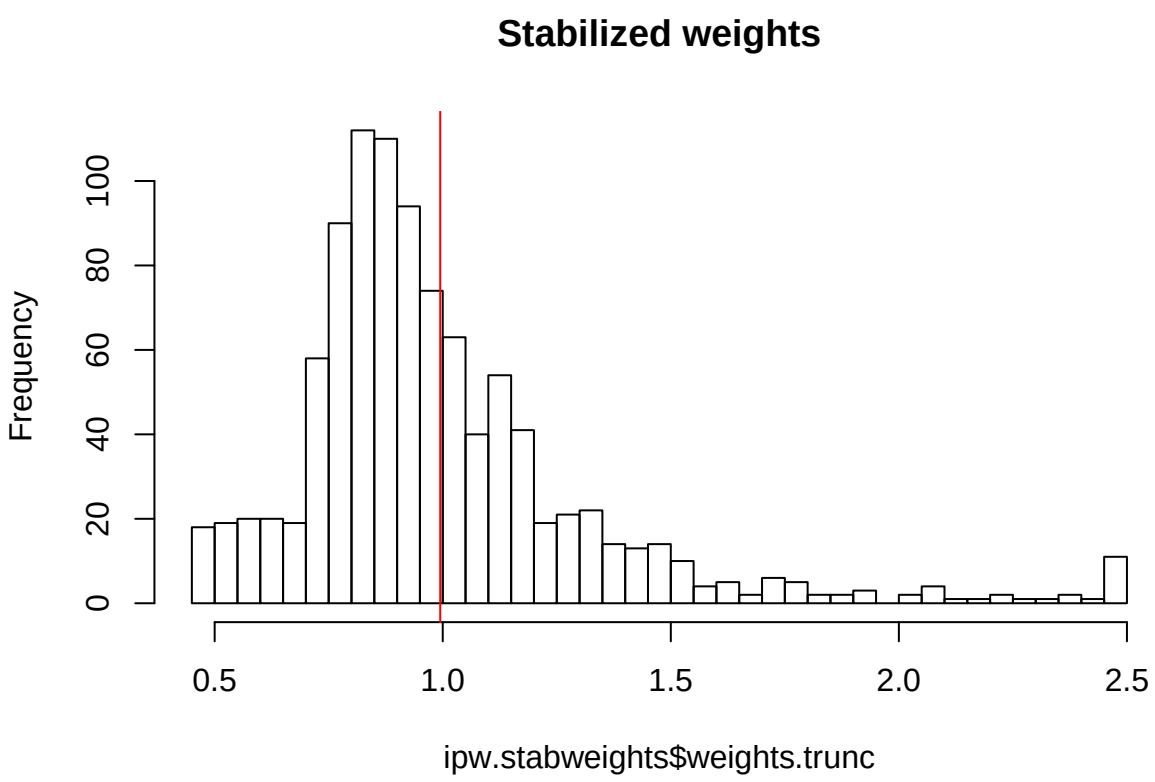
```
####simulation code#############################
set.seed(12345)
n <- 1000
u <- rnorm(n,0,1)
x1 <- .9*u + rnorm(n,2,.6)
x2 <- .9*u + rnorm(n,2,.6)
p <- (1/(1+exp(-.5+.1*x1+.5*x2)))
t <- rbinom(n,1,p)
y <- .5*t - .9*x1 - .9*x2 + .3*I(x1^2) - .3*I(x2^2) + .4*t*x1 - .3*t*x2 + .3*t*I(x1^2) + rnorm(n,0,.3)
t <- factor(t)
dfex5 <- data.frame(x1,x2,t,y)
dfex5$t <- factor(dfex5$t)
################################################

####analysis code##############################
library(ipw)
library(survey)

#first unstabilized weights
ipw.rawweights <- ipwpoint(
  exposure = t,
  family = "binomial",
  link = "logit",
  denominator = ~ x1 + x2,
  data = dfex5)

hist(ipw.rawweights$ipw.weights, main = "Raw weights",breaks=50)
abline(v=mean(ipw.rawweights$ipw.weights),col="red")
```




```
ipw.stabweights <- ipwpoint(
  exposure = t,
  family = "binomial",
  link = "logit",
  denominator = ~ x1 + x2,
  numerator = ~ 1,
  data = dfex5)

hist(ipw.stabweights$ipw.weights, main = "Stabilized weights",breaks=50)
abline(v=mean(ipw.stabweights$ipw.weights),col="red")
```




```
ipw.stabweights <- ipwpoint(
  exposure = t,
  family = "binomial",
  link = "logit",
  denominator = ~ x1 + x2,
  numerator = ~ 1,
  trunc = .01,
  data = dfex5)

hist(ipw.stabweights$weights.trunc, main = "Stabilized weights",breaks=50)
abline(v=mean(ipw.stabweights$weights.trunc),col="red")
```




```
#robust standard errors
lm1 <- svyglm(y~t, design = svydesign(~ 1, weights = ~ ipw.stabweights$weights.trunc, data = dfex5))
summary(lm1)
```

```
##
## Call:
## svyglm(formula = y ~ t, design = svydesign(~1, weights = ~ipw.stabweights$weights.trunc,
##     data = dfex5))
##
## Survey design:
## svydesign(~1, weights = ~ipw.stabweights$weights.trunc, data = dfex5)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.64275    0.08593  -42.39   <2e-16 ***
## t1           2.53297    0.16100   15.73   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 4.310063)
##
## Number of Fisher Scoring iterations: 2
```

Above, you can see a lot of code that performs the weighting. Let's dissect it a little. The ipwpoint function takes several arguments. First, exposure defines the treatment variable, or our putative cause. Family and link, define the model family and link function. For binary treatments, we typically use binomial and logit, to define a logistic regression model. Gaussian and identity would be used for a continuous putative cause.

The denominator and numerator lines define the weight equations. For the denominator, we put all variables that we want to adjust for. If we omit the numerator term, unstabilized weights are used. If we put the numerator along with a 1, stabilized weights are used. This is a bit confusing, because the help file describes it somewhat differently. If we want to truncate weights, we use the trunc option. If we put e.g., .01 then the 1% most extreme weights (on both sides of the distribution) are truncated.

I also use a very simple histogram, to plot the weights and inspect them for outliers, and their central tendency. Weights should typically be centered at 1.

Finally, after the IPTWs are generated, I run a simple outcome model. I am using the survey package to integrate robust standard errors that take into account the uncertainty associated with the weights.

Exercise:

1.) Download the file dfex3a again from github (https://raw.githubusercontent.com/felixthoemmes/IPN_workshop/master/dfex3a.csv). You can download this file directly into R (no need to navigate to github in a browser, using the following code snippet:

```
library(readr)
dfex3a <- read_csv("https://raw.githubusercontent.com/felixthoemmes/IPN_workshop/master/dfex3a.csv")
```

The file contains a treatment t, an outcome y, and covariates x1-x4. We assume that these variables are those that fulfill the back-door criterion. Obtain an *unadjusted* estimate for the effect of t on y. You already did this several times, and we just repeat it to get a bit of a baseline.

2.) First, construct unstabilized weights. Inspect them visually, and then estimate a treatment effect from them, using the survey package, for robust weights.

3.) Then, construct stabilized weights. Inspect them and compare them with the unstabilized weights. Again, estimate a treatment effect, using the weights.

4.) Truncate the weights at .1%, 1%, and 5%. For all inspect, and compare the weights, and estimate, and compare all resulting treatment effects.