aQMRA of ingested water in an IWS - Cajibio, Colombia

Angela Bayona-Valderrama

This is the analytical approach to estimating the potential health risks of ingesting fecally contaminated water in an intermittent supply system, the data comes from grab water samples taken in Cajibio, Colombia.

Install packages

```r
library(tidyverse)
library(dplyr)
library(here)
library(rriskDistributions)
library(VGAM)
library(ggridges)
library(scales)
library(hrbrthemes)
library(rstatix)
library(kableExtra)
library(EnvStats)
```

Load data

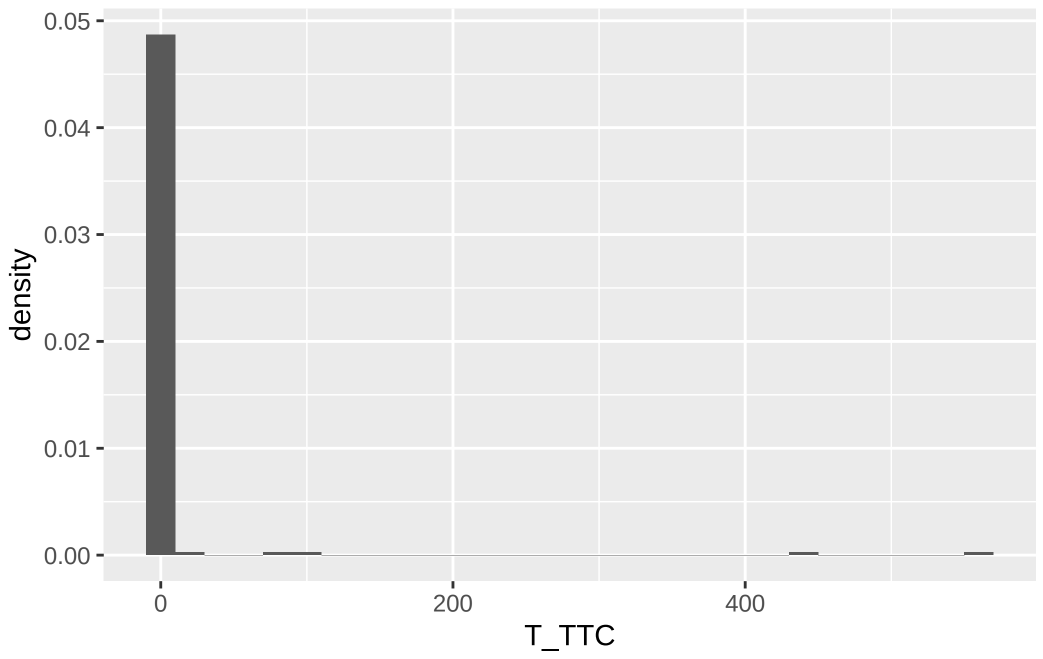
Loading collected data. It holds results from measuring Thermotolerant coliform (TTC) concentrations in drinking water in a cross-sectional study from 200 households in Cajibio, Cauca region, Colombia. The dataset includes water quality measurements of paired grab water samples taken at the tap and at the point of storage, and responses to household survey.

```r
data <- read.csv(here("data","clean_df.csv"))
```

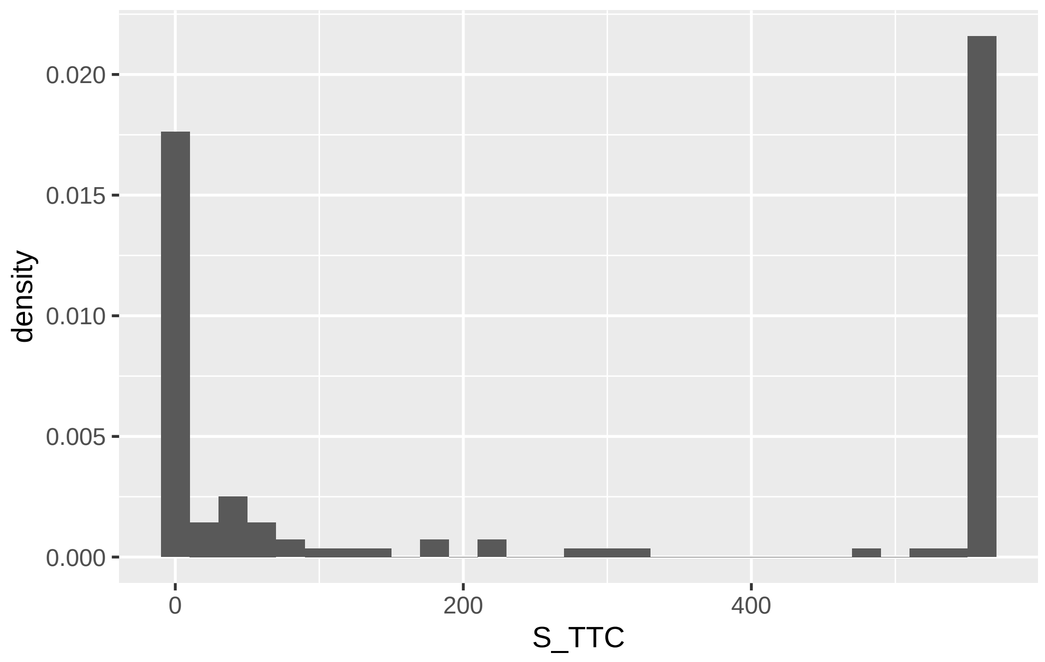
Exploratory data analysis

The data exploration shows an apparent log-normal distribution of TTC counts in tap water and bi-modal distribution of the stored water TTC counts.

```r
#frequency distribution of TTCs (CFU/100ml) in tap water (T_TTC)
data %>%
  drop_na(T_TTC) %>%
  ggplot(aes(T_TTC,y=after_stat(density)))+
  geom_histogram(binwidth = 20)
```



```r
#frequency distribution of TTCs (CFU/100ml) in stored water (S_TTC)
data %>%
  drop_na(S_TTC) %>%
  ggplot(aes(S_TTC,y=after_stat(density)))+
  geom_histogram(binwidth = 20)
```



Summary statistics for TTCs (CFU/100ml) in tap water

```
#tap water summary statistics
summary(data$T_TTC)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  0.000   0.000   0.000   6.398   0.000 552.000       9
```

```
mean(data$T_TTC, na.rm = TRUE)
```

```
[1] 6.397906
```

```
sd(data$T_TTC, na.rm = TRUE)
```

```
[1] 51.44705
```

Summary statistics for TTCs (CFU/100ml) in stored water

```
#stored water summary statistics
summary(data$S_TTC)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
    0.0     0.5   174.0   269.9   552.0   552.0      61
```

```
mean(data$S_TTC, na.rm = TRUE)
```

```
[1] 269.8777
```

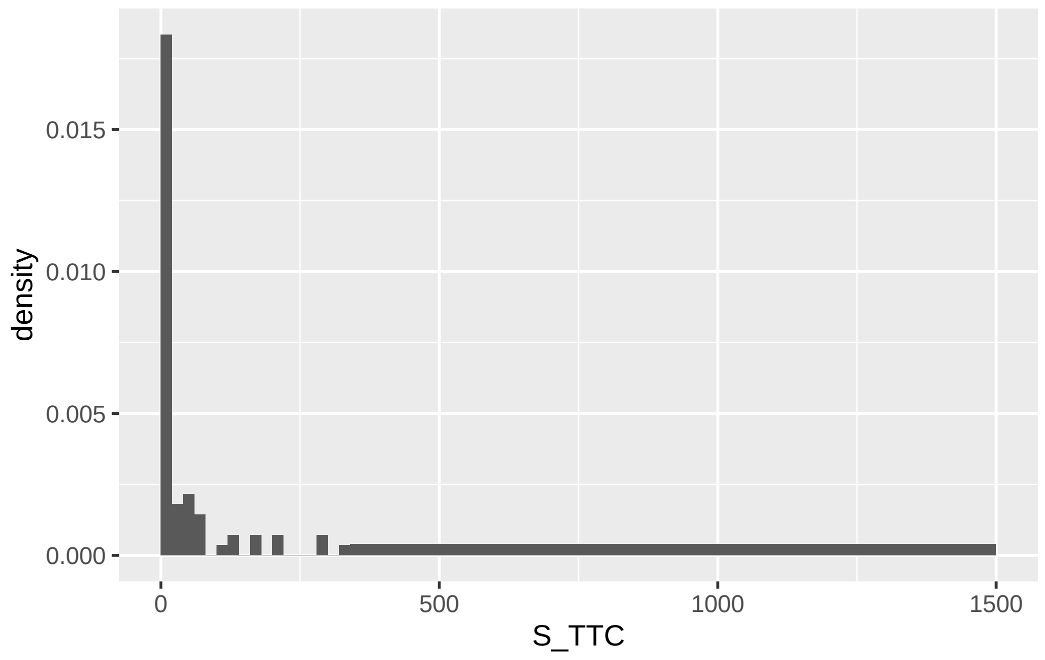
```
sd(data$S_TTC, na.rm = TRUE)
```

```
[1] 261.818
```

During data collection 62 observations had to be classified as too numerous to count (TNTC) and a value of 552 CFU (having a max count of 551, TNTC was set as max+1) was assigned to those observations. The second mode in the TTC counts of stored water (S_TTC) is actually representing the cumulative frequency of values >551.

In the following code the value of the last bin is assumed arbitrarily as 1500 to illustrate the accumulated frequencies of samples with values >551. The histogram below shows the frequency density. Visual inspection guides into fitting a log-normal distribution in the following Maximum Likelihood Estimation (MLE)

```
bins <- c(seq(0,340,20),1500) #1500 is an arbitrary number just to illustrate the cumulati

data  %>%
  drop_na(S_TTC) %>%
  ggplot(aes(S_TTC,
              y=after_stat(density)
              ))+
  geom_histogram(breaks = bins)
```



Maximum likelihood estimation

Vector for analysis of TTCs in stored water

Saving stored TTCs concentration data for MLE in two separate vectors. NA values are dropped. Zero values will be assumed as < 1 in the MLE.

```
#vector for tap water data
T_TTC <- data %>%
  drop_na(T_TTC) %>%
  pull(T_TTC)
```

```
#vector for stored water data
S_TTC <- data %>%
  drop_na(S_TTC) %>%
  pull(S_TTC)
```

Probability density function fitting

To perform a MLE in R one defines a negative log likelihood function and minimizes it, since R's standard optimisation, or gradient search, follows a minimisation routine.

The following code defines a negative log likelihood function for the TTC concentrations found in stored water. Note that zero values are subset to calculate the probability between 0 and 1 (sum0), while the TNTC values are subset and treated differently by calculating the probability of obtaining a value greater than 551 CFU (sum2).

```
negloglike = function(parameters,
                      threshold,
                      counts){

  mean = parameters[1]
  sd = parameters[2]

  counts0 = counts[counts < 1]
  sum0 = sum(-log(pnorm(log(counts0+1), mean=mean, sd = sd)))

  counts1 = counts[counts >= 1 & counts<=threshold]
  sum1 = sum(-log(dnorm(log(counts1), mean=mean, sd = sd)))

  # Probabilities and likelihoods for the TNC values (>551)
  counts2 = counts[counts>threshold]

  # Setting a flag for the counts using the threshold
  prob2 = 1 - pnorm(log(counts2), mean = mean, sd = sd)
```

```r
  sum2 = sum(-log(prob2))

  return(sum0+sum1+sum2)
}
```

```r
negloglike_zip <- function(parameters,
                           threshold,
                           counts){

  lambda = parameters[1]
  pstr0 = parameters[2]

  counts1 = counts[counts<=threshold]
  sum1 = sum(-log(dzipois(counts1,lambda = lambda,pstr0 = pstr0)))

  # Probabilities and likelihoods for the TNC values (>551)
  counts2 = counts[counts>threshold]

  # Setting a flag for the counts using the threshold
  prob2 = 1 - pzipois(counts2,lambda = lambda,pstr0 = pstr0)
  sum2 = sum(-log(prob2))

  return(sum1+sum2)
}
```

```r
negloglike_nbinom <- function(parameters,
                              threshold,
                              counts){

  size = parameters[1]
  mu = parameters[2]

  counts1 = counts[counts<=threshold]
  sum1 = sum(-log(dnbinom(counts1, size = size, mu = mu)))

  # Probabilities and likelihoods for the TNC values (>551)
  counts2 = counts[counts>threshold]

  # Setting a flag for the counts using the threshold
  prob2 = 1 - pnbinom(counts2, size = size, mu = mu)
  sum2 = sum(-log(prob2))

  return(sum1+sum2)
}
```

This is a test of the `negloglike` functions for the TTC data of stored water.

```
negloglike(parameters = c(mean(S_TTC),sd(S_TTC)),
           threshold = 551,
           counts =  2)
```

```
[1] 7.01512
```

```
negloglike_zip(parameters = c(600,0.9),
           threshold = 551,
           counts =  c(2,15,560))
```

```
[1] 1126.806
```

```
negloglike_nbinom(parameters = c(200,600),
           threshold = 551,
           counts =  c(2,15,560))
```

```
[1] 497.6503
```

This is a test of the `negloglike` function for the TTC data of tap water

```
negloglike(parameters=c(mean(T_TTC),sd(T_TTC)),
           threshold = 551,
           counts=2)
```

```
[1] 4.865639
```

To identify the MLE estimates, the negloglike function is minimised using the parameters that define the lognormal distribution and the `nlm` function. The following code defines this parameters for the concentration of stored TTCs

```
out_stored <- nlm(negloglike,
           p = c(mean(S_TTC,
                      na.rm = TRUE), sd(S_TTC, na.rm = TRUE)),
           hessian=TRUE,
           threshold = 551,
           counts = S_TTC)

out_stored
```

```
$minimum
[1] 232.1302

$estimate
[1] 4.999007 7.309606

$gradient
[1] -3.932640e-05 -1.675455e-05

$hessian
           [,1]       [,2]
[1,]  2.0841030 -0.3804906
[2,] -0.3804906  1.1862280

$code
[1] 1

$iterations
[1] 25
```

```
out_stored_zip <- nlm(negloglike_zip,
           p = c(500, 0.9),
           hessian=TRUE,
           threshold = 551,
           counts = S_TTC)

out_stored_zip
```

```
$minimum
[1] 12523.03

$estimate
[1] 389.8176122   0.9130024

$gradient
[1]  32.31459 478.92336

$hessian
             [,1]         [,2]
[1,] 3.253543e-02 4.666258e-07
[2,] 4.666258e-07 6.001301e+03

$code
[1] 2
```

```
$iterations
[1] 32
```

```
out_stored_nbinom <- nlm(negloglike_nbinom,
           p = c(200,600),
           hessian=TRUE,
           threshold = 551,
           counts = S_TTC)

out_stored_nbinom
```

```
$minimum
[1] 391.32

$estimate
[1]    0.1237315 9556.7858647

$gradient
[1] 2.478373e-05 5.293688e-10

$hessian
             [,1]         [,2]
[1,] 5.258360e+03 8.681463e-03
[2,] 8.681463e-03 3.119144e-08

$code
[1] 1

$iterations
[1] 37
```

As for the tap water concentrations, the log transformed data will be fitted to a normal distribution. The same `negloglike` function can be used for this set of data.

```
out_tap <- nlm(negloglike,
           p = c(mean(T_TTC, na.rm = TRUE),
                 sd(T_TTC, na.rm = TRUE)),
           hessian=TRUE,
           threshold = 551,
           counts = T_TTC)

out_tap
```

```
$minimum
```

```
[1] 82.16034

$estimate
[1] -6.906716  4.974429

$gradient
[1] 3.508108e-06 6.576310e-06

$hessian
         [,1]     [,2]
[1,] 2.466257 3.681995
[2,] 3.681995 6.295158

$code
[1] 1

$iterations
[1] 22
```

```
#| message: false

out_tap_zip <- nlm(negloglike_zip,
           p = c(500, 0.9),
           hessian=TRUE,
           threshold = 551,
           counts = T_TTC)

out_tap_zip
```

```
$minimum
[1] 4991.873

$estimate
[1] 395.2932652   0.9986901

$gradient
[1]    13.30506 11280.43552

$hessian
              [,1]      [,2]
[1,]  4.286957e-03 -133051.4
[2,] -1.330514e+05      -Inf

$code
[1] 2
```

```
$iterations
[1] 19
```

```
out_tap_nbinom <- nlm(negloglike_nbinom,
           p = c(200,600),
           hessian=TRUE,
           threshold = 551,
           counts = T_TTC)

out_tap_nbinom
```

```
$minimum
[1] 113.3665

$estimate
[1] 0.01335192 8.63330879

$gradient
[1] -1.354437e-04 -1.629589e-07

$hessian
             [,1]       [,2]
[1,] 71400.914084 4.51822796
[2,]     4.518228 0.02712418

$code
[1] 2

$iterations
[1] 64
```

Validation of fitted distributiones

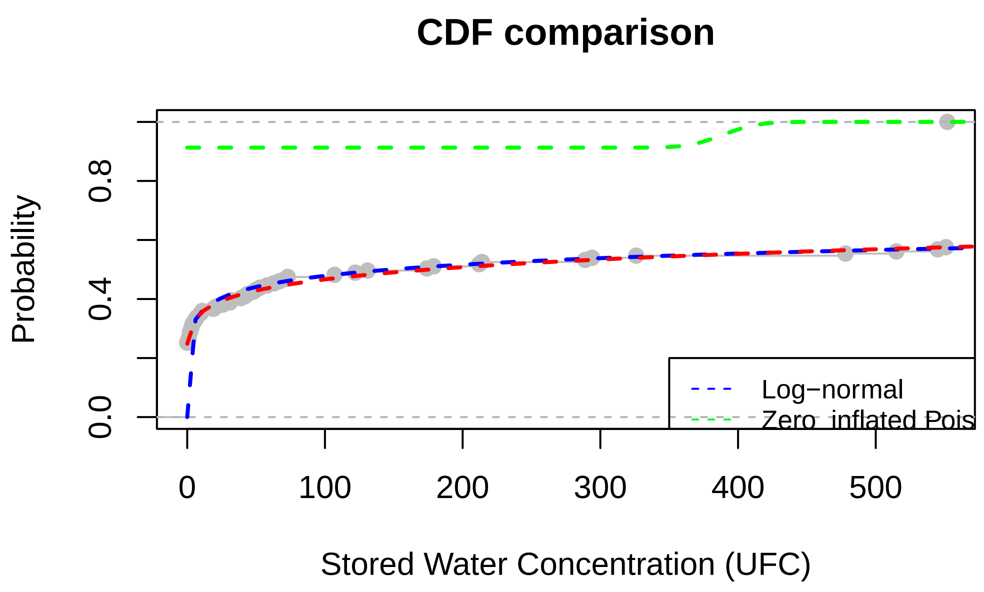
Stored

```
tibble(prob_function = c("Log-normal", "Zero_inflated Poisson","Negative Binomial"),
       Neg_log_likelihood = c(out_stored$minimum,
                              out_stored_zip$minimum,
                              out_stored_nbinom$minimum)) |>
kable(digits = 3) |>
  kable_paper()
```

prob_function	Neg_log_likelihood
Log-normal	232.13
Zero_inflated Poisson	12523.03
Negative Binomial	391.32

CDF comparison

```
plot(ecdf(S_TTC),col = "gray",
     main="CDF comparison",
     xlab = "Stored Water Concentration (UFC)",
     ylab = "Probability",xlim = c(0,550),)

curve(plnorm(x,meanlog = out_stored$estimate[1],sdlog = out_stored$estimate[2]),
      from = 0,
      to = 600,
      ylim =c(0,1),
      add = T,
      col = "blue",type="l", lty=2, lwd=2)

curve(pzipois(x,lambda = out_stored_zip$estimate[1],pstr0 = out_stored_zip$estimate[2]),
      from = 0,
      to = 600,
      ylim =c(0,1),
      add = T,
      col = "green",type="l", lty=2, lwd=2)

curve(pnbinom(x,size =out_stored_nbinom$estimate[1],mu = out_stored_nbinom$estimate[2]),
      from = 0,
      to = 600,
      ylim =c(0,1),
      add = T,
      col = "red",type="l", lty=2, lwd=2)

# Add a legend
legend(350, 0.2, legend=c("Log-normal", "Zero_inflated Poisson","Negative Binomial"),
       col=c("blue","green","red"), lty=2, cex=0.8)
```

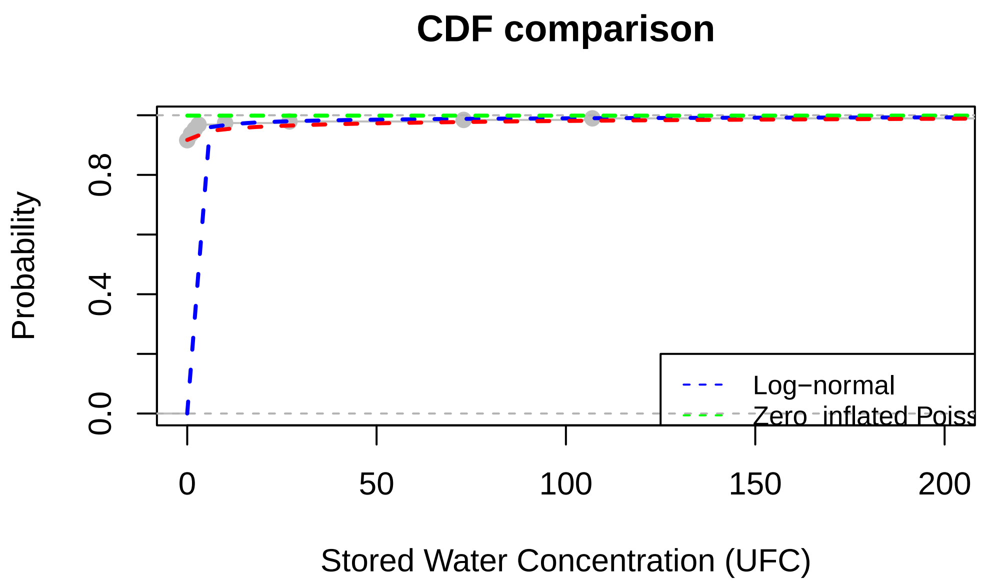


Tap

```
tibble(prob_function = c("Log-normal", "Zero_inflated Poisson","Negative Binomial"),
       Neg_log_likelihood = c(out_tap$minimum,
                              out_tap_zip$minimum,
                              out_tap_nbinom$minimum))|>
kable(digits = 3) |>
  kable_paper()
```

prob_function	Neg_log_likelihood
Log-normal	82.160
Zero_inflated Poisson	4991.873
Negative Binomial	113.366

```
plot(ecdf(T_TTC),col = "gray",
     main="CDF comparison",
     xlab = "Stored Water Concentration (UFC)",
     ylab = "Probability",xlim = c(0,200),)

curve(plnorm(x,meanlog = out_tap$estimate[1],sdlog = out_tap$estimate[2]),
      from = 0,
      to = 600,
      ylim =c(0,1),
      add = T,
      col = "blue",type="l", lty=2, lwd=2)

curve(pzipois(x,lambda = out_tap_zip$estimate[1],pstr0 = out_tap_zip$estimate[2]),
      from = 0,
      to = 600,
      ylim =c(0,1),
      add = T,
      col = "green",type="l", lty=2, lwd=2)

curve(pnbinom(x,size =out_tap_nbinom$estimate[1],mu = out_tap_nbinom$estimate[2]),
      from = 0,
      to = 600,
      ylim =c(0,1),
      add = T,
      col = "red",type="l", lty=2, lwd=2)

# Add a legend
legend(125, 0.2, legend=c("Log-normal", "Zero_inflated Poisson","Negative Binomial"),
       col=c("blue","green","red"), lty=2, cex=0.8)
```



The following code uses the outcome of the previous MLE, point and range estimates taken from literature, and Montecarlo tecniques to assess acute diahrreal disease (ADD) risk of infection and illness given the ingestion of microbiologically contaminated drinking water in an intermittent water supply system. The code is organised to follow the canonical QMRA framework: Exposure Assessment, Dose-Response calculation, and Risk Characterisation (Haas et al 1999, Haas et al 2014).

Exposure assessment

First, a random number generator seed and a predefined number of iterations are set

```
#set random number generator seed
set.seed(123)
iter=10^4
```

Second, point estimates for risk calculations (based on assumptions from literature) are defined

```
#####Input of point estimates

#pathogen info
morbidity <- 200/1000 # (80 to 200/1000inhab) national prevalence of ADD in children under
```

Pathogen concentrations

Pathogen concentrations are estimated using the results from the previous MLE and point estimates from literature (ETEC=Enterotoxigenic E Coli, Campy=Campylobacter jejuni, rota=rotavirus)

Estimation of TTC concentrations from MLE

```
#####Input of ranges
    ####Ranges of concentration of TTCs in tap and stored water are evaluated by
    ####sampling data from the distributions estimated in the MLE section

#concentrations in tap water from distribution fitted in MLE in previous section
tap = exp(rnorm(iter, mean = out_tap$estimate[1],sd = out_tap$estimate[2])) #conc of TTCs

#concentrations in stored water from distribution fitted in MLE in previous section
stored = exp(rnorm(iter, mean = out_stored$estimate[1],
                     sd= out_stored$estimate[2]))#conc of TTCs in stored water (CFU/100ml)
```

Pathogen ratios to estimate concentration

In the code below the concentration of pathogens is estimated using the TTC estimates from the MLE, the point estimates and PDFs taken from literature.

A dataframe of the estimated tap and stored water TTC concentrations is created under 'sim_conc'.

```
sim_conc <- rbind(tibble(w_source = "tap", conc_0 = tap),
      tibble(w_source = "stored", conc_0 = stored))

#providing this summary to do "reality check" of estimates
sim_conc %>%
  summarise(count = n(),
            min.conc=min(conc_0),
            mean.conc = mean(conc_0),
            median.conc = mean(conc_0),
            p95.conc = quantile(conc_0,0.95),
            p5.conc = quantile(conc_0,0.05),
            max.conc=max(conc_0),
            .by = w_source)
```

```
# A tibble: 2 x 8
  w_source count min.conc      mean.conc    median.conc  p95.conc p5.conc max.conc
  <chr>    <int>    <dbl>          <dbl>          <dbl>     <dbl>   <dbl>    <dbl>
1 tap      10000 4.93e-12           54.3           54.3    3.53e0 2.84e-7  2.06e 5
2 stored   10000 1.25e- 9 23565004341.   23565004341.      2.48e7 8.76e-4  1.27e14
```

In the code below we estimate concentration of pathogens from TTC concentrations in 'sim_conc_pathogens' dataframe by using previously published pathogen to TTC ratios that are relevant to IWS (Bivins et al 2017) and Colombian (Barragán et al 2021) research.

```
sim_conc_pathogens <- sim_conc %>%
  mutate(
    #estimation of concentration of ETEC
    #from tap and stored TTC MLE, using direct estimate from Barragan et al (2021)
    ETEC = (conc_0 * 0.076),

    #estimation of concentration of Campylobacter jejuni
    #from tap and stored TTC MLE, using lognormal values from Bivins et al (2017)      #an
    campy = conc_0/(1+1/rlnorm(nrow(sim_conc), 0.0089, 1.33)),


    #estimation of concentration of Rotavirus
    #from tap and stored TTC MLE, using lognormal values from Bivins et al (2017)      #an
    rota=conc_0/(1+1/rlnorm(nrow(sim_conc), 8.79e-7, 1.77e-6)))%>%

  # Converting the table to long format
pivot_longer(cols = ETEC:rota, names_to = "pathogen",values_to = "conc_p")

#summary table for "reality check"
sim_conc_pathogens %>%
  summarise(count = n(),
            min.conc_p = min(conc_p),
            mean.conc_p = mean(conc_p),
            median.conc_p = mean(conc_p),
            p95.conc_p = quantile(conc_p,0.95),
            p5.conc_p = quantile(conc_p,0.05),
            max.con_p= max(conc_p),
            .by = w_source)
```

```
# A tibble: 2 x 8
  w_source count min.conc_p  mean.conc_p median.conc_p p95.conc_p    p5.conc_p
  <chr>    <int>      <dbl>        <dbl>         <dbl>      <dbl>        <dbl>
1 tap      30000   3.75e-13         20.3          20.3       1.00 0.0000000578
2 stored   30000   9.48e-11 6933517435.   6933517435.  7041189.   0.000183
# i 1 more variable: max.con_p <dbl>
```

Volume of ingested water

Volume of ingested water is simulated using two sources of information: age segregated calculation using the EPA Exposure Handbook (2019) and the common assumption of of 1-2L range from WHO (2017).

Age segregated estimations of ingestion of water, EPA (2019)

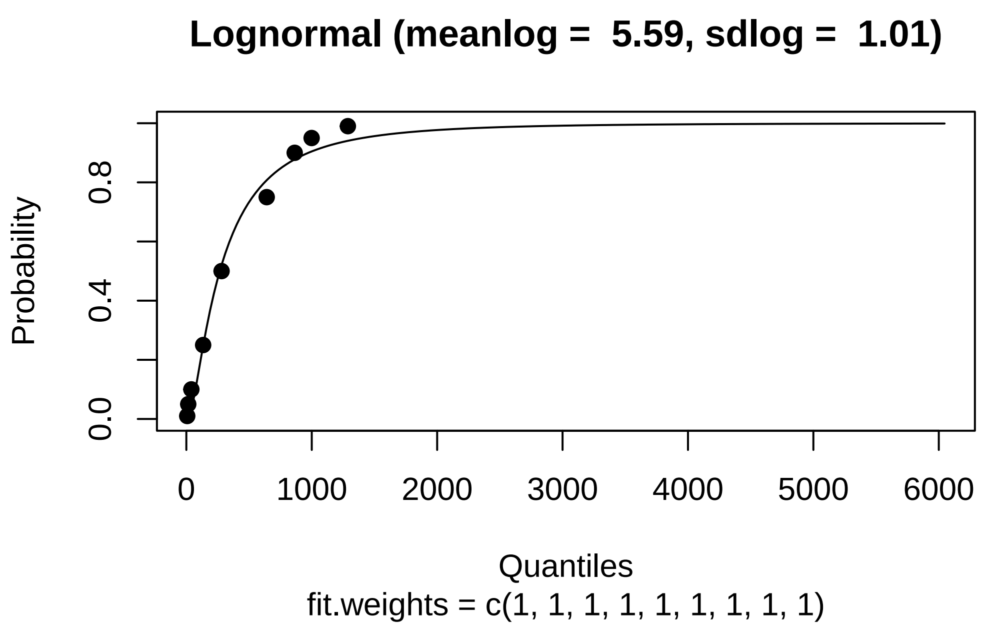
The code below is implemented to fit log-normal distributions to water ingestion data (from EPA exposure Handbook 2019). The reported volumes of ingested water are organised per percentiles in individual vectors to then obtain means and standard deviations, per age group, from the fitted distribution.

Through the code below we obtain estimated parameters for age-seggregated PDFs, these parameters will later on be used to sample random numbers for each age group.

```
#EPA Handbook 2019
#data comes fromTable 3-17 Two day average consumer only estimates of combined #direct and
#The code below fits truncated normal and lognormal distributions (previously #tested for

#creating a vector with percentiles
```

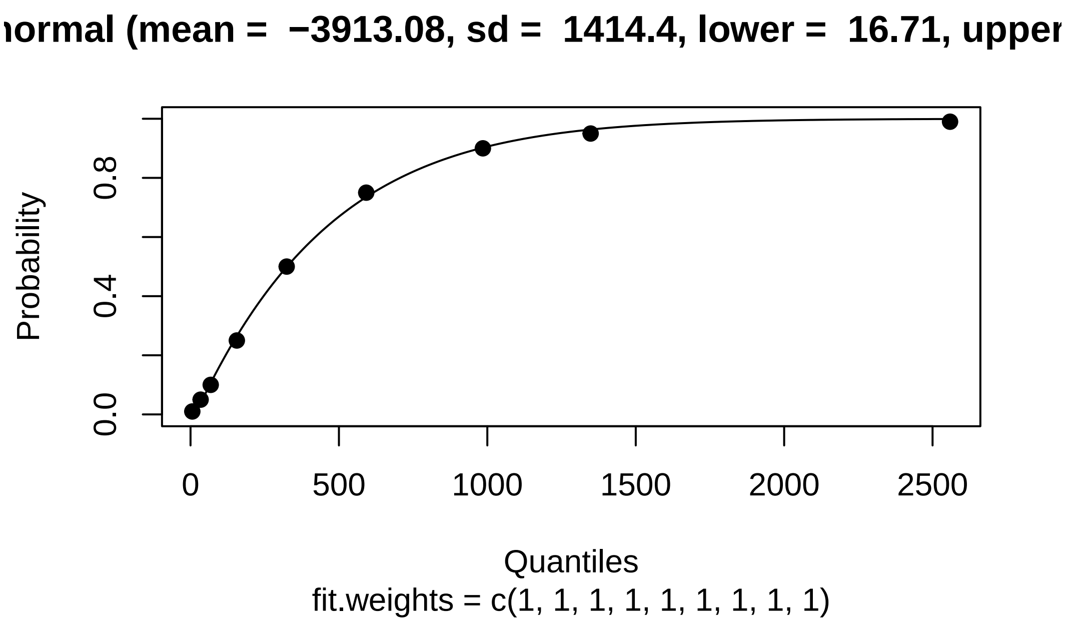
```r
p=c(0.01,0.05,0.10,0.25,0.5,0.75,0.90,0.95,0.99)

#truncated normal pdf fitted to ingestion of water age Birth to <2 years
vol_fitted_lnorm_2=get.lnorm.par(p = p, q = c(7,15,40,134,281,641,864,999,1288))
```

```
Warning: The fitting procedure 'L-BFGS-B' has failed (convergence error
occurred or specified tolerance not achieved)!


The fitting procedure 'Nelder-Mead' was successful!
(Used this fallback optimization method because 'L-BFGS-B' has failed...)


$par
[1] 5.586054 1.010009

$value
[1] 0.000197485

$counts
function gradient
      71       NA

$convergence
[1] 0

$message
NULL
```



```
#truncated normal pdf to estimate ingestion of water age 2 to 16 years
vol_fitted_tnorm_16=get.tnorm.par(p = p,
                  q = c(6,  34, 68, 156,    324,    592,    985,    1348,   2559))
```

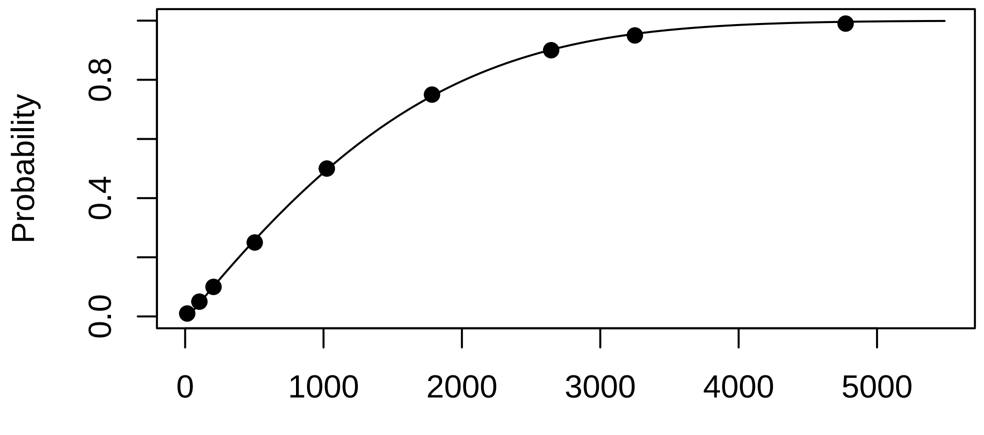
```
Warning: The fitting procedure 'L-BFGS-B' has failed (convergence error
occurred or specified tolerance not achieved)!


The fitting procedure 'Nelder-Mead' was successful!
(Used this fallback optimization method because 'L-BFGS-B' has failed...)


$par
[1] -3913.08418  1414.39918    16.70938  4360.74911

$value
[1] 1.256922e-05

$counts
function gradient
     395       NA

$convergence
[1] 0

$message
NULL
```



```
#truncated normal pdf to estimate ingestion of water age 16 to 70 years
vol_fitted_tnorm_70=get.tnorm.par(p = p,
                  q = c(15, 103,    205,    503,    1024,    1784,    2645,    3250,    4773))
```

```
Warning: The fitting procedure 'L-BFGS-B' has failed (convergence error
occurred or specified tolerance not achieved)!

The fitting procedure 'Nelder-Mead' was successful!
(Used this fallback optimization method because 'L-BFGS-B' has failed...)

$par
[1] -601.56319 1802.49549   24.13796 9192.98249

$value
[1] 3.855001e-06

$counts
function gradient
     477       NA

$convergence
[1] 0

$message
NULL
```

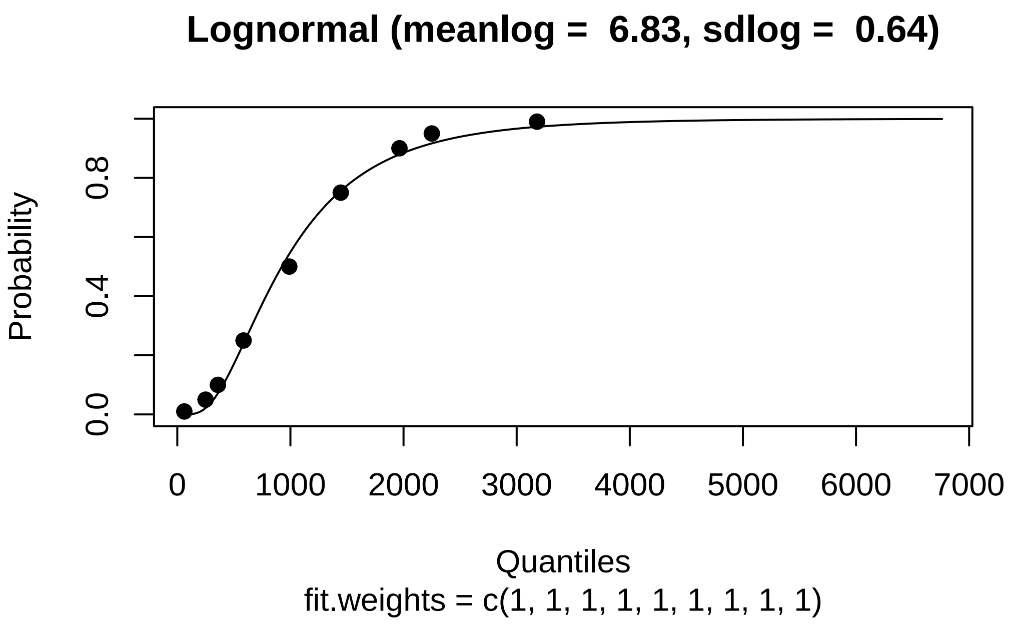
normal (mean = −601.56, sd = 1802.5, lower = 24.14, upper



fit.weights = c(1, 1, 1, 1, 1, 1, 1, 1, 1)

```
#truncated normal pdf to estimate ingestion of water age 80+ years
vol_fitted_lnorm_80=get.lnorm.par(p = p,
                    q = c(62, 250, 359,   586,    990,    1445,    1964,    2250,    3180))
```

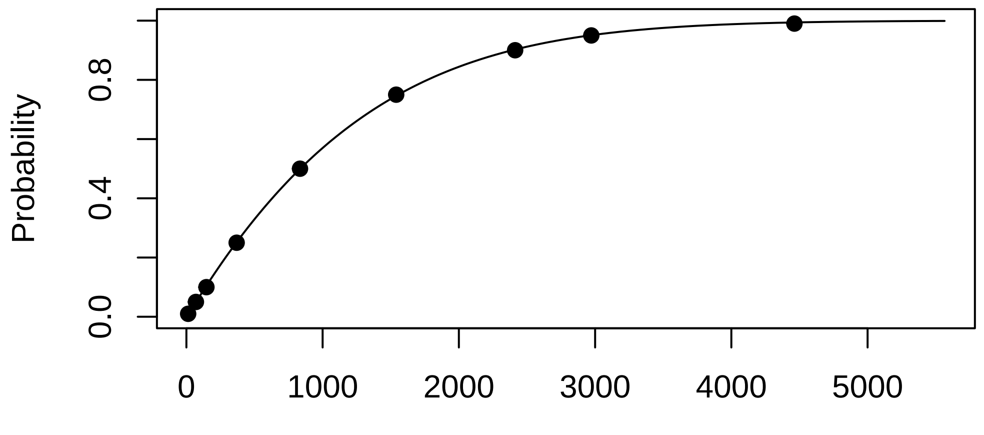
```
Warning: The fitting procedure 'L-BFGS-B' has failed (convergence error
occurred or specified tolerance not achieved)!

The fitting procedure 'Nelder-Mead' was successful!
(Used this fallback optimization method because 'L-BFGS-B' has failed...)

$par
[1] 6.8284280 0.6441207

$value
[1] 6.905619e-05

$counts
function gradient
      93       NA

$convergence
[1] 0

$message
NULL
```



```r
#truncated normal pdf to estimate ingestion of water all ages
vol_fitted_tnorm_all=get.tnorm.par(p = p,
                  q = c(13, 70, 147,    369,    834,    1540,    2413,    2972,    4463))
```

```
Warning: The fitting procedure 'L-BFGS-B' has failed (convergence error
occurred or specified tolerance not achieved)!

The fitting procedure 'Nelder-Mead' was successful!
(Used this fallback optimization method because 'L-BFGS-B' has failed...)

$par
[1] -2882.262020  2274.835068     4.797037 11460.142813

$value
[1] 9.694903e-07

$counts
function gradient
     221       NA

$convergence
[1] 0

$message
NULL
```

ıormal (mean = −2882.26, sd = 2274.84, lower = 4.8, upper :



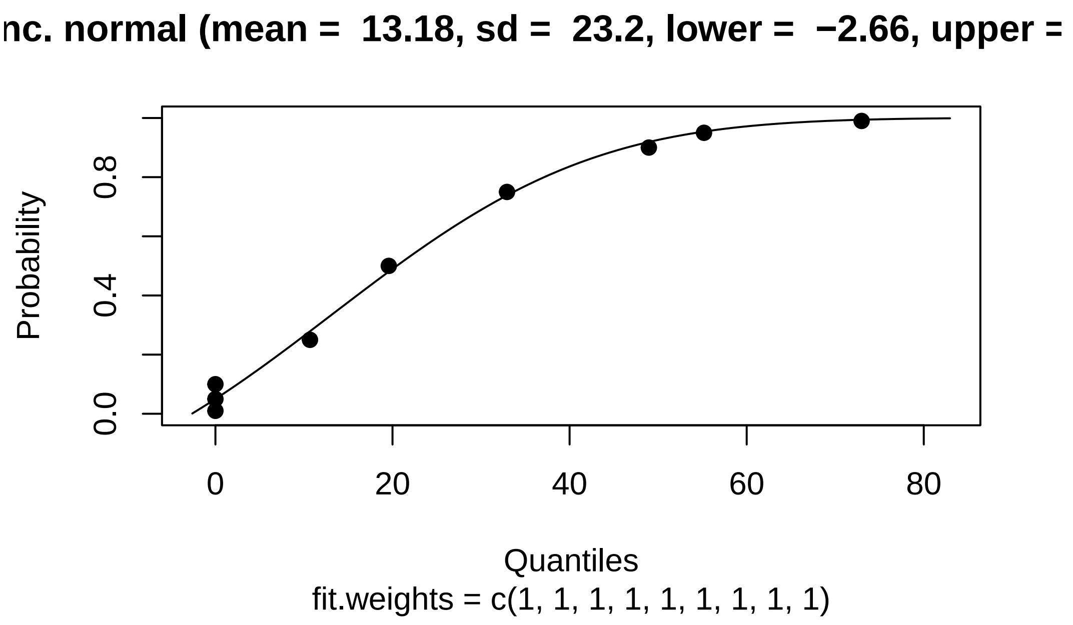
fit.weights = c(1, 1, 1, 1, 1, 1, 1, 1, 1)

Through the code below we obtain estimated parameters for the chosen PDFs (per age group), these parameters will later on be used to to sample random numbers for intake factors by age group.

```
#EPA Handbook 2019
#Table 3-31 Total tap water intake (as percentage of total water intake)
#Table 3-32 General Dietary sources of tap water for both sexes

#The code below fits truncated normal distributions to the data reported in
#Table 3-31. The resulting values are then multplied by an estimated percentage
#factor (i.e., intake factor) obtained from the sum of means of Drinking water and Other b

### <1 years, equivalent to 0_to_2
Intake_fitted_2 = get.tnorm.par(p = p,
                                q = c(0, 0, 0, 12, 22, 37, 55, 62, 82)*.89)/100
```

```
The fitting procedure 'L-BFGS-B' was successful!


$par
[1] 13.183353 23.200243 -2.663588 88.745444

$value
[1] 7.211702e-05

$counts
```

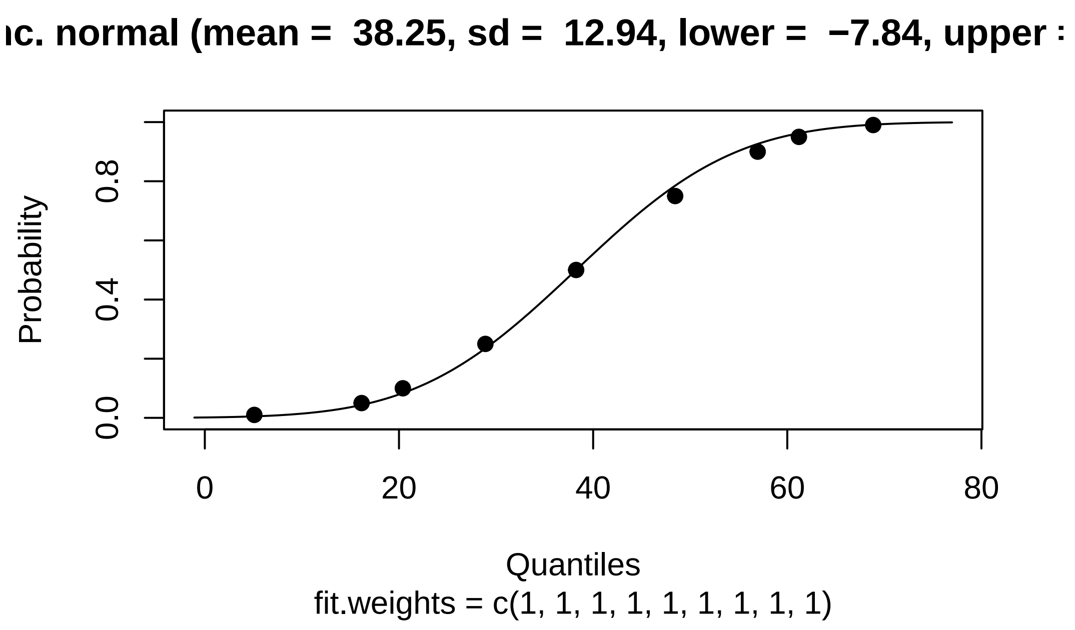
```
function gradient
      24       24

$convergence
[1] 0

$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```



```
### 1 to 10 years of age, equivalent to 2_to_16
Intake_fitted_16 = get.tnorm.par(p = p,
                                 q = c(6, 19, 24, 34, 45, 57, 67, 72, 81)*.85)/100
```

```
The fitting procedure 'L-BFGS-B' was successful!

$par
[1] 38.250014 12.938605 -7.838561 81.788561

$value
[1] 3.236196e-05

$counts
function gradient
       2        2
```

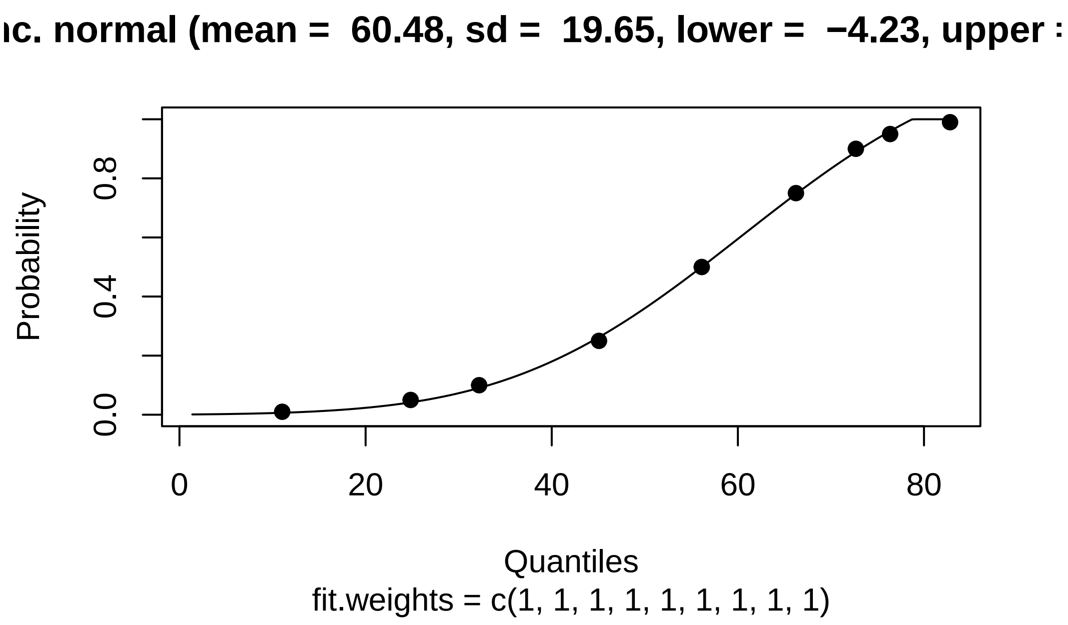
```
$convergence
[1] 0

$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```



```r
## 20 to 64 years of age, equivalent to 16_to_70
Intake_fitted_70 = get.tnorm.par(p = p,
                                 q = c(12, 27, 35, 49, 61, 72, 79, 83, 90)*.92)/100
```

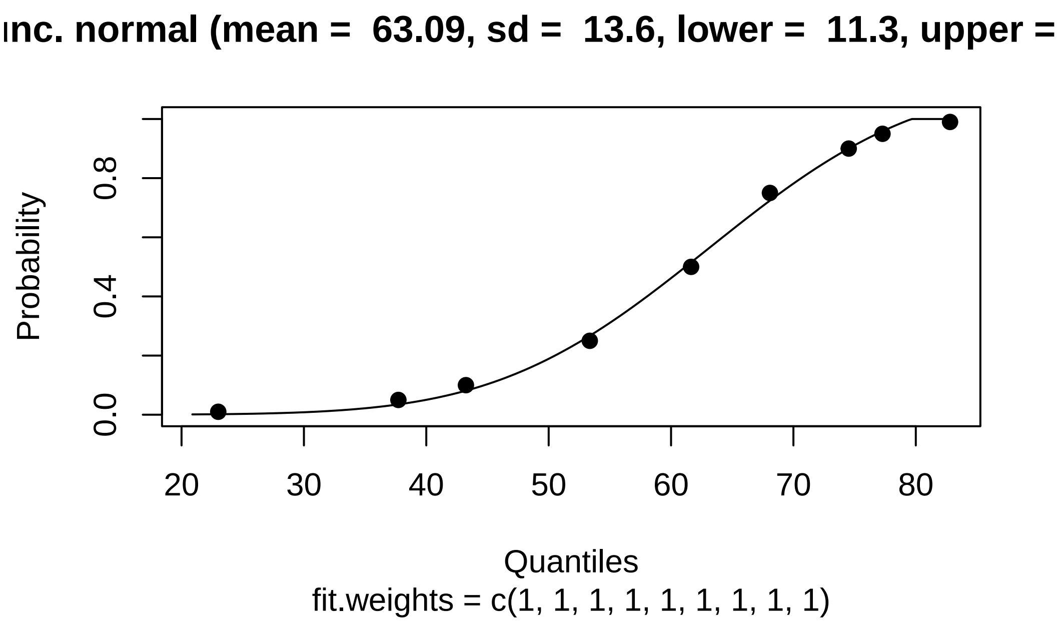
```
The fitting procedure 'L-BFGS-B' was successful!

$par
[1] 60.479951 19.653804 -4.233975 78.738009

$value
[1] 7.915663e-06

$counts
function gradient
      30       30

$convergence
[1] 0
```

```
$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```



```r
## <65 years of age, equivalent to 80Plus
Intake_fitted_80 = get.tnorm.par(p = p,
                                 q = c(25, 41, 47, 58, 67, 74, 81, 84, 90)*.92)/100
```

```
The fitting procedure 'L-BFGS-B' was successful!

$par
[1] 63.08547 13.60241 11.29892 79.65177

$value
[1] 2.486357e-05

$counts
function gradient
      27       27

$convergence
[1] 0

$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```



```
## All ages, using 20 to 64 since the table does not have this category
Intake_fitted_all = Intake_fitted_70
```

Three dataframes (`IntakeFactor`, `vol_fitted_EPA_lnorm`, and `vol_fitted_EPA_tnorm`) are created to store fitted means, standard deviations, upper and low values, logmeans, and logsd of truncated normal and lognormal distributions (per age group).

```
IntakeFactor <- tibble(
  age =c("0_to_2",
         "2_to_16",
         "16_to_70",
         "80Plus",
         "All_ages"),
  bind_rows(Intake_fitted_2,
            Intake_fitted_16,
            Intake_fitted_70,
            Intake_fitted_80,
            Intake_fitted_all)) %>%
  rename_with(.cols = c(mean:upper),.fn = \(x) paste0(x,".fac")) %>%
  mutate(lower.fac = if_else(lower.fac<0,0,lower.fac))

vol_fitted_EPA_lnorm <- tibble(
  age = c("0_to_2",
          "80Plus"),
  bind_rows(vol_fitted_lnorm_2,
            vol_fitted_lnorm_80)
  )%>%
  rename_with(.cols = c(meanlog:sdlog),.fn = \(x) paste0(x,".vol")) %>%
  left_join( IntakeFactor,
  by = "age")

vol_fitted_EPA_tnorm <- tibble(
  age = c("2_to_16",
          "16_to_70",
          "All_ages"),
  bind_rows(vol_fitted_tnorm_16,
            vol_fitted_tnorm_70,
            vol_fitted_tnorm_all)) %>%
  rename_with(.cols = c(mean:upper),.fn = \(x) paste0(x,".vol")) %>%
  left_join( IntakeFactor,
  by = "age")
```

Through the code below, the volume of water intake values are sampled from distributions defined by the parameters in `vol_fitted_EPA_lnorm`, by age group

```
#Volume of consumed water in litres
sim_volumes_EPA_lnorm = list()
for(i in seq_along(vol_fitted_EPA_lnorm$age)){
 age_vol_EPA = tibble(vol_type = paste0("EPA_",vol_fitted_EPA_lnorm$age[i]),
                      vol_L =  rlnorm(
                        n = nrow(sim_conc_pathogens),
                        mean = vol_fitted_EPA_lnorm$meanlog.vol[i],
                        sd = vol_fitted_EPA_lnorm$sdlog.vol[i]
                      ) / 1000,
                      in.factor =  rnormTrunc(
                        n = nrow(sim_conc_pathogens),
                        mean = vol_fitted_EPA_lnorm$mean.fac[i],
                        sd = vol_fitted_EPA_lnorm$sd.fac[i],
                        min = vol_fitted_EPA_lnorm$lower.fac[i],
                        max = vol_fitted_EPA_lnorm$upper.fac[i]
                      )) %>%  bind_cols(sim_conc_pathogens)
 sim_volumes_EPA_lnorm[[i]] <- age_vol_EPA
}
sim_volumes_EPA_lnorm <- do.call(rbind,sim_volumes_EPA_lnorm)
```

Through the code below, the volume of water intake values are sampled from distributions defined by the parameters in `vol_fitted_EPA_tnorm` by age group

```
#Volume of consumed water in litres
sim_volumes_EPA_tnorm = list()
for(i in seq_along(vol_fitted_EPA_tnorm$age)){
 age_vol_EPA = tibble(vol_type = paste0("EPA_",vol_fitted_EPA_tnorm$age[i]),
                     vol_L =  rnormTrunc(
                       n = nrow(sim_conc_pathogens),
                       mean = vol_fitted_EPA_tnorm$mean.vol[i],
                       sd = vol_fitted_EPA_tnorm$sd.vol[i],
                       min = vol_fitted_EPA_tnorm$lower.vol[i],
                       max = vol_fitted_EPA_tnorm$upper.vol[i]
                     ) / 1000,
                     in.factor =  rnormTrunc(
                       n = nrow(sim_conc_pathogens),
                       mean = vol_fitted_EPA_tnorm$mean.fac[i],
                       sd = vol_fitted_EPA_tnorm$sd.fac[i],
                       min = vol_fitted_EPA_tnorm$lower.fac[i],
                       max = vol_fitted_EPA_tnorm$upper.fac[i]
                     )) %>%
   bind_cols(sim_conc_pathogens)
 sim_volumes_EPA_tnorm[[i]] <- age_vol_EPA
}
sim_volumes_EPA_tnorm <- do.call(rbind,sim_volumes_EPA_tnorm)
```

Aggregated estimation of ingestion of water, WHO (2017)

In the code below a uniform probability distribution function is assumed to sample the ingestion volume of water, following common use of uniform PDF and assumption of 1-2L ppd range (WHO 2017). Included the same `IntakeFactor` data as in the above estimations.

```
#Volume of consumed water for adults, range given by WHO(2017)

sim_volumes_WHO = tibble(vol_type = "WHO_all",
                         vol_L =
                           (runif(
                             nrow(sim_conc_pathogens), min = 1, max = 2)),
                         in.factor =  rnormTrunc(
                             n = nrow(sim_conc_pathogens),
                             mean = vol_fitted_EPA_tnorm$mean.fac[i],
                             sd = vol_fitted_EPA_tnorm$sd.fac[i],
                             min = vol_fitted_EPA_tnorm$lower.fac[i],
                             max = vol_fitted_EPA_tnorm$upper.fac[i]
                           ))  %>%
    bind_cols(sim_conc_pathogens)
```

Joining the volume of ingested water simulations into one dataframe

```
df_simulation_0 = rbind(sim_volumes_EPA_lnorm,sim_volumes_EPA_tnorm,
                        sim_volumes_WHO)

#A summary table for "reality check"
df_simulation_0 %>%
  summarise(count = n(),
            min.vol= min(vol_L),
            mean.vol = mean(vol_L),
            median.vol = mean(vol_L),
            p95.vol = quantile(vol_L,0.95),
            p5.vol = quantile(vol_L,0.05),
            max.vol= max(vol_L),
            .by =vol_type)
```

```
# A tibble: 6 x 8
  vol_type      count min.vol mean.vol median.vol p95.vol p5.vol max.vol
  <chr>         <int>   <dbl>    <dbl>      <dbl>   <dbl>  <dbl>   <dbl>
1 EPA_0_to_2    60000 0.00319    0.444      0.444    1.40 0.0507   18.7
2 EPA_80Plus    60000 0.0583     1.14       1.14     2.69 0.319    16.1
3 EPA_2_to_16   60000 0.0167     0.437      0.437    1.22 0.0400    3.61
4 EPA_16_to_70  60000 0.0242     1.26       1.26     3.18 0.112     8.05
5 EPA_All_ages  60000 0.00484    1.08       1.08     2.94 0.0718    8.95
6 WHO_all       60000 1.00       1.50       1.50     1.95 1.05      2.00
```

Dose calculations

Volume of ingested water and pathogen concentrations are used to calculate the dose ingested. The value for volume ingested, `vol_L` , is multiplied by the intake factor, `in.factor`.

```
df_dose <- df_simulation_0 %>%
  mutate(dose = conc_p*(vol_L*in.factor))

df_dose %>%
  summarise(count = n(),
            min.dose= min(dose),
            mean.dose = mean(dose),
            median.dose = mean(dose),
            p95.dose = quantile(dose,0.95),
            p5.dose = quantile(dose,0.05),
            max.dose= max(dose),
            .by =pathogen)
```

```
# A tibble: 3 x 8
```

```
  pathogen   count min.dose   mean.dose median.dose p95.dose     p5.dose max.dose
  <chr>      <int>    <dbl>       <dbl>       <dbl>    <dbl>       <dbl>    <dbl>
1 ETEC      120000 2.91e-16  519801109.  519801109.   39887.     2.08e-8  1.36e13
2 campy     120000 5.46e-15 2267856584. 2267856584.  222599.     1.06e-7  4.80e13
3 rota      120000 8.89e-14 2227367821. 2227367821.  259506.     1.36e-7  5.00e13
```

Dose-response assessment

The probability of infection (daily) given a dose of pathogen (ETEC, Campy, Rota), is estimated using a Beta-Poisson model

$$risk = 1 - \left[1 + dose\frac{2^{1/a} - 1}{N50}\right]^{-a}$$

In the code below the point/range/PDF estimates for a and $N50$ are defined for the pathogens of study (ETEC, Campy and rota)

```
df_dose_resp <- df_dose %>%
  mutate(N50 = case_when(
            pathogen == "ETEC" ~ 1.7e06,#from Moncada-Barragan (2021)
            #pathogen == "campy" ~ rlnorm(nrow(df_dose), 1.69e03, 2.78e03),#from Bivins et
            pathogen == "campy" ~ 6.68e4,#from QMRA wiki
            pathogen == "rota" ~ rlnorm(nrow(df_dose), 8.16, 6.65)),#from Bivins et al (20
              #pathogen == "rota" ~ 6.17),#from QMRA wiki
         a = case_when(
            pathogen == "ETEC" ~ 0.0754,#from Moncada-Barragan (2021)
            #pathogen == "campy" ~ rlnorm(nrow(df_dose), 1.51e-01, 5.90e-02),#from Bivins
            pathogen == "campy" ~ 3.19e-01, #from QMRA wiki
            pathogen == "rota" ~ rlnorm(nrow(df_dose), 2.48e-01, 1.46e-1))) #from Bivins e
            #pathogen == "rota" ~2.5e-02)) #from QMRA wiki
```

Risk characterisation

Daily infection risk

In the code below, I use the Beta-Poisson equation to estimate probability of infection given an ingested dose. All results are stored in an appended dataframe.

```
df_risk <- df_dose_resp |>
  mutate( risk = 1-(1+(dose/N50)*((2^(1/a)-1)))^-a)

df_risk %>%
  summarise(count = n(),
            min.risk= min(risk),
            mean.risk = mean(risk),
            median.risk = mean(risk),
            p95.risk = quantile(risk,0.95),
            p5.risk = quantile(risk,0.05),
            max.risk= max(risk),
            .by =c(pathogen, w_source))
```

```
# A tibble: 6 x 9
  pathogen w_source count min.risk mean.risk median.risk  p95.risk  p5.risk
  <chr>    <chr>    <int>    <dbl>     <dbl>       <dbl>     <dbl>    <dbl>
1 ETEC     tap      60000 0         0.000221    0.000221 0.0000403 1.52e-12
2 campy    tap      60000 0         0.000211    0.000211 0.0000208 6.56e-13
3 rota     tap      60000 0         0.0239      0.0239   0.0388    3.00e-14
4 ETEC     stored   60000 5.55e-16  0.0835      0.0835   0.460     5.38e- 9
5 campy    stored   60000 2.22e-16  0.135       0.135    0.857     2.46e- 9
6 rota     stored   60000 0         0.300       0.300    1.00      3.56e-10
# i 1 more variable: max.risk <dbl>
```

Yearly infection risk

In the code below, I calculate yearly infection risk using the the previously defined daily infection risk. All results are stored in an appended dataframe.

```
df_yearly_riks <-
  df_risk |>
  mutate(yearly_risk = 1-(1-risk)^365)

df_yearly_riks %>%
  summarise(count = n(),
            min.riskyear= min(yearly_risk),
            mean.riskyear = mean(yearly_risk),
            median.riskyear = mean(yearly_risk),
            p95.riskyear = quantile(yearly_risk,0.95),
            p5.riskyear = quantile(yearly_risk,0.05),
            max.riskyear= max(yearly_risk),
            .by =pathogen)
```

```
# A tibble: 3 x 8
  pathogen  count min.riskyear mean.riskyear median.riskyear p95.riskyear
  <chr>     <int>        <dbl>         <dbl>           <dbl>        <dbl>
1 ETEC     120000            0         0.247           0.247            1
2 campy    120000            0         0.226           0.226            1
3 rota     120000            0         0.324           0.324            1
# i 2 more variables: p5.riskyear <dbl>, max.riskyear <dbl>
```

Daily illness risk

Finally, we calculate illness risk multiplying daily infection risk by the Colombia's ADD (acute diarrheal disease) morbidity rate (reported by the INS in 2023).

```r
df_illness_risk <- df_risk |>
  mutate(ill_risk = risk * morbidity)

df_illness_risk %>%
  summarise(count = n(),
            min.riskIll= min(ill_risk),
            mean.riskIll = mean(ill_risk),
            median.riskIll = mean(ill_risk),
            p95.riskIll = quantile(ill_risk,0.95),
            p5.riskIll = quantile(ill_risk,0.05),
            max.riskIll= max(ill_risk),
            .by =pathogen)
```

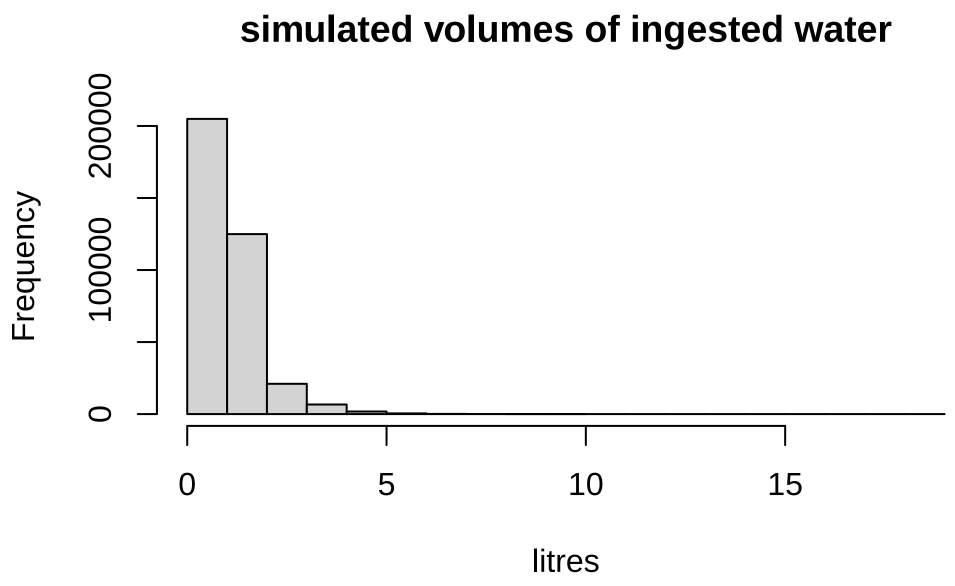
```
# A tibble: 3 x 8
  pathogen  count min.riskIll mean.riskIll median.riskIll p95.riskIll p5.riskIll
  <chr>     <int>       <dbl>        <dbl>          <dbl>       <dbl>      <dbl>
1 ETEC     120000           0      0.00837        0.00837      0.0673   1.81e-12
2 campy    120000           0      0.0135         0.0135       0.130    7.87e-13
3 rota     120000           0      0.0324         0.0324       0.200    8.75e-14
# i 1 more variable: max.riskIll <dbl>
```

Visualisation of results

Volume of ingested water

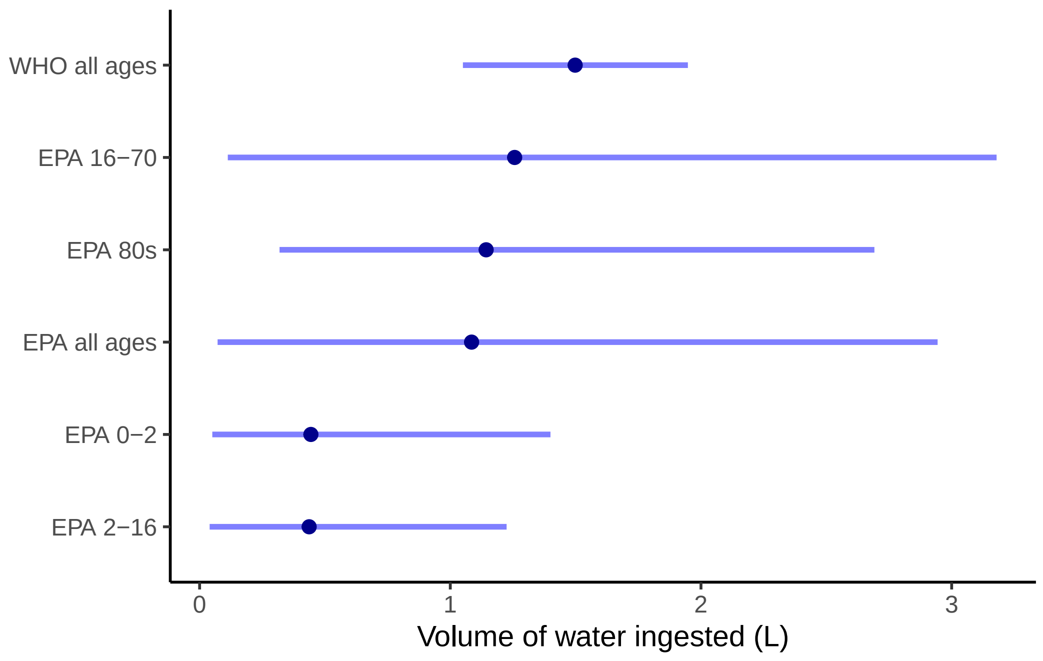
Below, an histogram of the overall simulated volumes of ingested water

```r
hist(df_risk$vol_L, main="simulated volumes of ingested water", xlab="litres")
```



Through the code below we visualize the simulation of volumes of ingested water, while using the WHO (2017) values, sampled using uniform distribution; and the EPA (2007) values, sampled using log normal distributions.

```
vol_ing <- df_risk %>%
  summarise(count = n(),
            mean.vol_L = mean(vol_L),
            median.vol_L = mean(vol_L),
            p95.vol_L = quantile(vol_L,0.95),
            p5.vol_L = quantile(vol_L,0.05),
            .by = vol_type) %>%
    mutate(vol_type = case_when(
    vol_type=="WHO_all"~"WHO all ages",
    vol_type=="EPA_16_to_70"~"EPA 16-70",
    vol_type=="EPA_80Plus"~"EPA 80s",
    vol_type=="EPA_All_ages"~ "EPA all ages",
    vol_type=="EPA_0_to_2"~"EPA 0-2",
    vol_type=="EPA_2_to_16"~"EPA 2-16"
    )) %>%
  ggplot(aes(x=fct_reorder(vol_type,median.vol_L,mean)))+
  geom_linerange(aes(ymin = p5.vol_L,ymax = p95.vol_L),
                 col="blue", lwd = 1,alpha = 0.5)+
  geom_point(aes(y=mean.vol_L), col="darkblue", size=2)+
  coord_flip()+
  theme_classic()+
  labs(x = NULL ,
       y = "Volume of water ingested (L)")

print(vol_ing)
```



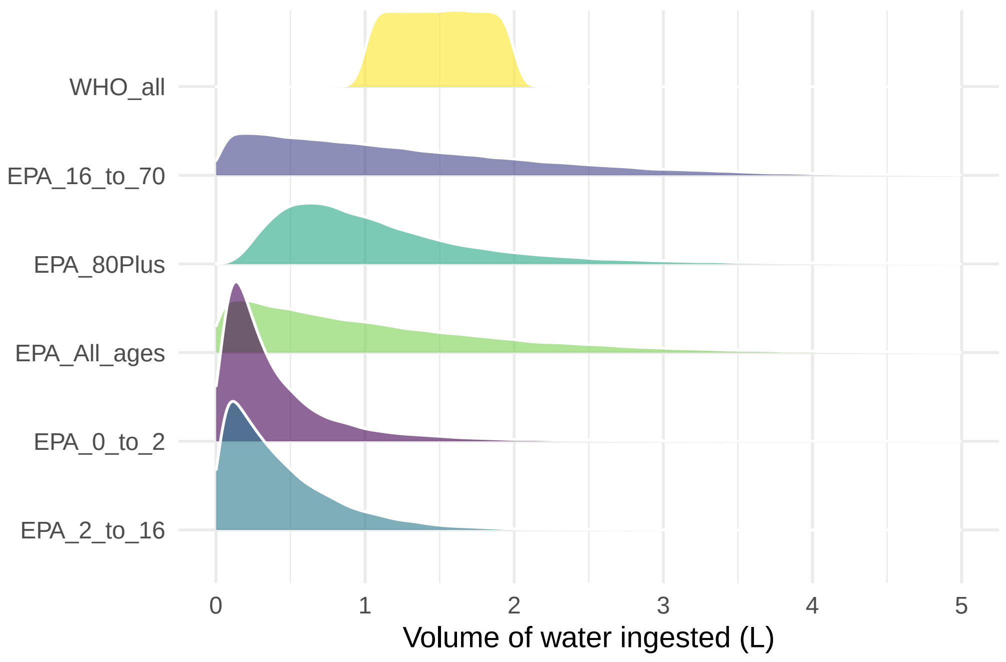
```
ggsave("vol_ing.png", vol_ing, dpi = "retina", units = "cm", width = 12, height = 10)
```

The code below shows the same results as above, the only difference is the display of the distributions.

```
df_risk %>%
  ggplot(aes(y=fct_reorder(vol_type,vol_L,mean), x=vol_L,fill=vol_type))+
  geom_density_ridges(alpha = 0.6,col = "white")+
  scale_x_continuous(limits = c(0,5))+
  scale_fill_viridis_d()+
  theme_minimal()+
  labs(y="",x="Volume of water ingested (L)")+
  theme(legend.position = "none")
```

```
Picking joint bandwidth of 0.0561
```

```
Warning: Removed 720 rows containing non-finite outside the scale range
(`stat_density_ridges()`).
```

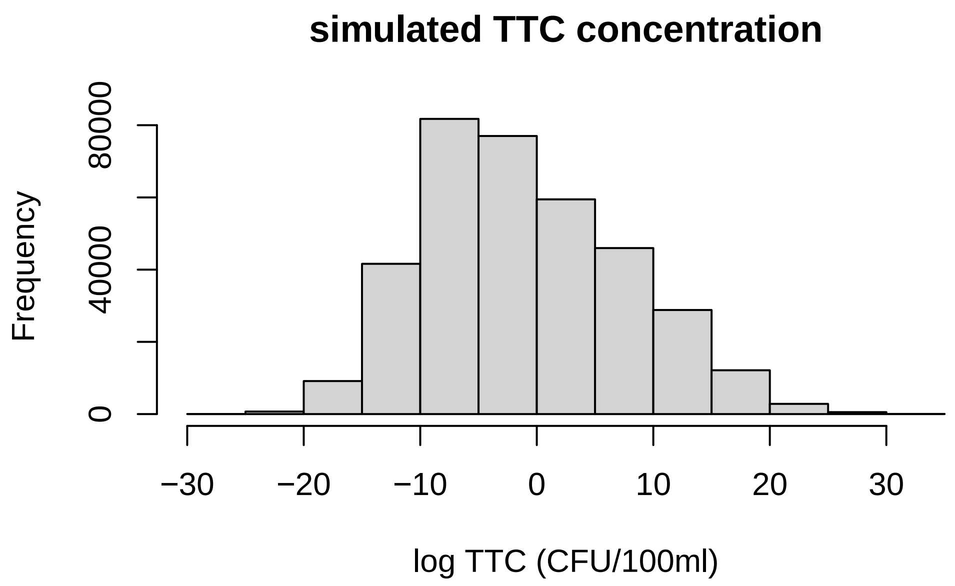


Concentration of pathogens

Thermotolerant coliforms

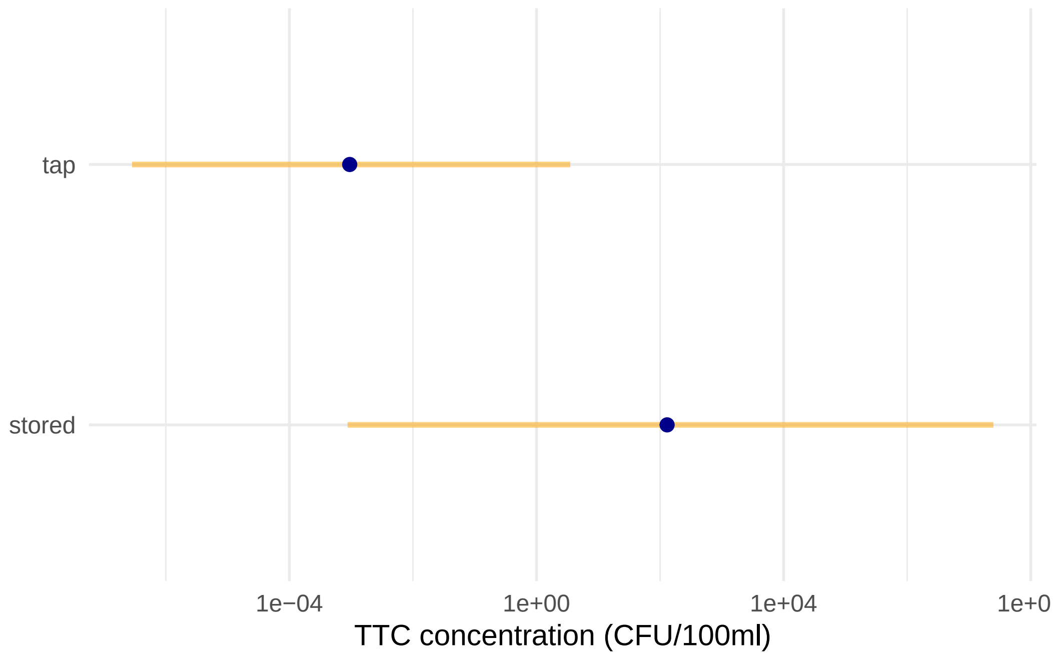
Below, an histogram of the overall simulated TTC concentrations. The values were sampled using the MLE estimates.

```
hist(log(df_risk$conc_0), main = "simulated TTC concentration", xlab = "log TTC (CFU/100ml
```



Through the code below we visualize the simulation of concentration of TTCs in stored and tap water.

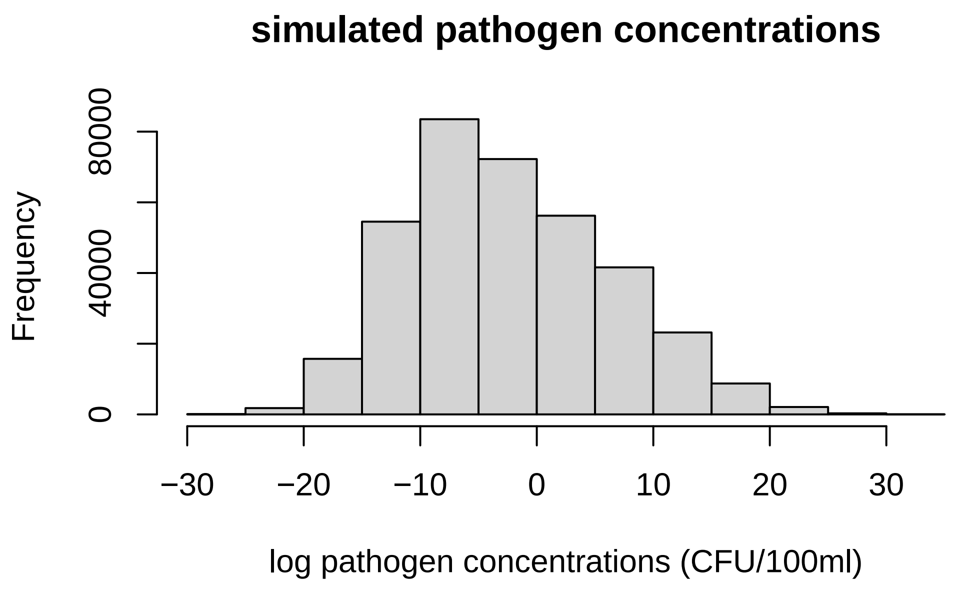
```
  df_risk |> summarise(across(conc_0,
                              list(mean = mean,
                                   median = median,
                                   p5 = \(x) quantile(x,0.05),
                                   p95 = \(x) quantile(x,0.95)
                                   )
                              ),
                       .by = w_source
                       ) |>
  ggplot(aes(x = w_source))+
  geom_linerange(aes(ymin = conc_0_p5,ymax = conc_0_p95),
                 col="orange", lwd = 1,alpha = 0.5)+
  geom_point(aes(y=conc_0_median), col="darkblue", size=2)+
  coord_flip()+
  scale_y_log10()+
  theme_minimal()+
labs(x = NULL , y = "TTC concentration (CFU/100ml)")
```



Enteropathogens

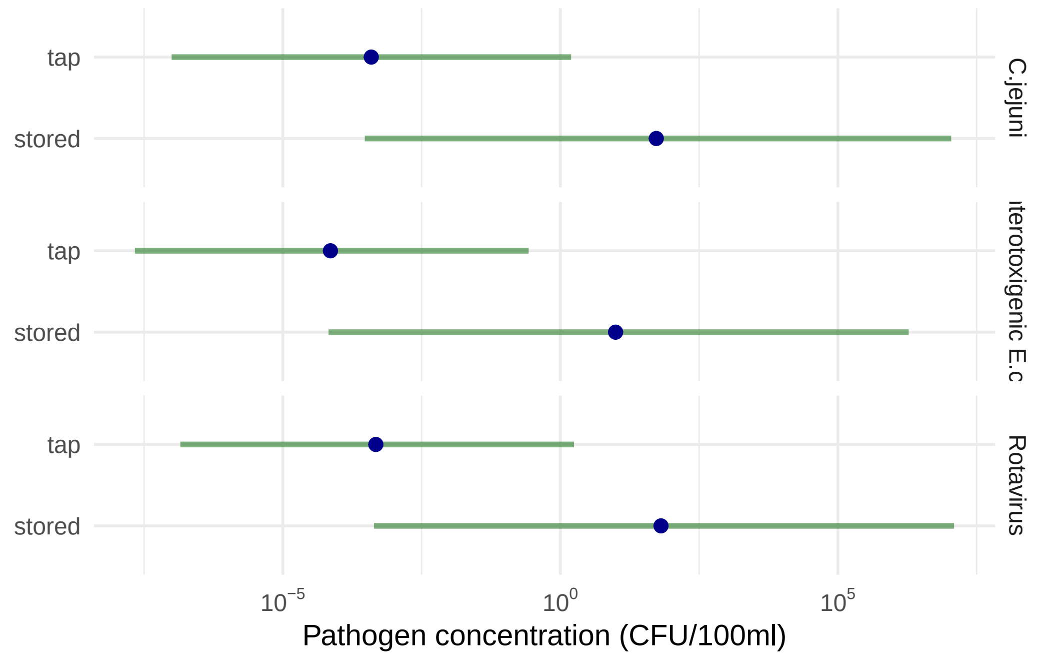
Below, an histogram of the overall simulated pathogen concentrations. The values were sampled using the MLE estimates obtained previously.

```
hist(log(df_risk$conc_p), main="simulated pathogen concentrations", xlab = "log pathogen c
```



Through the code below we visualise the simulation of concentration of three pathogens (ETEC, Campylobacter jejuni, and rotavirus), in sampled tap and stored water

```r
path_conc <-   df_risk %>%  summarise(across(
  conc_p,
  list(
    mean = mean,
    median = median,
    p5 = \(x) quantile(x, 0.05),
    p95 = \(x) quantile(x, 0.95)
  )
), .by = c(w_source, pathogen)) %>%
  mutate(
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  ) %>%
  ggplot(aes(x = w_source)) +
  geom_linerange(
    aes(ymin = conc_p_p5, ymax = conc_p_p95),
    col = "darkgreen",
    lwd = 1,
    alpha = 0.5
  ) +
  geom_point(aes(y = conc_p_median),
             col = "darkblue",
             size = 2) +
  coord_flip() +
  #scale_y_log10()+
  theme_minimal() +
  facet_grid(pathogen ~ .) +
  labs(x = NULL , y = "Pathogen concentration (CFU/100ml)") +
  scale_y_log10(
    breaks = scales::trans_breaks("log10", function(x)
      10 ^ x),
    labels = scales::trans_format("log10", scales::math_format(10 ^ .x))
  )

print(path_conc)
```



```
ggsave("path_conc.png", path_conc, dpi = "retina", units = "cm", width = 12, height = 10)
```

```
df_risk |> summarise(across(conc_p,
                            list(mean = mean,
                                 median = median,
                                 p5 = \(x) quantile(x,0.05),
                                 p95 = \(x) quantile(x,0.95)
                                 )
                            ),
                     .by = c(w_source, pathogen))
```

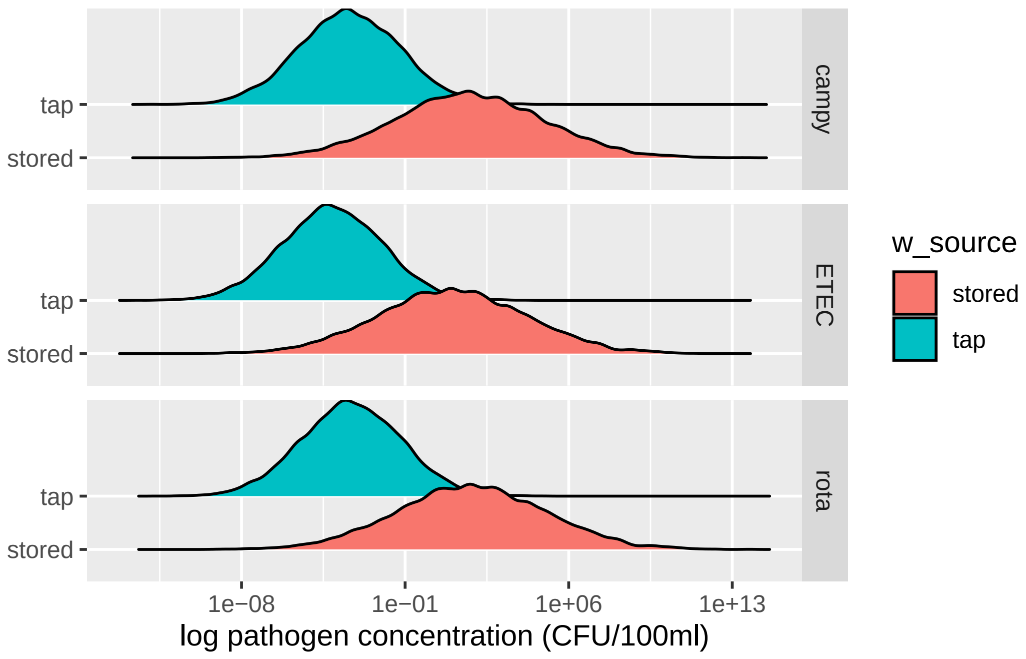
```
# A tibble: 6 x 6
  w_source pathogen conc_p_mean conc_p_median    conc_p_p5   conc_p_p95
  <chr>    <chr>          <dbl>         <dbl>        <dbl>        <dbl>
1 tap      ETEC         4.13e 0     0.0000720 0.0000000216        0.268
2 tap      campy        2.95e 1     0.000392  0.0000000991        1.56
3 tap      rota         2.71e 1     0.000474  0.000000142         1.76
4 stored   ETEC         1.79e 9     9.89      0.0000666     1886444.
5 stored   campy        7.23e 9    53.5       0.000300     11012491.
6 stored   rota         1.18e10    65.0       0.000438     12410827.
```

The code below shows the same results as above, the only difference is the display of the distributions.

```r
df_risk |>
  ggplot(aes(x = conc_p,
             y = w_source,
             fill = w_source
             ))+
  geom_density_ridges()+
  scale_x_log10()+
  facet_grid(pathogen~.)+
  labs(y = NULL , x = "log pathogen concentration (CFU/100ml)")
```

```
Picking joint bandwidth of 0.267
```

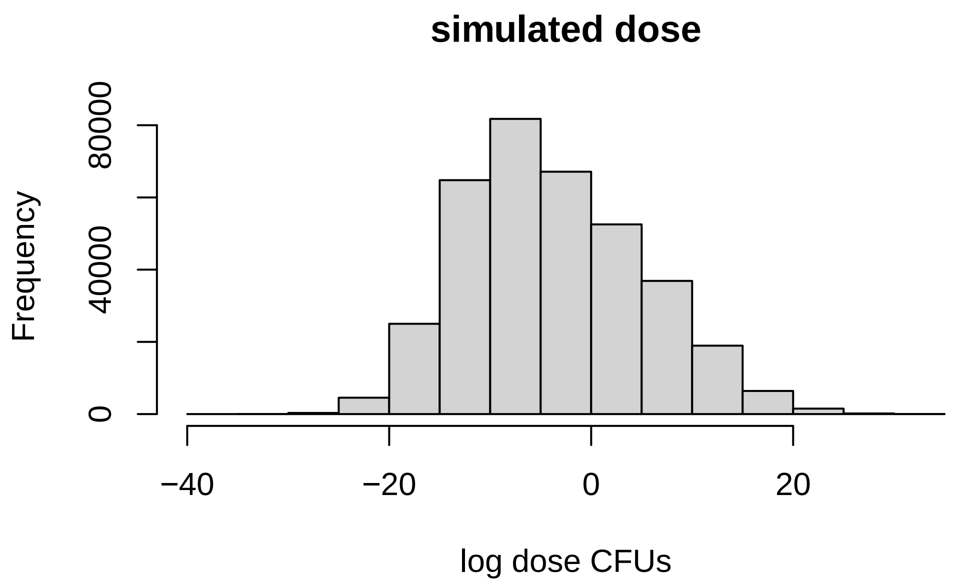
```
Picking joint bandwidth of 0.266
Picking joint bandwidth of 0.266
```



Dose of pathogens

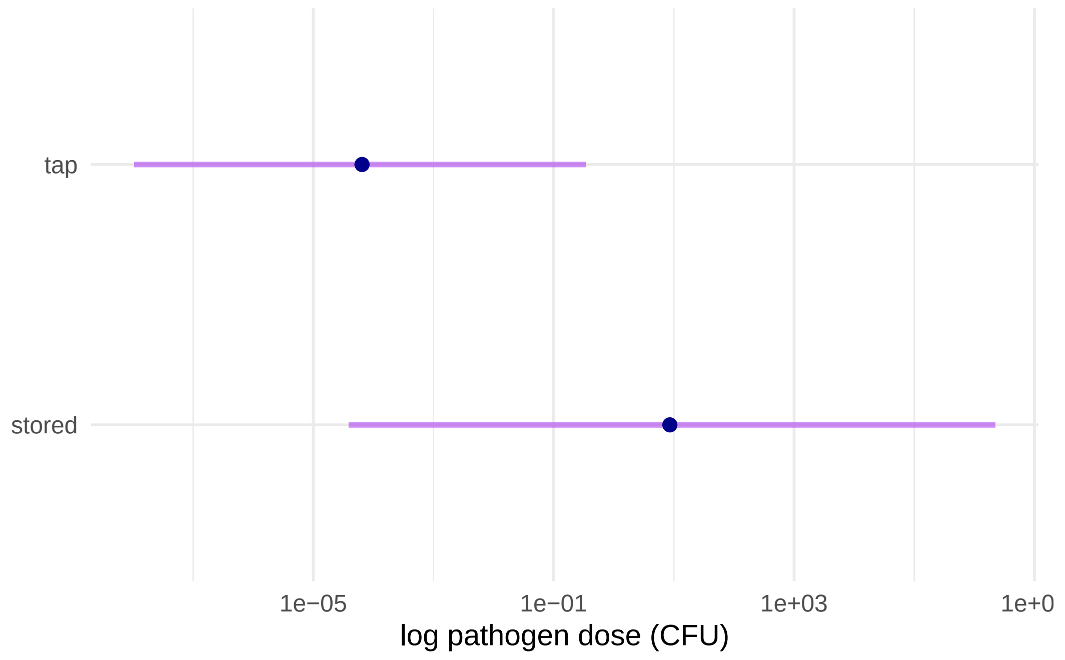
Below, an histogram of the overall simulated dose. The values were obtained from multiplying the simulations of concentration of pathogens and volume of water consumed (both coming from the previously described MLE)

```r
hist(log(df_risk$dose), main = "simulated dose", xlab = "log dose CFUs")
```



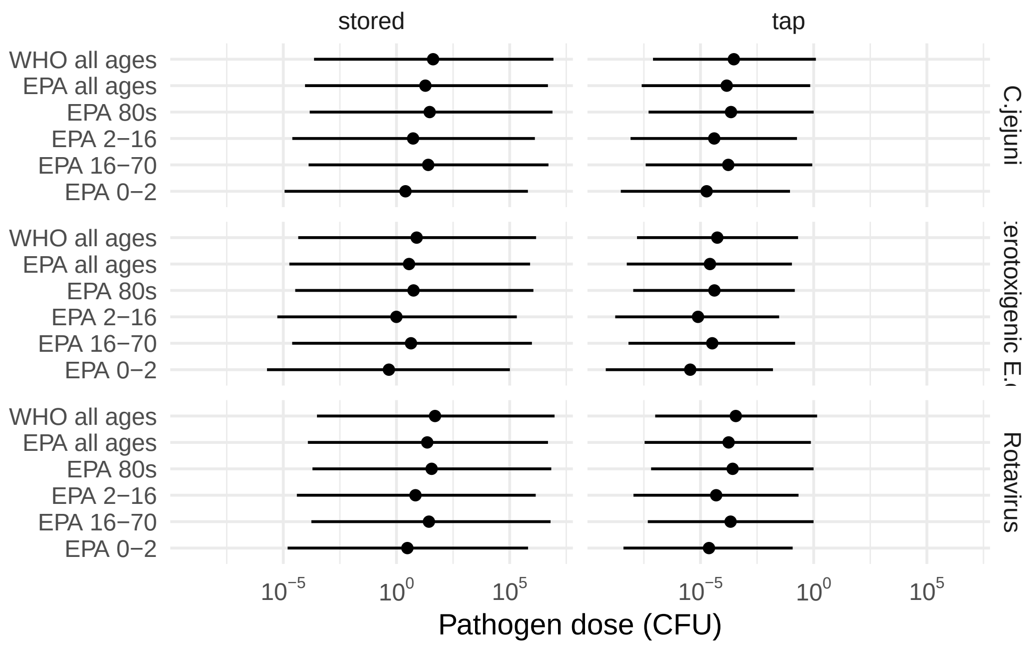
Through the code below we visualize the dose simulation in stored and tap water.

```
  df_risk |> summarise(across(dose,
                              list(mean = mean,
                                   median = median,
                                   p5 = \(x) quantile(x,0.05),
                                   p95 = \(x) quantile(x,0.95)
                                   )
                              ),
                       .by = w_source
                       ) |>
  ggplot(aes(x = w_source))+
  geom_linerange(aes(ymin = dose_p5,ymax = dose_p95),
                 col="purple", lwd = 1,alpha = 0.5)+
  geom_point(aes(y=dose_median), col="darkblue", size=2)+
  coord_flip()+
  scale_y_log10()+
  theme_minimal()+
labs(x = NULL , y = "log pathogen dose (CFU)")
```



Through the code below we visualize the results for dose calculation per type of water sample and pathogens

```r
path_dose <-   df_risk |> summarise(across(
  dose,
  list(
    mean = mean,
    median = median,
    p5 = \(x) quantile(x, 0.05),
    p95 = \(x) quantile(x, 0.95)
  )
), .by = c(vol_type, w_source, pathogen)) |>
  mutate(
    vol_type = case_when(
      vol_type == "WHO_all" ~ "WHO all ages",
      vol_type == "EPA_16_to_70" ~ "EPA 16-70",
      vol_type == "EPA_80Plus" ~ "EPA 80s",
      vol_type == "EPA_All_ages" ~ "EPA all ages",
      vol_type == "EPA_0_to_2" ~ "EPA 0-2",
      vol_type == "EPA_2_to_16" ~ "EPA 2-16"
    ),
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  ) %>%
  ggplot(aes(x = vol_type)) +
  geom_linerange(aes(ymin = dose_p5, ymax = dose_p95)) +
  geom_point(aes(y = dose_median)) +
  facet_grid(pathogen ~ w_source) +
  coord_flip() +
  #scale_y_log10()+
  theme_minimal() +
  labs(x = NULL , y = "Pathogen dose (CFU)") +
  scale_y_log10(
    breaks = scales::trans_breaks("log10", function(x)
      10 ^ x),
    labels = scales::trans_format("log10", scales::math_format(10 ^ .x))
  )


print(path_dose)
```



```
ggsave("path_dose.png", path_dose, dpi = "retina", units = "cm", width = 15, height = 15)
```

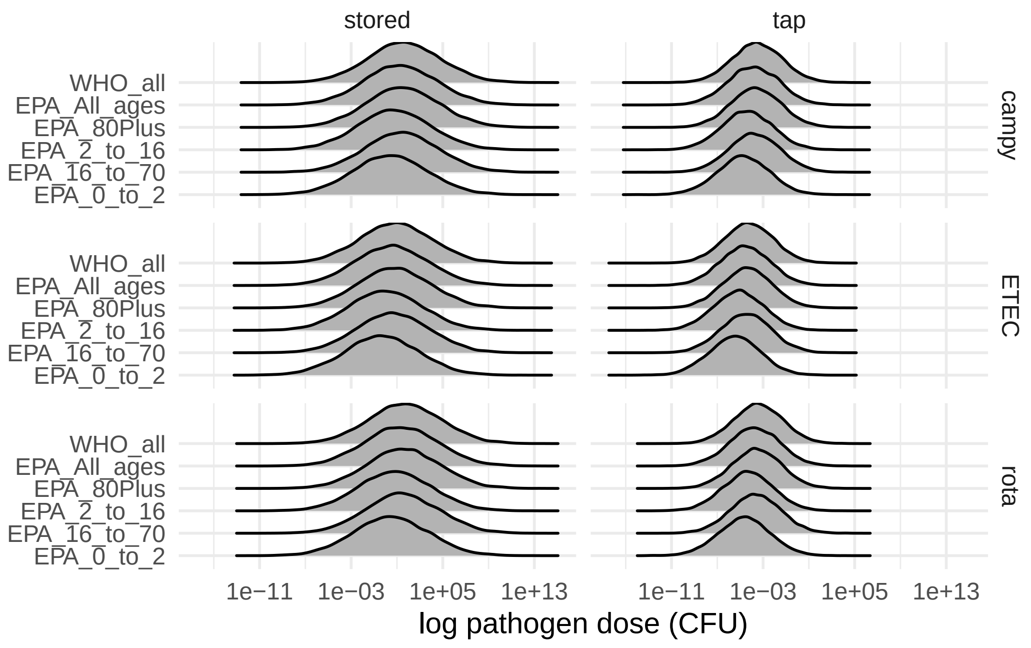
```
df_risk %>%  summarise(across(dose,
                             list(mean = mean,
                                  median = median,
                                  p5 = \(x) quantile(x,0.05),
                                  p95 = \(x) quantile(x,0.95)
                                  )
                             ),
                      .by = c(w_source,pathogen))
```

```
# A tibble: 6 x 6
  w_source pathogen      dose_mean dose_median        dose_p5      dose_p95
  <chr>    <chr>             <dbl>       <dbl>          <dbl>         <dbl>
1 tap      ETEC               2.14   0.0000194 0.00000000348         0.0926
2 tap      campy             14.8    0.000106  0.0000000176          0.559
3 tap      rota              18.7    0.000130  0.0000000232          0.590
4 stored   ETEC      1039602216.     2.63      0.0000123        610682.
5 stored   campy     4535713153.    14.3       0.0000662       3812945.
6 stored   rota      4454735624.    16.9       0.0000859       4072509.
```

The code below shows the same results as above, the only difference is the display of the distributions.

```
  df_risk %>%
  ggplot(aes(x = dose, y=vol_type))+
  geom_density_ridges()+
  facet_grid(pathogen~w_source)+
  scale_x_log10()+
  theme_minimal()+
  labs(y = NULL , x = "log pathogen dose (CFU)")
```

```
Picking joint bandwidth of 0.459

Picking joint bandwidth of 0.318

Picking joint bandwidth of 0.458

Picking joint bandwidth of 0.314

Picking joint bandwidth of 0.458

Picking joint bandwidth of 0.314
```

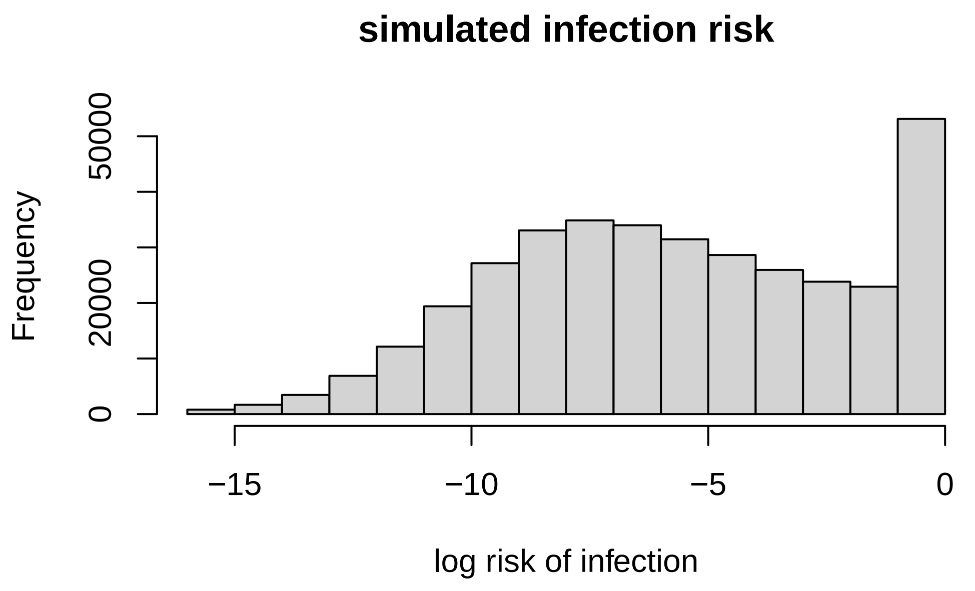


Risks

Infection

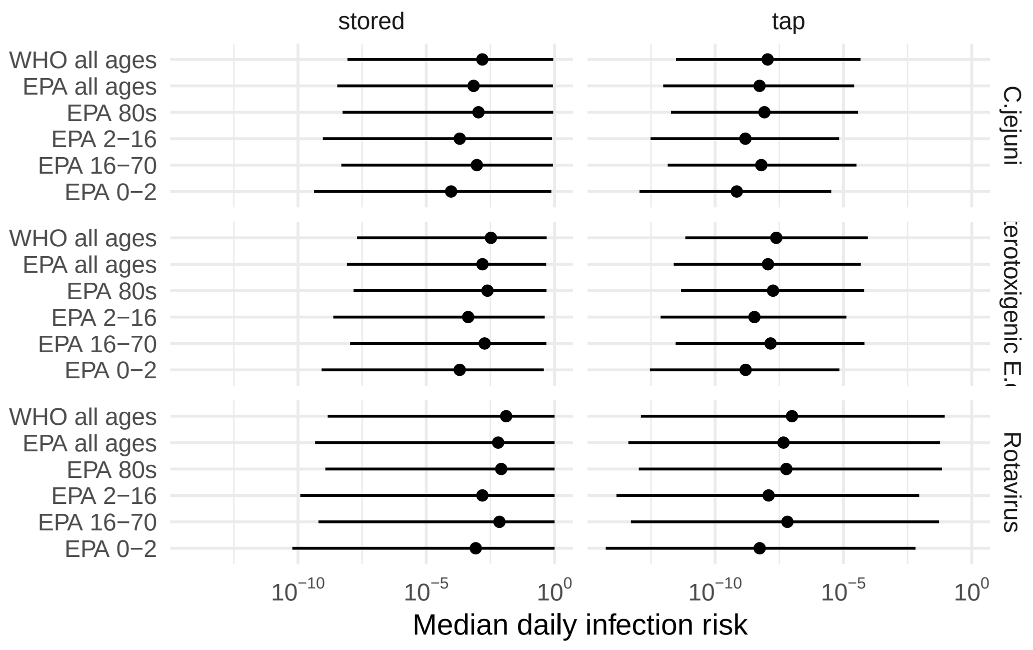
First, an overview of the overall risk simulations by plotting an histogram

```
hist(log10(df_risk$risk), main = "simulated infection risk", xlab = "log risk of infection
```



Through the code below we visualize the results of median daily risk of infection for the 3 pathogens of study.

```r
risk_daily <- df_risk |> summarise(across(risk,
                            list(mean = mean,
                                 median = median,
                                 p5 = \(x) quantile(x,0.05),
                                 p95 = \(x) quantile(x,0.95)
                                 )
                            ),
                     .by = c(vol_type, w_source, pathogen)
                     ) |>
  mutate(
    vol_type = case_when(
      vol_type == "WHO_all" ~ "WHO all ages",
      vol_type == "EPA_16_to_70" ~ "EPA 16-70",
      vol_type == "EPA_80Plus" ~ "EPA 80s",
      vol_type == "EPA_All_ages" ~ "EPA all ages",
      vol_type == "EPA_0_to_2" ~ "EPA 0-2",
      vol_type == "EPA_2_to_16" ~ "EPA 2-16"
    ),
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  ) %>%
  ggplot(aes(x = vol_type))+
  geom_linerange(aes(ymin = risk_p5,ymax = risk_p95))+
  geom_point(aes(y=risk_median))+
  facet_grid(pathogen~w_source)+
  coord_flip()+
  #scale_y_log10(breaks = scales::trans_breaks
              #("log10", function(x) 10^x),
              #labels = scales::trans_format
              #("log10", scales::math_format(10^.x)))+
  theme_minimal()+
  labs(x = NULL , y = "Median daily infection risk")+
  scale_y_log10(
    breaks = scales::trans_breaks("log10", function(x)
      10 ^ x),
    labels = scales::trans_format("log10", scales::math_format(10 ^ .x))
  )

print(risk_daily)
```



```
ggsave("risk_daily.png", risk_daily, dpi = "retina", units = "cm", width = 15, height = 15
```

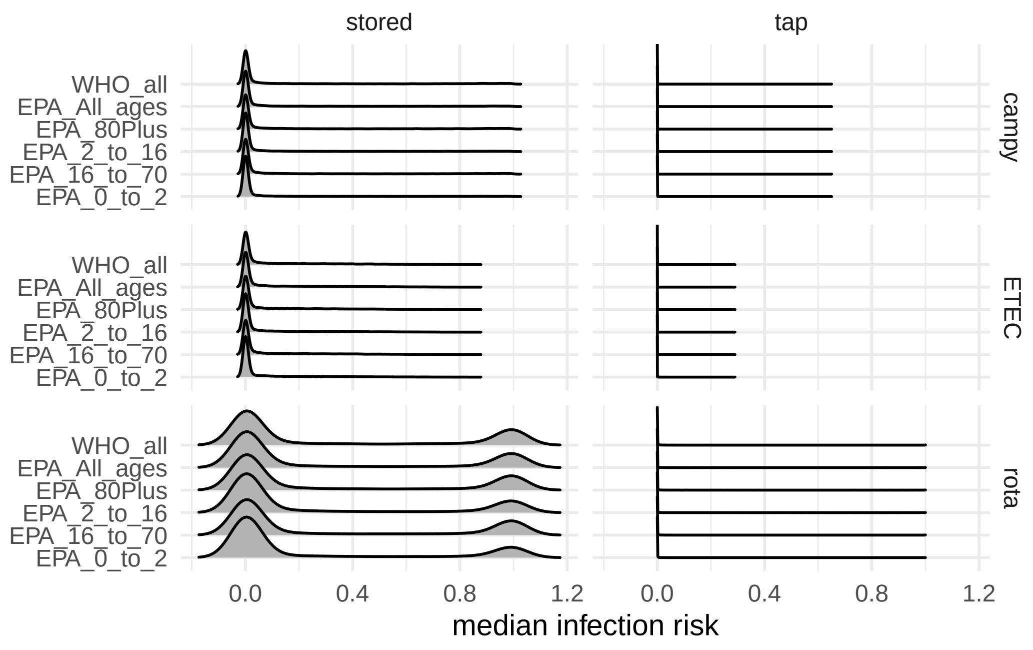
```
df_risk %>%  summarise(across(risk,
                              list(mean = mean,
                                   median = median,
                                   p5 = \(x) quantile(x,0.05),
                                   p95 = \(x) quantile(x,0.95)
                                   )
                              ),
                       .by = c(w_source, pathogen))
```

```
# A tibble: 6 x 6
  w_source pathogen risk_mean   risk_median  risk_p5  risk_p95
  <chr>    <chr>        <dbl>         <dbl>    <dbl>     <dbl>
1 tap      ETEC      0.000221 0.00000000846 1.52e-12 0.0000403
2 tap      campy     0.000211 0.00000000392 6.56e-13 0.0000208
3 tap      rota      0.0239   0.0000000330  3.00e-14 0.0388
4 stored   ETEC      0.0835   0.00114       5.38e- 9 0.460
5 stored   campy     0.135    0.000532      2.46e- 9 0.857
6 stored   rota      0.300    0.00446       3.56e-10 1.00
```

The code below shows the same results as above, the only difference is the display of the distributions.

```
  df_risk %>%
  ggplot(aes(x = risk, y=vol_type))+
  geom_density_ridges()+
  facet_grid(pathogen~w_source)+
  #scale_x_log10()+
  theme_minimal()+
  labs(y = NULL , x = "median infection risk")
```

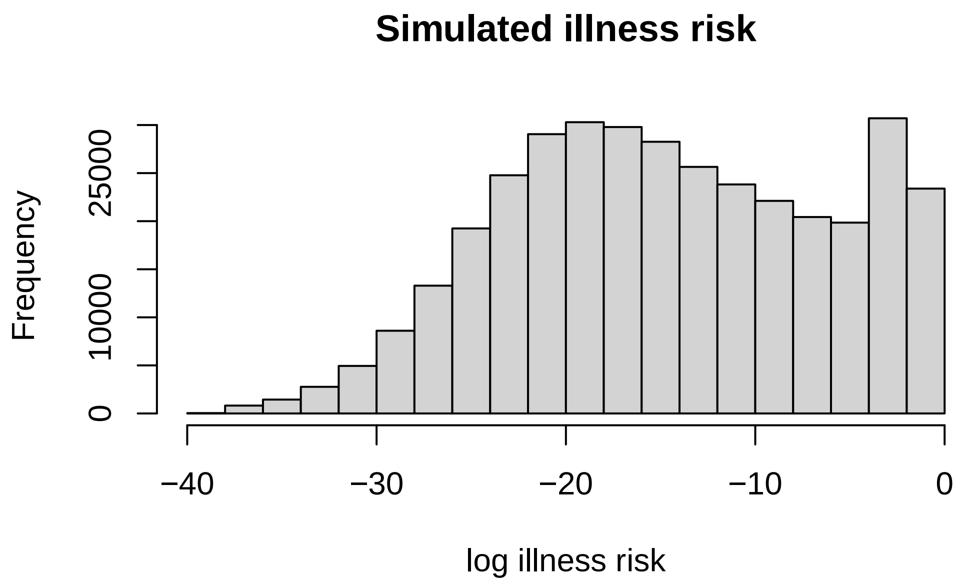
```
Picking joint bandwidth of 0.00923

Picking joint bandwidth of 1.97e-08

Picking joint bandwidth of 0.00973

Picking joint bandwidth of 3.92e-08

Picking joint bandwidth of 0.0578

Picking joint bandwidth of 1.46e-06
```



Illness

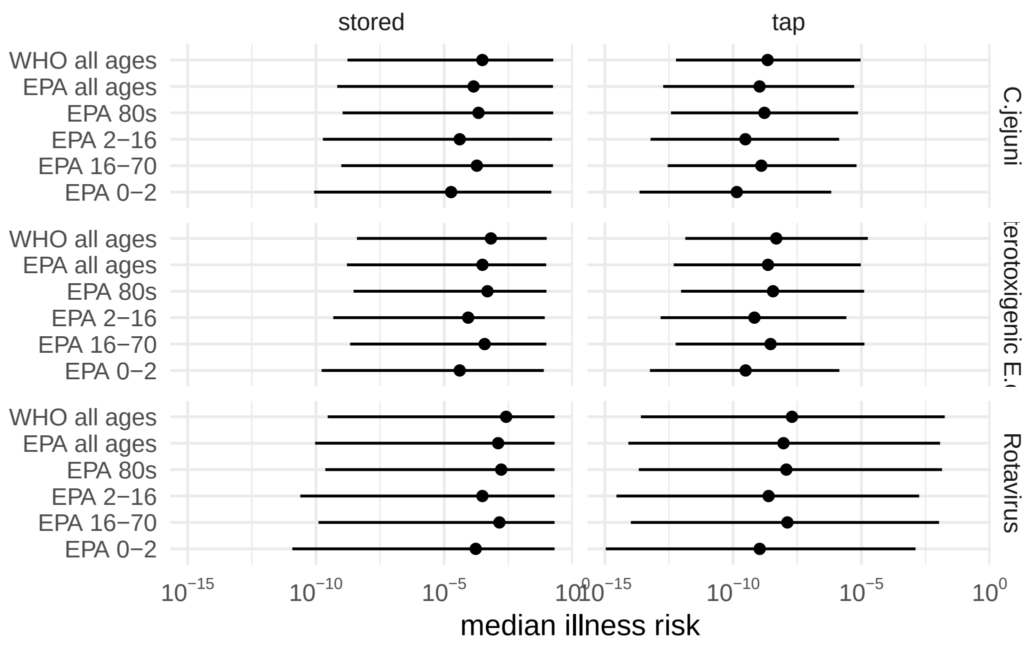
First, an overview of the overall illness risk simulations by plotting an histogram

```
hist(log(df_illness_risk$ill_risk), main = "Simulated illness risk", xlab = "log illness r
```



Through the code below we visualize the results of estimating the daily risk of illness using the Colombian morbidity ratio and the beta-poisson model of the 3 pathogens of study.

```r
risk_illn <- df_illness_risk |> summarise(across(
  ill_risk,
  list(
    mean = mean,
    median = median,
    p5 = \(x) quantile(x, 0.05),
    p95 = \(x) quantile(x, 0.95)
  )
), .by = c(vol_type, w_source, pathogen)) |>
  mutate(
    vol_type = case_when(
      vol_type == "WHO_all" ~ "WHO all ages",
      vol_type == "EPA_16_to_70" ~ "EPA 16-70",
      vol_type == "EPA_80Plus" ~ "EPA 80s",
      vol_type == "EPA_All_ages" ~ "EPA all ages",
      vol_type == "EPA_0_to_2" ~ "EPA 0-2",
      vol_type == "EPA_2_to_16" ~ "EPA 2-16"
    ),
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  ) %>%
  ggplot(aes(x = vol_type)) +
  geom_linerange(aes(ymin = ill_risk_p5, ymax = ill_risk_p95)) +
  geom_point(aes(y = ill_risk_median)) +
  facet_grid(pathogen ~ w_source) +
  coord_flip() +
  #scale_y_log10()+
  theme_minimal() +
  labs(x = NULL , y = "median illness risk") +
  scale_y_log10(
    breaks = scales::trans_breaks("log10", function(x)
      10 ^ x),
    labels = scales::trans_format("log10", scales::math_format(10 ^ .x))
  )

print(risk_illn)
```



```r
ggsave("risk_illn.png", risk_illn, dpi = "retina", units = "cm", width = 15, height = 15)
```

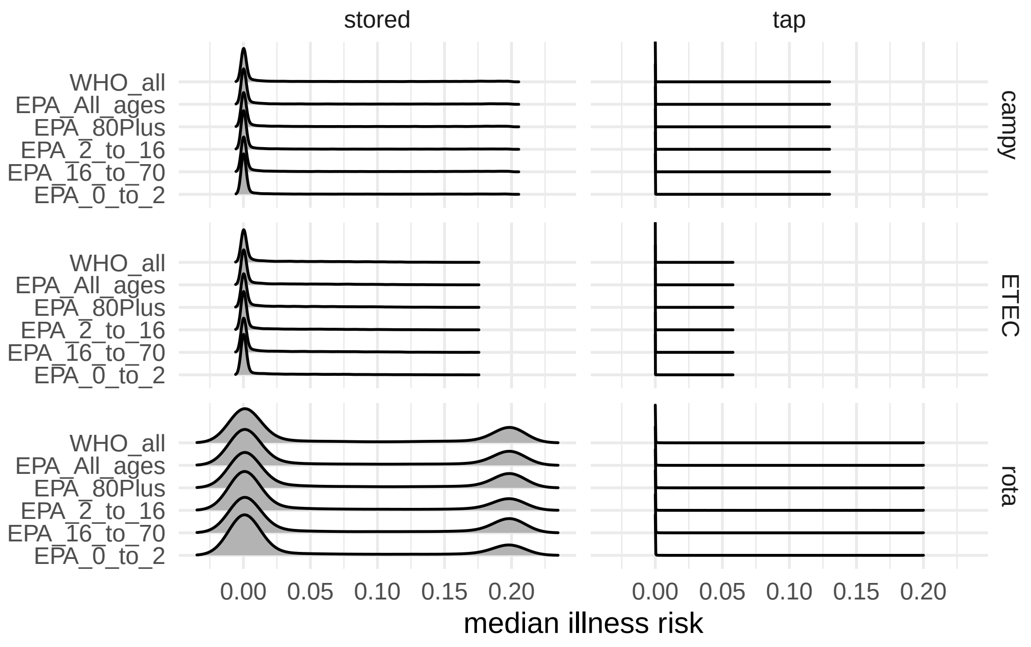
```r
df_illness_risk %>%  summarise(across(ill_risk,
                                      list(mean = mean,
                                           median = median,
                                           p5 = \(x) quantile(x,0.05),
                                           p95 = \(x) quantile(x,0.95)
                                           )
                                      ),
                               .by = c(w_source, pathogen))
```

```
# A tibble: 6 x 6
  w_source pathogen ill_risk_mean ill_risk_median ill_risk_p5 ill_risk_p95
  <chr>    <chr>            <dbl>           <dbl>       <dbl>        <dbl>
1 tap      ETEC         0.0000443        1.69e- 9    3.04e-13   0.00000807
2 tap      campy        0.0000423        7.85e-10    1.31e-13   0.00000416
3 tap      rota         0.00478          6.60e- 9    6.00e-15   0.00776
4 stored   ETEC         0.0167           2.27e- 4    1.08e- 9   0.0920
5 stored   campy        0.0270           1.06e- 4    4.92e-10   0.171
6 stored   rota         0.0600           8.91e- 4    7.12e-11   0.200
```

The code below shows the same results as above, the only difference is the display of the distributions.

```r
df_illness_risk |>
ggplot(aes(x = ill_risk, y=vol_type))+
geom_density_ridges()+
facet_grid(pathogen~w_source)+
#scale_x_log10()+
theme_minimal()+
labs(y = NULL , x = "median illness risk")
```

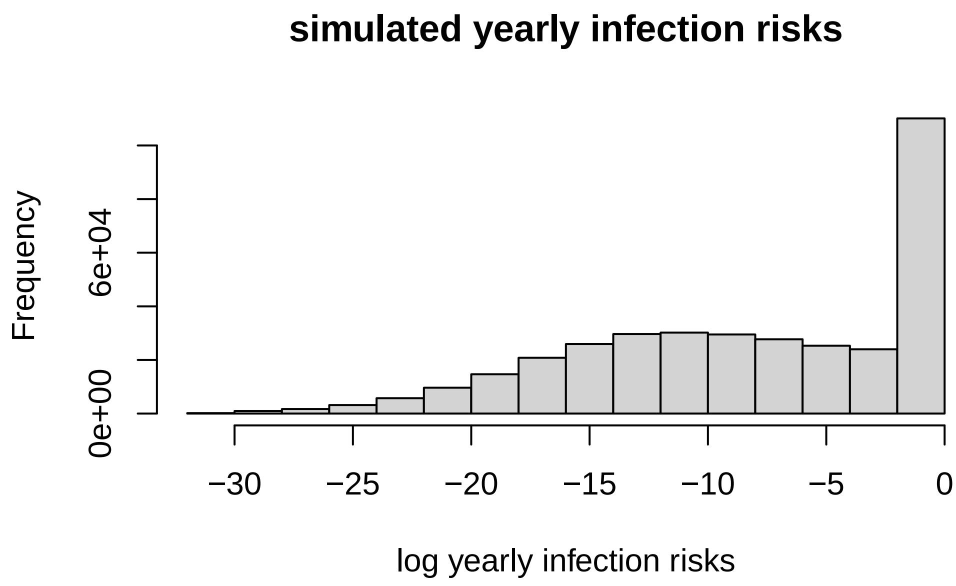
```
Picking joint bandwidth of 0.00185

Picking joint bandwidth of 3.94e-09

Picking joint bandwidth of 0.00195

Picking joint bandwidth of 7.83e-09

Picking joint bandwidth of 0.0116

Picking joint bandwidth of 2.92e-07
```



Yearly infection risk

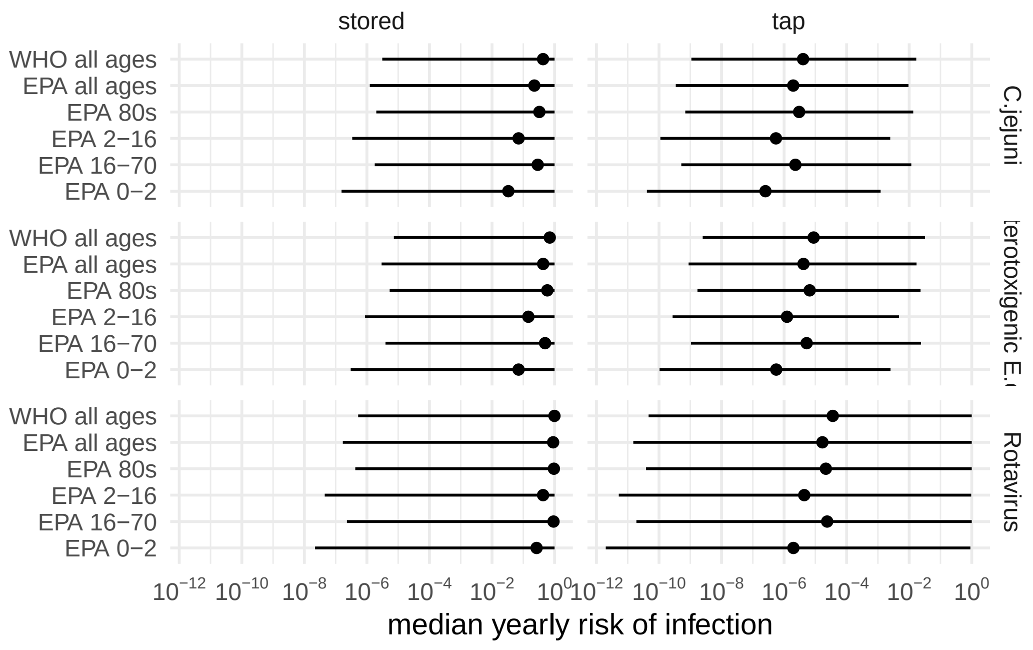
First, an overview of the overall illness risk simulations by plotting an histogram

```
hist(log(df_yearly_riks$yearly_risk), main="simulated yearly infection risks", xlab = "log
```



Through the code below we visualize the results of estimating the yearly risk of infection using the beta-poisson model of the 3 pathogens of study.

```
risk_year <- df_yearly_riks |> summarise(across(
  yearly_risk,
  list(
    mean = mean,
    median = median,
    p5 = \(x) quantile(x, 0.05),
    p95 = \(x) quantile(x, 0.95)
  )
), .by = c(vol_type, w_source, pathogen)) |>
  mutate(
    vol_type = case_when(
      vol_type == "WHO_all" ~ "WHO all ages",
      vol_type == "EPA_16_to_70" ~ "EPA 16-70",
      vol_type == "EPA_80Plus" ~ "EPA 80s",
      vol_type == "EPA_All_ages" ~ "EPA all ages",
      vol_type == "EPA_0_to_2" ~ "EPA 0-2",
      vol_type == "EPA_2_to_16" ~ "EPA 2-16"
    ),
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  ) %>%
  ggplot(aes(x = vol_type)) +
  geom_linerange(aes(ymin = yearly_risk_p5, ymax = yearly_risk_p95)) +
  geom_point(aes(y = yearly_risk_median)) +
  facet_grid(pathogen ~ w_source) +
  coord_flip() +
  #scale_y_log10()+
  theme_minimal() +
  labs(x = NULL , y = "median yearly risk of infection") +
  scale_y_log10(
    breaks = scales::trans_breaks("log10", function(x)
      10 ^ x),
    labels = scales::trans_format("log10", scales::math_format(10 ^ .x))
  )

print(risk_year)
```



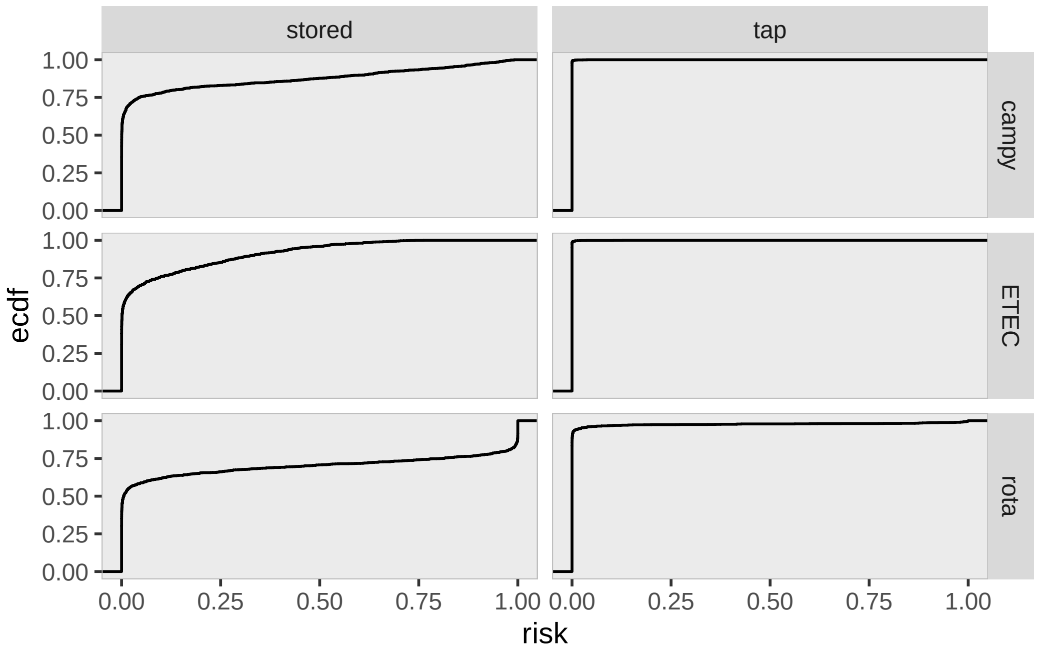
```
ggsave("risk_year.png", risk_year, dpi = "retina", units = "cm", width = 15, height = 15)
```

```
df_yearly_riks %>% summarise(across(yearly_risk,
                             list(mean = mean,
                                  median = median,
                                  p5 = \(x) quantile(x,0.05),
                                  p95 = \(x) quantile(x,0.95)
                                  )
                             ),
                      .by = c(w_source, pathogen))
```

```
# A tibble: 6 x 6
  w_source pathogen yearly_risk_mean yearly_risk_median yearly_risk_p5
  <chr>    <chr>               <dbl>              <dbl>          <dbl>
1 tap      ETEC              0.0117          0.00000309       5.54e-10
2 tap      campy             0.00869         0.00000143       2.39e-10
3 tap      rota              0.105           0.0000120        1.09e-11
4 stored   ETEC              0.483           0.340            1.96e- 6
5 stored   campy             0.443           0.177            8.97e- 7
6 stored   rota              0.543           0.804            1.30e- 7
# i 1 more variable: yearly_risk_p95 <dbl>
```

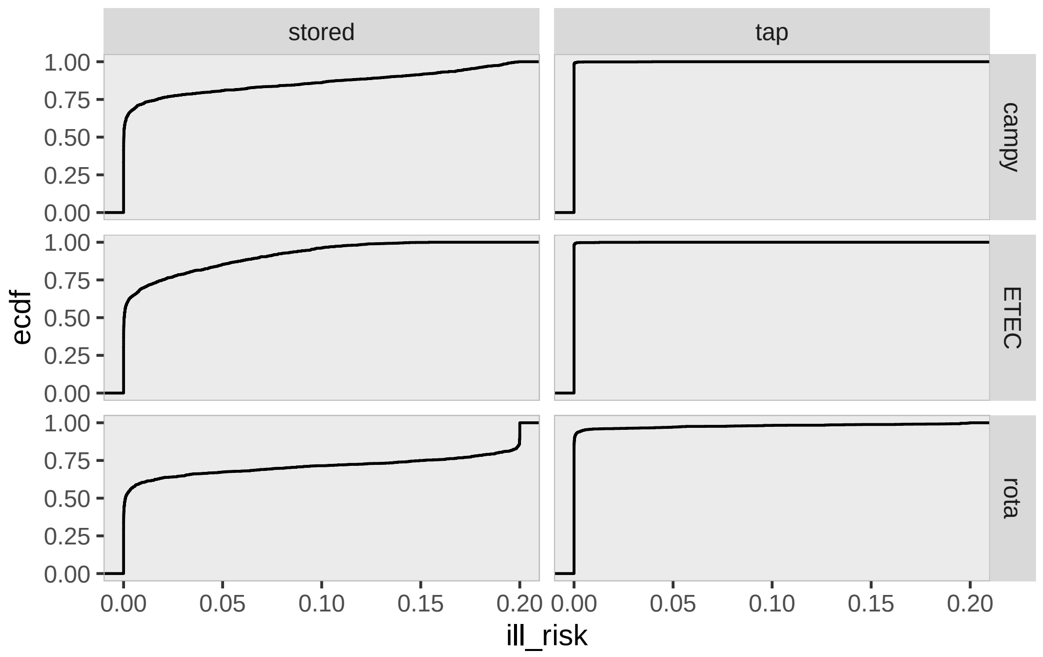
Daily risk CDFs per pathogen

```
df_risk %>%
  sample_n(1e4) %>%
  ggplot(aes(x=risk))+
  stat_ecdf(geom = "step")+
  facet_grid(pathogen~w_source)+
  theme(panel.grid = element_blank(),
        panel.border = element_rect(linewidth = 0.3,colour = "grey",fill = NA))
```



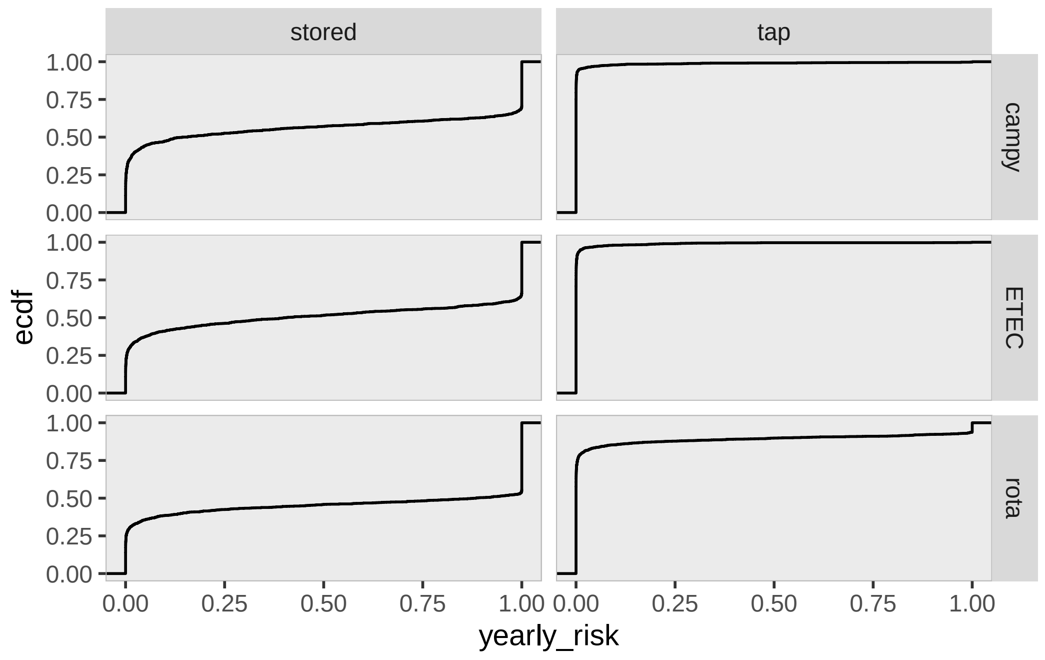
Daily illness risk CDFs per pathogen

```
df_illness_risk %>%
  sample_n(1e4) %>%
  ggplot(aes(x=ill_risk))+
  stat_ecdf(geom = "step")+
  facet_grid(pathogen~w_source)+
  theme(panel.grid = element_blank(),
        panel.border = element_rect(linewidth = 0.3,colour = "grey",fill = NA))
```



Yearly risk of infection CDFs per pathogen

```
df_yearly_riks %>%
  sample_n(1e4) %>%
  ggplot(aes(x=yearly_risk))+
  stat_ecdf(geom = "step")+
  facet_grid(pathogen~w_source)+
  theme(panel.grid = element_blank(),
        panel.border = element_rect(linewidth = 0.3,colour = "grey",fill = NA))
```



Sensitivity analysis

```
cor1 <- df_risk |>
  summarise(cor_Risk_Vol = cor(risk,vol_L, method = "spearman"),
            cor_Risk_Conc_P = cor(risk,conc_p, method = "spearman"),
            cor_Risk_dose = cor(risk,dose, method = "spearman"),
            #cor_Risk_N50 = cor(risk,N50),
            #cor_Risk_a = cor(risk,a),
            .by = c(w_source,pathogen)) |>
    mutate(
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  )
```

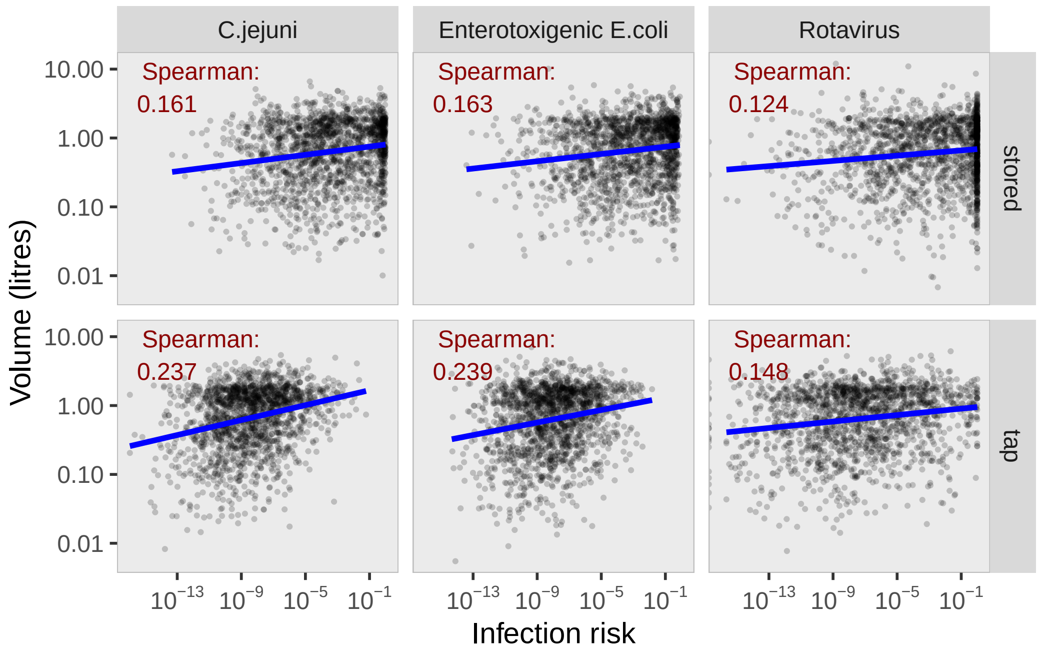
Correlation plots

risk vs vol

```
corr_riskvsvol <- df_risk %>%
  sample_n(1e4) |>
```

```
    mutate(
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  ) %>%
  ggplot(aes(x = risk, y = vol_L)) +
  geom_point(shape = 19,
             alpha = 0.2,
             size = 0.4) +
  geom_smooth(method = "lm",
              se = FALSE,
              color = "blue") +
  geom_text(
    data = cor1,
    aes(label = paste("Spearman: \n", round(cor_Risk_Vol, 3))),
    x = -Inf,
    y = Inf,
    size = 3,
    hjust = -0.2,
    vjust = 1.2,
    color = "darkred"
  ) +
  scale_x_log10(labels = trans_format("log10", math_format(10 ^ .x))) +
  scale_y_log10() +
  # theme_ipsum_rc()+
  labs(y = "Volume (litres)", x = "Infection risk") +
  facet_grid(w_source ~ pathogen) +
  theme(
    panel.grid = element_blank(),
    panel.border = element_rect(
      linewidth = 0.3,
      colour = "grey",
      fill = NA
    )
  )

print(corr_riskvsvol)
```

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



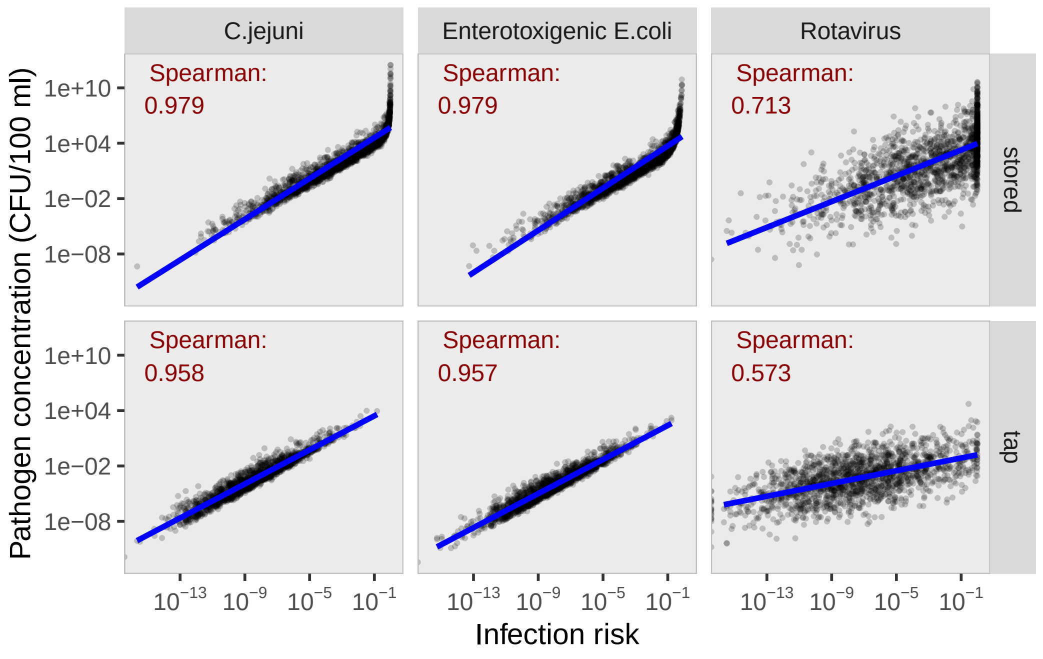
```
ggsave("corr_riskvsvol.png", corr_riskvsvol, dpi = "retina", units = "cm", width = 15, hei
```

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

risk vs pathogen concentration

```
corr_riskvsconcP <- df_risk %>%
  sample_n(1e4) |>
      mutate(
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  ) %>%
  ggplot(aes(x=risk, y=conc_p))+
  geom_point(shape = 19,alpha = 0.2,size = 0.4)+
  geom_smooth(method = "lm", se = FALSE, color = "blue")+
  geom_text(data = cor1,
            aes(label = paste("Spearman: \n", round(cor_Risk_Conc_P, 3))),
            x = -Inf, y = Inf, size = 3,
            hjust = -0.2, vjust = 1.2, color = "darkred") +
  scale_x_log10(labels = trans_format("log10", math_format(10^.x)))+
  scale_y_log10()+
  # theme_ipsum_rc()+
  labs(y = "Pathogen concentration (CFU/100 ml)",x = "Infection risk")+
  facet_grid(w_source~pathogen)+
  theme(panel.grid = element_blank(),
        panel.border = element_rect(linewidth = 0.3,colour = "grey",fill = NA)
        )
print(corr_riskvsconcP)
```

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



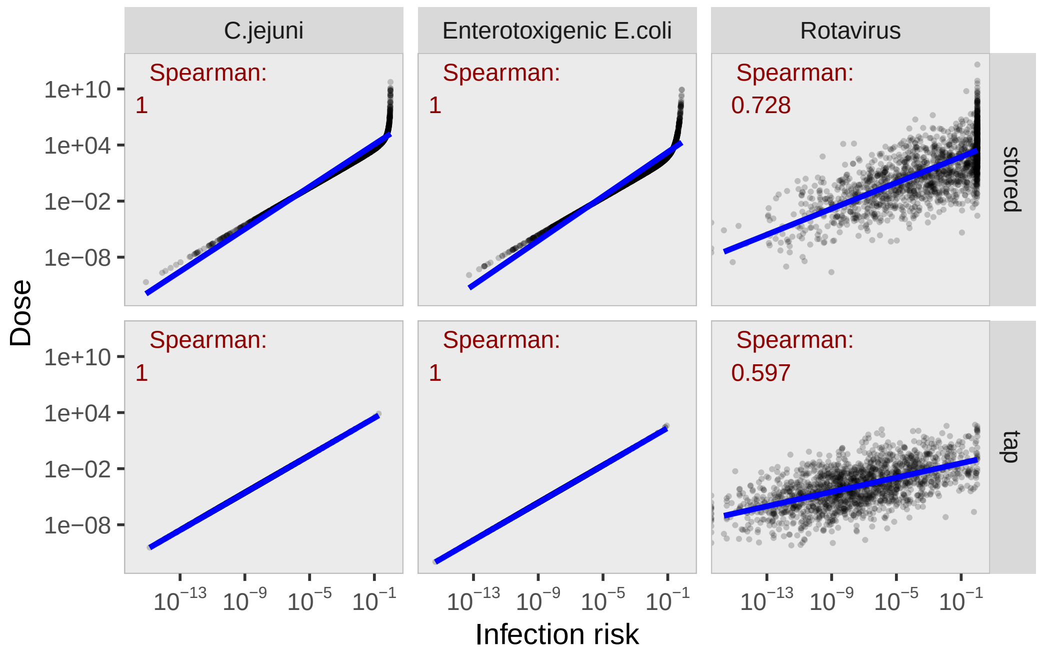
```
ggsave("corr_riskvsconcP.png", corr_riskvsconcP, dpi = "retina", units = "cm", width = 15,
```

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

risk vs dose

```
df_risk %>%
  sample_n(1e4) |>
      mutate(
    pathogen = case_when(
      pathogen == "ETEC" ~ "Enterotoxigenic E.coli",
      pathogen == "campy" ~ "C.jejuni",
      pathogen == "rota" ~ "Rotavirus"
    )
  ) %>%
  ggplot(aes(x=risk, y=dose))+
  geom_point(shape = 19,alpha = 0.2,size = 0.4)+
  geom_smooth(method = "lm", se = FALSE, color = "blue")+
  geom_text(data = cor1,
            aes(label = paste("Spearman: \n", round(cor_Risk_dose, 3))),
            x = -Inf, y = Inf, size = 3,
            hjust = -0.2, vjust = 1.2, color = "darkred") +
  scale_x_log10(labels = trans_format("log10", math_format(10^.x)))+
  scale_y_log10()+
  # theme_ipsum_rc()+
  labs(y = "Dose",x = "Infection risk")+
  facet_grid(w_source~pathogen)+
  theme(panel.grid = element_blank(),
        panel.border = element_rect(linewidth = 0.3,colour = "grey",fill = NA)
        )
```

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



risk vs N50

```
# df_risk %>%
#   sample_n(1e4) |>
#   ggplot(aes(x=risk, y=N50))+
#   geom_point(shape = 19,alpha = 0.2,size = 0.4)+
#   geom_smooth(method = "lm", se = FALSE, color = "blue")+
#   geom_text(data = cor1,
#             aes(label = paste("Spearman: \n", round(cor_Risk_N50, 3))),
#             x = -Inf, y = Inf, size = 3,
#             hjust = -0.2, vjust = 1.2, color = "darkred") +
#   scale_x_log10(labels = trans_format("log10", math_format(10^.x)))+
#   scale_y_log10()+
#   theme_ipsum_rc()+
#   labs(y = "Pathogen concentration (CFU/100 ml))",x = "Infection risk")+
#   facet_grid(w_source~pathogen)+
#   theme(panel.grid = element_blank(),
#         panel.border = element_rect(linewidth = 0.3,colour = "grey",fill = NA)
#         )
```

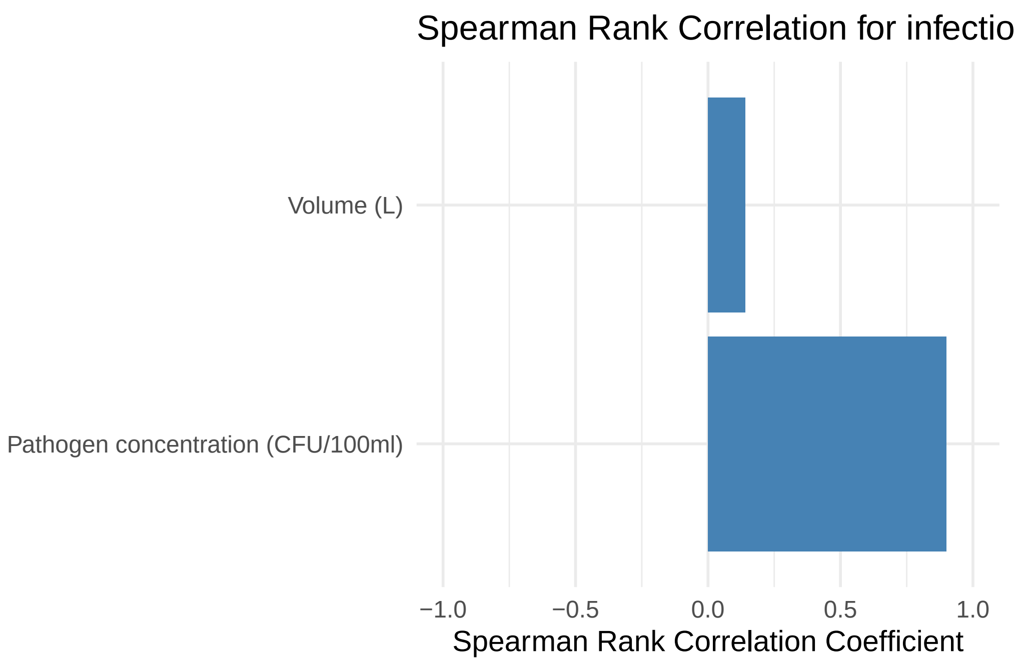
risk vs a

```
# df_risk %>%
#   sample_n(1e4) |>
#   ggplot(aes(x=risk, y=a))+
#   geom_point(shape = 19,alpha = 0.2,size = 0.4)+
#   geom_smooth(method = "lm", se = FALSE, color = "blue")+
#   geom_text(data = cor1,
#             aes(label = paste("Spearman: \n", round(cor_Risk_a, 3))),
#             x = -Inf, y = Inf, size = 3,
#             hjust = -0.2, vjust = 1.2, color = "darkred") +
#   scale_x_log10(labels = trans_format("log10", math_format(10^.x)))+
#   scale_y_log10()+
#   theme_ipsum_rc()+
#   labs(y = "a",x = "Infection risk")+
#   facet_grid(w_source~pathogen)+
#   theme(panel.grid = element_blank(),
#         panel.border = element_rect(linewidth = 0.3,colour = "grey",fill = NA)
#         )
```

Tornado plots

```
cor2 <- df_risk %>%
  sample_n(1e4) %>%
```

```r
  cor_test(risk, vol_L, method = "spearman")

cor3 <- df_risk %>%
  sample_n(1e4) %>%
  cor_test(risk, conc_p, method = "spearman")

# cor4 <- df_risk %>%
#   sample(1e4) %>%
#   cor_test(risk, N50, method="spearman")


cor_all <- data.frame(names=c("Volume (L)", "Pathogen concentration (CFU/100ml)"),
                      spearman=c(cor2$cor, cor3$cor))

ggplot(cor_all, aes(x = names, y = spearman)) +
  geom_bar(stat = "identity", position = "identity", fill = "steelblue") +
  labs(title = "Spearman Rank Correlation for infection risk",
       x = " ",
       y = "Spearman Rank Correlation Coefficient") +
  scale_y_continuous(limits = c(-1, 1))+
  theme_minimal() +
  coord_flip()
```


Spearman Rank Correlation for infectio
Volume (L)
Pathogen concentration (CFU/100ml)
Spearman Rank Correlation Coefficient