# Problem Set 1

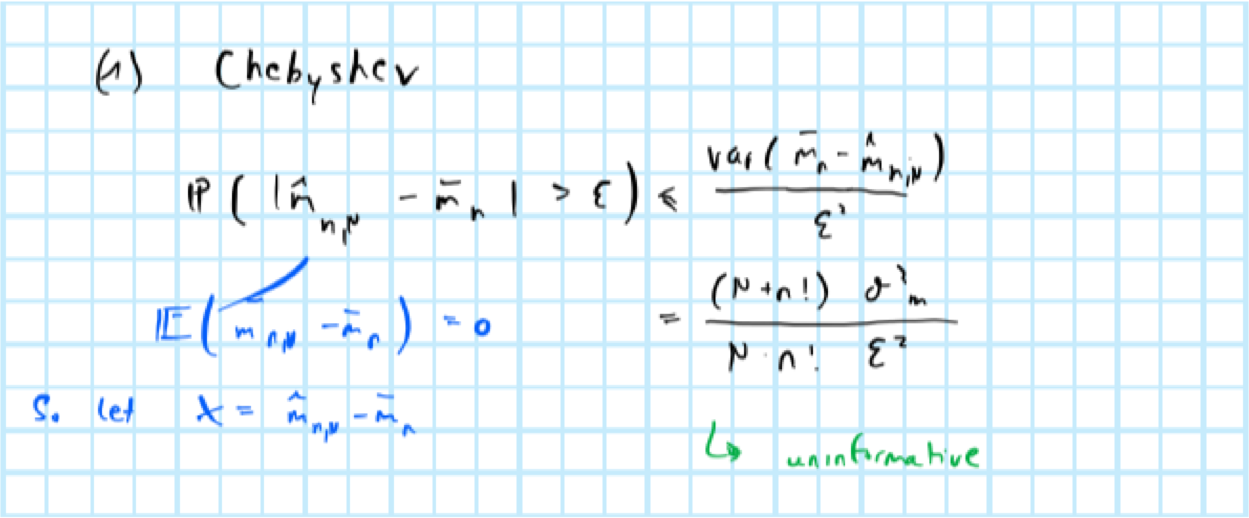
Patrick Altmeyer

31 January, 2021

## 1 Problem 1

### 1.1 Chebyshev

We can derive Chebyshev's inequality as follows



(1) Chebyshev

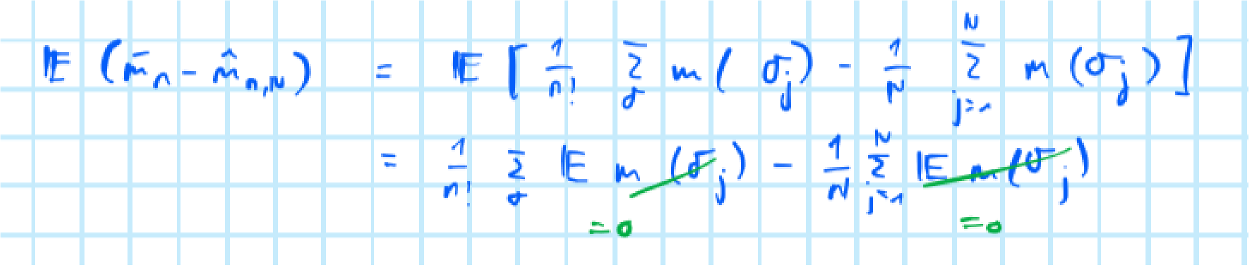
$$\mathbb{P}\left(|\hat{m}_{n,N} - \bar{m}_n| > \varepsilon\right) \leq \frac{var(\bar{m}_n - \hat{m}_{n,N})}{\varepsilon^2}$$

$$\mathbb{E}\left(\hat{m}_{n,N} - \bar{m}_n\right) = 0 \qquad = \frac{(N+n!)\,\sigma^2_m}{N \cdot n!\ \varepsilon^2}$$

So let $X = \hat{m}_{n,N} - \bar{m}_n$

↳ uninformative

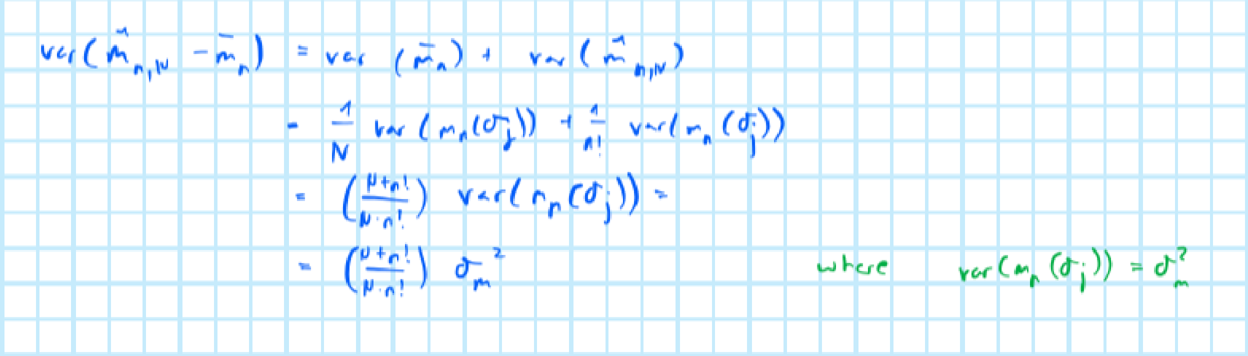
where the derivation for the expected value of the differences is as follows



$$\mathbb{E}\left(\bar{m}_n - \hat{m}_{n,N}\right) = \mathbb{E}\left[\frac{1}{n!}\sum_{\sigma} m(\sigma_j) - \frac{1}{N}\sum_{j=1}^{N} m(\sigma_j)\right]$$

$$= \frac{1}{n!}\sum_{\sigma} \mathbb{E}\,\underbrace{\cancel{m(\sigma_j)}}_{=0} - \frac{1}{N}\sum_{j=1}^{N} \mathbb{E}\,\underbrace{\cancel{m(\sigma_j)}}_{=0}$$

and the derivation of the variance term is as follows:



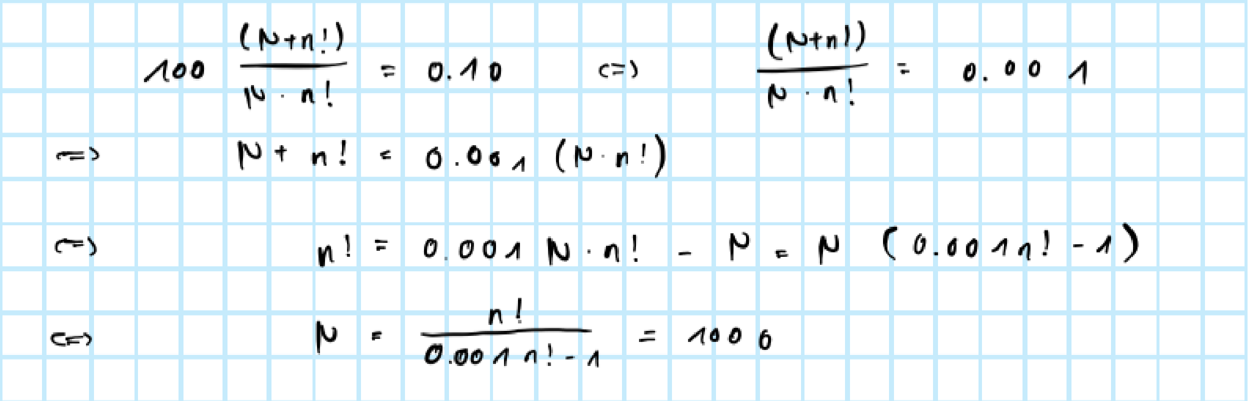
$$\mathrm{var}(\hat{m}_{n,N} - \bar{m}_n) = \mathrm{var}(\bar{m}_n) + \mathrm{var}(\hat{m}_{n,N})$$

$$= \frac{1}{N}\mathrm{var}(m_n(\sigma_j)) + \frac{1}{n!}\mathrm{var}(m_n(\sigma_j))$$

$$= \left(\frac{N+n!}{N \cdot n!}\right)\mathrm{var}(m_n(\sigma_j)) =$$

$$= \left(\frac{N+n!}{N \cdot n!}\right)\sigma_m^2 \qquad \text{where} \qquad \mathrm{var}(m_n(\sigma_j)) = \sigma_m^2$$

For a fixed distance $\varepsilon$, variance $\sigma_m^2$ and sample size $n$ we can simply solve right-hand side of the concentration inequality for $N$ by equating it to some $\delta$ of our choice. For example, suppose we are interested $\delta = 0.10$ and let $n = 10000$, $\sigma_m^2 = 1$, $\varepsilon = 0.1$. Then a good choice of $N$ would be 1000:



$$100\,\frac{(N+n!)}{N \cdot n!} = 0.10 \iff \frac{(N+n!)}{N \cdot n!} = 0.001$$

$$\iff N + n! = 0.001\,(N \cdot n!)$$

$$\iff n! = 0.001\,N \cdot n! - N = N\,(0.001n! - 1)$$

$$\iff N = \frac{n!}{0.001n! - 1} = 1000$$

In other words the choice of $N$ is basically independent of $n$ as $n$ gets large.

## 1.2 Hoeffding

It is easy to see that for large choices of $N$ and values of $n$ both estimators have the same expected value. Let $\mu$ denote that expected value. Then noting that each $m(\sigma_j)$ is bounded by $m(\sigma_j) \in [a_i = \min(\mathbf{X}), b_i = \max(\mathbf{X})]$ we can apply Hoeffding. For the ideal, infeasible estimator we have

$$p(|\bar{m}_n - \mu| \geq \varepsilon) = 2\exp\left(\frac{2n^2\varepsilon^2}{\sum_i (b_i - a_i)^2}\right)$$

and the expression for $\hat{m}_{n,N}$ is analogous. Noting that this expression depends on $\varepsilon$ we can let this distance to the expected value be different for $\hat{m}_{n,N}$ and $\bar{m}_n$. This enables us to make the following claim: with a probability of at least $1 - \delta$ we have

$$|\bar{m}_n - \mu| \leq \sqrt{\frac{(b-a)^2 \log\left(\frac{2}{\delta_{n!}}\right)}{2n!}} = \varepsilon_{n!}$$

$$|\hat{m}_{n,N} - \mu| \leq \sqrt{\frac{(b-a)^2 \log\left(\frac{2}{\delta_N}\right)}{2N}} = \varepsilon_N$$

which we can rewrite as:

$$-\sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{\delta}\right)}{2n!}} \leq \bar{m}_n - \mu \leq \sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{\delta}\right)}{2n!}}$$

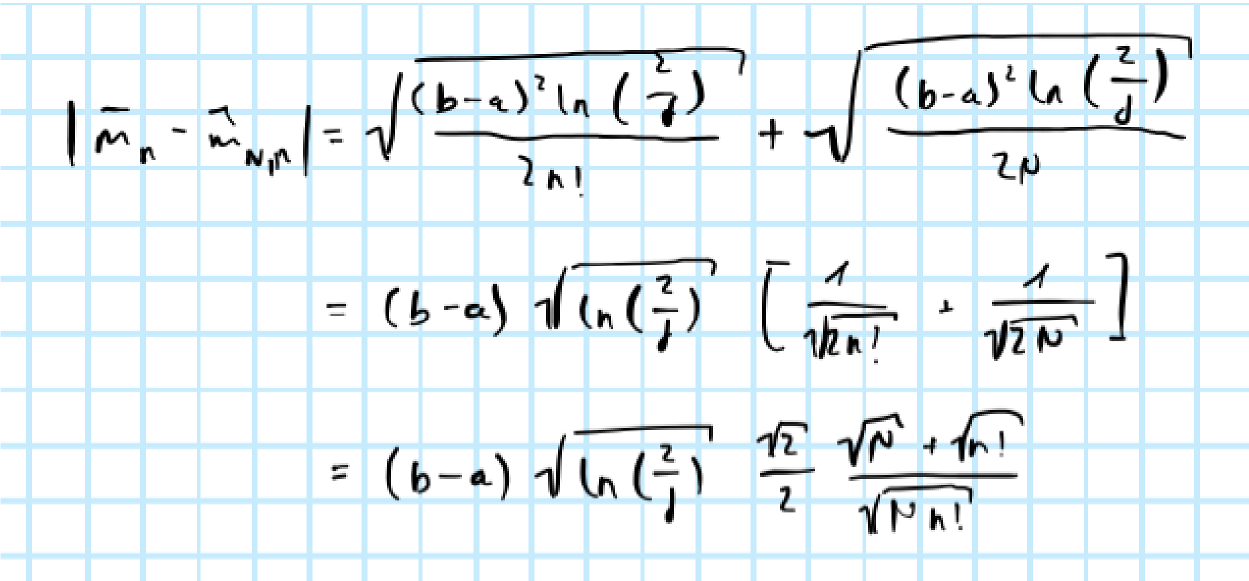
$$-\sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{\delta}\right)}{2N}} \leq \hat{m}_{N,n} - \mu \leq \sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{\delta}\right)}{2N}}$$

There are two worst-case scenarios that would maximise the distance between the to estimators: either the first inequality is binding on the left side and the second inequality is binding on the right side or the other way around. Under the first case we can write

$$\bar{m}_n - \mu = \sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{\delta}\right)}{2n!}} \quad , \quad \hat{m}_{N,n} - \mu = -\sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{d}\right)}{2N}}$$

$$\Leftrightarrow \quad (\bar{m}_n - \mu) - (\hat{m}_{N,N} - \mu) = \bar{m}_n - \hat{m}_{N,n} = \sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{\delta}\right)}{2n!}} + \sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{d}\right)}{2N}}$$

and since the same logic applies to the second case it follows

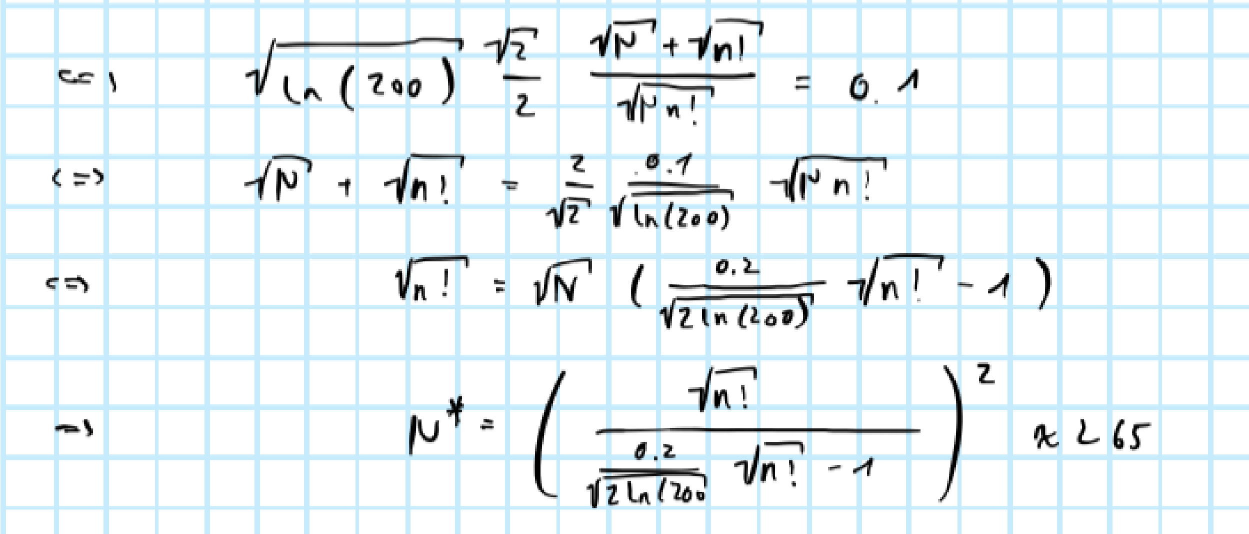


$$|\bar{m}_n - \tilde{m}_{N,n}| = \sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{\delta}\right)}{2n!}} + \sqrt{\frac{(b-a)^2 \ln\left(\frac{2}{\delta}\right)}{2N}}$$

$$= (b-a)\sqrt{\ln\left(\frac{2}{\delta}\right)} \left[ \frac{1}{\sqrt{2n!}} + \frac{1}{\sqrt{2N}} \right]$$

$$= (b-a)\sqrt{\ln\left(\frac{2}{\delta}\right)} \frac{\sqrt{2}}{2} \frac{\sqrt{N} + \sqrt{n!}}{\sqrt{N n!}}$$

which is the distance we were originally after. Now notice that this is very similar to what we obtained earlier when we applied Chebyshev: for given $\delta$, $n$, $\varepsilon$ and $\mathbf{X}$ (given $\mathbf{X}$ both $a$ and $b$ are deterministic) we once again can simply solve for $N$. Using the same parameter choice as before and assuming $a = 0$, $b = 1$ this solves to $N \approx 265$:



$$\Leftrightarrow \quad \sqrt{\ln(200)} \frac{\sqrt{2}}{2} \frac{\sqrt{N} + \sqrt{n!}}{\sqrt{N n!}} = 0.1$$

$$\Leftrightarrow \quad \sqrt{N} + \sqrt{n!} = \frac{2}{\sqrt{2}} \frac{0.1}{\sqrt{\ln(200)}} \sqrt{N n!}$$

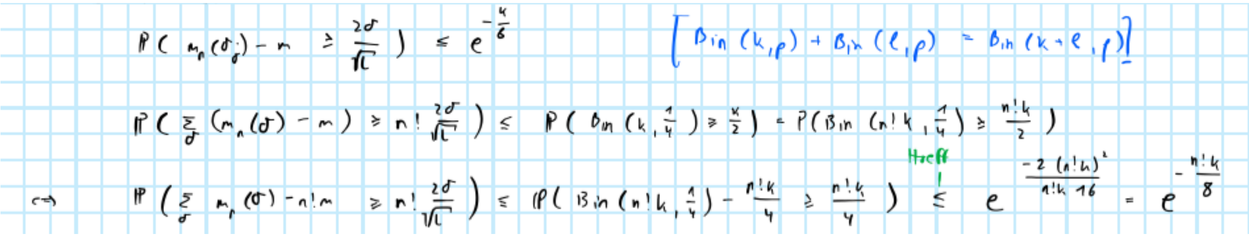
$$\Leftrightarrow \quad \sqrt{n!} = \sqrt{N} \left( \frac{0.2}{\sqrt{2\ln(200)}} \sqrt{n!} - 1 \right)$$

$$\Rightarrow \quad N^* = \left( \frac{\sqrt{n!}}{\frac{0.2}{\sqrt{2\ln(200}} \sqrt{n!} - 1} \right)^2 \approx 265$$

Hoeffding's inequality consequently prescribes a lower choice of $N$, which is intuitive since the inequality is tighter.

## 1.3 Chernoff

We can apply Chernoff bounds to both estimators. For the ideal, infeasible one we have



$$\mathbb{P}\left( m_n(\sigma_\sigma) - m \geq \frac{2\sigma}{\sqrt{L}} \right) \leq e^{-\frac{k}{6}} \qquad \left[ \mathrm{Bin}(k,p) + \mathrm{Bin}(\ell,p) = \mathrm{Bin}(k+\ell,p) \right]$$

$$\mathbb{P}\left( \sum_\sigma (m_n(\sigma) - m) \geq n!\frac{2\sigma}{\sqrt{L}} \right) \leq \mathbb{P}\left( \mathrm{Bin}(k,\tfrac{1}{4}) \geq \tfrac{k}{2} \right) = \mathbb{P}\left( \mathrm{Bin}(n!k,\tfrac{1}{4}) \geq \frac{n!k}{2} \right)$$

$$\hookrightarrow \quad \mathbb{P}\left( \sum_\sigma m_n(\sigma) - n!m \geq n!\frac{2\sigma}{\sqrt{L}} \right) \leq \mathbb{P}\left( \mathrm{Bin}(n!k,\tfrac{1}{4}) - \frac{n!k}{4} \geq \frac{n!k}{4} \right) \overset{\text{Hoeff}}{\overset{!}{\leq}} e^{\frac{-2(n!k)^2}{n!k\,16}} = e^{-\frac{n!k}{8}}$$

and equivalently applying Chernoff to $\hat{m}_{n,N}$ yields the same expression with $N$ instead of $n!$ in the denominator of the exponential. These expressions to not depend on the distance $\varepsilon$ as with Hoeffding, so we cannot apply the same logic as before where we let the distance vary by estimator. However we know that as long as $N < n!$ the concentration inequality for $\bar{m}_n$ will be tighter than for $\hat{m}_{n,N}$. Hence, we can always just look at $\delta_N$ (corresponding to the wider bound) and know that if the inequality holds for $\hat{m}_{n,N}$ then for the same $\delta_N$ it must also hold for $\bar{m}_n$. Hence we could proceed as we did before for Hoeffding, which I omit here.

## 2 Problem 2

The median-of-means estimator and its permutation-invariant Monte-Carlo alternative can be implemented in R as follows:

```r
median_of_means <- function(x, block_size=NULL, delta=.05, permutation_invariant=F, N=1000) {
  n <- length(x)
  if (is.null(block_size)) {
    block_size = round(n/(8 * log(1/delta))) # default block size
  }
  if (!permutation_invariant) {
    x <- x[sample.int(n,n)] # shuffle
    blocks <- split(x, ceiling(seq_along(x)/block_size))
    median_of_means <- median(sapply(blocks, mean))
  } else {
    median_of_means <- mean(
      sapply(
        1:N,
        function(i) {
          x <- x[sample.int(n,n)] # shuffle
          blocks <- split(x, ceiling(seq_along(x)/block_size))
          median_of_means <- median(sapply(blocks, mean))
          return(median_of_means)
        }
      )
    )
  }
  return(median_of_means)
}
```

To run the simulation I first set up a grid of parameter combinations. Samples will be drawn from four distribution: two distributions with light tails – Gaussian and Laplace – and two heavy-tailed student-$t$ distributions with one and five degrees of freedom, respectively.

```r
library(rmutil)
library(data.table)
# Parameters
n <- round(exp(5:10))
block_size_ratio <- c(.01,.1,0.5) # block size ratio of sample size n
```

```r
N <- c(10, 100) # number of random permutations
# Distribution functions:
t1 <- function(...,df=1) {
  rt(...,df=df)
}
t5 <- function(...,df=5) {
  rt(...,df=df)
}
dist_fun <- list(
  "gauss" = rnorm,
  "laplace" = rlaplace,
  "t1" = t1,
  "t5" = t5
)
grid <- data.table(
  expand.grid(
    n = n,
    dist = names(dist_fun),
    block_size_ratio = block_size_ratio,
    N = N
  )
)
```

I then run across the rows of the grid and for each parameter combination/row run the experiment $J = 100$ times each time computing the the three different estimators. Since I generate data with $\mathbb{E}(\mathbf{X}) = 0$, the absolute values of the obtained point estimates correspond to the mean absolute error (MAE). Hence, in order to estimate error probabilities for a given distance $\varepsilon$ it suffices to compute the proportion of draws for which the MAE exceeds $\varepsilon$. Below follows the code that runs experiment:

```r
J <- 100 # number of independent samples
P <- nrow(grid) # number of parameter combinations
performance <- rbindlist(
  lapply(
    1:P,
    function(p) {
      list2env(c(grid[p,]), envir = environment()) # load parameters into function scope
      performance <- rbindlist(
        lapply( # loop over J samples
          1:J,
          function(j) {
            x <- dist_fun[[dist]](n=n) # n random draws from given distribution
            block_size <- block_size_ratio * n
            # Compute the estimates:
            performance <- data.table(
              n = n,
              estimator = c(
                "emp_mean",
                "median_of_means",
                "mom_permutation_inv"
              ),
              estimate = c(
                mean(x),
                median_of_means(x, block_size = block_size),
                median_of_means(x, block_size = block_size, permutation_invariant = TRUE, N=N)
              )
            )
            performance <- merge(grid[p,],performance)
            performance[,sample:=j]
            return(performance)
          }
        )
      )
      return(performance)
    }
  )
)
eps <- data.table(eps=c(0.1,0.025,0.01))
performance <- setkey(performance[,c(k=1,.SD)],k)[eps[,c(k=1,.SD)],allow.cartesian=TRUE][,k:=NULL]
performance[,mae:=abs(estimate)]
performance[,higher_than_eps:=mae>eps]
error_rates <- performance[
  ,
  .(error_rate=sum(higher_than_eps)/.N, mae=mean(abs(estimate))), # error percentage and mean absolute
  by=.(n,eps,dist,block_size_ratio,N,estimator)
]
saveRDS(performance, file = "data/mean_estimators_performance.rds")
saveRDS(error_rates, file = "data/mean_estimators_error_rates.rds")
```

## 2.1 Error probabilities

The first plot provides an overall picture of the effects of sample size and distribution as well as insight into how the performacme differs across the different estimators (Figure 1; block size and $N$ are both held constant). A few unsurprising observations stand out immediately: (i) error rates decrease with sample size, (ii) the probability of decreases with the distance $\varepsilon$ and (iii) error probabilities are generally higher for heavy-tailed distributions due to the presence of outliers. Another observation that holds here and across the following illustrations is that the median-of-means (non permutation-invariant) generally does worse than the other two estimators, except for the very heavy-tailed $t$-distribution with 1 degree of freedom ($t1$). And intuitive result is also that both median-of-means estimators outperform the empirical mean for the $t1$-distribution: they are more robust to outliers than the empirical mean.

```
performance <- readRDS(file = "data/mean_estimators_performance.rds")
error_rates <- readRDS(file = "data/mean_estimators_error_rates.rds")
library(ggplot2)
p <- ggplot(data = error_rates[block_size_ratio==median(block_size_ratio) & N==max(N)], aes(x=log(n), y
  geom_point() +
  geom_line() +
  facet_grid(
    rows = vars(eps),
    cols = vars(dist)
  ) +
  scale_color_discrete(name="Estimator:") +
  labs(
    x="Sample size (logs)",
    y="Probability of error"
  )
p
```

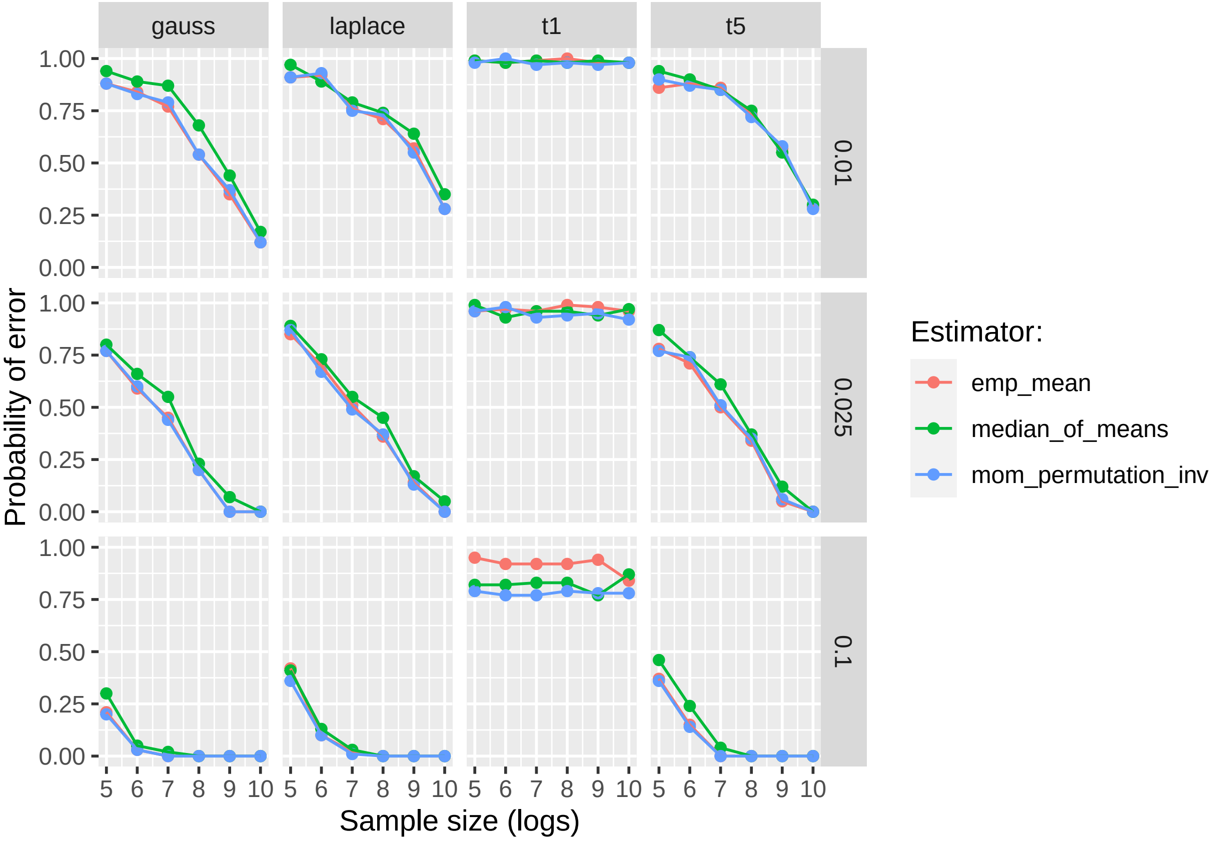


Figure 1: Probability of error by distribution (columns) and distance from expected value (rows).

## 2.2 Mean absolute error

Figure 2 instead plots the mean absolute error rates directly. The same picture emerges overall. As the bottom-left panel illustrates, it can be very dangerous to use the empirical mean in the presence of outliers. Bother median-of-mean estimators fair much better.

```
library(ggplot2)
p <- ggplot(data = error_rates[block_size_ratio==median(block_size_ratio) & N==max(N)], aes(x=log(n), y
  geom_point() +
  geom_line() +
  facet_wrap(~dist, ncol=2, scales = "free") +
  scale_color_discrete(name="Estimator:") +
  labs(
    x="Sample size (logs)",
    y="Mean absolute error"
  )
p
```

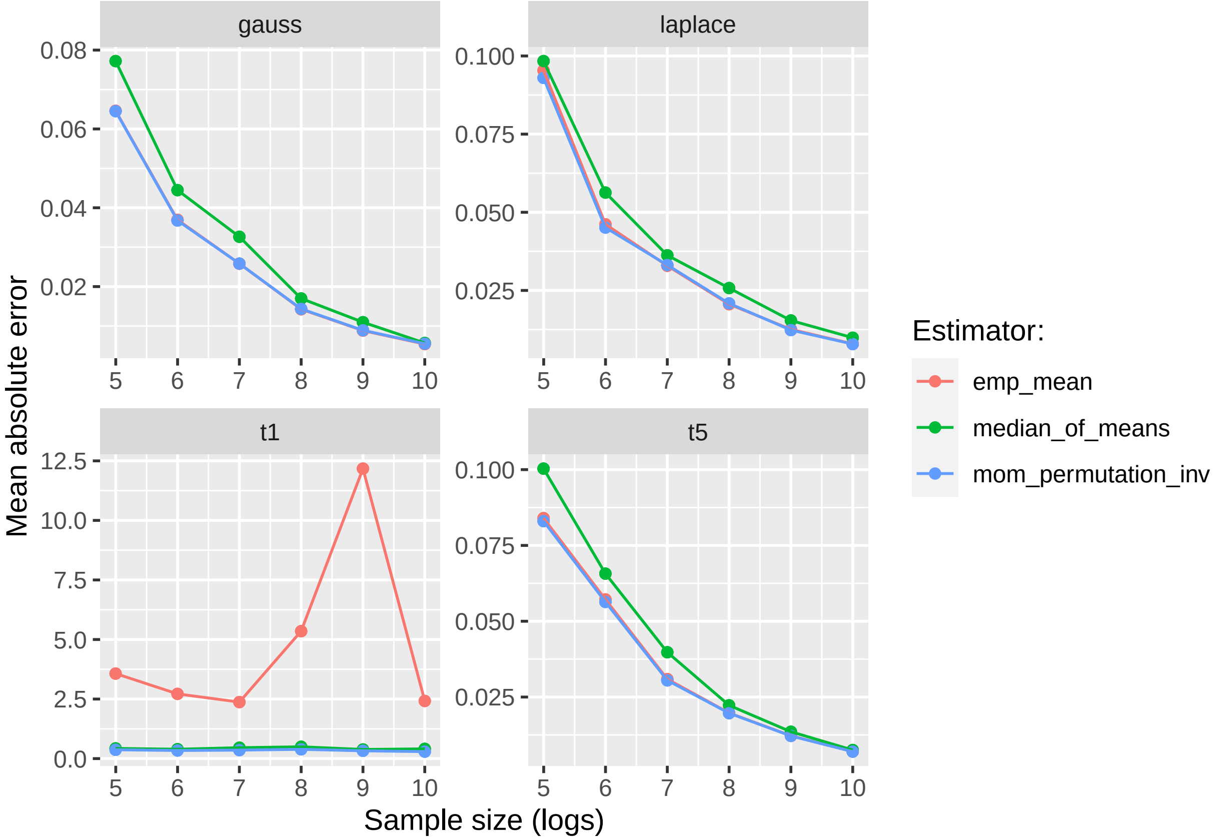


Figure 2: Mean absolute errors.

## 2.3 Block size and $N$

The next two illustrations focus on the median-of-means. Figure 3 shows the mean absolute error rates for the permutation-invariant median-of-means. I let $N$ vary across rows. Different colours correspond to different block sizes. Errors in the bottom row are generally lower, indicative of the fact that higher values of $N$ mitigates the bias associated with the median-of-means estimator. Smaller block sizes tend to have a similar effect, although this does not always hold.

```
p <- ggplot(data = error_rates[estimator=="mom_permutation_inv"], aes(x=log(n), y=mae, colour=factor(bl
  geom_point() +
  geom_line() +
  facet_wrap(
    N~dist,
    scales = "free_y",
    ncol = 4
  ) +
  scale_color_discrete(name="Block size ratio:") +
  labs(
    x="Sample size (logs)",
    y="Mean absolute error"
  )
p
```

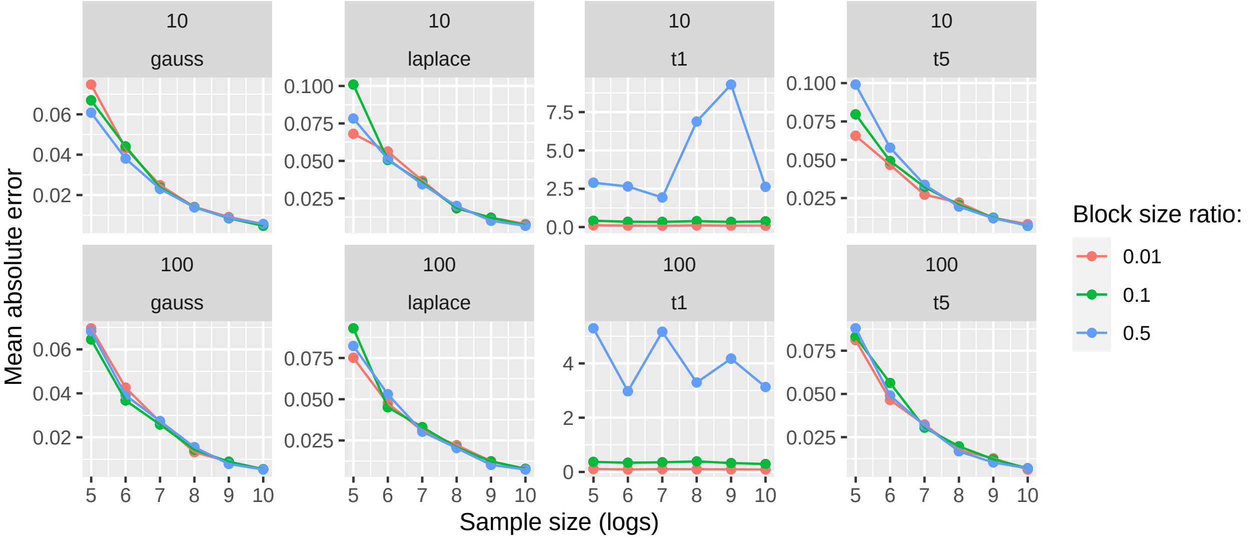


Figure 3: Permutation-invariant median-of-means for different choices of $N$ and different block sizes.

Interestingly, block size appears to have the opposite effect on the median-of-means estimator that does not take into account the effect of permutations. Excpet for the very heavy-tailed $t1$-distribution, smaller block sizes tend to produce better results. This is simply due to the fact that as the block size approaches 1, the median-of-means estimator approaches the empirical mean. As we have seen in the previous charts, the empirical mean is unbiased and hence a better choice whenever outliers are not a big deal.

```
p <- ggplot(data = error_rates[estimator=="median_of_means" & N==max(N)], aes(x=log(n), y=mae, colour=fa
  geom_point() +
  geom_line() +
  facet_wrap(
    ~dist,
    scales = "free_y",
    ncol = 2
  ) +
  scale_color_discrete(name="Block size ratio:") +
  labs(
    x="Sample size (logs)",
    y="Mean absolute error"
  )
p
```

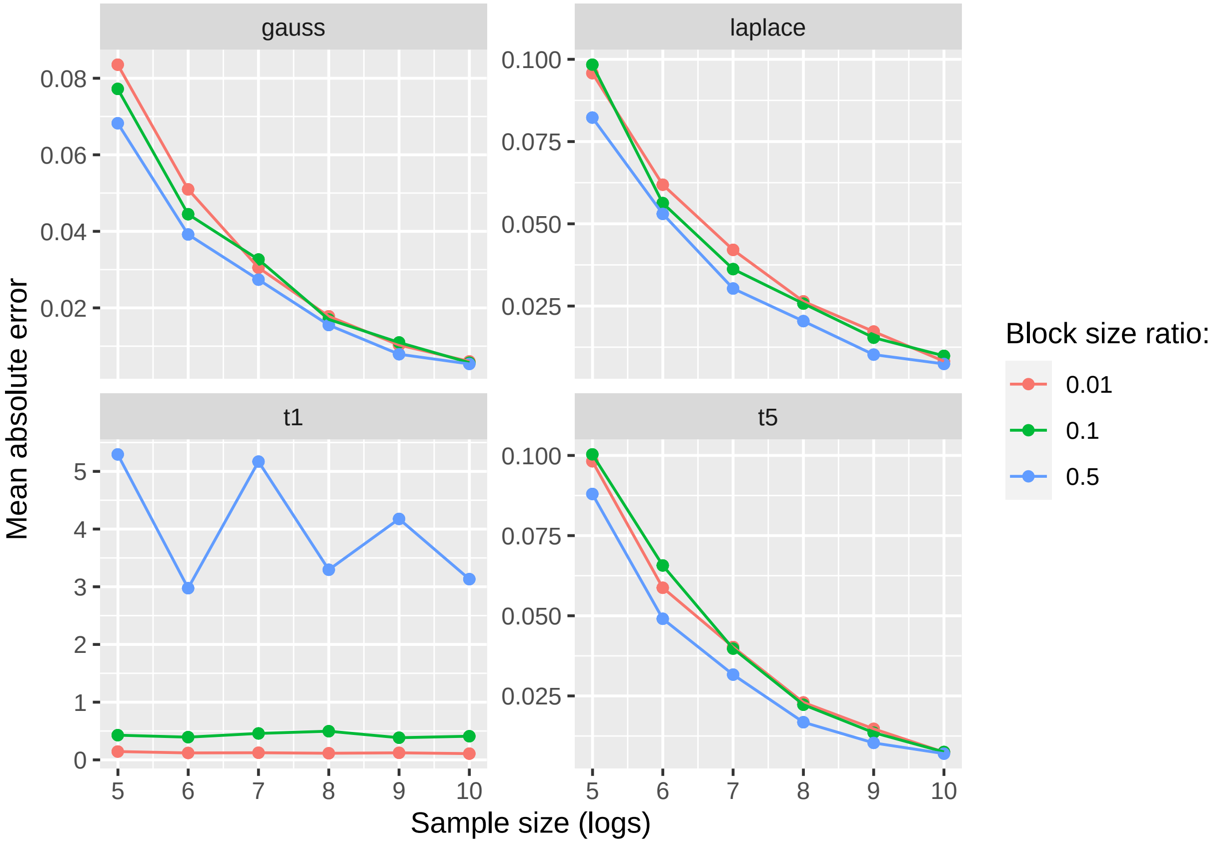
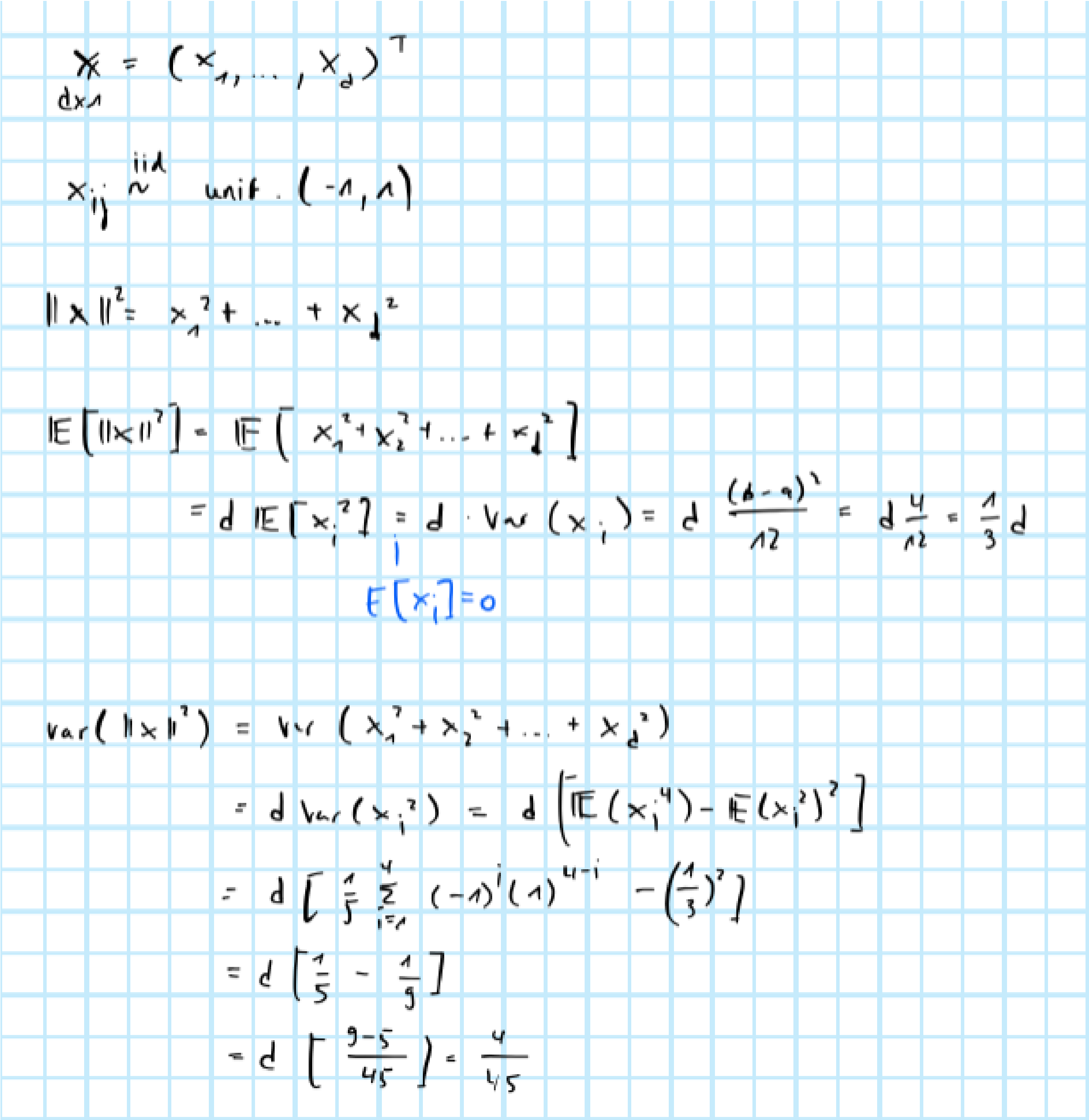


Figure 4: Median-of-means without bias correction.

# 3 Problem 3

The first two moments can be derived as follows:



$$\underset{d\times 1}{x} = (x_1, \dots, x_d)^T$$

$$x_{ij} \overset{iid}{\sim} \text{unif.}(-1,1)$$

$$\|x\|^2 = x_1^2 + \dots + x_d^2$$

$$\mathbb{E}[\|x\|^2] = \mathbb{E}[x_1^2 + x_2^2 + \dots + x_d^2]$$

$$= d\,\mathbb{E}[x_i^2] \underset{\mathbb{E}[x_i]=0}{=} d \cdot \text{Var}(x_i) = d\,\frac{(b-a)^2}{12} = d\,\frac{4}{12} = \frac{1}{3}d$$

$$\text{var}(\|x\|^2) = \text{Var}(x_1^2 + x_2^2 + \dots + x_d^2)$$

$$= d\,\text{Var}(x_i^2) = d\left[\mathbb{E}(x_i^4) - \mathbb{E}(x_i^2)^2\right]$$

$$= d\left[\frac{1}{5}\sum_{i=1}^{4}(-1)^i(1)^{4-i} - \left(\frac{1}{3}\right)^2\right]$$

$$= d\left[\frac{1}{5} - \frac{1}{9}\right]$$

$$= d\left[\frac{9-5}{45}\right] = \frac{4}{45}$$

In order to derive a concentration inequality when can apply Chernoff:

$$\mathbb{P}\left( \|x\|^2 - \frac{d}{3} \geq t \right) \leq \frac{\mathbb{E}\left[ e^{\lambda\left[x_1^2+x_i^2+\dots+x_d^2 - \frac{d}{3}\right]} \right]}{e^{\lambda \cdot t}} = \frac{\prod_{i=1}^{d} \mathbb{E}\left[e^{\lambda x_i}\right]}{e^{\lambda t + \frac{d}{3}}} = \frac{\mathbb{E}\left[ e^{\lambda x_i^2} \right]^d}{e^{\lambda t + \frac{d}{3}}} = \frac{\mathbb{E}\left[ e^{\lambda(x_i^2 - \frac{1}{3})} \right]^d}{e^{\lambda t}} \leq e^{\frac{\lambda^2}{8} d - \lambda t}$$

Optimize : $\frac{2\lambda}{8} d - t = 0$

$$\lambda^* = \frac{4t}{d}$$

$$= e^{\left( \frac{t^2 16}{d^2} \cdot \frac{d}{8} - \frac{t4}{d} t \right)}$$

$$= \exp\left( t^2 \left[ \frac{2}{d} - \frac{4}{d} \right] \right)$$

$$= \exp\left( t^2 \left[ - \frac{2}{d} \right] \right)$$

$$\Leftrightarrow \quad \mathbb{P}\left( \|x\|^2 - \frac{d}{3} \geq t \right) \leq e^{-\frac{2t^2}{d}}$$

On order to determine the “typical” magnitude of the cosine of the angle between the two independent vectors, it may suffice to specify the first two moments of the distribution of the cosine.

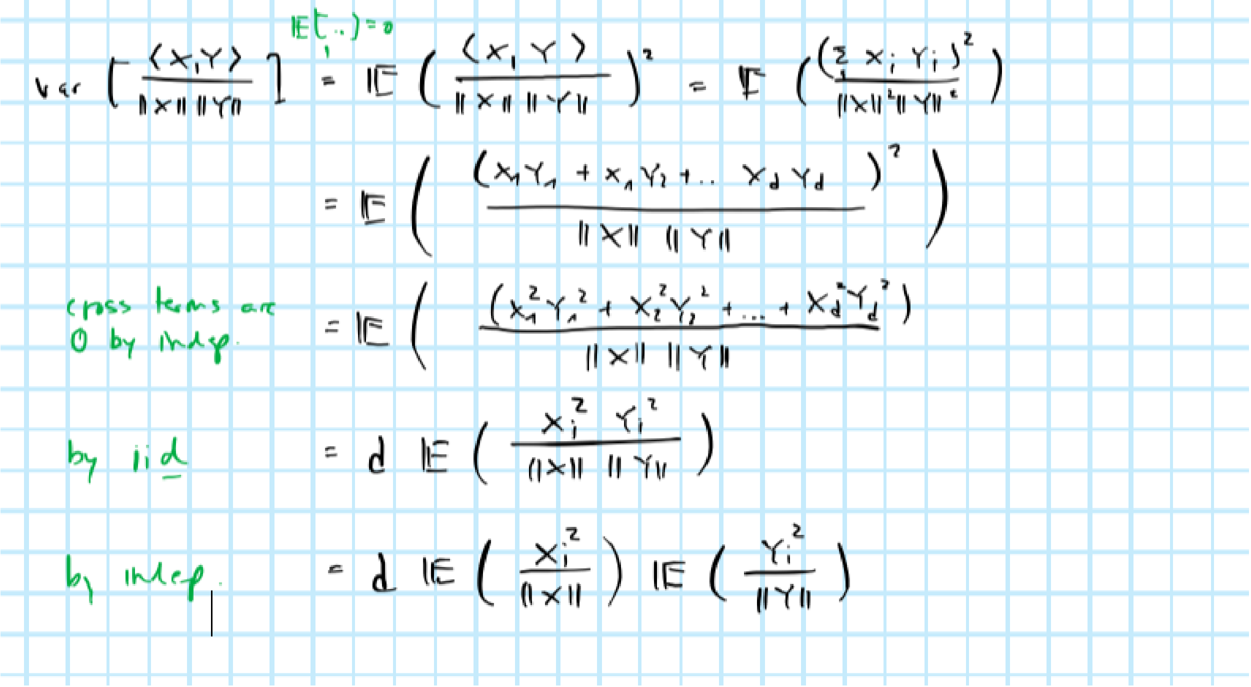
## 3.1 Unconditonal moments of cosine

In order to derive the expected value of the cosine, assume the norm of both **X** and **Y** is equal to 1. Then we have the following:

$$\mathbb{E}\left[ \frac{\langle X, Y \rangle}{\|X\| \|Y\|} \right] = \mathbb{E}\left[ \langle X, Y \rangle \right] \overset{\text{lin.}}{=} \sum_{i=1}^{d} \mathbb{E}\left[ X_i Y_i \right] \overset{\text{iid}}{=} \sum_{i=1}^{d} \underset{=0}{\mathbb{E}[\cancel{X}_i]} \, \underset{=0}{\mathbb{E}[\cancel{Y}_i]} = 0$$

Of course, this only holds under the assumption we made. But notice that we can always find some vector **a** such that $||\mathbf{a}|| = c||\mathbf{X}||$ and **b** such that $||\mathbf{b}|| = k||\mathbf{Y}||$ while the angle between **a** and **b** may still be the same as before. In other words we can stretch or squeeze both **X** and **Y** such that their norm changes, but the angle between remains the same. For such **a** and **b** we still have that the expected value of the cosine is zero.

For the variance we can proceeds as follows:



$$\text{var}\left[\frac{\langle x,Y\rangle}{\|x\|\|Y\|}\right] \overset{\mathbb{E}[..]=0}{=} \mathbb{E}\left(\frac{\langle x,Y\rangle}{\|x\|\|Y\|}\right)^2 = \mathbb{E}\left(\frac{(\sum x_i Y_i)^2}{\|x\|^2\|Y\|^2}\right)$$

$$= \mathbb{E}\left(\frac{(x_1Y_1 + x_1Y_2 + \ldots x_dY_d)^2}{\|X\|\ \|Y\|}\right)$$

cross terms are 0 by indep.
$$= \mathbb{E}\left(\frac{(x_1^2Y_1^2 + x_2^2Y_2^2 + \ldots + x_d^2Y_d^2)}{\|x\|\ \|Y\|}\right)$$

by iid
$$= d\ \mathbb{E}\left(\frac{x_i^2\ Y_i^2}{\|x\|\ \|Y\|}\right)$$

by indep.
$$= d\ \mathbb{E}\left(\frac{x_i^2}{\|x\|}\right) \mathbb{E}\left(\frac{Y_i^2}{\|Y\|}\right)$$

*Here I got stuck and did not know how to proceed.*

# 4 Problem 4

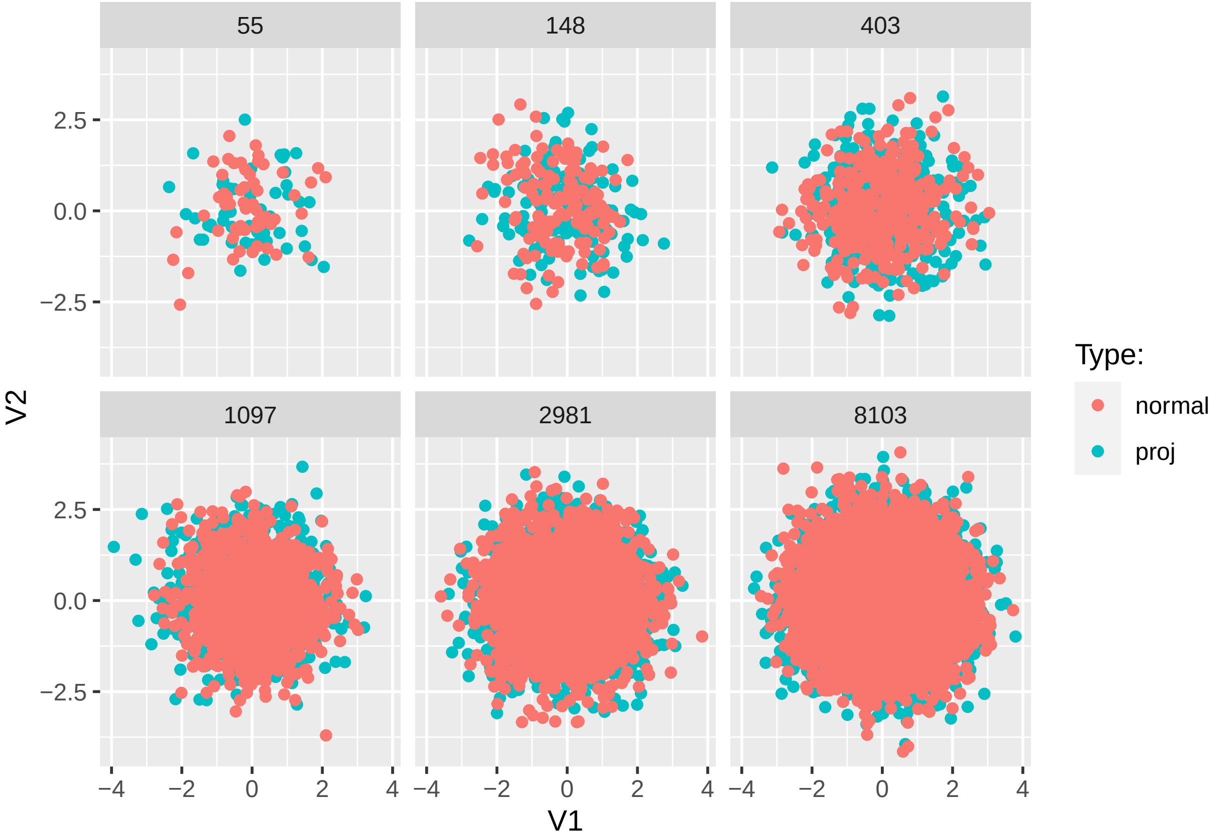
The program can be implemented in R as follows:

```r
random_projection <- function(n,d=2) {
  I <- diag(n) # matrix of basis vectors
  w <- matrix(rnorm(2*n), nrow=n) # random 2-d vector of weights
  A <- I %*% w # linear projection
  A_stand <- (A - mean(A[,1]))/sd(A[,1]) # center and rescale
  dt <- data.table(A_stand, type="proj")
  v <- matrix(rnorm(2*n), nrow=n) # some random Gaussian 2-d vector
  dt <- rbind(dt, data.table(v, type="normal"))
  dt[,n:=n]
  return(dt)
}
```

Applying this for different values of $n$ yields the picture below. It turns out that the random projection of $n$-dimesional basis vectors onto the 2-dimensional plane is also random and hence resembles random draws from a bivariate Gaussian distribution.

```r
n <- round(exp(4:9))
dt <- rbindlist(
  lapply(
    n,
    function(i) {
      random_projection(i)
    }
  )
)
```

```r
p <- ggplot(data = dt, aes(x=V1, y=V2, colour=type)) +
  geom_point() +
  facet_wrap(~factor(n), ncol=3) +
  scale_color_discrete(name="Type:")
p
```


V2
V1
Type:
● normal
● proj