varameta::

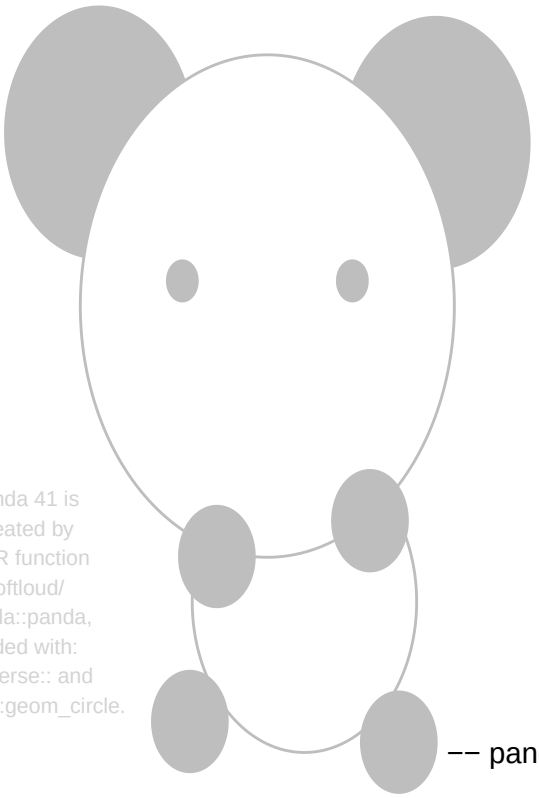
Contents


```
# for reproducibility
set.seed(39)

# packages
library(varameta)
library(tidyverse)
library(simeta)
library(panda)

# move to parameter when done with chunk-wise?
sample_size <- 10
```

1 Objective of the `varameta::` package

The `varameta::` package aims to bridge the toolchain gap from a dataset containing medians to a software package, such as `metafor::` (Viechtbauer 2010), for meta-analysis.

```
panda("varameta:: is for meta-analysing medians")
#> Set panda = 41 to reproduce this panda.
```



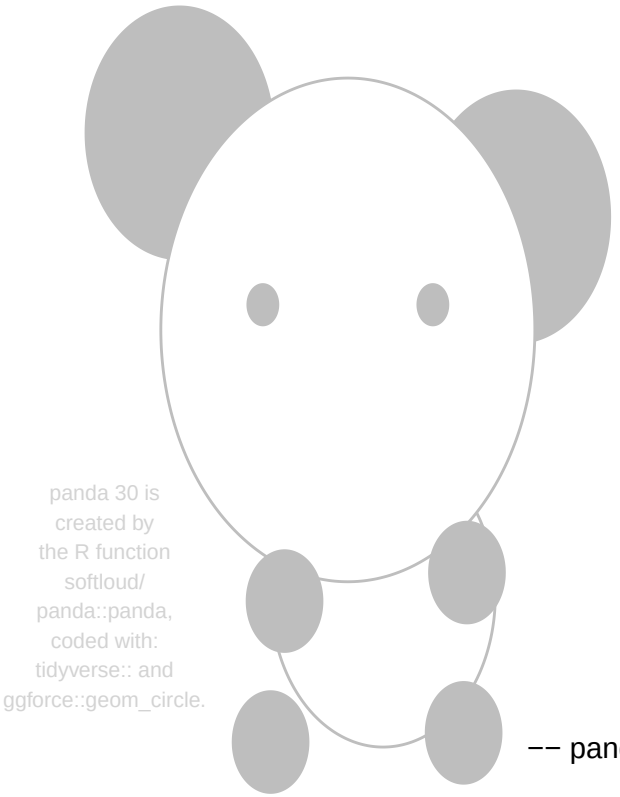
varameta:: is for meta−
analysing medians

panda 41 is
created by
the R function
softloud/
panda::panda,
coded with:
tidyverse:: and
ggforce::geom_circle.

−− panda 41

2 A minimal demonstration of calculating the variance of the sample median

```
panda("show me the code!")
#> Set panda = 30 to reproduce this panda.
```



show me the code!

-- panda 30

2.1 Compute standard error of the sample median

```
# get a sample
a_sample <- rexp(sample_size)

# get standard error of the sample median
effect_se(
  centre = median(a_sample),
  spread = IQR(a_sample),
  n = length(a_sample),
  centre_type = "median",
  spread_type = "iqr"
)
#> [1] 0.1400956
```

2.2 Vectorised calculations for dataframes

Here we borrow a function from the companion `simeta::` package. See below for details.

```
# generate random meta-analysis dataset
# one row per study
(ma_sample <- sim_stats() %>%
  # filter down to one group per study
  dplyr::filter(group == "control"))
#> # A tibble: 3 x 5
#>   study   group   effect effect_spread     n
#>   <chr>   <chr>    <dbl>         <dbl> <dbl>
```

```
#> 1 study_1 control   48.4         0.310    93
#> 2 study_2 control   47.3         0.350    36
#> 3 study_3 control   59.2         0.275    65
```

```
ma_sample %>%
  # append a column with the standard error of the median for each study
  mutate(effect_se = pmap_dbl(
    list(centre = effect,
         spread = effect_spread,
         n = n),
    effect_se,
    centre_type = "median",
    spread_type = "iqr"
  ))
#> # A tibble: 3 x 6
#>   study   group   effect effect_spread     n effect_se
#>   <chr>   <chr>    <dbl>         <dbl> <dbl>     <dbl>
#> 1 study_1 control   48.4         0.310    93    0.0298
#> 2 study_2 control   47.3         0.350    36    0.0542
#> 3 study_3 control   59.2         0.275    65    0.0317
```

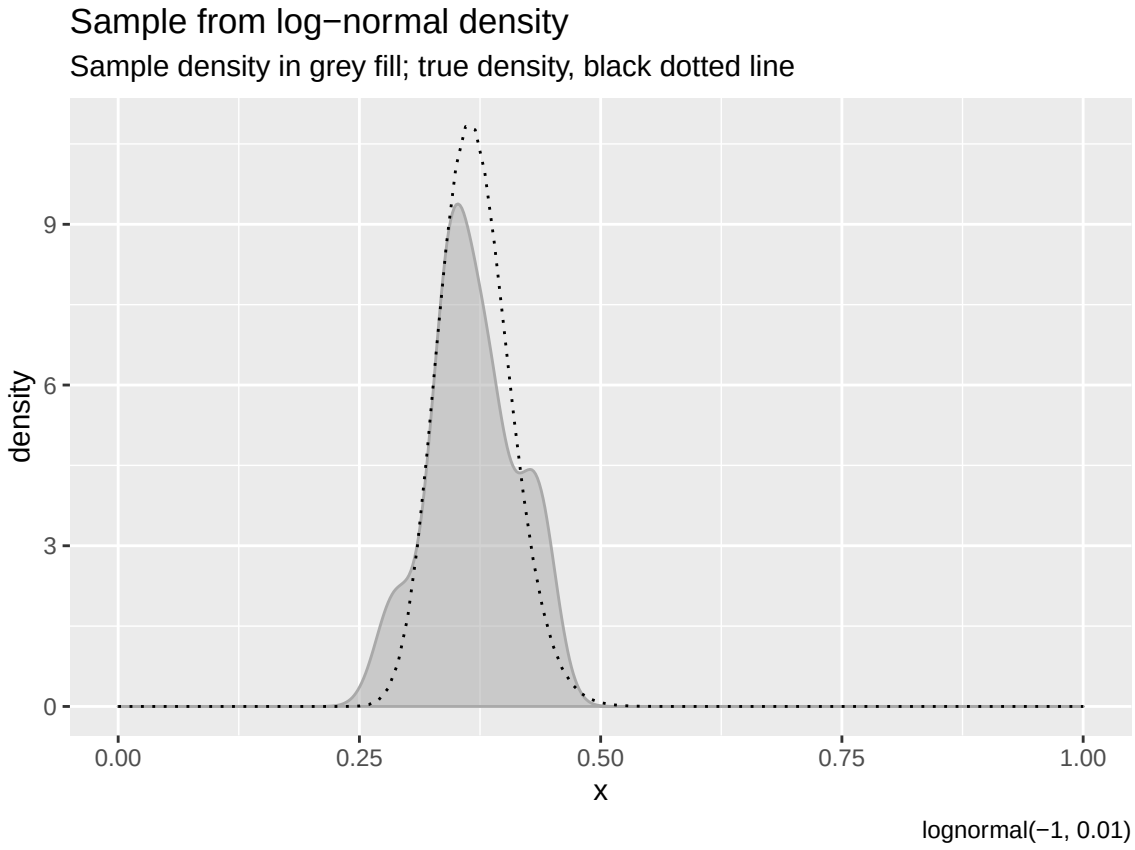
3 Calculate the standard error of mean or median based on effect, spread, and sample size

A wrapper function `effect_se` provides a quick method of calculating the error of an effect based on its measure of effect, spread, sample size.

Consider a randomly generated sample.

```
(a_sample <- rlnorm(sample_size,-1, 0.1))
#>  [1] 0.3534391 0.3395523 0.3876237 0.2869194 0.3688613 0.3874923 0.4324285 0.4337460 0.3388299 0.344

# summary statistics for this sample
a_sample %>% log() %>% summary()
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#> -1.2486 -1.0762 -1.0187 -1.0082 -0.9478 -0.8353
```

Taken from a lognormal distribution lognormal$(-1, 0.1^2)$.



3.1 Estimate the standard error of the sample median

With the `varameta::` package we execute the following code to estimate the standard error of the sample median, with sample summary statistics: `median`; `IQR` (interquartile range); `length` for sample size.

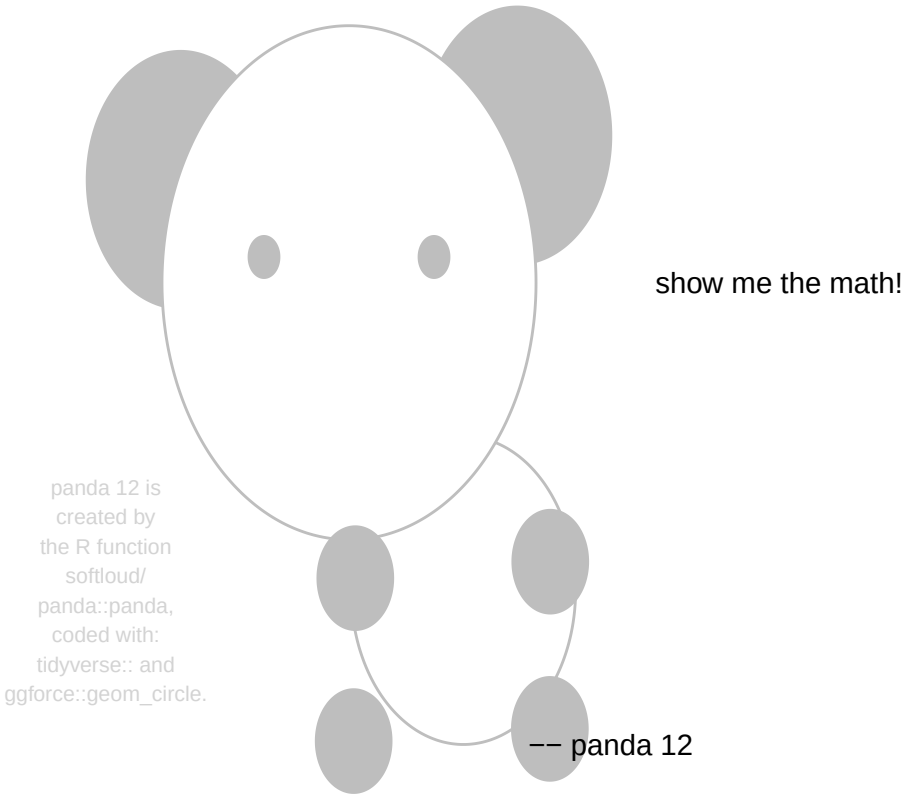
```
# standard error based on the median, interquartile range, and sample size
effect_se(
  centre = median(a_sample),
  spread = IQR(a_sample),
  n = length(a_sample),
  centre_type = "median",
  spread_type = "iqr"
)
#> [1] 0.01370849
```

The calculation can be adapted for the range.

```
# standard error based on the median, range, and sample size
effect_se(
  centre = median(a_sample),
  spread = abs(diff(range(a_sample))),
  n = length(a_sample),
  centre_type = "median",
  spread_type = "range"
)
#> [1] 0.01756955
```

```
panda("show me the math!")
```

```
#> Set panda = 12 to reproduce this panda.
```



This calculation

todo: eqn to finish off section! words around eqns to finish off section blah blah see companion manuscript..

$$\mathrm{v}(M) := \frac{1}{4n\left[g\left(M;\hat{\theta}\right)\right]^2}$$

$$\hat{\mu} := \log(M).$$

$$G^{-1}(p;\mu,\sigma) = \exp(\sigma\Phi^{-1}(p) + \mu).$$

$$\hat{\sigma}^{(1)} := \frac{1}{\Phi^{-1}\left(\frac{3}{4}\right)} \log\left(\frac{\mathrm{iqr}e^{-\hat{\mu}} \pm \sqrt{\mathrm{iqr}^2 e^{-2\hat{\mu}} + 4}}{2}\right)$$

$$\hat{\sigma}^{(2)} := \frac{1}{\Phi^{-1}\left(\frac{n-\frac{1}{2}}{n}\right)} \log\left[\frac{(x_{[n]} - x_{[1]})e^{-\hat{\mu}} \pm \sqrt{(x_{[n]} - x_{[1]})^2 e^{-2\hat{\mu}} + 4}}{2}\right].$$

3.2 Calculate the standard error of the sample mean

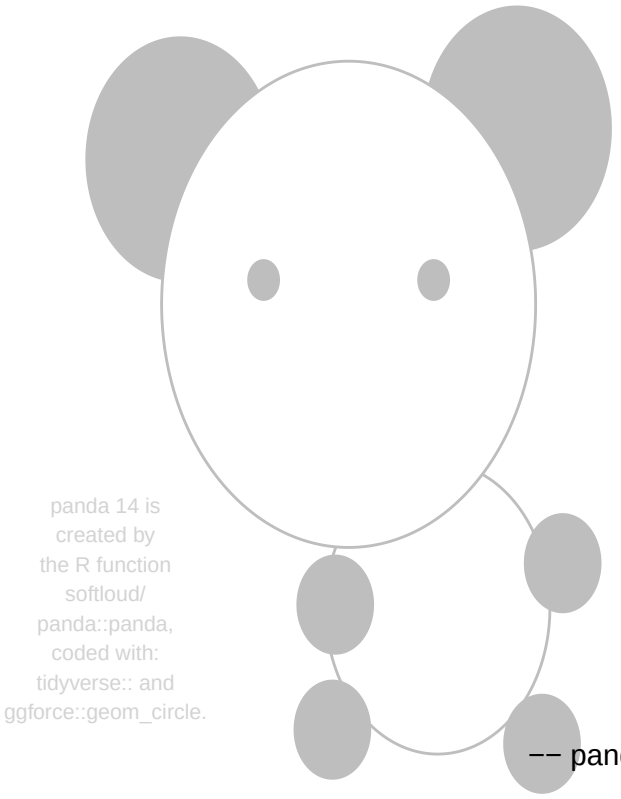
```
# take a sample
se_mean_eg_sample <- rexp(sample_size)
```

Now, we wish to calculate the standard error $s/\sqrt{n}$ of the sample mean, calcualted with the the sample standard deviation and the squared-root of the sample size.

```
# mean and sd
effect_se(
  centre = mean(se_mean_eg_sample),
  spread = sd(se_mean_eg_sample),
  n = length(se_mean_eg_sample),
  centre_type = "mean",
  spread_type = "sd"
)
#> [1] 0.1548837

# compare
sd(se_mean_eg_sample) /
  sqrt(length(se_mean_eg_sample))
#> [1] 0.1548837

# mean and var
effect_se(
  centre = mean(se_mean_eg_sample),
  spread = var(se_mean_eg_sample),
  n = length(se_mean_eg_sample),
  centre_type = "mean",
  spread_type = "var"
)
#> [1] 0.1548837

# compare
sqrt(var(se_mean_eg_sample) /
  length(se_mean_eg_sample))
#> [1] 0.1548837
```

4 Vectorised calculations for meta-analysis datasets

```
panda("vectorisation is my favourite part of R", panda = 14)
#> Set panda = 14 to reproduce this panda.
```



vectorisation is my
favourite part of R

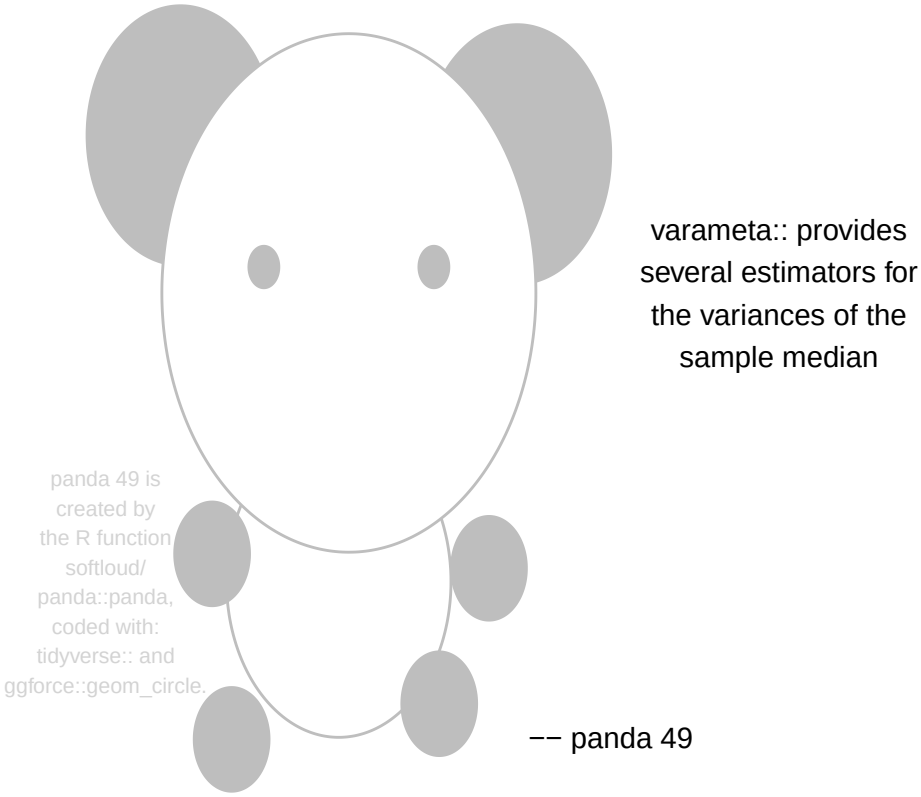
panda 14 is
created by
the R function
softloud/
panda::panda,
coded with:
tidyverse:: and
ggforce::geom_circle.

-- panda 14

```
# borrowing from sister package simeta:: to simulate a dataset
(meta_data <- sim_stats() %>%
  dplyr::filter(group == "control"))
#> # A tibble: 3 x 5
#>   study   group   effect effect_spread     n
#>   <chr>   <chr>    <dbl>         <dbl> <dbl>
#> 1 study_1 control   64.4         0.189    19
#> 2 study_2 control   52.0         0.273    99
#> 3 study_3 control   48.6         0.190    27

# add a vecorised
# todo function this (after report)
meta_data %>%
  mutate(
    effect_se = pmap_dbl(
      list(centre = effect, spread = effect_spread, n = n),
      effect_se,
      centre_type = "median",
      spread_type = "iqr"
    )
  )
#> # A tibble: 3 x 6
#>   study   group   effect effect_spread     n effect_se
#>   <chr>   <chr>    <dbl>         <dbl> <dbl>     <dbl>
#> 1 study_1 control   64.4         0.189    19    0.0403
#> 2 study_2 control   52.0         0.273    99    0.0255
#> 3 study_3 control   48.6         0.190    27    0.0339
```

5 Other estimators

```
panda(
  panda = 49,
  "varameta:: provides several estimators for the variances of the sample median")
#> Set panda = 49 to reproduce this panda.
```



All other estimators available for meta-analysis are provided in `varameta::` for use in a compative analysis.

References

Viechtbauer, Wolfgang. 2010. "Conducting Meta-Analyses in R with the metafor Package." *Journal of Statistical Software* 36 (3): 1–48.