Bounding CDFs

Derumigny, A. , Guyonvarch Y. & Girard, L.

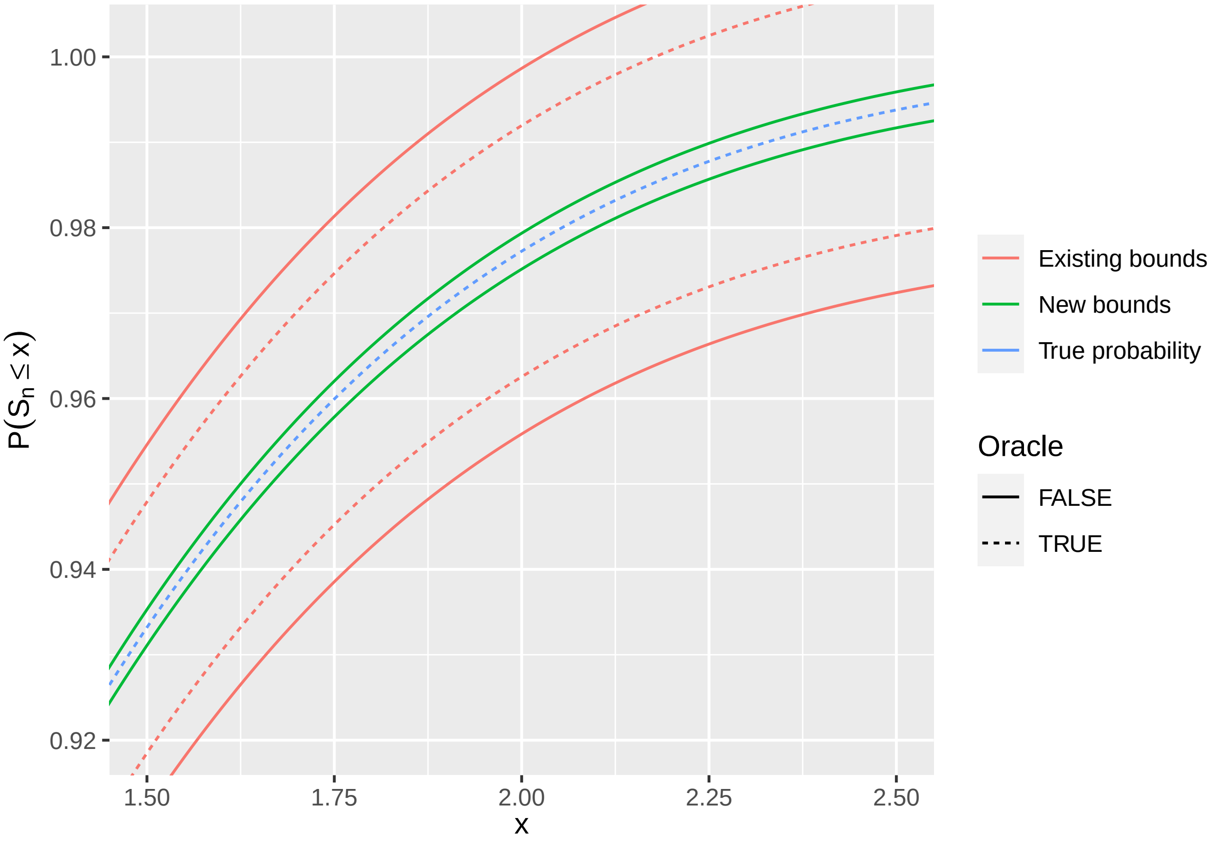
May 12, 2023

```r
Nrep = 10000
```

Student-t distribution with 5 degrees of freedom

```r
cdf_Student_5df = read.csv("cdf_Student_n5000_5df.csv", header = TRUE, sep = ",", dec = ".")

lambda3n_5df = mean(scale(rt(Nrep, df = 5))^3)
K3n_5df = mean(abs(scale(rt(Nrep, df = 5)))^3)
```

```r
n_5df = 5000

x_5df = seq(-5, 5, by = 0.01)

DeltanE_5df = BoundEdgeworth::Bound_EE1(
  list(continuity = TRUE, iid = TRUE, no_skewness = TRUE),
  n = n_5df, K4 = 9)

PreviousBound_5df_oracle = 0.4690 * K3n_5df / sqrt(n_5df)

PreviousBound_5df = 0.4690 * 0.621 * (9^(3/4)) / sqrt(n_5df)

df_bounds_Student_5df = data.frame(
  x = c(cdf_Student_5df$x, rep(x_5df, times = 6)),
  y = c(cdf_Student_5df$cdf,
        pnorm(x_5df) - DeltanE_5df,
        pnorm(x_5df) + DeltanE_5df,
        pnorm(x_5df) - PreviousBound_5df,
        pnorm(x_5df) + PreviousBound_5df,
        pnorm(x_5df) - PreviousBound_5df_oracle,
        pnorm(x_5df) + PreviousBound_5df_oracle),
  type = c(rep("True probability", nrow(cdf_Student_5df)),
           rep(c("New bounds", "New bounds",
                 "Existing bounds", "Existing bounds",
                 "Existing bounds", "Existing bounds"),
               each = length(x_5df))),
  uniqid = c(rep("True", nrow(cdf_Student_5df)),
             rep(c("New_min", "New_max", "Existing_min", "Existing_max",
                   "Existing_min_oracle", "Existing_max_oracle"), each = length(x_5df))),

  Oracle = c(rep(c(TRUE,
                   FALSE, FALSE,
                     FALSE, FALSE, TRUE, TRUE), each = length(x_5df) ) )
)

# plot(cdf_Student_5df$x, cdf_Student_5df$cdf, xlim = c(-5, 5), type = "l")
# ymin = pnorm(x) - DeltanE
# ymax = pnorm(x) + DeltanE
#
# lines(x, ymin, col = "red")
# lines(x, ymax, col = "red")
#
# lines(x, pnorm(x) - PreviousBound, col = "blue")
# lines(x, pnorm(x) + PreviousBound, col = "blue")

plot_Student_5df <- ggplot(df_bounds_Student_5df) +
  geom_line(aes(x = x, y = y, color = type, group = uniqid, linetype = Oracle)) +
  coord_cartesian(xlim = c(1.5, 2.5), ylim = c(0.92, 1.002)) +
  xlab(label = expression(x)) +
  ylab(label = expression(P(S[n] <= x))) +
  guides(color = guide_legend(title = element_blank()) )

print(plot_Student_5df)
```

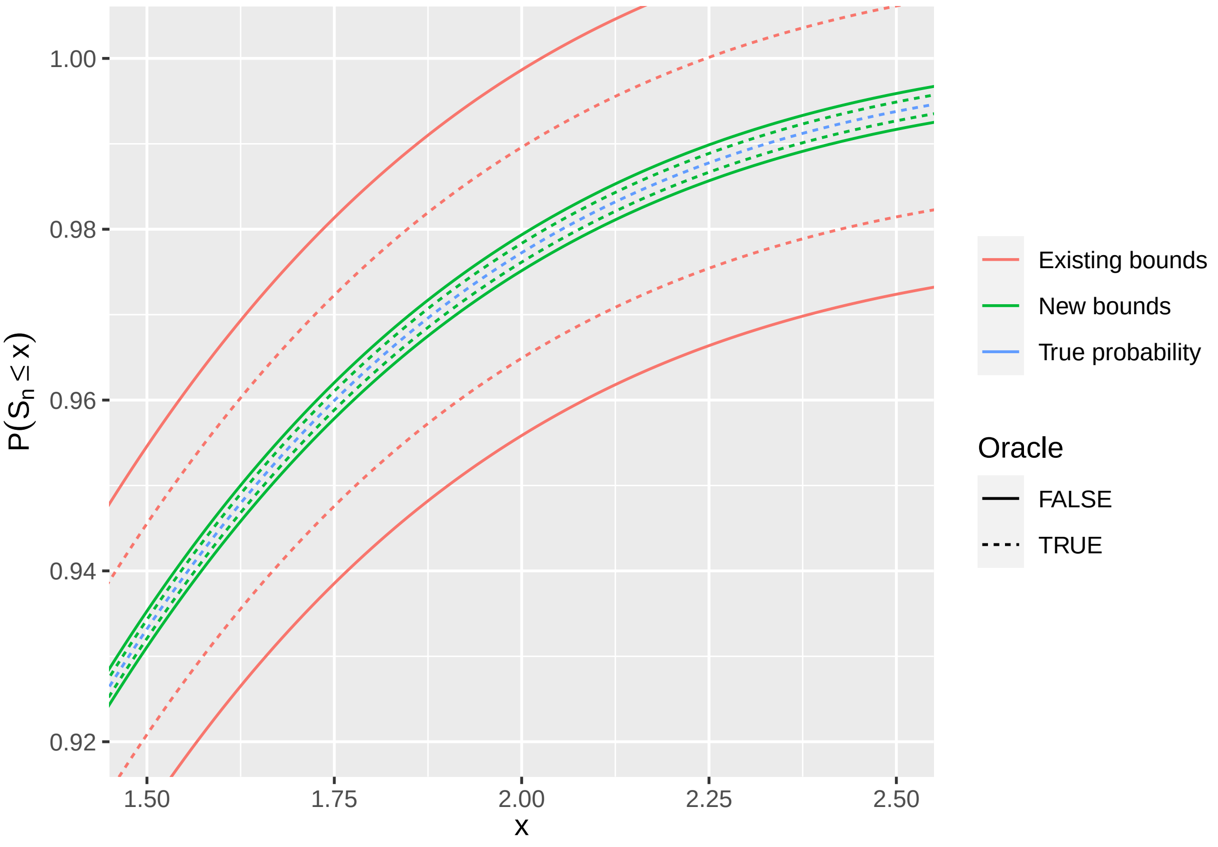


```
ggsave("plot_bounds_Student_5df.pdf", plot = plot_Student_5df, width = 10, height = 5)
```

Student-t distribution with 8 degrees of freedom

```r
cdf_Student_8df = read.csv("cdf_Student_n5000_8df.csv", header = TRUE, sep = ",", dec = ".")

lambda3n_8df = mean(scale(rt(Nrep, df = 8))^3)
K3n_8df = mean(abs(scale(rt(Nrep, df = 8)))^3)
K4n_8df = 3 + 6 / (8 - 4)
```

```r
n_8df = 5000

x_8df = seq(-5, 5, by = 0.01)

DeltanE_8df = BoundEdgeworth::Bound_EE1(
  list(continuity = TRUE, iid = TRUE, no_skewness = TRUE),
  n = n_8df, K4 = 9)

DeltanE_8df_oracle = BoundEdgeworth::Bound_EE1(
  list(continuity = TRUE, iid = TRUE, no_skewness = TRUE),
  n = n_8df, K4 = K4n_8df)

PreviousBound_8df_oracle = 0.4690 * K3n_8df / sqrt(n_8df)

PreviousBound_8df = 0.4690 * 0.621 * (9^(3/4)) / sqrt(n_8df)

df_bounds_Student_8df = data.frame(
  x = c(cdf_Student_8df$x, rep(x_8df, times = 8)),
  y = c(cdf_Student_8df$cdf,
        pnorm(x_8df) - DeltanE_8df,
        pnorm(x_8df) + DeltanE_8df,
        pnorm(x_8df) - DeltanE_8df_oracle,
        pnorm(x_8df) + DeltanE_8df_oracle,
        pnorm(x_8df) - PreviousBound_8df,
        pnorm(x_8df) + PreviousBound_8df,
        pnorm(x_8df) - PreviousBound_8df_oracle,
        pnorm(x_8df) + PreviousBound_8df_oracle),
  type = c(rep("True probability", nrow(cdf_Student_8df)),
           rep(c("New bounds", "New bounds",
                 "New bounds", "New bounds",
                 "Existing bounds", "Existing bounds",
                 "Existing bounds", "Existing bounds"),
               each = length(x_8df))),
  uniqid = c(rep("True", nrow(cdf_Student_8df)),
             rep(c("New_min", "New_max",
                   "New_min_oracle", "New_max_oracle",
                   "Existing_min", "Existing_max",
                   "Existing_min_oracle", "Existing_max_oracle"), each = length(x_8df))),

  Oracle = c(rep(c(TRUE,
                   FALSE, FALSE, TRUE, TRUE,
                   FALSE, FALSE, TRUE, TRUE), each = length(x_8df) )
  )
)

plot_Student <- ggplot(df_bounds_Student_8df) +
  geom_line(aes(x = x, y = y, color = type, group = uniqid, linetype = Oracle)) +
  coord_cartesian(xlim = c(1.5, 2.5), ylim = c(0.92, 1.002)) +
  xlab(label = expression(x)) +
  ylab(label = expression(P(S[n] <= x))) +
  guides(color = guide_legend(title = element_blank()) )


print(plot_Student)
```



```
ggsave("plot_bounds_Student_8df.pdf", plot = plot_Student, width = 10, height = 5)
```

Gamma distribution

If $X_i \sim Gamma(1, scale = 1)$, $mu = 1$, $sigma = 1$,

$$S_n = \sum_{i=1}^{n}(X_i - 1)/\sqrt{\sigma * n} = (\sum_{i=1}^{n} X_i/\sqrt{n}) - \sqrt{n}$$

$$P(S_n \leq x) = P(Gamma(n, 1)/sqrtn - \sqrt{n} \leq x) = P(Gamma(n, 1) \leq x * sqrt(n) + n)$$

```
library(Rmpfr)
```

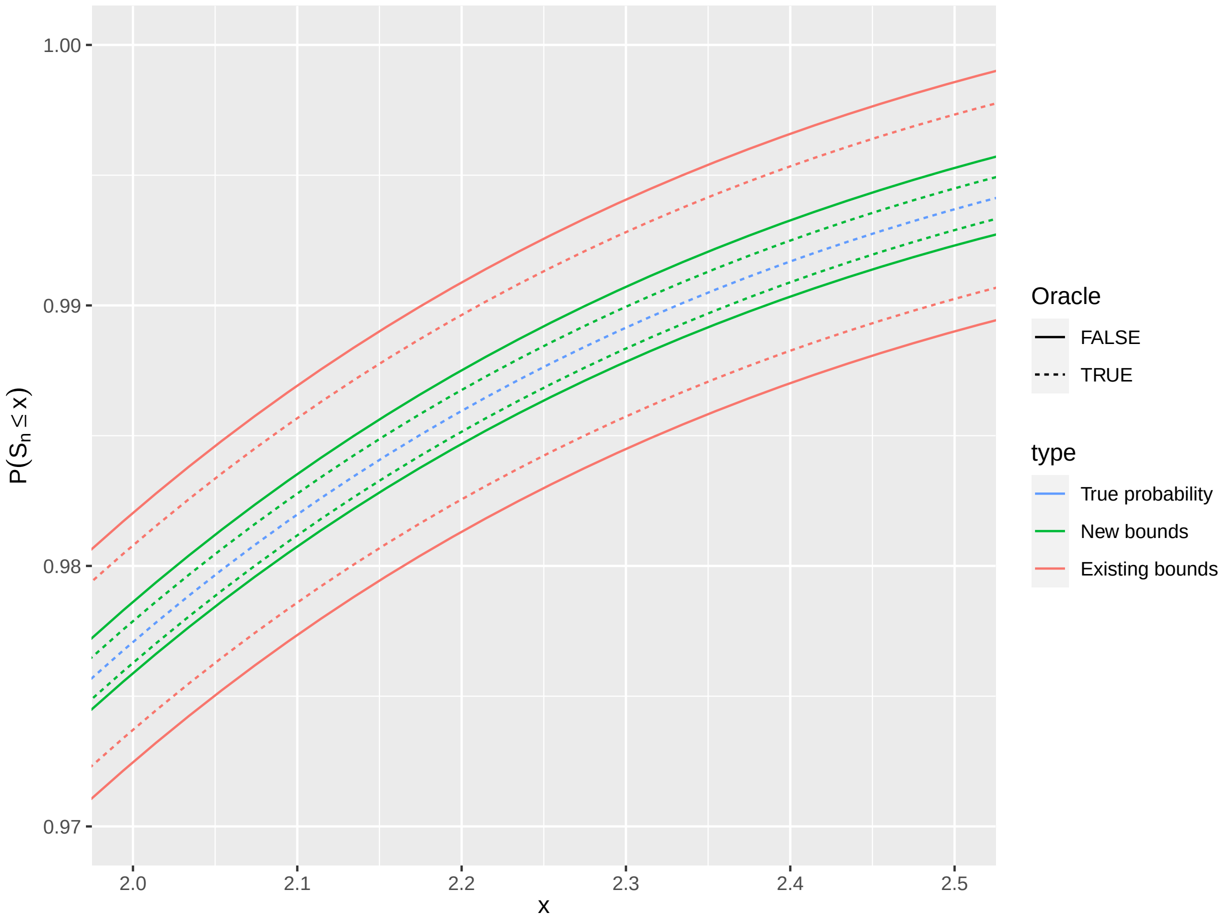
```
## Le chargement a nécessité le package : gmp
```

```
##
## Attachement du package : 'gmp'

## Les objets suivants sont masqués depuis 'package:base':
##
##     %*%, apply, crossprod, matrix, tcrossprod

## C code of R package 'Rmpfr': GMP using 64 bits per limb

##
## Attachement du package : 'Rmpfr'

## L'objet suivant est masqué depuis 'package:gmp':
##
##     outer

## Les objets suivants sont masqués depuis 'package:stats':
##
##     dbinom, dgamma, dnbinom, dnorm, dpois, dt, pnorm

## Les objets suivants sont masqués depuis 'package:base':
##
##     cbind, pmax, pmin, rbind
```

```
precBits = 500

n = 100000
meanx = sqrt(n)
x_mpfr = seqMpfr(from = mpfr(-5, precBits),
                 to = mpfr(5, precBits), length.out = 500)

shape = mpfr(n, precBits) # shape parameter
gamma_a = Rmpfr::igamma(a = shape, 0)
cdf_Gamma = mpfr(1, precBits) - ( Rmpfr::igamma(a = shape, x = x_mpfr * sqrt(n) + n) / gamma_a )

# plot(x, cdf_Gamma, type = "l")
```

```
x = as.numeric(x_mpfr)

# plot(x, cdf_Gamma, type = "l", xlim = c(-5, 5), ylim = c(0,1.02))

df_mc = scale(rexp(Nrep, rate = 1))
lambda3n = mean(df_mc^3)
K3 = mean(abs(df_mc)^3)
K4 = 9

DeltanE = BoundEdgeworth::Bound_EE1(
  list(continuity = TRUE, iid = TRUE, no_skewness = FALSE),
  n = n, K4 = 9)

DeltanE_oracle = BoundEdgeworth::Bound_EE1(
  list(continuity = TRUE, iid = TRUE, no_skewness = FALSE),
  n = n, K3 = K3, K4 = K4, lambda3 = lambda3n)


# ymin_unif = pnorm(x) - 0.621 * 9^(3/4) / (6 * sqrt(n)) * 0.067 - DeltanE
# ymax_unif = pnorm(x) + 0.621 * 9^(3/4) / (6 * sqrt(n)) * 0.067 + DeltanE
```

```
ymin = pnorm(x) - 0.621 * 9^(3/4) / (6 * sqrt(n)) * (1 - x^2) * dnorm(x) - DeltanE
ymax = pnorm(x) + 0.621 * 9^(3/4) / (6 * sqrt(n)) * (1 - x^2) * dnorm(x) + DeltanE

ymin_oracle = pnorm(x) + lambda3n / (6 * sqrt(n)) * (1 - x^2) * dnorm(x) - DeltanE_oracle
ymax_oracle = pnorm(x) + lambda3n / (6 * sqrt(n)) * (1 - x^2) * dnorm(x) + DeltanE_oracle

PreviousBound = 0.4690 * 0.621 * 9^(3/4) / sqrt(n)
y_previous_min = pnorm(x) - PreviousBound
y_previous_max = pnorm(x) + PreviousBound

PreviousBound_oracle = 0.4690 * K3 / sqrt(n)
y_previous_oracle_min = pnorm(x) - PreviousBound_oracle
y_previous_oracle_max = pnorm(x) + PreviousBound_oracle

df_bounds_Gamma = data.frame(
  x = c(rep(x, times = 9)),
  y = c(as.numeric(cdf_Gamma),
        ymin, ymax,
        # ymin_unif, ymax_unif,
        ymin_oracle, ymax_oracle,
        y_previous_min, y_previous_max,
        y_previous_oracle_min, y_previous_oracle_max),
  type = factor(c(rep(c("True probability",
                        "New bounds", "New bounds",
                        "New bounds", "New bounds",
                        "Existing bounds", "Existing bounds",
                        "Existing bounds", "Existing bounds"), each = length(x))),
                levels = c("True probability", "New bounds", "Existing bounds")),
  uniqid = rep(c("True",
                 "K4=9_min", "K4=9_max", "Oracle_min", "Oracle_max",
                 "Existing_min", "Existing_max",
                 "Existing_oracle_min", "Existing__oracle_max"), each = length(x) ),
  Oracle = c(rep(c(TRUE,
                   FALSE, FALSE, TRUE, TRUE,
                   FALSE, FALSE, TRUE, TRUE), each = length(x) )  )
)

plot_Gamma <- ggplot(df_bounds_Gamma) +
  geom_line(aes(x = x, y = y, color = type, group = uniqid, linetype = Oracle)) +
  # coord_cartesian(xlim = c(1.5, 2.5), ylim = c(0.92, 1.002))
  coord_cartesian(xlim = c(2, 2.5), ylim = c(0.97, 1)) +
  xlab(label = expression(x)) +
  ylab(label = expression(P(S[n] <= x))) +
  scale_color_manual(
    values = c("True probability" = scales::hue_pal()(3)[3],
               "New bounds"       = scales::hue_pal()(3)[2],
               "Existing bounds"  = scales::hue_pal()(3)[1] ) )

print(plot_Gamma)
```



```
ggsave("plot_bounds_Gamma.pdf", plot = plot_Gamma, width = 10, height = 5)
```