Introductory Applied Statistical Methods

Vasileios (Bill) Katsianos

September 2023

Contents

1 Point Estimation **1**

2 Confidence Intervals **40**

3 Statistical Hypothesis Testing **58**

4 Linear Regression **80**

5 Analysis of Variance **80**

6 Nonparametric Methods **80**

1 Point Estimation

Maximum Likelihood Estimation

Direct maximization of the likelihood function isn't always possible. In those situations, one might want to numerically optimize the likelihood function in order to obtain the MLE of the unknown parameter. For distribution families with just one unknown parameter, we can make use of R's built-in optimize function in order to maximize the log-likelihood function. In cases where the MLE of the unknown parameter is tractable, the optimize function should always lead to a maximum value which is very close to the theoretical maximum likelihood estimate. By plotting the curve of the log-likelihood function as a function of the unknown parameter, we can verify that its maximization has been correctly performed and that its maximum is achieved very close to the true value of the unknown parameter.

Example 1.1. Let $X_1, \dots, X_n \sim \text{Poisson}(\lambda)$ be a random sample. Then, we know that:

$$\ell(\lambda \mid x) = -n\lambda + \sum_{i=1}^{n} x_i \log \lambda - \sum_{i=1}^{n} \log x_i!, \quad \widehat{\lambda}(X) = \overline{X}.$$

```r
loglik = function(lambda, x) {
    -length(x) * lambda + sum(x) * log(lambda) - sum(lfactorial(x))
}
```

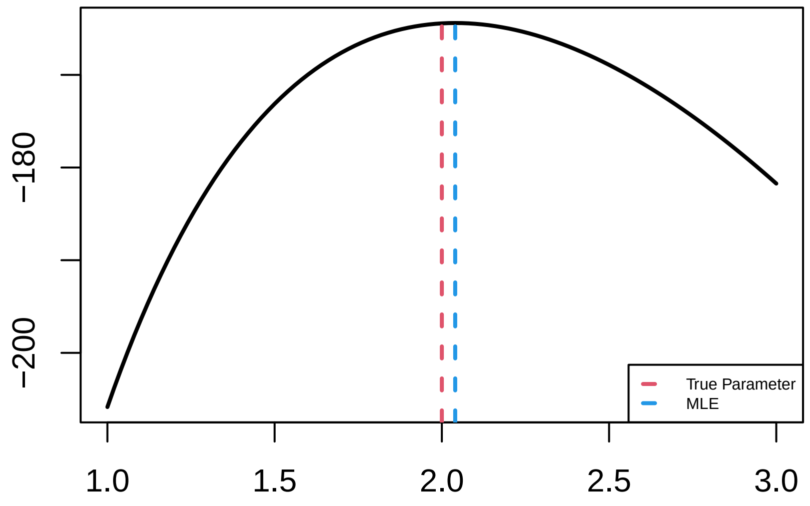
```r
n = 100
lambda = 2
X = rpois(n, lambda)
MLE = mean(X)
print(MLE)
```

```
## [1] 2.04
```

```r
optimize(loglik, c(0, 1e+100), maximum = TRUE, x = X)
```

```
## $maximum
## [1] 2.040006
##
## $objective
## [1] -164.4036
```

```r
curve(loglik(x, X), xlab = NA, ylab = NA, xlim = c(1, 3), lwd = 2)
abline(v = lambda, col = 2, lty = 2, lwd = 2)
abline(v = MLE, col = 4, lty = 2, lwd = 2)
legend("bottomright", c("True Parameter", "MLE"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```



Example 1.2. Let $X_1, \dots, X_n \sim \text{Exp}(\lambda)$ and $Y_1, \dots, Y_n \sim \text{Exp}(1/\lambda)$ be 2 independent random samples. Then, we know that:

$$\ell(\lambda \mid x, y) = -\lambda \sum_{i=1}^{n} x_i - \frac{1}{\lambda} \sum_{i=1}^{n} y_i, \quad \widehat{\lambda}(X, Y) = \sqrt{\frac{\overline{Y}}{\overline{X}}}.$$

```r
loglik = function(lambda, x, y) {
    -lambda * sum(x) - sum(y)/lambda
}

X = rexp(n, lambda)
Y = rexp(n, lambda^(-1))
MLE = sqrt(sum(Y)/sum(X))
```

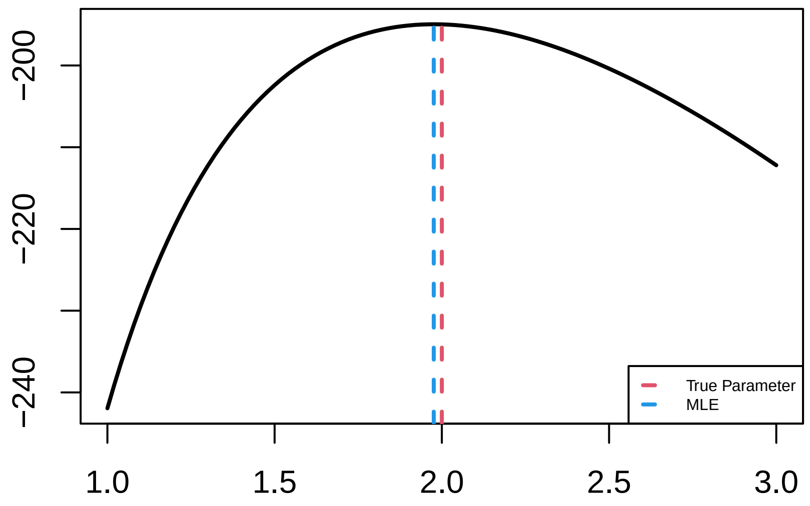
```r
print(MLE)
```

```
## [1] 1.97584
```

```r
optimize(loglik, c(0, 1e+100), maximum = TRUE, x = X, y = Y)
```

```
## $maximum
## [1] 1.975825
##
## $objective
## [1] -194.959
```

```r
curve(loglik(x, X, Y), xlab = NA, ylab = NA, xlim = c(1, 3), lwd = 2)
abline(v = lambda, col = 2, lty = 2, lwd = 2)
abline(v = MLE, col = 4, lty = 2, lwd = 2)
legend("bottomright", c("True Parameter", "MLE"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```



Example 1.3. Let $X_1, X_2, \ldots, X_n$ be a random sample with PDF $f(x;\vartheta) = \frac{1}{\vartheta}e^{-(x-\vartheta)/\vartheta}$ for $x \geqslant \vartheta$ and $\vartheta > 0$. Then, we know that:

$$\ell(\vartheta \mid x) = \begin{cases} \vartheta^{-n}e^{-n\overline{x}/\vartheta+n}, & \vartheta \leqslant x_{(1)} \\ 0, & \vartheta > x_{(1)} \end{cases}, \quad \widehat{\vartheta}(X) = X_{(1)}.$$

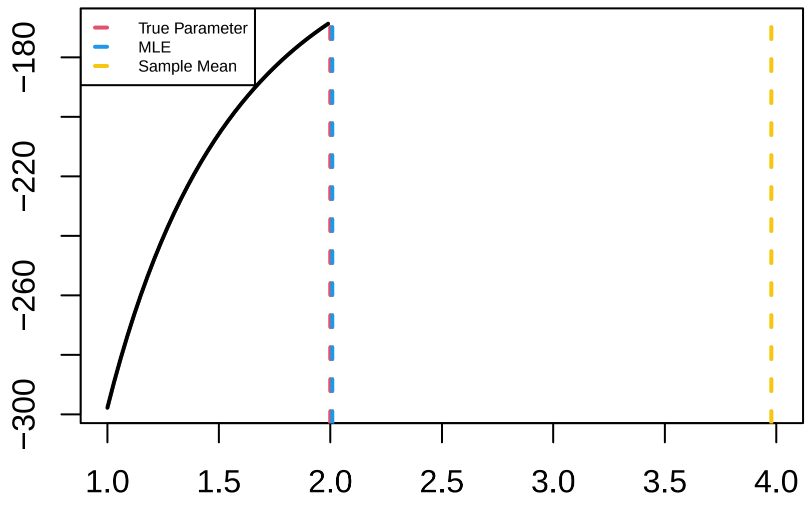
```r
loglik = function(theta, x) {
    ifelse(theta < min(x), length(x) * (1 - log(theta)) - sum(x)/theta,
        -Inf)
}

theta = 2
X = rexp(n, theta^(-1)) + theta
MLE = min(X)
print(MLE)
```

```
## [1] 2.009003
```

```r
optimize(loglik, c(0, 5), maximum = TRUE, x = X)
```

```
## $maximum
## [1] 2.00897
##
## $objective
## [1] -167.7648
```

```r
curve(loglik(x, X), xlab = NA, ylab = NA, xlim = c(1, 4), lwd = 2)
abline(v = theta, col = 2, lty = 2, lwd = 2)
abline(v = MLE, col = 4, lty = 2, lwd = 2)
abline(v = mean(X), col = 7, lty = 2, lwd = 2)
legend("topleft", c("True Parameter", "MLE", "Sample Mean"), col = c(2,
    4, 7), lty = rep(2, 3), lwd = rep(2, 3), cex = 0.5)
```



Example 1.4. Let $X_1, X_2, \ldots, X_n$ be a random sample with PDF $f(x;\vartheta) = e^{-(x-\vartheta)}$ for $x \geqslant \vartheta$ and $\vartheta < 0$. We want to estimate the parametric function $g(\vartheta) = \mathbb{P}_\vartheta(X_1 < 0) = 1 - e^{\vartheta}$. Suppose that we only observe the values of the random variable $W = \sum_{i=1}^n \mathbb{1}_{[\vartheta,0)}(X_i)$ and those of the random variables $X_1, X_2, \ldots, X_n$ which are negative. Then, we know that:

$$\ell(\vartheta \mid x, w) = \begin{cases} \binom{n}{w} e^{n\vartheta} \exp\left\{-\sum_{i:\, x_i<0} x_i\right\}, & \vartheta \leqslant x_{(1)} \\ 0, & \vartheta > x_{(1)} \end{cases}, \quad \widehat{g(\vartheta)} = 1 - e^{\min\{X_{(1)},0\}}.$$

```r
loglik = function(theta, n, x, w) {
    ifelse(theta < min(x), lfactorial(n) - lfactorial(w) - lfactorial(n -
        w) + n * theta - sum(x), -Inf)
}

n = 1000
theta = -1
print(1 - exp(theta))
```

```
## [1] 0.6321206
```

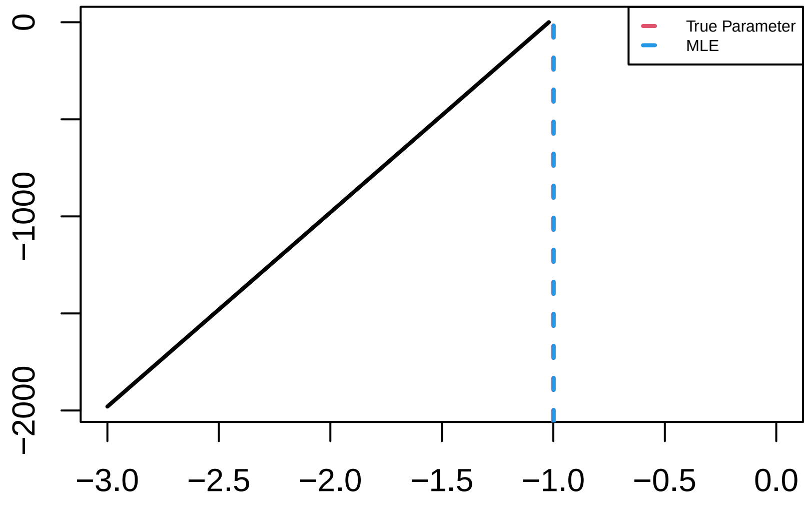
```
X = rexp(n, 1) + theta
W = sum(X < 0)
MLE = 1 - exp(min(min(X), 0))
print(MLE)
```

```
## [1] 0.6314108
```

```
opt = optimize(loglik, c(-4, 1), maximum = TRUE, n = n, x = X, w = W)
print(1 - exp(opt$maximum))
```

```
## [1] 0.6314346
```

```
curve(loglik(x, n, X[X < 0], W), xlab = NA, ylab = NA, xlim = c(-3, 0),
    lwd = 2)
abline(v = theta, col = 2, lty = 2, lwd = 2)
abline(v = log(1 - MLE), col = 4, lty = 2, lwd = 2)
legend("topright", c("True Parameter", "MLE"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```



Example 1.5. In the setting of the previous example, suppose that we only observe the values of the random variable W and those of the random variables $X_1, X_2, \ldots, X_n$ which are positive. Then, we know that:

$$\ell(\vartheta \mid x, w) = \binom{n}{w} [g(\vartheta)]^w [1 - g(\vartheta)]^{n-w} \exp\left\{ - \sum_{i:\, x_i > 0} x_i \right\}, \quad \widehat{g(\vartheta)} = \frac{1}{n} W.$$

We observe that the estimate of the previous example is much closer to the true value of the parametric function than that of this example.

```
loglik = function(theta, n, x, w) {
    lfactorial(n) - lfactorial(w) - lfactorial(n - w) + w * log(1 - exp(theta)) +
        (n - w) * theta - sum(x)
}
```

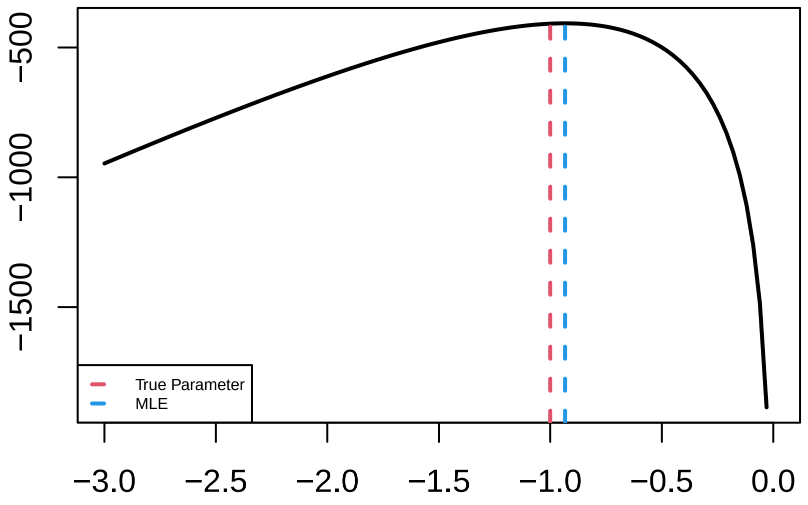
```
MLE = W/n
print(MLE)
```

```
## [1] 0.607
```

```
opt = optimize(loglik, c(-4, 1), maximum = TRUE, n = n, x = X, w = W)
print(1 - exp(opt$maximum))
```

```
## [1] 0.6069996
```

```
curve(loglik(x, n, X[X > 0], W), xlab = NA, ylab = NA, xlim = c(-3, 0),
    lwd = 2)
abline(v = theta, col = 2, lty = 2, lwd = 2)
abline(v = log(1 - MLE), col = 4, lty = 2, lwd = 2)
legend("bottomleft", c("True Parameter", "MLE"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```



Example 1.6. Let $X_1, \dots, X_n \sim \mathcal{N}(\vartheta, \vartheta)$ be a random sample with $\vartheta > 0$. Then, we know that:

$$\ell(\vartheta \mid x) = -\frac{n}{2}\log(2\pi\vartheta) - \frac{1}{2\vartheta}\sum_{i=1}^{n}(x_i - \vartheta)^2, \quad \widehat{\vartheta}(X) = \frac{1}{2}\sqrt{1 + \frac{4}{n}\sum_{i=1}^{n} X_i^2} - \frac{1}{2}.$$

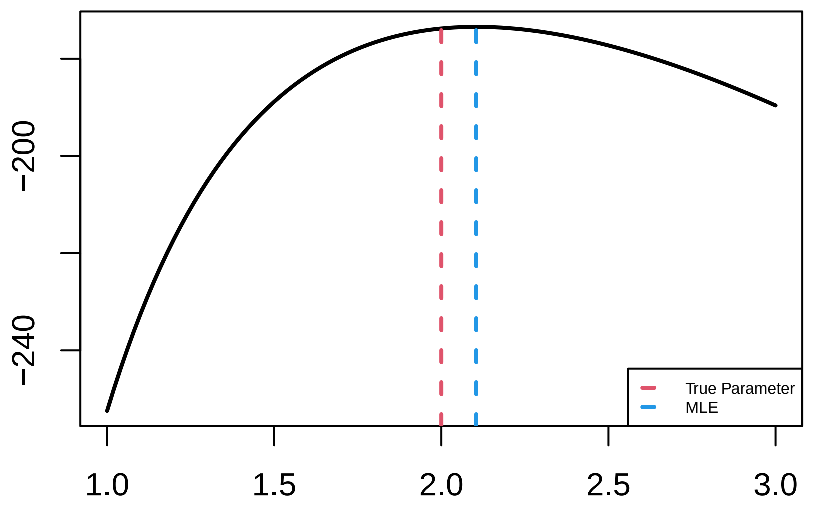
```
loglik = function(theta, x) {
    -log(2 * pi * theta) * length(x)/2 - colSums(outer(x, theta, "-")^2)/(2 *
        theta)
}

n = 100
theta = 2
X = rnorm(n, theta, sqrt(theta))
MLE = (sqrt(1 + 4 * mean(X^2)) - 1)/2
print(MLE)
```

```
## [1] 2.104385
```

```r
optimize(loglik, c(0, 1e+100), maximum = TRUE, x = X)
```

```
## $maximum
## [1] 2.104386
##
## $objective
## [1] -173.444
```

```r
curve(loglik(x, X), xlab = NA, ylab = NA, xlim = c(1, 3), lwd = 2)
abline(v = theta, col = 2, lty = 2, lwd = 2)
abline(v = MLE, col = 4, lty = 2, lwd = 2)
legend("bottomright", c("True Parameter", "MLE"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```



For distribution families with more than one unknown parameter, we can make use of R's built-in optim function in order to maximize the log-likelihood function. Optimization should be performed with respect to transformations of the parameters, so that all transformed parameters take values on the entire real line. For example, a parameter $\lambda > 0$ should be transformed to $\vartheta = \log \lambda \in \mathbb{R}$ and a parameter $p \in (0, 1)$ should be transformed to $\vartheta = \operatorname{logit} p = \log \frac{p}{1-p}$.

Example 1.7. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu, \sigma^2\right)$ be a random sample. Then, we know that:

$$\ell\left(\mu, \sigma^2 \mid x\right) = -\frac{n}{2} \log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (x_i - \mu)^2, \quad \widehat{\mu}(X) = \overline{X}, \quad \widehat{\sigma}^2(X) = \frac{1}{n} \sum_{i=1}^{n} (X_i - \overline{X})^2.$$

```r
loglik = function(param, x) {
    mu = param[1]
    sigma = exp(param[2])
    -log(2 * pi * sigma^2) * length(x)/2 - sum((x - mu)^2)/(2 * sigma^2)
}

n = 10000
mu = 1
sigma = 2
```

```r
X = rnorm(n, mu, sigma)
MLE = c(mean(X), mean((X - mean(X))^2))
print(MLE)
```

```
## [1] 1.019362 3.990510
```

```r
opt = optim(c(0, 0), loglik, x = X, control = list(fnscale = -1))
c(opt$par[1], exp(2 * opt$par[2]))
```

```
## [1] 1.019041 3.991276
```

Example 1.8. Let $(X_1, Y_1), \ldots, (X_n, Y_n)$ be a random sample with $X_1 \sim \text{Exp}(\lambda)$ and $(Y_1 \mid X_1 = x) \sim \text{Poisson}(\mu x)$ for $\mu > 0$ and $x > 0$. Then, we know that:

$$\ell(\lambda, \mu \mid x, y) = n \log \lambda - (\lambda + \mu) \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i \log \mu + \sum_{i=1}^{n} y_i \log x_i - \sum_{i=1}^{n} \log y_i!,$$

$$\widehat{\lambda}(X, Y) = \frac{1}{\overline{X}}, \quad \widehat{\mu}(X, Y) = \frac{\overline{Y}}{\overline{X}}.$$

```r
loglik = function(param, x, y) {
    lambda = exp(param[1])
    mu = exp(param[2])
    length(x) * log(lambda) - (lambda + mu) * sum(x) + sum(y) * log(mu) +
        sum(y * log(x)) - sum(lfactorial(y))
}

lambda = 2
mu = 3
X = rexp(n, lambda)
Y = rpois(n, mu * X)
MLE = c(1/mean(X), mean(Y)/mean(X))
print(MLE)
```

```
## [1] 2.033974 3.019637
```

```r
opt = optim(c(0, 0), loglik, x = X, y = Y, control = list(fnscale = -1))
exp(opt$par)
```

```
## [1] 2.034212 3.019866
```

Example 1.9. Let $X_1, X_2, \ldots, X_n$ be a random sample with PDF $f(x; \lambda, \mu) = \lambda e^{-\lambda(x-\mu)}$ for $x \geqslant \mu$, $\lambda > 0$ and $\mu \in \mathbb{R}$. Then, we know that:

$$\ell(\lambda, \mu \mid x) = \begin{cases} \lambda^n e^{-n\lambda\overline{x} + n\lambda\mu}, & \mu \leqslant x_{(1)} \\ 0, & \mu > x_{(1)} \end{cases}, \quad \widehat{\lambda}(X) = \frac{1}{\overline{X} - X_{(1)}}, \quad \widehat{\mu}(X) = X_{(1)}.$$

```r
loglik = function(param, x) {
    mu = param[1]
    lambda = exp(param[2])
    ifelse(mu < min(x), length(x) * (log(lambda) + mu * lambda) - lambda *
        sum(x), -Inf)
}

mu = 1
X = rexp(n, lambda) + mu
MLE = c(min(X), 1/(mean(X) - min(X)))
print(MLE)
```

```
## [1] 1.000042 1.987717
```

```r
opt = optim(c(0, 0), loglik, x = X, control = list(fnscale = -1))
c(opt$par[1], exp(opt$par[2]))
```

```
## [1] 1.000042 1.987686
```

Example 1.10. Let $X_1, X_2, \ldots, X_n \sim \text{Laplace}(\mu, \lambda)$ be a random sample with PDF $f(x; \mu, \lambda) = \frac{\lambda}{2} e^{-\lambda|x-\mu|}$ for $\mu \in \mathbb{R}$, $\lambda > 0$ and $x \in \mathbb{R}$. Then, we know that:

$$\ell(\mu, \lambda \mid x) = n \log \frac{\lambda}{2} - \lambda \sum_{i=1}^{n} |x_i - \mu|, \quad \widehat{\mu}(X) = \text{median}(X), \quad \widehat{\lambda}(X) = \frac{n}{\sum_{i=1}^{n} |X_i - \text{median}(X)|}.$$

```r
loglik = function(param, x) {
    mu = param[1]
    lambda = exp(param[2])
    length(x) * log(lambda/2) - lambda * sum(abs(x - mu))
}

X = (2 * rbinom(n, 1, 0.5) - 1) * rexp(n, lambda) + mu
MLE = c(median(X), 1/mean(abs(X - median(X))))
print(MLE)
```

```
## [1] 1.002650 1.974653
```

```r
opt = optim(c(0, 0), loglik, x = X, control = list(fnscale = -1))
c(opt$par[1], exp(opt$par[2]))
```

```
## [1] 1.002578 1.974500
```

Example 1.11. Let $X_1, X_2, \ldots, X_n$ be a random sample with the following PDF:

$$f(x; p, \lambda) = \begin{cases} p\lambda e^{-\lambda x}, & x > 0 \\ (1-p)\lambda e^{\lambda x}, & x \leqslant 0 \end{cases}, \quad x \in \mathbb{R}, \quad p \in (0,1), \quad \lambda > 0.$$

Then, we know that:

$$\ell(p, \lambda \mid x) = \sum_{i=1}^n \mathbb{1}_{(0,\infty)}(x_i) \log p + \left[n - \sum_{i=1}^n \mathbb{1}_{(0,\infty)}(x_i) \right] \log(1-p) + n \log \lambda - \lambda \sum_{i=1}^n |x_i|,$$

$$\widehat{p}(X) = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{(0,\infty)}(X_i), \quad \widehat{\lambda}(X) = \frac{1}{\overline{X}}.$$

```
loglik = function(param, x) {
    p = (1 + exp(-param[1]))^(-1)
    lambda = exp(param[2])
    sum(x > 0) * (log(p) - log(1 - p)) + length(x) * (log(1 - p) + log(lambda)) -
        lambda * sum(abs(x))
}


p = 0.3
X = (2 * rbinom(n, 1, p) - 1) * rexp(n, lambda)
MLE = c(mean(X > 0), 1/mean(abs(X)))
print(MLE)
```

```
## [1] 0.300500 2.009993
```

```
opt = optim(c(0, 0), loglik, x = X, control = list(fnscale = -1))
c((1 + exp(-opt$par[1]))^(-1), exp(opt$par[2]))
```

```
## [1] 0.3005525 2.0099568
```

Example 1.12. Let $X_1, \dots, X_n \sim \text{Exp}(\lambda)$ and $Y_1, \dots, Y_n \sim \text{Exp}(\mu)$ be 2 independent random variables. Suppose that we observe the following random variables:

$$Z_i = \min\{X_i, Y_i\}, \quad W_i = \begin{cases} 1, & Z_i = X_i \\ 0, & Z_i = Y_i \end{cases}, \quad i = 1, 2, \dots, n.$$

Then, we know that:

$$\ell(\lambda, \mu \mid z, w) = -(\lambda + \mu) \sum_{i=1}^n z_i + \sum_{i=1}^n w_i \log \lambda + \left(n - \sum_{i=1}^n w_i \right) \log \mu,$$

$$\widehat{\lambda}(Z, W) = \frac{\overline{W}}{\overline{Z}}, \quad \widehat{\mu}(Z, W) = \frac{1 - \overline{W}}{\overline{Z}}.$$

```
loglik = function(param, z, w) {
    lambda = exp(param[1])
    mu = exp(param[2])
    -(lambda + mu) * sum(z) + sum(w) * (log(lambda) - log(mu)) + n * log(mu)
}


mu = 3
```

```r
X = rexp(n, lambda)
Y = rexp(n, mu)
Z = apply(cbind(X, Y), 1, min)
W = 2 - apply(cbind(X, Y), 1, which.min)
MLE = c(mean(X), mean(Y))^(-1)
print(MLE)
```

```
## [1] 1.997255 2.995005
```

```r
MLE = c(sum(W)/sum(Z), (n - sum(W))/sum(Z))
print(MLE)
```

```
## [1] 2.010357 3.011769
```

```r
opt = optim(c(0, 0), loglik, z = Z, w = W, control = list(fnscale = -1))
exp(opt$par)
```

```
## [1] 2.010136 3.011512
```

Example 1.13. Let $X_1, \dots, X_n \sim \text{Gamma}\,(\alpha, \lambda)$ be a random sample. Then, we know that:

$$\ell\,(\alpha, \lambda \mid x) = n\alpha \log \lambda - n \log \Gamma(\alpha) + (\alpha - 1) \sum_{i=1}^{n} \log x_i - \lambda \sum_{i=1}^{n} x_i,$$

but there's no closed form solution to the maximization problem when the parameter α is unknown.

```r
loglik = function(param, x) {
    alpha = exp(param[1])
    lambda = exp(param[2])
    length(x) * (alpha * log(lambda) - lgamma(alpha)) + (alpha - 1) * sum(log(x)) -
        lambda * sum(x)
}

alpha = 3
X = rgamma(n, alpha, lambda)
opt = optim(c(0, 0), loglik, x = X, control = list(fnscale = -1))
exp(opt$par)
```

```
## [1] 3.052034 2.059584
```

Example 1.14. Let $X_1, \dots, X_n \sim \text{Beta}\,(\alpha, \beta)$ be a random sample. Then, we know that:

$$\ell\,(\alpha, \beta \mid x) = -n \log B(\alpha, \beta) + (\alpha - 1) \sum_{i=1}^{n} \log x_i + (\beta - 1) \sum_{i=1}^{n} \log(1 - x_i),$$

but there's no closed form solution to the maximization problem.

```r
loglik = function(param, x) {
    alpha = exp(param[1])
    beta = exp(param[2])
    -length(x) * lbeta(alpha, beta) + (alpha - 1) * sum(log(x)) + (beta -
        1) * sum(log(1 - x))
}

beta = 2
X = rbeta(n, alpha, beta)
opt = optim(c(0, 0), loglik, x = X, control = list(fnscale = -1))
exp(opt$par)
```

```
## [1] 2.992604 1.997590
```

Mean Squared Error

We want to empirically compare the bias, the variance and the mean squared error of different estimators of the same parameter ϑ. In order to achieve that, we have to generate n_{sim} independent random samples following the same distribution and compute the value $\widehat{\vartheta}^{(k)}$ of our candidate estimator $\widehat{\vartheta}$ for each of the generated samples. Then, we can estimate the bias, the variance and the MSE of $\widehat{\vartheta}$ as follows:

$$\widehat{\text{Bias}}\left(\widehat{\vartheta}\right) = \frac{1}{n_{\text{sim}}} \sum_{k=1}^{n_{\text{sim}}} \widehat{\vartheta}^{(k)} - \vartheta,$$

$$\widehat{\text{Var}}\left(\widehat{\vartheta}\right) = \frac{1}{n_{\text{sim}}} \sum_{k=1}^{n_{\text{sim}}} \left[\widehat{\vartheta}^{(k)} - \frac{1}{n_{\text{sim}}} \sum_{\ell=1}^{n_{\text{sim}}} \widehat{\vartheta}^{(\ell)}\right]^2,$$

$$\widehat{\text{MSE}}\left(\widehat{\vartheta}\right) = \frac{1}{n_{\text{sim}}} \sum_{k=1}^{n_{\text{sim}}} \left[\widehat{\vartheta}^{(k)} - \vartheta\right]^2.$$

Additionally, we can calculate the Cramér - Rao lower bound for an unbiased estimator of the parameter ϑ and compare it against the empirical variance of any of our unbiased estimators. Finally, we can plot histograms of the computed values of our candidate estimators, in order to get a better sense of how their values are distributed around the true value of the unknown parameter ϑ and how the distributions of different estimators of the same parameter compare against each other.

Example 1.15. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu, \sigma^2\right)$ be a random sample with known μ. Then, we know that the MLE $\widehat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \mu)^2$ of σ^2 is also the UMVUE of σ^2 and an efficient estimator of σ^2 with $\frac{n}{\sigma^2}\widehat{\sigma}^2 \sim \chi^2_n$, whereas the sample variance $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2$ is another unbiased estimator of σ^2 with larger variance than that of $\widehat{\sigma}^2$ and $\frac{n-1}{\sigma^2}S^2 \sim \chi^2_{n-1}$.

We observe that the estimated bias of both estimators is close to 0. The estimated variance of the MLE is approximately equal to the Cramér - Rao lower bound and smaller than that of the sample variance. The distributions of the 2 estimators almost coincide even for a sample size of $n = 10$ observations, but the distribution of the sample variance displays slightly higher variation than that of the MLE.

```r
library(xtable)
n = 10
nsim = 10000
```

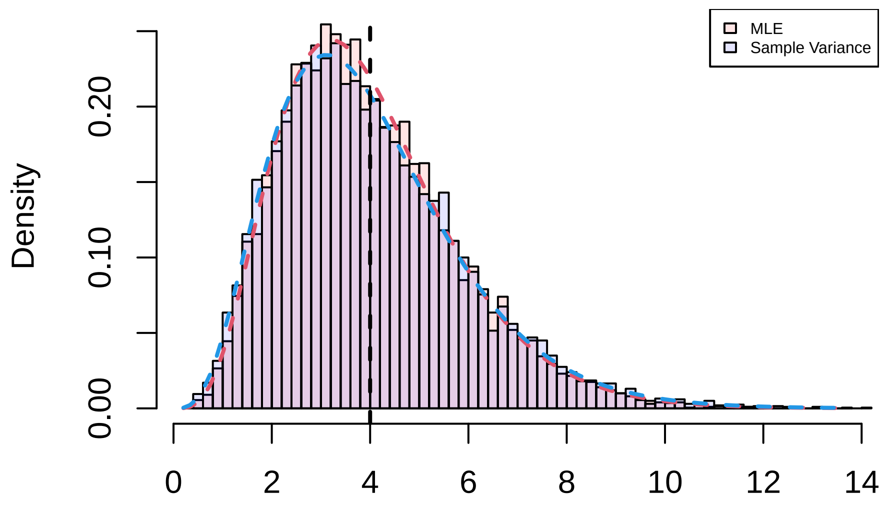
```r
mu = 1
sigma = 2
X = matrix(rnorm(n * nsim, mu, sigma), n)
MLE = colMeans((X - mu)^2)
UE = apply(X, 2, var)
mse = matrix(0, 2, 3)
rownames(mse) = c("MLE", "Sample Variance")
colnames(mse) = c("Bias", "Variance", "MSE")
mse[1, 1] = mean(MLE) - sigma^2
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - sigma^2)^2)
mse[2, 1] = mean(UE) - sigma^2
mse[2, 2] = mean((UE - mean(UE))^2)
mse[2, 3] = mean((UE - sigma^2)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	0.0082	3.1363	3.1364
Sample Variance	0.0150	3.5000	3.5002

```r
CRLB = 2 * sigma^4/n
print(CRLB)
```

```
[1] 3.2
```

```r
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), main = NA, xlab = NA)
hist(UE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), add = TRUE)
curve(dchisq(x * n/sigma^2, n) * n/sigma^2, add = TRUE, col = 2, lty = 2,
    lwd = 2)
curve(dchisq(x * (n - 1)/sigma^2, n - 1) * (n - 1)/sigma^2, add = TRUE,
    col = 4, lty = 2, lwd = 2)
abline(v = sigma^2, lty = 2, lwd = 2)
legend("topright", c("MLE", "Sample Variance"), fill = c(rgb(1, 0, 0, 0.1),
    rgb(0, 0, 1, 0.1)), cex = 0.5)
```



Example 1.16. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu, \sigma^2\right)$ be a random sample with unknown μ. Then, we know that the MLE $\widehat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X})^2$ is a biased estimator of σ^2 with $\frac{n}{\sigma^2}\widehat{\sigma}^2 \sim \chi^2_{n-1}$ and smaller MSE than the sample variance $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2$, which is the UMVUE but not an efficient estimator of σ^2 with $\frac{n-1}{\sigma^2}S^2 \sim \chi^2_{n-1}$.

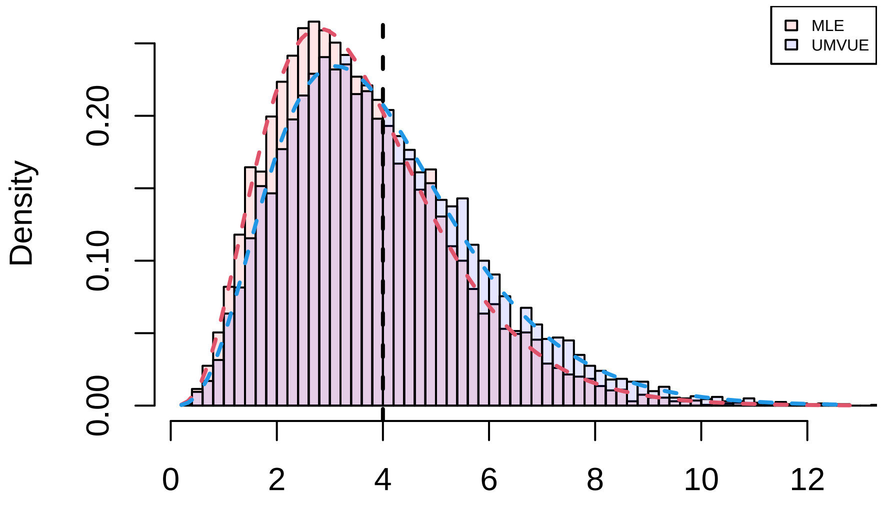
We observe that the estimated bias of the UMVUE is close to 0, whereas the MLE tends to underestimate the true value of σ^2. The estimated variance of the UMVUE is larger than both the estimated variance of the MLE and the Cramér - Rao lower bound. The estimated MSE of the MLE is smaller than that of the sample variance. Looking at the observed distributions of the two estimators, the MLE tends to take smaller values than the true value of σ^2 on average, whereas the distribution of the sample variance displays higher variation than that of the MLE.

```r
MLE = colMeans(t(t(X) - colMeans(X))^2)
UMVUE = apply(X, 2, var)
rownames(mse) = c("MLE", "UMVUE")
mse[1, 1] = mean(MLE) - sigma^2
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - sigma^2)^2)
mse[2, 1] = mean(UMVUE) - sigma^2
mse[2, 2] = mean((UMVUE - mean(UMVUE))^2)
mse[2, 3] = mean((UMVUE - sigma^2)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	-0.3865	2.8350	2.9844
UMVUE	0.0150	3.5000	3.5002

```r
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), main = NA, xlab = NA)
hist(UMVUE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), add = TRUE)
curve(dchisq(x * n/sigma^2, n - 1) * n/sigma^2, add = TRUE, col = 2, lty = 2,
    lwd = 2)
curve(dchisq(x * (n - 1)/sigma^2, n - 1) * (n - 1)/sigma^2, add = TRUE,
    col = 4, lty = 2, lwd = 2)
abline(v = sigma^2, lty = 2, lwd = 2)
```

```
legend("topright", c("MLE", "UMVUE"), fill = c(rgb(1, 0, 0, 0.1), rgb(0,
    0, 1, 0.1)), cex = 0.5)
```



Example 1.17. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu, \sigma^2\right)$ be a random sample with known σ^2. Then, we are aware that $e^{\overline{X}} \sim \text{Lognormal}\left(\mu, \frac{1}{n}\sigma^2\right)$ is the MLE of of $g(\mu) = e^{\mu}$, whereas $e^{\overline{X} - \sigma^2/2n} \sim \text{Lognormal}\left(\mu - \frac{1}{2n}\sigma^2, \frac{1}{n}\sigma^2\right)$ is the UMVUE of e^{μ}.

We observe that the estimated bias of the UMVUE is close to 0, whereas the MLE tends to overestimate the true value of e^{μ}. The estimated variance of the UMVUE is larger than the Cramér - Rao lower bound but much smaller than that of the MLE. The observed distribution of the MLE accordingly displays much higher variation than that of the UMVUE.

```
MLE = exp(colMeans(X))
UMVUE = exp(colMeans(X) - sigma^2/(2 * n))
mse[1, 1] = mean(MLE) - exp(mu)
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - exp(mu))^2)
mse[2, 1] = mean(UMVUE) - exp(mu)
mse[2, 2] = mean((UMVUE - mean(UMVUE))^2)
mse[2, 3] = mean((UMVUE - exp(mu))^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

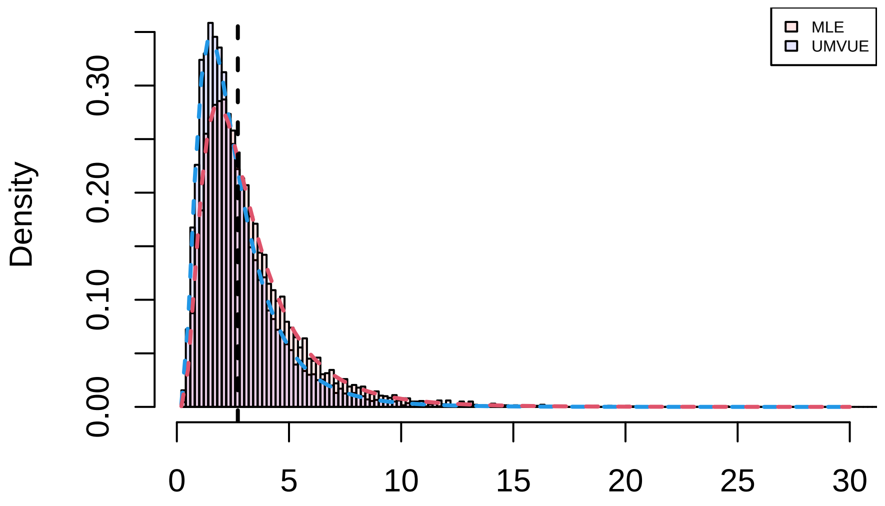
	Bias	Variance	MSE
MLE	0.6084	5.5495	5.9196
UMVUE	0.0053	3.7199	3.7200

```
CRLB = exp(2 * mu) * sigma^2/n
print(CRLB)
```

```
[1] 2.955622
```

```
hist(UMVUE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), main = NA, xlab = NA)
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), add = TRUE)
```

```r
curve(dlnorm(x, mu - sigma^2/(2 * n), sigma/sqrt(n)), add = TRUE, col = 4,
    lty = 2, lwd = 2)
curve(dlnorm(x, mu, sigma/sqrt(n)), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = exp(mu), lty = 2, lwd = 2)
legend("topright", c("MLE", "UMVUE"), fill = c(rgb(1, 0, 0, 0.1), rgb(0,
    0, 1, 0.1)), cex = 0.5)
```



Example 1.18. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu, \sigma^2\right)$ be a random sample with unknown σ^2. Then, we know that $\overline{X}^2$ is the MLE of of $g(\mu) = \mu^2$, whereas $\overline{X}^2 - \frac{1}{n}S^2$ is the UMVUE of μ^2.

We observe that the estimated bias of the UMVUE is close to 0, whereas the MLE tends to overestimate the true value of μ^2. The estimated variance of the UMVUE is larger than both the estimated variance of the MLE and the Cramér - Rao lower bound, but the estimated MSE of the UMVUE is still smaller than that of the MLE.

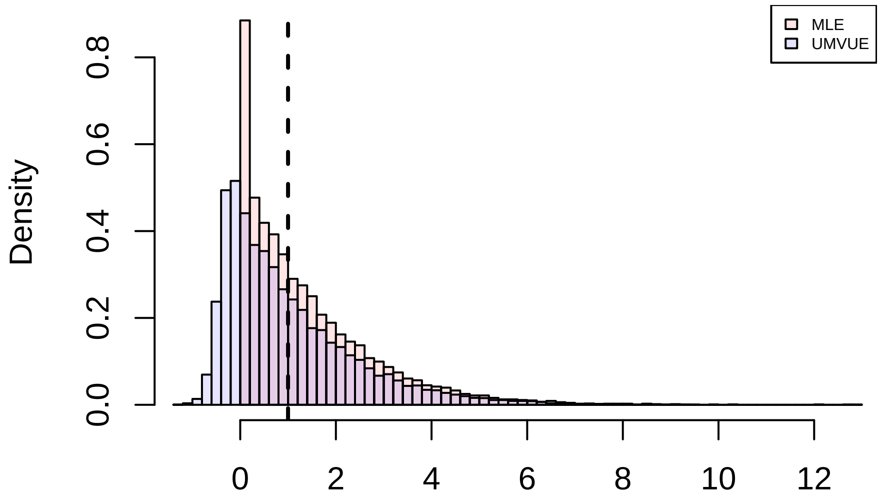
```r
MLE = colMeans(X)^2
UMVUE = colMeans(X)^2 - apply(X, 2, var)/n
mse[1, 1] = mean(MLE) - mu^2
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - mu^2)^2)
mse[2, 1] = mean(UMVUE) - mu^2
mse[2, 2] = mean((UMVUE - mean(UMVUE))^2)
mse[2, 3] = mean((UMVUE - mu^2)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	0.4004	1.9430	2.1033
UMVUE	-0.0011	1.9759	1.9759

```r
CRLB = 4 * mu^2 * sigma^2/n
print(CRLB)
```

```
[1] 1.6
```

```r
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), main = NA, xlim = range(UMVUE),
    xlab = NA)
hist(UMVUE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), add = TRUE)
abline(v = mu^2, lty = 2, lwd = 2)
legend("topright", c("MLE", "UMVUE"), fill = c(rgb(1, 0, 0, 0.1), rgb(0,
    0, 1, 0.1)), cex = 0.5)
```



Example 1.19. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu, \sigma^2\right)$ be a random sample. Then, we know that $\sqrt{\frac{n}{n-1}}\frac{\overline{X}}{S}$ is the MLE of $g\left(\mu, \sigma^2\right) = \frac{\mu}{\sigma}$, whereas $\sqrt{\frac{2}{n-1}}\frac{\Gamma\left(\frac{n-1}{2}\right)}{\Gamma\left(\frac{n-2}{2}\right)}\frac{\overline{X}}{S}$ is the UMVUE of $\frac{\mu}{\sigma}$.

We observe that the estimated bias of the UMVUE is close to 0, whereas the MLE tends to overestimate the true value of $\frac{\mu}{\sigma}$. The estimated variance of the UMVUE is larger than the Cramér - Rao lower bound but smaller than that of the MLE. The observed distribution of the MLE accordingly displays much higher variation than that of the UMVUE.

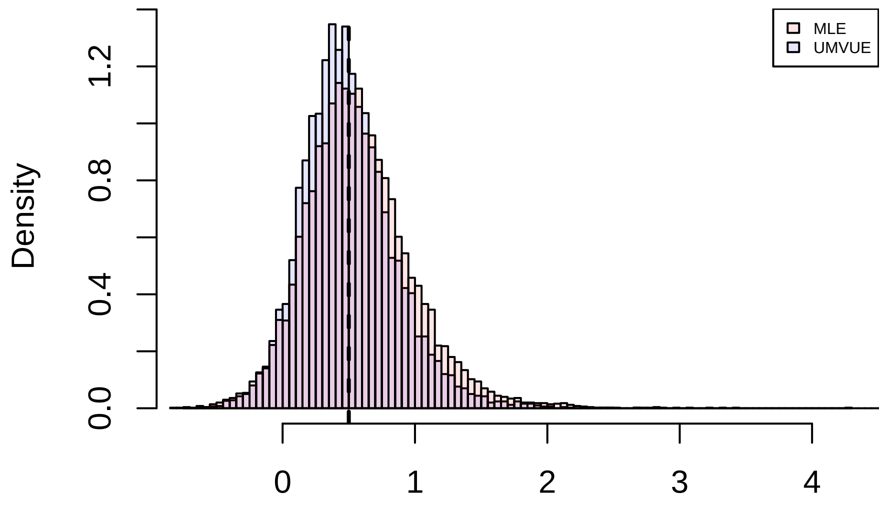
```r
MLE = sqrt(n/(n - 1)) * colMeans(X)/apply(X, 2, sd)
UMVUE = sqrt(2/(n - 1)) * gamma((n - 1)/2)/gamma((n - 2)/2) * colMeans(X)/apply(X,
    2, sd)
mse[1, 1] = mean(MLE) - mu/sigma
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - mu/sigma)^2)
mse[2, 1] = mean(UMVUE) - mu/sigma
mse[2, 2] = mean((UMVUE - mean(UMVUE))^2)
mse[2, 3] = mean((UMVUE - mu/sigma)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	0.0767	0.1671	0.1729
UMVUE	-0.0000	0.1256	0.1256

```
CRLB = (mu^2 + 2 * sigma^2)/(2 * n * sigma^2)
print(CRLB)
```

[1] 0.1125

```
hist(UMVUE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), main = NA, xlab = NA)
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), add = TRUE)
abline(v = mu/sigma, lty = 2, lwd = 2)
legend("topright", c("MLE", "UMVUE"), fill = c(rgb(1, 0, 0, 0.1), rgb(0,
    0, 1, 0.1)), cex = 0.5)
```



Example 1.20. Let $X_1, \ldots, X_n \sim \text{Exp}(\lambda)$ be a random sample. Then, we know that $\widehat{\lambda} = \frac{1}{\overline{X}} \sim \text{Inv-Gamma}\,(n, n\lambda)$ is the MLE of λ, whereas $\frac{n-1}{n\overline{X}} \sim \text{Inv-Gamma}\,(n, (n-1)\lambda)$ is the UMVUE of λ.

We observe that the estimated bias of the UMVUE is close to 0, whereas the MLE tends to overestimate the true value of λ. The estimated variance of the UMVUE is larger than the Cramér - Rao lower bound but smaller than that of the MLE. The observed distribution of the MLE accordingly displays higher variation than that of the UMVUE.

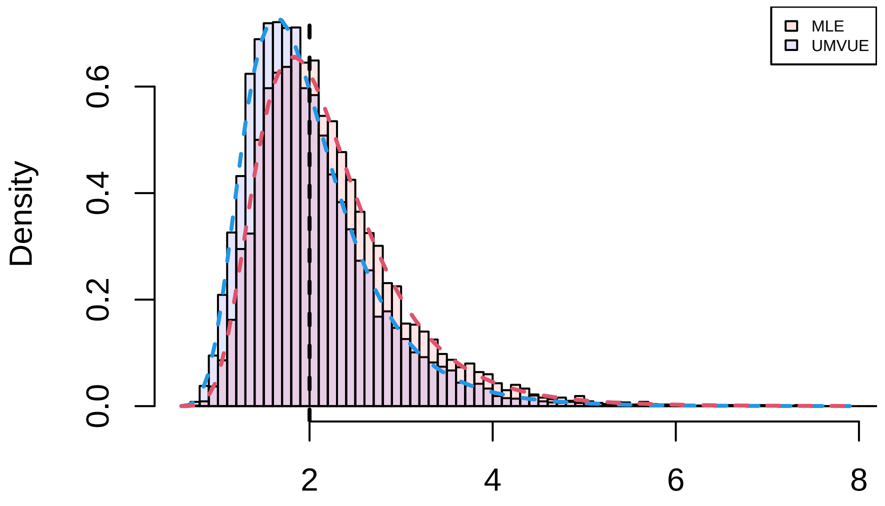
```
lambda = 2
X = matrix(rexp(n * nsim, lambda), n)
MLE = 1/colMeans(X)
UMVUE = (n - 1)/colSums(X)
mse[1, 1] = mean(MLE) - lambda
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - lambda)^2)
mse[2, 1] = mean(UMVUE) - lambda
mse[2, 2] = mean((UMVUE - mean(UMVUE))^2)
mse[2, 3] = mean((UMVUE - lambda)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	0.2218	0.6081	0.6572
UMVUE	-0.0004	0.4925	0.4925

```r
CRLB = lambda^2/n
print(CRLB)
```

```
[1] 0.4
```

```r
hist(UMVUE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), main = NA, xlab = NA)
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), add = TRUE)
curve(dgamma(x^(-1), n, (n - 1) * lambda)/x^2, add = TRUE, col = 4, lty = 2,
    lwd = 2)
curve(dgamma(x^(-1), n, n * lambda)/x^2, add = TRUE, col = 2, lty = 2,
    lwd = 2)
abline(v = lambda, lty = 2, lwd = 2)
legend("topright", c("MLE", "UMVUE"), fill = c(rgb(1, 0, 0, 0.1), rgb(0,
    0, 1, 0.1)), cex = 0.5)
```



Example 1.21. Let $X_1, \dots, X_n \sim \text{Laplace}(\mu, \lambda)$ be a random sample with PDF $f(x; \mu, \lambda) = \frac{\lambda}{2} e^{-\lambda|x-\mu|}$ for $x \in \mathbb{R}$, $\mu \in \mathbb{R}$ and $\lambda > 0$. Then, we know that $\widehat{\mu} = \text{median}(X)$ is the MLE of μ, whereas the sample mean $\overline{X}$ is an unbiased estimator of μ.

We observe that the estimated bias of both estimators is close to 0. The estimated variance of the MLE is larger than the Cramér - Rao lower bound but smaller than that of the sample mean. The observed distribution of the sample mean accordingly displays much higher variation than that of the MLE.

```r
mu = 1
X = matrix((2 * rbinom(n * nsim, 1, 0.5) - 1) * rexp(n * nsim, lambda),
    n) + mu
MLE = apply(X, 2, median)
UE = colMeans(X)
rownames(mse) = c("MLE", "Unbiased Estimator")
```

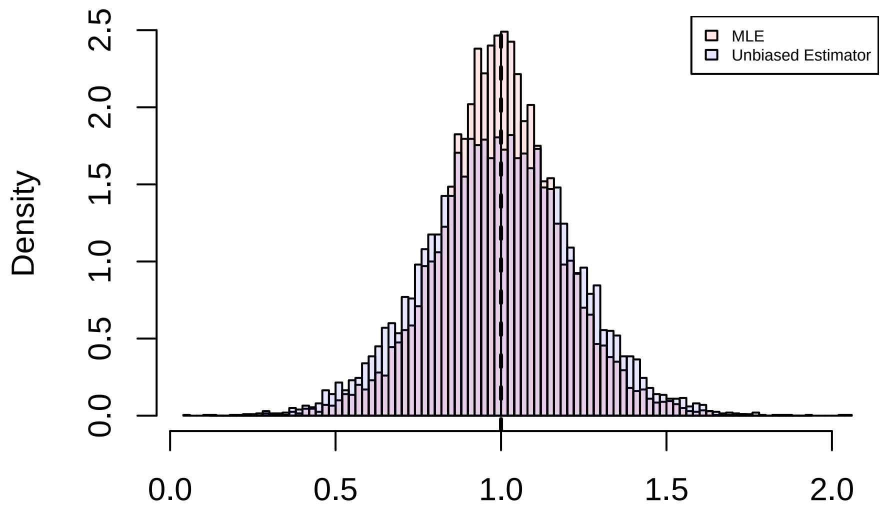
```r
mse[1, 1] = mean(MLE) - mu
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - mu)^2)
mse[2, 1] = mean(UE) - mu
mse[2, 2] = mean((UE - mean(UE))^2)
mse[2, 3] = mean((UE - mu)^2)
CRLB = 1/(n * lambda^2)
print(CRLB)
```

```
[1] 0.025
```

```r
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	-0.0029	0.0354	0.0354
Unbiased Estimator	-0.0028	0.0497	0.0497

```r
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), main = NA, xlab = NA)
hist(UE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), add = TRUE)
abline(v = mu, lty = 2, lwd = 2)
legend("topright", c("MLE", "Unbiased Estimator"), fill = c(rgb(1, 0, 0,
    0.1), rgb(0, 0, 1, 0.1)), cex = 0.5)
```



Example 1.22. Let $X_1, \ldots, X_n \sim \text{Poisson}(\lambda)$ be a random sample. Then, we know that the MLE $\widehat{\lambda} = \overline{X}$ of λ is also the UMVUE of λ and an efficient estimator of λ, whereas the sample variance S^2 is another unbiased estimator of λ with larger variance than that of $\widehat{\lambda}$.

We observe that the estimated bias of both estimators is close to 0. The estimated variance of the MLE is approximately equal to the Cramér - Rao lower bound and smaller than that of the sample variance. The observed distribution of the sample variance accordingly displays much higher variation than that of the MLE.

```r
n = 1000
X = matrix(rpois(n * nsim, lambda), n)
```

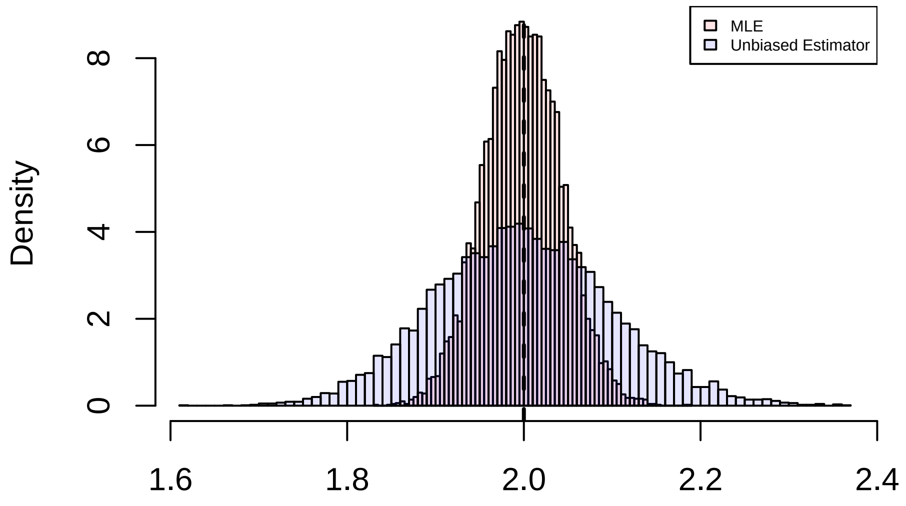
```r
MLE = colMeans(X)
UE = apply(X, 2, var)
mse[1, 1] = mean(MLE) - lambda
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - lambda)^2)
mse[2, 1] = mean(UE) - lambda
mse[2, 2] = mean((UE - mean(UE))^2)
mse[2, 3] = mean((UE - lambda)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	-0.0008	0.0020	0.0020
Unbiased Estimator	-0.0008	0.0098	0.0098

```r
CRLB = lambda/n
print(CRLB)
```

```
[1] 0.002
```

```r
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), main = NA, xlim = range(UE),
    xlab = NA)
hist(UE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), add = TRUE)
abline(v = lambda, lty = 2, lwd = 2)
legend("topright", c("MLE", "Unbiased Estimator"), fill = c(rgb(1, 0, 0,
    0.1), rgb(0, 0, 1, 0.1)), cex = 0.5)
```



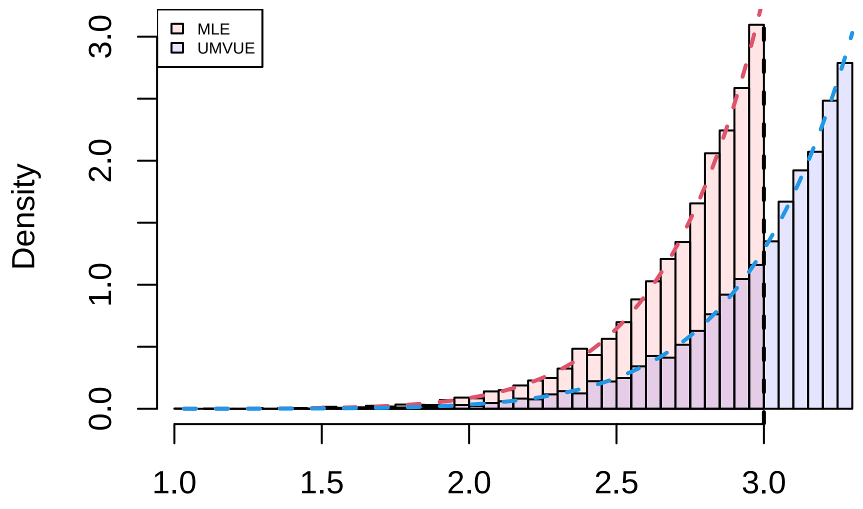
Example 1.23. Let $X_1, \ldots, X_n \sim \mathcal{U}(0, \vartheta)$ be a random sample. Then, we know that the MLE $\widehat{\vartheta} = X_{(n)}$ of ϑ is a biased estimator of ϑ with $f_{X_{(n)}}(x) = \frac{n}{\vartheta^n} x^{n-1}$ and $\frac{n+1}{n} X_{(n)}$ is the UMVUE of ϑ.

We observe that the estimated bias of the UMVUE is close to 0, whereas the MLE tends to underestimate the true value of ϑ. On the other hand, the estimated variance of the UMVUE is larger than that of the MLE, but the estimated MSE of the UMVUE is still smaller than that of the MLE. Looking at the observed distributions of the two estimators, the MLE always takes smaller values than the true value of ϑ.

```
n = 10
theta = 3
X = matrix(runif(n * nsim, 0, theta), n)
MLE = apply(X, 2, max)
UMVUE = MLE * (n + 1)/n
rownames(mse) = c("MLE", "UMVUE")
mse[1, 1] = mean(MLE) - theta
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - theta)^2)
mse[2, 1] = mean(UMVUE) - theta
mse[2, 2] = mean((UMVUE - mean(UMVUE))^2)
mse[2, 3] = mean((UMVUE - theta)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	-0.2733	0.0622	0.1369
UMVUE	-0.0006	0.0753	0.0753

```
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), main = NA, xlim = range(c(MLE,
    UMVUE)), xlab = NA)
hist(UMVUE, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), add = TRUE)
curve(n * x^(n - 1)/theta^n, add = TRUE, col = 2, lty = 2, lwd = 2)
curve(n * x^(n - 1) * (n/(theta * (n + 1)))^n, add = TRUE, col = 4, lty = 2,
    lwd = 2)
abline(v = theta, lty = 2, lwd = 2)
legend("topleft", c("MLE", "UMVUE"), fill = c(rgb(1, 0, 0, 0.1), rgb(0,
    0, 1, 0.1)), cex = 0.5)
```



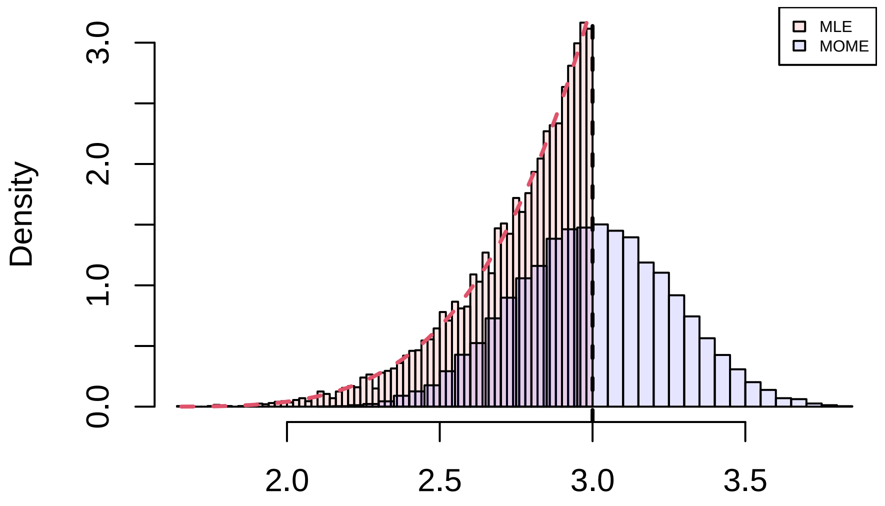
Example 1.24. Let $X_1, \dots, X_n \sim \mathcal{U}(\vartheta, 2\vartheta)$ be a random sample. Then, we know that the MLE $\widehat{\vartheta} = \frac{1}{2}X_{(n)}$ of ϑ is a biased estimator of ϑ with $f_{X_{(n)}}(x) = \frac{n}{\vartheta^n}(x-\vartheta)^{n-1}$, whereas the method of moments estimator $\frac{2}{3}\overline{X}$ is an unbiased estimator of ϑ.

For $n = 5$, we observe that the estimated bias of the MOME is close to 0, whereas the MLE tends to underestimate the true value of ϑ. On the other hand, the estimated variance of the MOME is larger than that of the MLE, but the estimated MSE of the MOME is still smaller than that of the MLE. Looking at the observed distributions of the two estimators, the MLE always takes smaller values than the true value of ϑ.

```r
n = 5
X = matrix(runif(n * nsim, theta, 2 * theta), n)
MLE = apply(X, 2, max)/2
MOME = 2 * colMeans(X)/3
rownames(mse) = c("MLE", "MOME")
mse[1, 1] = mean(MLE) - theta
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - theta)^2)
mse[2, 1] = mean(MOME) - theta
mse[2, 2] = mean((MOME - mean(MOME))^2)
mse[2, 3] = mean((MOME - theta)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	-0.2488	0.0441	0.1060
MOME	0.0037	0.0661	0.0661

```r
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), main = NA, xlim = range(c(MLE,
    MOME)), xlab = NA)
hist(MOME, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), add = TRUE)
curve(2 * n * (2 * x - theta)^(n - 1)/theta^n, add = TRUE, col = 2, lty = 2,
    lwd = 2)
abline(v = theta, lty = 2, lwd = 2)
legend("topright", c("MLE", "MOME"), fill = c(rgb(1, 0, 0, 0.1), rgb(0,
    0, 1, 0.1)), cex = 0.5)
```

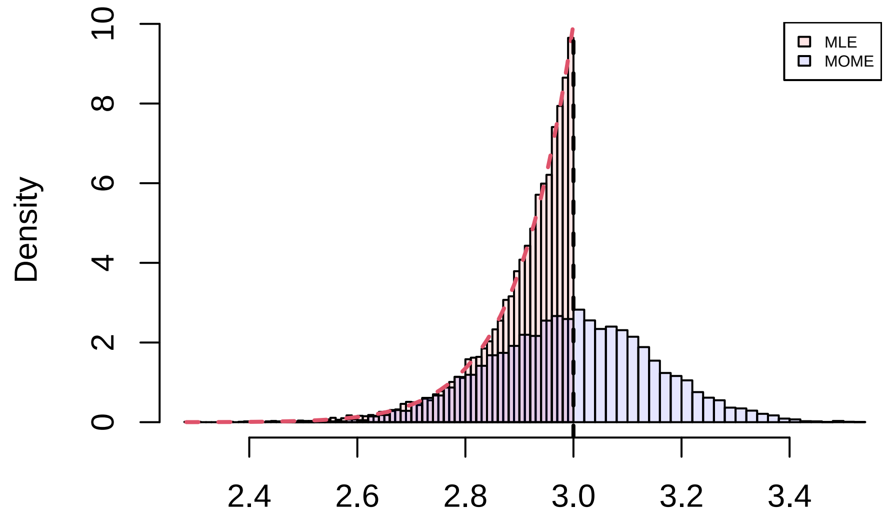


For $n = 15$, we observe that the estimated MSE of the MOME becomes larger than that of the MLE.

```r
n = 15
X = matrix(runif(n * nsim, theta, 2 * theta), n)
MLE = apply(X, 2, max)/2
MOME = 2 * colMeans(X)/3
mse[1, 1] = mean(MLE) - theta
mse[1, 2] = mean((MLE - mean(MLE))^2)
mse[1, 3] = mean((MLE - theta)^2)
mse[2, 1] = mean(MOME) - theta
mse[2, 2] = mean((MOME - mean(MOME))^2)
mse[2, 3] = mean((MOME - theta)^2)
print(xtable(mse, digits = c(0, rep(4, 3))), comment = FALSE)
```

	Bias	Variance	MSE
MLE	-0.0933	0.0078	0.0165
MOME	-0.0010	0.0228	0.0228

```r
hist(MLE, "FD", freq = FALSE, col = rgb(1, 0, 0, 0.1), main = NA, xlim = range(c(MLE,
    MOME)), xlab = NA)
hist(MOME, "FD", freq = FALSE, col = rgb(0, 0, 1, 0.1), add = TRUE)
curve(2 * n * (2 * x - theta)^(n - 1)/theta^n, add = TRUE, col = 2, lty = 2,
    lwd = 2)
abline(v = theta, lty = 2, lwd = 2)
legend("topright", c("MLE", "MOME"), fill = c(rgb(1, 0, 0, 0.1), rgb(0,
    0, 1, 0.1)), cex = 0.5)
```



Asymptotic Distribution of Estimators

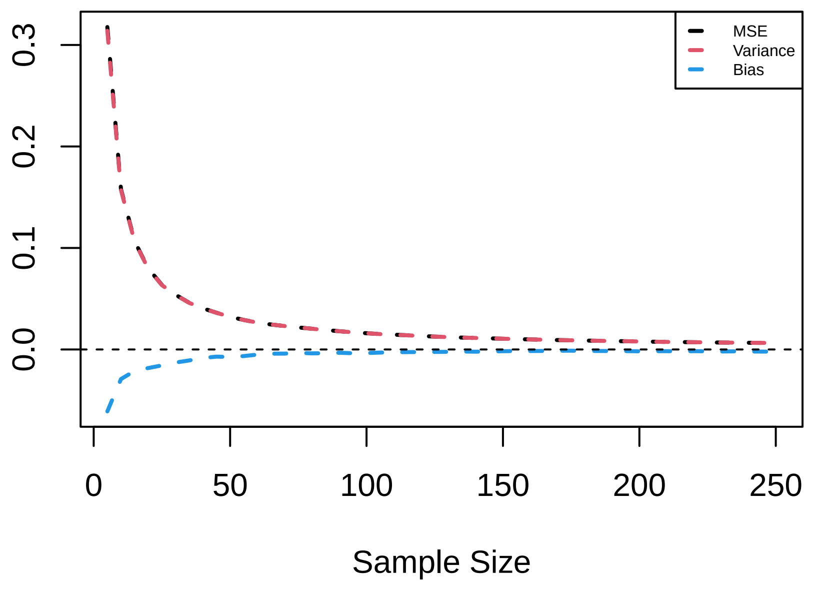
Now, we want to check what happens to the bias, the variance and the MSE of an estimator as we generate more and more observations for our sample. Furthermore, we can check how the finite sample distribution and the asymptotic distribution of the estimator compare against the empirical distribution of the estimator for small vs. large sample sizes.

Example 1.25. Let $X_1, \dots, X_n \sim \mathcal{N}(\vartheta, \vartheta)$ be a random sample with $\vartheta > 0$. Then, we know that:

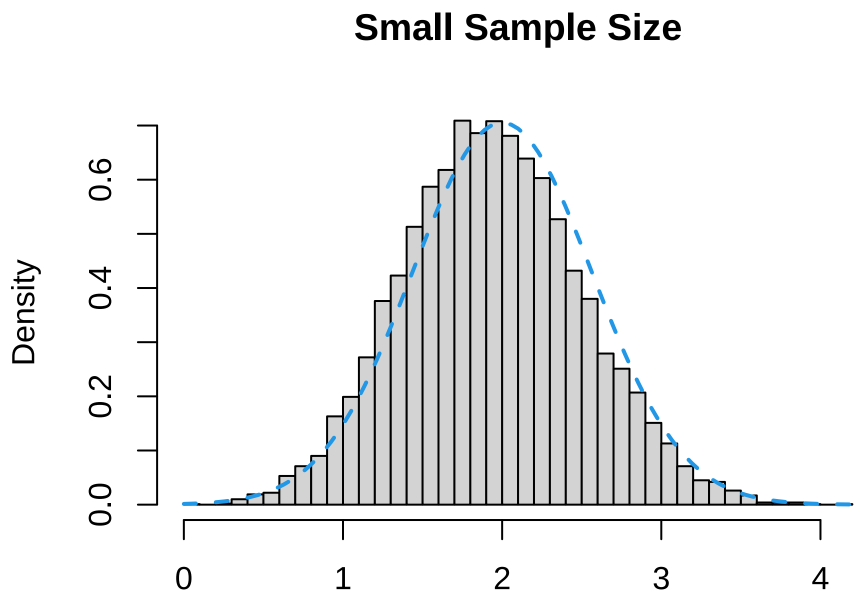
$$\widehat{\vartheta}_n = \frac{1}{2}\sqrt{1 + \frac{4}{n}\sum_{i=1}^n X_i^2} - \frac{1}{2}, \quad \sqrt{n}\left(\widehat{\vartheta}_n - \vartheta\right) \xrightarrow{d} Y \sim \mathcal{N}\left(0, \frac{2\vartheta^2}{2\vartheta + 1}\right).$$

We start off with a sample size of just $n = 5$ observations from the above distribution and continuously generate another step $= 5$ observations for that sample until we reach a total number of $n = 250$ observations. We observe that the MLE slightly underestimates the true value of ϑ for small sample sizes, but that bias converges to 0 as the sample size increases. Similarly, the MLE displays high variance and MSE for small sample sizes, but those quantities also converge to 0 as the sample size increases. The asymptotic distribution of the MLE almost coincides with its corresponding empirical distribution even for a sample size of just $n = 5$ observations, while it's an actual perfect fit for a large sample size of $n = 250$ observations.

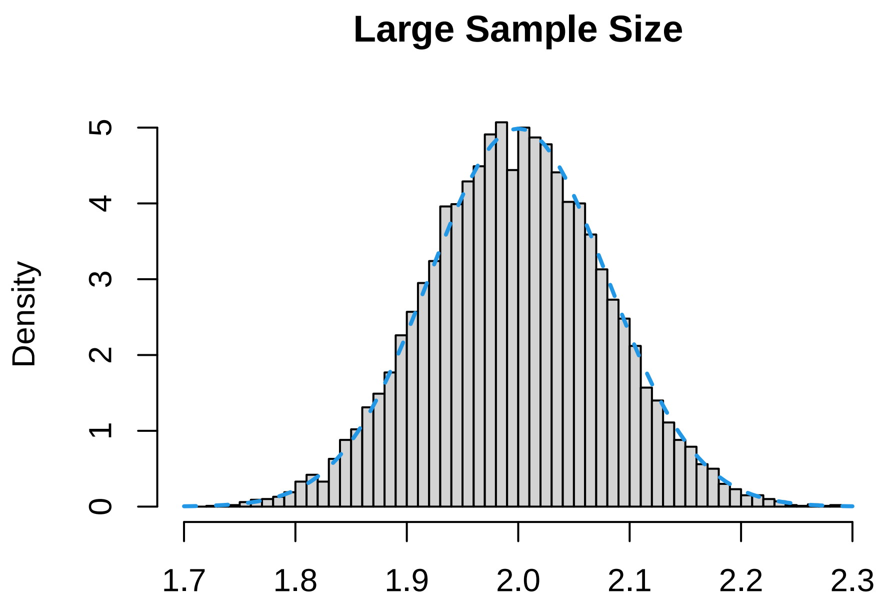
```r
step = 5
n = seq(5, 250, step)
nsim = 10000
theta = 2
X = matrix(0, 0, nsim)
MLE = matrix(0, 50, nsim)
Bias = numeric(50)
Variance = numeric(50)
MSE = numeric(50)
for (k in 1:50) {
    X = rbind(X, matrix(rnorm(step * nsim, theta, sqrt(theta)), step))
    MLE[k, ] = (sqrt(1 + 4 * colMeans(X^2)) - 1)/2
    Bias[k] = mean(MLE[k, ]) - theta
    Variance[k] = var(MLE[k, ])
    MSE[k] = mean((MLE[k, ] - theta)^2)
}
plot(n, MSE, "l", ylim = range(c(MSE, Bias)), xlab = "Sample Size", ylab = NA,
    lty = 2, lwd = 2)
lines(n, Variance, col = 2, lty = 2, lwd = 2)
lines(n, Bias, col = 4, lty = 2, lwd = 2)
abline(h = 0, lty = 2)
legend("topright", c("MSE", "Variance", "Bias"), col = c(1, 2, 4), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



```r
hist(MLE[1, ], "FD", freq = FALSE, main = "Small Sample Size", xlab = NA)
curve(dnorm(x, theta, sqrt(2 * theta^2/(n[1] * (2 * theta + 1)))), add = TRUE,
    col = 4, lty = 2, lwd = 2)
```



```r
hist(MLE[50, ], "FD", freq = FALSE, main = "Large Sample Size", xlab = NA)
curve(dnorm(x, theta, sqrt(2 * theta^2/(n[50] * (2 * theta + 1)))), add = TRUE,
    col = 4, lty = 2, lwd = 2)
```

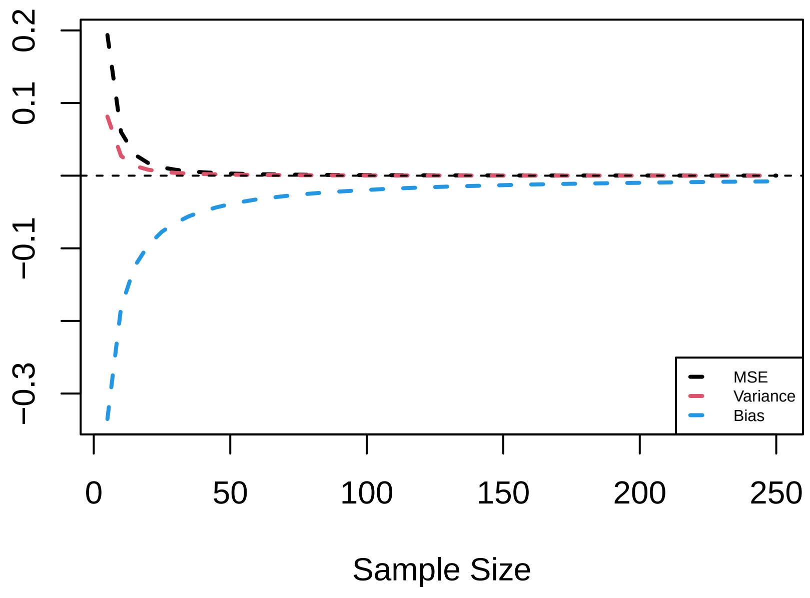


Example 1.26. Let $X_1, \ldots, X_n \sim \mathcal{U}(0, \vartheta)$ be a random sample. Then, we know that:

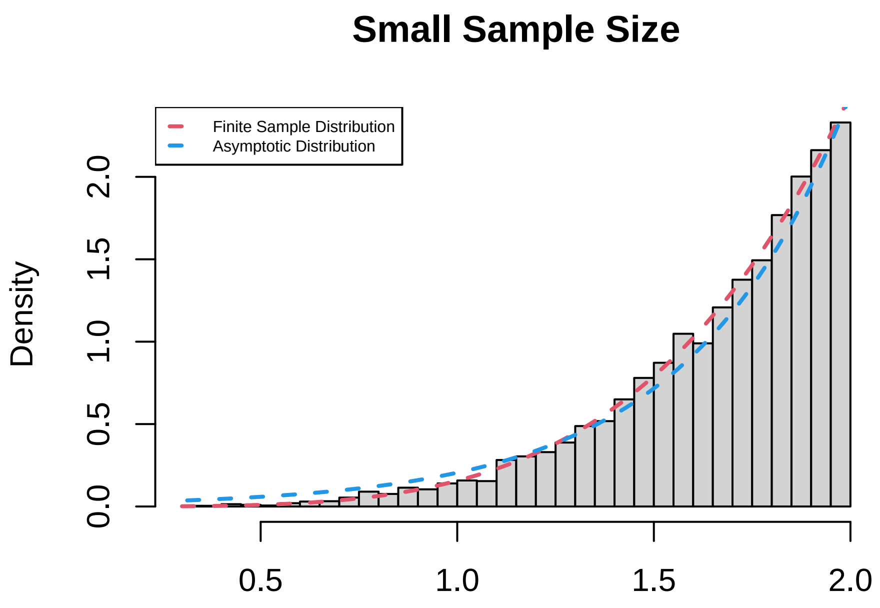
$$\widehat{\vartheta}_n = X_{(n)}, \quad f_{X_{(n)}} = \frac{n}{\vartheta^n} x^{n-1}, \quad n \left[\vartheta - X_{(n)}\right] \xrightarrow{d} Y \sim \operatorname{Exp}(1/\vartheta).$$

The asymptotic distribution of the MLE almost coincides with its corresponding finite sample distribution even for a sample size of just $n = 5$ observations, while it's an actual perfect fit for a large sample size of $n = 250$ observations.

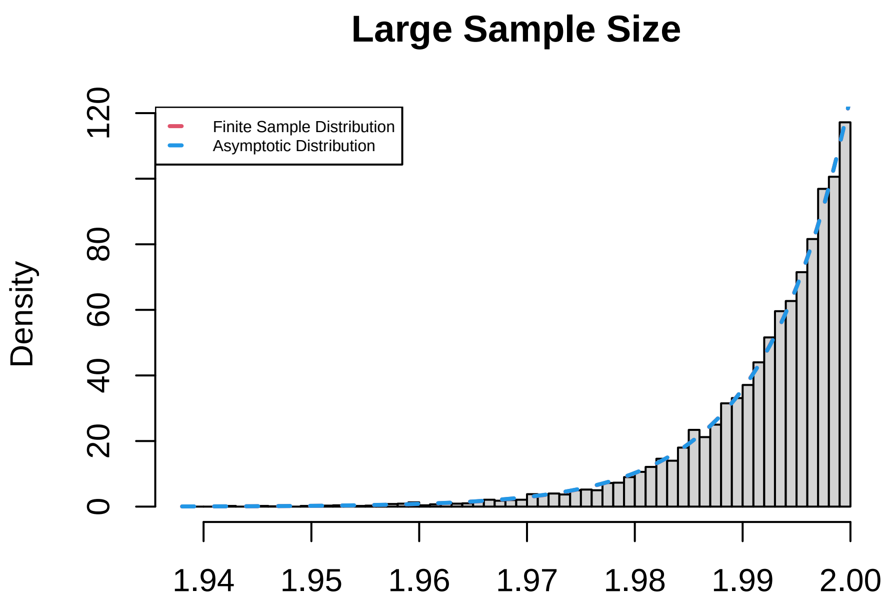
```r
X = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, matrix(runif(step * nsim, max = theta), step))
    MLE[k, ] = apply(X, 2, max)
    Bias[k] = mean(MLE[k, ]) - theta
    Variance[k] = var(MLE[k, ])
    MSE[k] = mean((MLE[k, ] - theta)^2)
}
plot(n, MSE, "l", ylim = range(c(MSE, Bias)), xlab = "Sample Size", ylab = NA,
    lty = 2, lwd = 2)
lines(n, Variance, col = 2, lty = 2, lwd = 2)
lines(n, Bias, col = 4, lty = 2, lwd = 2)
abline(h = 0, lty = 2)
legend("bottomright", c("MSE", "Variance", "Bias"), col = c(1, 2, 4), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



```
hist(MLE[1, ], "FD", freq = FALSE, main = "Small Sample Size", xlab = NA)
curve(n[1] * x^(n[1] - 1)/theta^n[1], add = TRUE, col = 2, lty = 2, lwd = 2)
curve(dexp(theta - x, n[1]/theta), add = TRUE, xlim = c(min(MLE[1, ]),
    theta), col = 4, lty = 2, lwd = 2)
legend("topleft", c("Finite Sample Distribution", "Asymptotic Distribution"),
    col = c(2, 4), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```



```
hist(MLE[50, ], "FD", freq = FALSE, main = "Large Sample Size", xlab = NA)
curve(n[50] * x^(n[50] - 1)/theta^n[50], add = TRUE, col = 2, lty = 2,
    lwd = 2)
curve(dexp(theta - x, n[50]/theta), add = TRUE, xlim = c(min(MLE[50, ]),
    theta), col = 4, lty = 2, lwd = 2)
legend("topleft", c("Finite Sample Distribution", "Asymptotic Distribution"),
    col = c(2, 4), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```

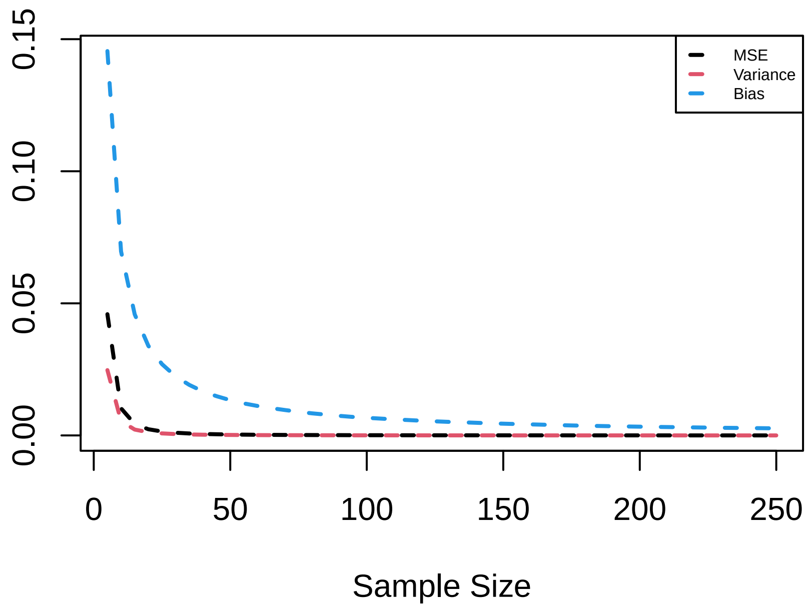


Example 1.27. Let $X_1, \ldots, X_n \sim \text{Pareto}(\vartheta, \lambda)$ be a random sample with $f(x; \vartheta) = \frac{\lambda \vartheta^\lambda}{x^{\lambda+1}}$ for $\vartheta > 0$, known $\lambda > 2$ and $x \geqslant \vartheta$. Then, we know that:

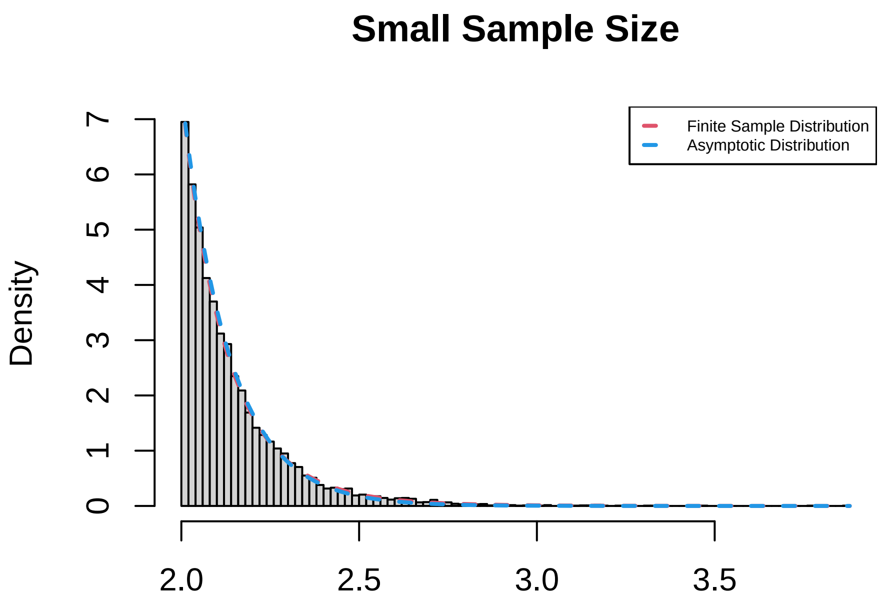
$$\widehat{\vartheta}_n = X_{(1)}, \quad f_{X_{(1)}} = \frac{n\lambda\vartheta^{n\lambda}}{x^{n\lambda+1}}, \quad n\left[X_{(1)} - \vartheta\right] \xrightarrow{d} Y \sim \text{Exp}\left(\lambda/\vartheta\right).$$

We observe that the MLE severely overestimates the true value of ϑ for small sample sizes, but that bias converges to 0 as the sample size increases. Similarly, the MLE displays high variance and MSE for small sample sizes, but those quantities also converge to 0 as the sample size increases. The asymptotic distribution of the MLE is an actual perfect fit for its corresponding finite sample distribution even for a sample size of just $n = 5$ observations.

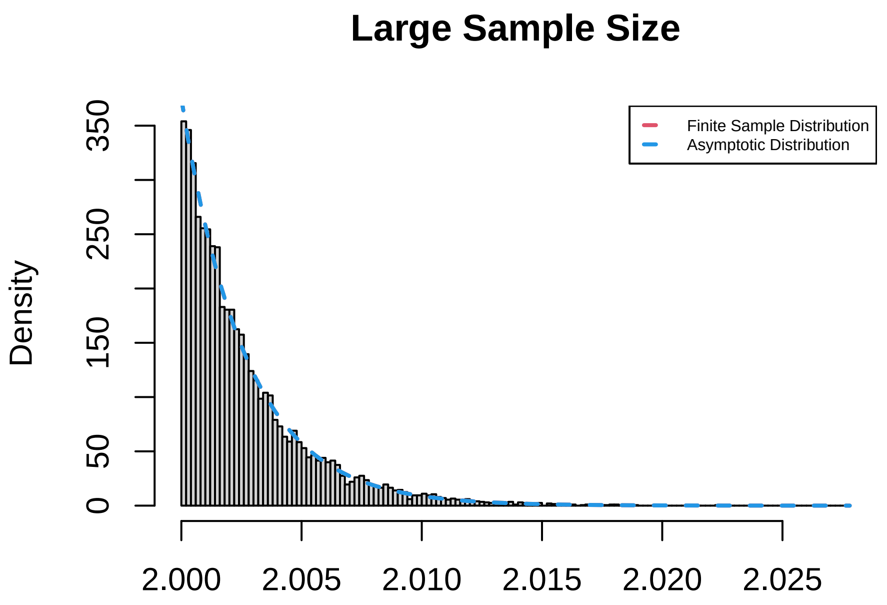
```r
lambda = 3
X = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, theta * matrix((1 - runif(step * nsim))^(-lambda^(-1)),
        step))
    MLE[k, ] = apply(X, 2, min)
    Bias[k] = mean(MLE[k, ]) - theta
    Variance[k] = var(MLE[k, ])
    MSE[k] = mean((MLE[k, ] - theta)^2)
}
plot(n, MSE, "l", ylim = range(c(MSE, Bias)), xlab = "Sample Size", ylab = NA,
    lty = 2, lwd = 2)
lines(n, Variance, col = 2, lty = 2, lwd = 2)
lines(n, Bias, col = 4, lty = 2, lwd = 2)
legend("topright", c("MSE", "Variance", "Bias"), col = c(1, 2, 4), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



```
hist(MLE[1, ], "FD", freq = FALSE, main = "Small Sample Size", xlab = NA)
curve(n[1] * lambda * theta^(n[1] * lambda)/x^(n[1] * lambda + 1), add = TRUE,
    col = 2, lty = 2, lwd = 2)
curve(dexp(x - theta, n[1] * lambda/theta), add = TRUE, col = 4, lty = 2,
    lwd = 2)
legend("topright", c("Finite Sample Distribution", "Asymptotic Distribution"),
    col = c(2, 4), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```



```
hist(MLE[50, ], "FD", freq = FALSE, main = "Large Sample Size", xlab = NA)
curve(n[50] * lambda * theta^(n[50] * lambda)/x^(n[50] * lambda + 1), add = TRUE,
    col = 2, lty = 2, lwd = 2)
curve(dexp(x - theta, n[50] * lambda/theta), add = TRUE, col = 4, lty = 2,
    lwd = 2)
legend("topright", c("Finite Sample Distribution", "Asymptotic Distribution"),
    col = c(2, 4), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```

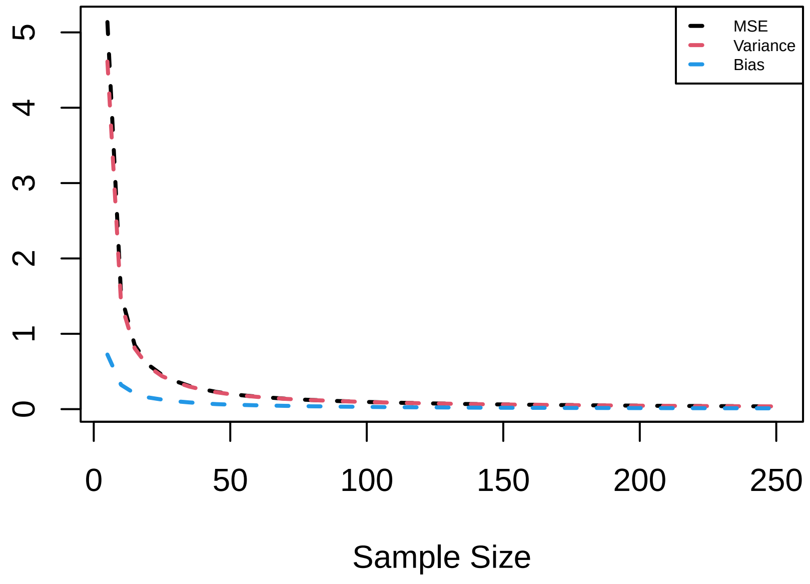


Example 1.28. Let $X_1, \ldots, X_n \sim \text{Exp}(\lambda)$ be a random sample. Then, we know that:

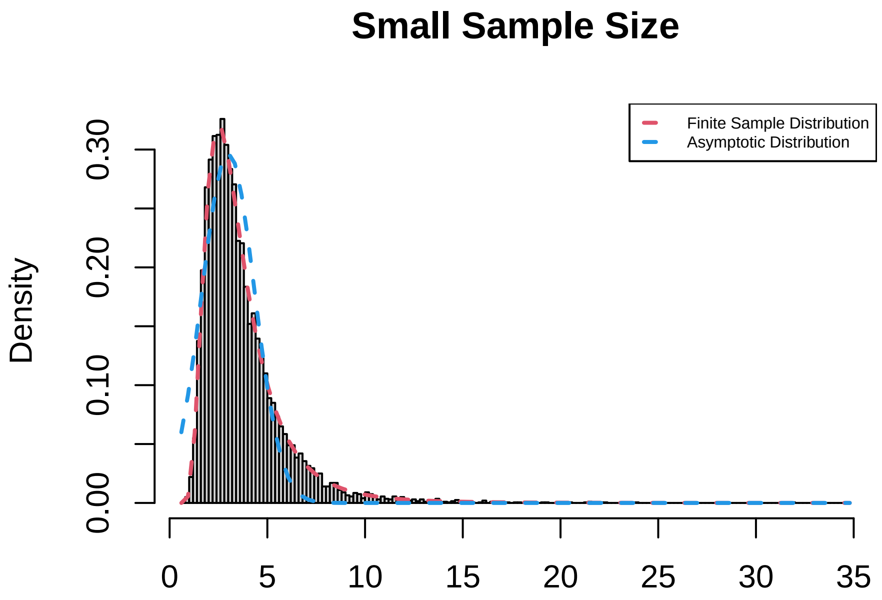
$$\widehat{\lambda}_n = \frac{1}{\overline{X}_n} \sim \text{Inv-Gamma}(n, n\lambda), \quad \sqrt{n}\left(\frac{1}{\overline{X}_n} - \lambda\right) \xrightarrow{d} Y \sim \mathcal{N}\left(0, \lambda^2\right).$$

The asymptotic distribution of the MLE is quite far off from its corresponding finite sample distribution for a small sample size of just $n = 5$ observations, but the 2 distributions almost perfectly coincide for a large sample size of $n = 250$ observations.

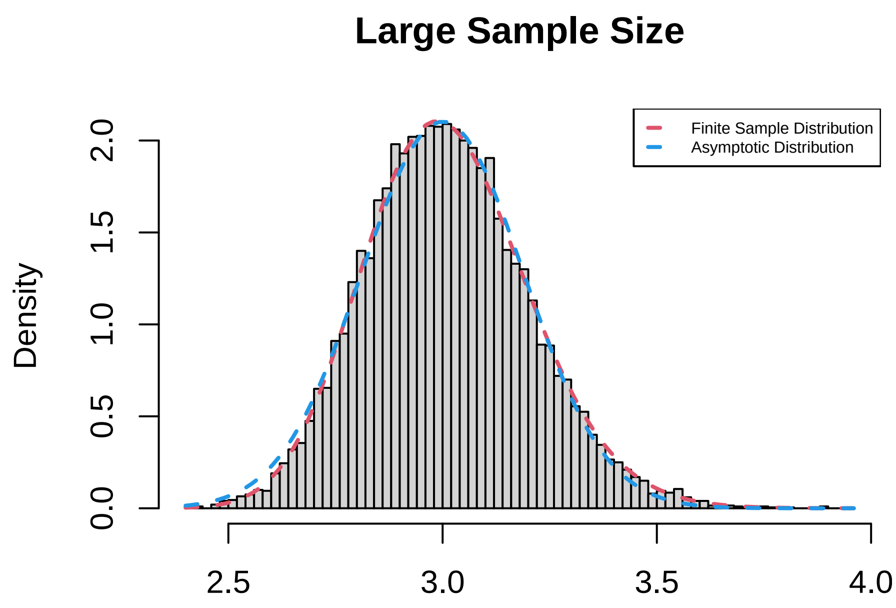
```
X = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, matrix(rexp(step * nsim, lambda), step))
    MLE[k, ] = 1/colMeans(X)
    Bias[k] = mean(MLE[k, ]) - lambda
    Variance[k] = var(MLE[k, ])
    MSE[k] = mean((MLE[k, ] - lambda)^2)
}
plot(n, MSE, "l", xlab = "Sample Size", ylab = NA, lty = 2, lwd = 2)
lines(n, Variance, col = 2, lty = 2, lwd = 2)
lines(n, Bias, col = 4, lty = 2, lwd = 2)
legend("topright", c("MSE", "Variance", "Bias"), col = c(1, 2, 4), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



```
hist(MLE[1, ], "FD", freq = FALSE, main = "Small Sample Size", xlab = NA)
curve(dgamma(x^(-1), n[1], n[1] * lambda)/x^2, add = TRUE, col = 2, lty = 2,
    lwd = 2)
curve(dnorm(x, lambda, lambda/sqrt(n[1])), add = TRUE, col = 4, lty = 2,
    lwd = 2)
legend("topright", c("Finite Sample Distribution", "Asymptotic Distribution"),
    col = c(2, 4), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```



```
hist(MLE[50, ], "FD", freq = FALSE, main = "Large Sample Size", xlab = NA)
curve(dgamma(x^(-1), n[50], n[50] * lambda)/x^2, add = TRUE, col = 2, lty = 2,
    lwd = 2)
curve(dnorm(x, lambda, lambda/sqrt(n[50])), add = TRUE, col = 4, lty = 2,
    lwd = 2)
legend("topright", c("Finite Sample Distribution", "Asymptotic Distribution"),
    col = c(2, 4), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```

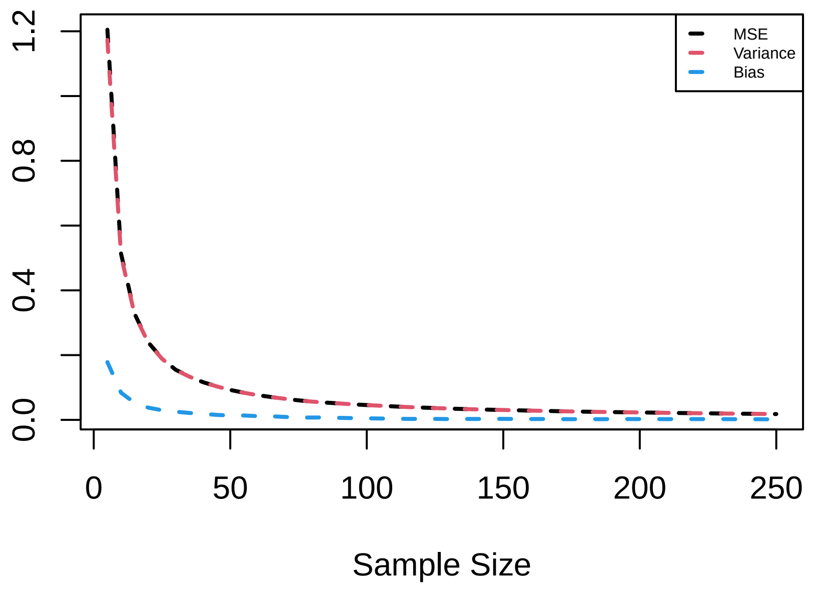


Example 1.29. Let $X_1, \ldots, X_n \sim \text{Exp}(\lambda)$ and $Y_1, \ldots, Y_n \sim \text{Exp}(1/\lambda)$ be 2 independent random samples. Then, we know that:

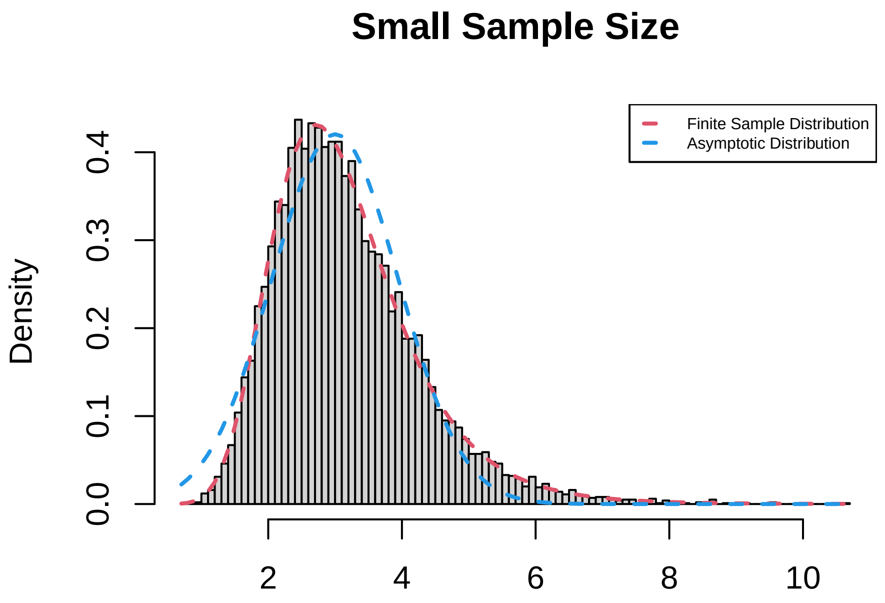
$$\widehat{\lambda}_n = \sqrt{\frac{\overline{Y}_n}{\overline{X}_n}}, \quad \frac{1}{\lambda^2}\widehat{\lambda}_n^2 \sim F(2n, 2n), \quad \sqrt{n}\left(\widehat{\lambda}_n - \lambda\right) \xrightarrow{d} Y \sim \mathcal{N}\left(0, \frac{1}{2}\lambda^2\right).$$

The asymptotic distribution of the MLE is quite far off from its corresponding finite sample distribution for a small sample size of just $n = 5$ observations, but the 2 distributions almost perfectly coincide for a large sample size of $n = 250$ observations.

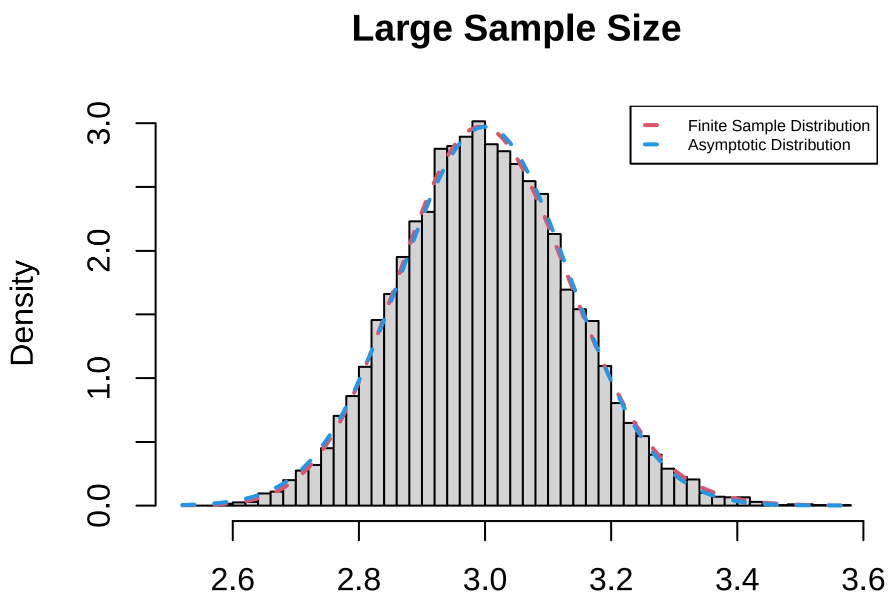
```r
X = matrix(0, 0, nsim)
Y = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, matrix(rexp(step * nsim, lambda), step))
    Y = rbind(Y, matrix(rexp(step * nsim, lambda^(-1)), step))
    MLE[k, ] = sqrt(colSums(Y)/colSums(X))
    Bias[k] = mean(MLE[k, ]) - lambda
    Variance[k] = var(MLE[k, ])
    MSE[k] = mean((MLE[k, ] - lambda)^2)
}
plot(n, MSE, "l", xlab = "Sample Size", ylab = NA, lty = 2, lwd = 2)
lines(n, Variance, col = 2, lty = 2, lwd = 2)
lines(n, Bias, col = 4, lty = 2, lwd = 2)
legend("topright", c("MSE", "Variance", "Bias"), col = c(1, 2, 4), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



```r
hist(MLE[1, ], "FD", freq = FALSE, main = "Small Sample Size", xlab = NA)
curve(2 * x * df(x^2/lambda^2, 2 * n[1], 2 * n[1])/lambda^2, add = TRUE,
    col = 2, lty = 2, lwd = 2)
curve(dnorm(x, lambda, lambda/sqrt(2 * n[1])), add = TRUE, col = 4, lty = 2,
    lwd = 2)
legend("topright", c("Finite Sample Distribution", "Asymptotic Distribution"),
    col = c(2, 4), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```



```r
hist(MLE[50, ], "FD", freq = FALSE, main = "Large Sample Size", xlab = NA)
curve(2 * x * df(x^2/lambda^2, 2 * n[50], 2 * n[50])/lambda^2, add = TRUE,
    col = 2, lty = 2, lwd = 2)
curve(dnorm(x, lambda, lambda/sqrt(2 * n[50])), add = TRUE, col = 4, lty = 2,
    lwd = 2)
legend("topright", c("Finite Sample Distribution", "Asymptotic Distribution"),
    col = c(2, 4), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```

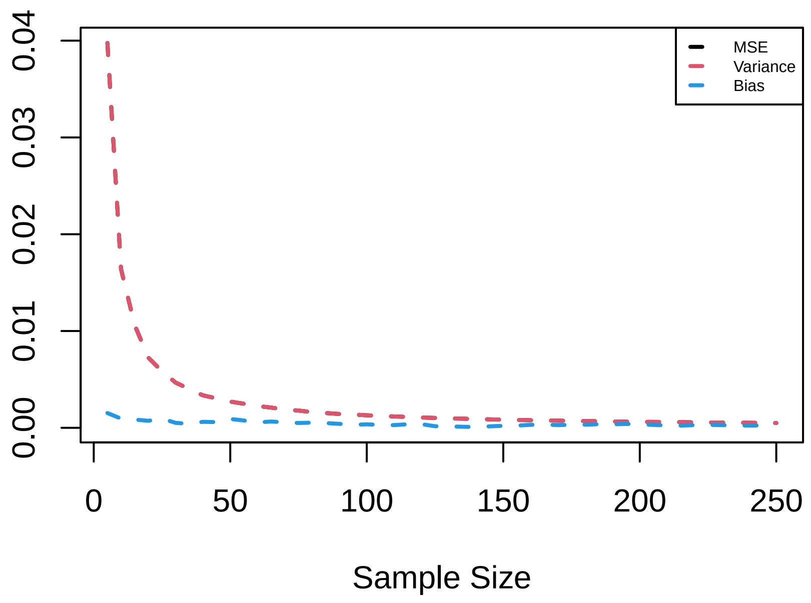


Example 1.30. Let $X_1, \ldots, X_n \sim \text{Laplace}(\mu, \lambda)$ be a random sample with $f(x;\mu) = \frac{\lambda}{2}e^{-\lambda|x-\mu|}$ for $\mu \in \mathbb{R}$, known $\lambda > 0$ and $x \in \mathbb{R}$. Then, we know that:

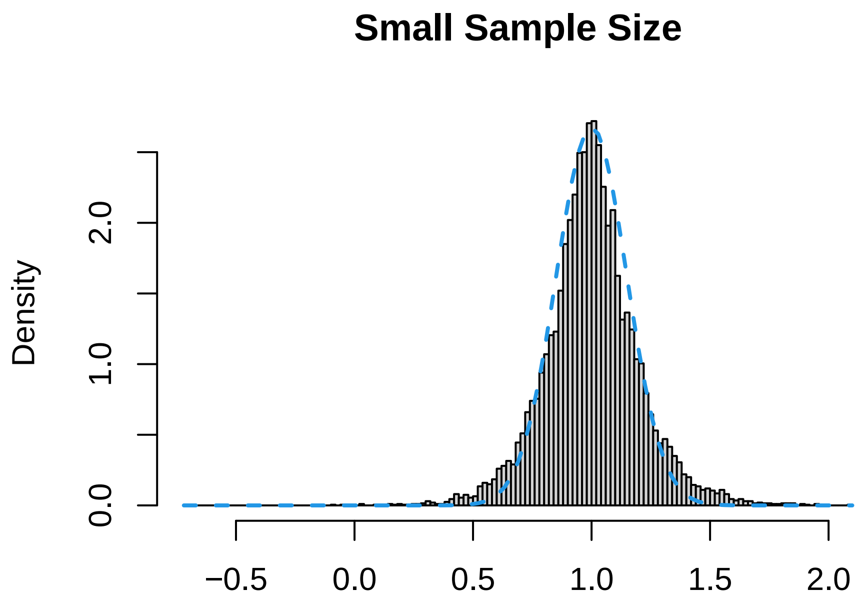
$$\widehat{\mu}_n = \text{median}(X), \quad \sqrt{n}\,[\text{median}(X) - \mu] \xrightarrow{d} Y \sim \mathcal{N}\left(0, \frac{1}{\lambda^2}\right).$$

Since the MLE $\widehat{\mu}_n$ is an unbiased estimator of μ, its estimated bias is always close to 0, even for a sample size of just $n = 5$ observations. The asymptotic distribution of the MLE is quite far off from its corresponding empirical distribution for a small sample size of just $n = 5$ observations, but the 2 distributions almost perfectly coincide for a large sample size of $n = 250$ observations.

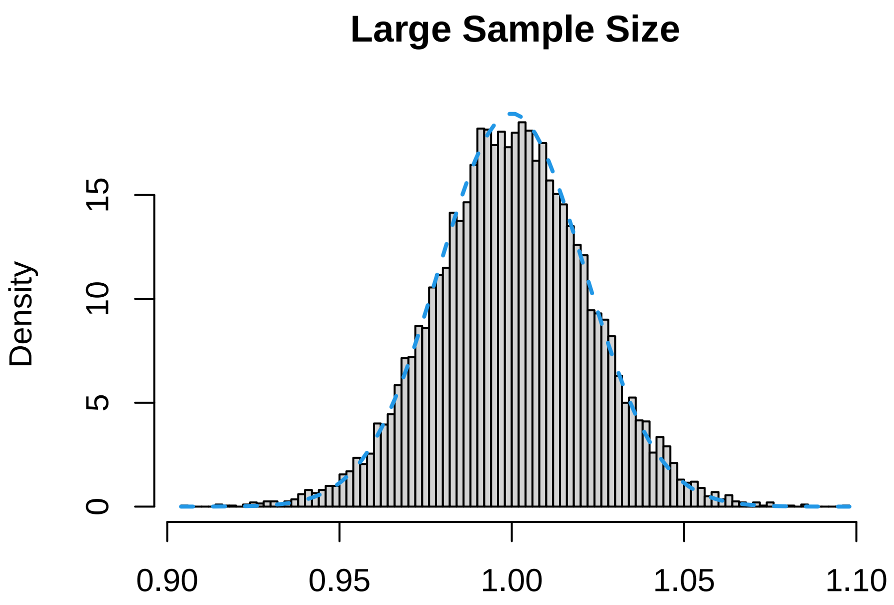
```
mu = 1
X = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, matrix((2 * rbinom(step * nsim, 1, 0.5) - 1) * rexp(step *
        nsim, lambda), step) + mu)
    MLE[k, ] = apply(X, 2, median)
    Bias[k] = mean(MLE[k, ]) - mu
    Variance[k] = var(MLE[k, ])
    MSE[k] = mean((MLE[k, ] - mu)^2)
}
plot(n, MSE, "l", ylim = range(c(MSE, Bias)), xlab = "Sample Size", ylab = NA,
    lty = 2, lwd = 2)
lines(n, Variance, col = 2, lty = 2, lwd = 2)
lines(n, Bias, col = 4, lty = 2, lwd = 2)
legend("topright", c("MSE", "Variance", "Bias"), col = c(1, 2, 4), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



```r
hist(MLE[1, ], "FD", freq = FALSE, main = "Small Sample Size", xlab = NA)
curve(dnorm(x, mu, 1/(sqrt(n[1]) * lambda)), add = TRUE, col = 4, lty = 2,
    lwd = 2)
```



```r
hist(MLE[50, ], "FD", freq = FALSE, main = "Large Sample Size", xlab = NA)
curve(dnorm(x, mu, 1/(sqrt(n[50]) * lambda)), add = TRUE, col = 4, lty = 2,
    lwd = 2)
```

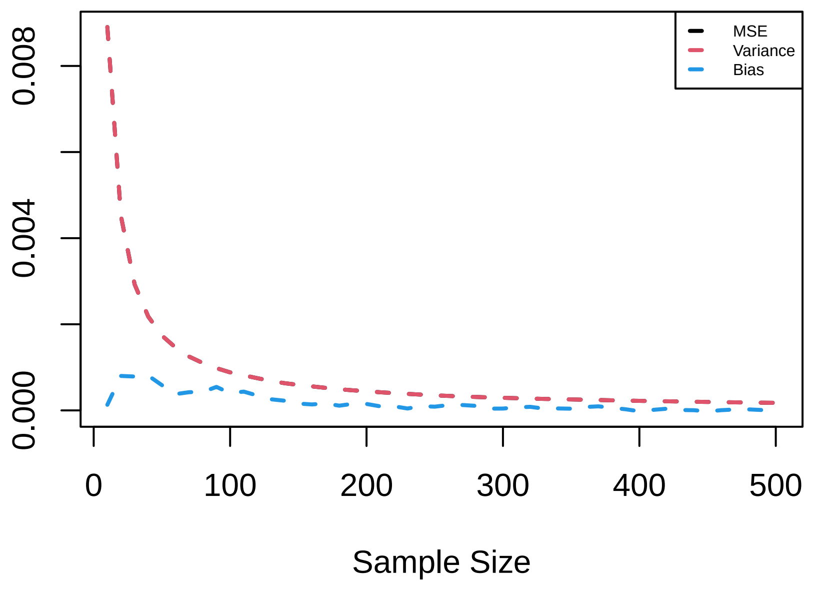


Example 1.31. Let $X_1, \ldots, X_n \sim \text{Bernoulli}(p)$ be a random sample and $g(p) = \min\{p, 1-p\}$. For $p \neq \frac{1}{2}$, we know that:

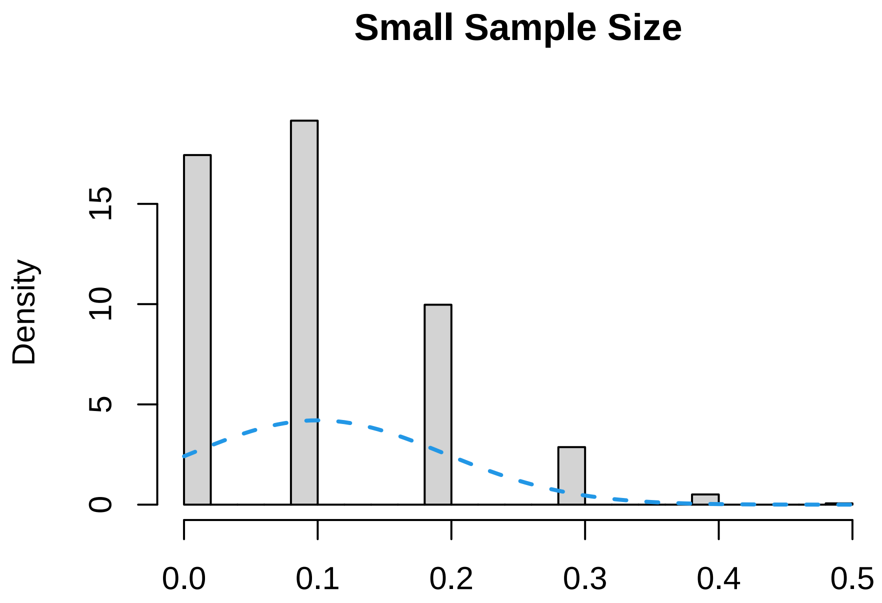
$$\widehat{g_n(p)} = \min\left\{\overline{X}_n, 1 - \overline{X}_n\right\}, \quad \sqrt{n}\left[\widehat{g_n(p)} - g(p)\right] \xrightarrow{d} Y \sim \mathcal{N}\left(0, p(1-p)\right).$$

Since the sampling distribution is discrete, the asymptotic distribution of the MLE is very far off from its corresponding empirical distribution for a small sample size of just $n = 10$ observations, but the 2 distributions closely match for a large sample size of $n = 500$ observations.

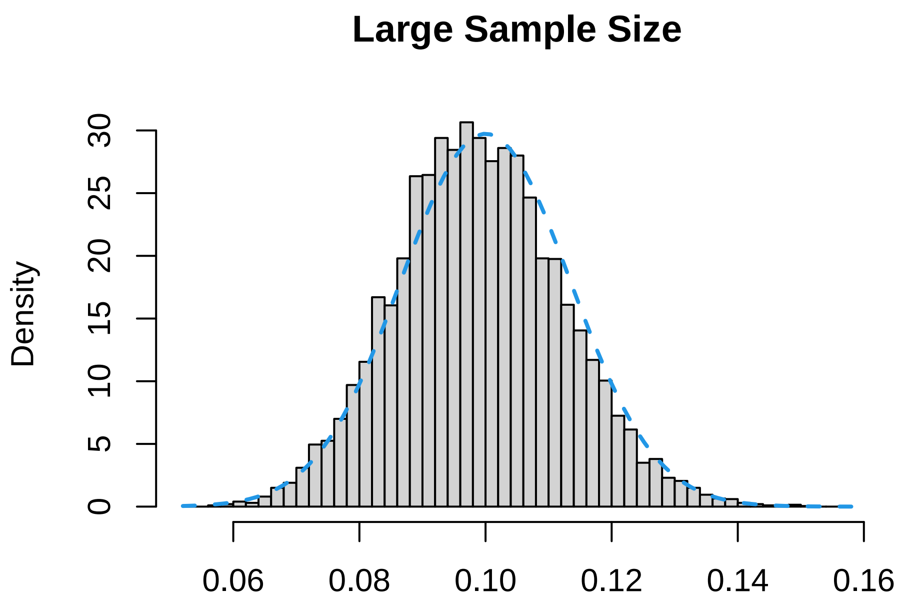
```r
step = 10
n = seq(10, 500, step)
p = 0.1
X = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, matrix(rbinom(step * nsim, 1, p), step))
    MLE[k, ] = pmin(colMeans(X), 1 - colMeans(X))
    Bias[k] = mean(MLE[k, ]) - min(p, 1 - p)
    Variance[k] = var(MLE[k, ])
    MSE[k] = mean((MLE[k, ] - min(p, 1 - p))^2)
}
plot(n, MSE, "l", ylim = range(c(MSE, Bias)), xlab = "Sample Size", ylab = NA,
    lty = 2, lwd = 2)
lines(n, Variance, col = 2, lty = 2, lwd = 2)
lines(n, Bias, col = 4, lty = 2, lwd = 2)
legend("topright", c("MSE", "Variance", "Bias"), col = c(1, 2, 4), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



```r
hist(MLE[1, ], "FD", freq = FALSE, main = "Small Sample Size", xlab = NA)
curve(dnorm(x, min(p, 1 - p), sqrt(p * (1 - p)/n[1])), add = TRUE, col = 4,
    lty = 2, lwd = 2)
```



```r
hist(MLE[50, ], "FD", freq = FALSE, main = "Large Sample Size", xlab = NA)
curve(dnorm(x, min(p, 1 - p), sqrt(p * (1 - p)/n[50])), add = TRUE, col = 4,
    lty = 2, lwd = 2)
```

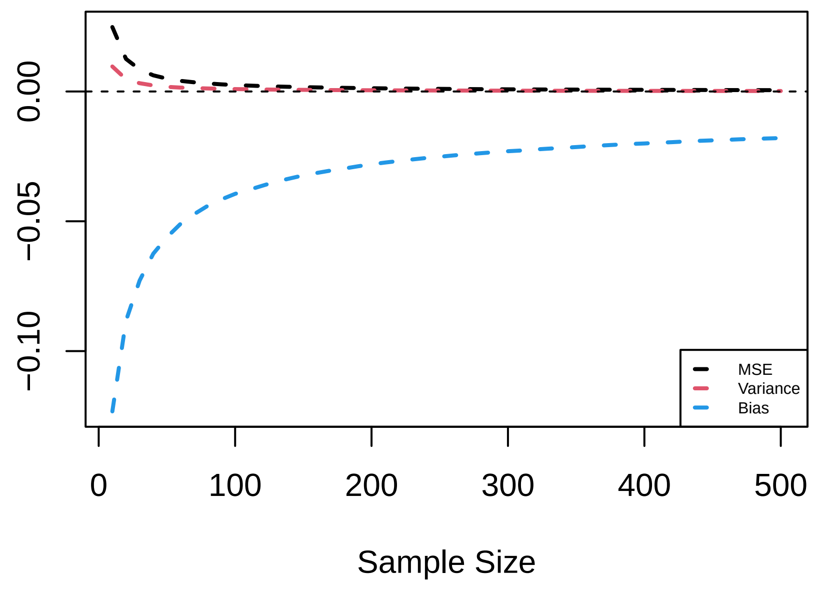


For $p = \frac{1}{2}$, we know that:

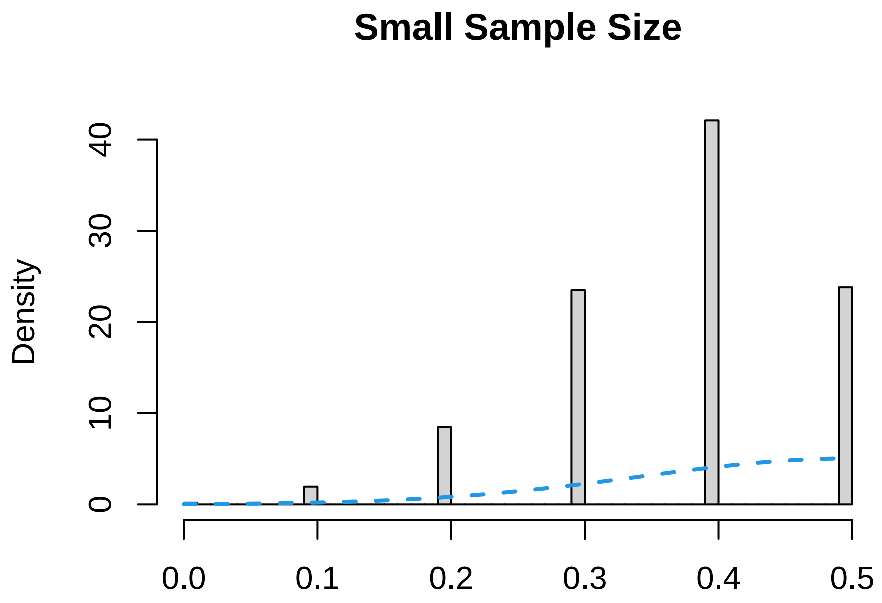
$$\sqrt{n}\left[\widehat{g_n(p)} - \frac{1}{2}\right] \xrightarrow{d} -|Y|, \quad Y \sim \mathcal{N}\left(0, \frac{1}{4}\right).$$

Since the MLE always underestimates the true value of p, its bias starts off significantly below 0 and very slowly converges towards 0.

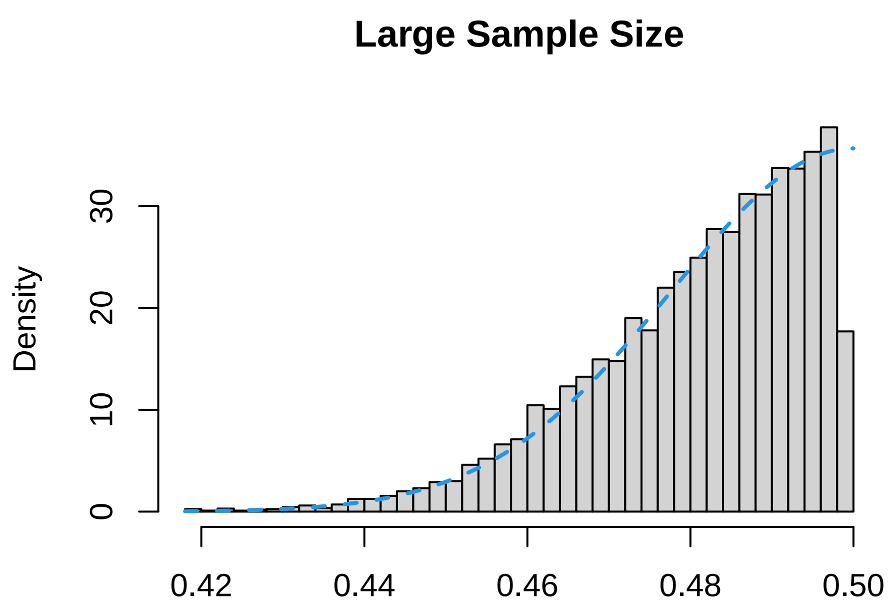
```
p = 0.5
X = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, matrix(rbinom(step * nsim, 1, p), step))
    MLE[k, ] = pmin(colMeans(X), 1 - colMeans(X))
    Bias[k] = mean(MLE[k, ]) - min(p, 1 - p)
    Variance[k] = var(MLE[k, ])
    MSE[k] = mean((MLE[k, ] - min(p, 1 - p))^2)
}
plot(n, MSE, "l", ylim = range(c(MSE, Bias)), xlab = "Sample Size", ylab = NA,
    lty = 2, lwd = 2)
lines(n, Variance, col = 2, lty = 2, lwd = 2)
lines(n, Bias, col = 4, lty = 2, lwd = 2)
abline(h = 0, lty = 2)
legend("bottomright", c("MSE", "Variance", "Bias"), col = c(1, 2, 4), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



```
hist(MLE[1, ], "FD", freq = FALSE, main = "Small Sample Size", xlab = NA)
curve(dnorm(x - p, 0, sqrt(p * (1 - p)/n[1])) + dnorm(p - x, 0, sqrt(p *
    (1 - p)/n[1])), add = TRUE, col = 4, lty = 2, lwd = 2)
```



```
hist(MLE[50, ], "FD", freq = FALSE, main = "Large Sample Size", xlab = NA)
curve(dnorm(x - p, 0, sqrt(p * (1 - p)/n[50])) + dnorm(p - x, 0, sqrt(p *
    (1 - p)/n[50])), add = TRUE, col = 4, lty = 2, lwd = 2)
```



2 Confidence Intervals

We want to verify that the confidence intervals which we're constructing have the correct coverage rate and to compare the average length of different types of confidence intervals for the same parameter. In order to achieve that, we have to generate n_{sim} independent random samples following the same distribution and compute a confidence interval for each of the generated samples. Then, we can calculate the empirical coverage rate of the confidence interval as the percentage of the computed confidence intervals which contain the true value of the unknown parameter.

Example 2.1. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu, \sigma^2\right)$ be a random sample. Then, we know that:

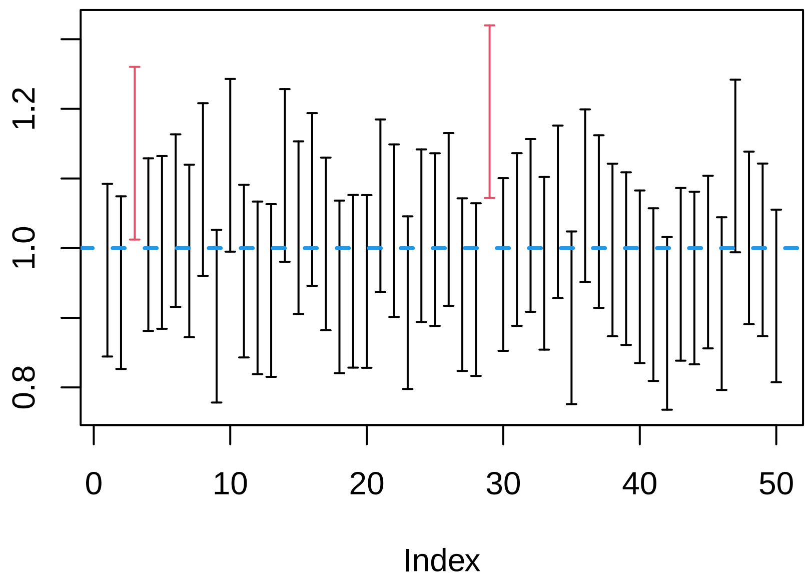
$$\mathcal{I}_{\mu;\,1-\alpha}\left(X; \sigma^2\right) = \left[\overline{X} - Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}, \overline{X} + Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right].$$

If we generate $n_{\text{sim}} = 50$ samples of size $n = 1000$ from this distribution and calculate the corresponding 95% confidence intervals for μ assuming that σ^2 is known, we observe that 2 out of the 50 computed confidence intervals don't cover the true value of μ. This leads to an empirical coverage rate of 96%, which is very close to the nominal coverage rate of 95%.

```
n = 1000
nsim = 50
mu = 1
sigma = 2
alpha = 0.05
X = matrix(rnorm(n * nsim, mu, sigma), n)
MLE = colMeans(X)
CI = cbind(MLE - qnorm(1 - alpha/2) * sigma/sqrt(n), MLE + qnorm(1 - alpha/2) *
    sigma/sqrt(n))
mean(CI[, 1] < mu & mu < CI[, 2])
```

```
## [1] 0.96
```

```r
plot(1:nsim, type = "n", ylim = range(CI), ylab = NA)
arrows(1:nsim, CI[, 1], 1:nsim, CI[, 2], 0.025, 90, 3, col = ifelse(CI[,
    1] < mu & mu < CI[, 2], 1, 2))
abline(h = mu, col = 4, lty = 2, lwd = 2)
```



If we generate $n_{\text{sim}} = 100000$ samples of size $n = 100$ from this distribution and calculate the corresponding 95% confidence intervals for μ, we observe that the empirical coverage rate is again really close to nominal and the average length of the confidence interval is approximately equal to 0.78.

```r
n = 100
nsim = 1e+05
X = matrix(rnorm(n * nsim, mu, sigma), n)
MLE = colMeans(X)
CI = cbind(MLE - qnorm(1 - alpha/2) * sigma/sqrt(n), MLE + qnorm(1 - alpha/2) *
    sigma/sqrt(n))
mean(CI[, 1] < mu & mu < CI[, 2])
```

```
## [1] 0.94977
```

```r
mean(CI[, 2] - CI[, 1])
```

```
## [1] 0.7839856
```

If we instead calculate the corresponding 95% confidence intervals for μ assuming that σ^2 is unknown, we observe that the empirical coverage rate is still really close to nominal, but the average length of the confidence interval ends up being slightly greater. We can verify this confidence interval calculation by use of R's built-in t.test function.

```r
S = apply(X, 2, sd)
CI = cbind(MLE - qt(1 - alpha/2, n - 1) * S/sqrt(n), MLE + qt(1 - alpha/2,
    n - 1) * S/sqrt(n))
mean(CI[, 1] < mu & mu < CI[, 2])
```

```
## [1] 0.95074
```

```
mean(CI[, 2] - CI[, 1])
```

```
## [1] 0.7917938
```

```
print(CI[1, ])
```

```
## [1] 0.4695112 1.2107723
```

```
as.vector(t.test(X[, 1])$conf.int)
```

```
## [1] 0.4695112 1.2107723
```

Example 2.2. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu_1, \sigma^2\right)$ and $Y_1, \ldots, Y_m \sim \mathcal{N}\left(\mu_2, \sigma^2\right)$ be 2 independent random samples. Then, we know that:

$$S_p^2 = \frac{1}{n+m-2}\left[\sum_{i=1}^{n}\left(X_i - \overline{X}\right)^2 + \sum_{i=1}^{m}\left(Y_i - \overline{Y}\right)^2\right],$$

$$\mathcal{I}_{\mu_1-\mu_2;\,1-\alpha}(X,Y) = \left[\overline{X} - \overline{Y} - t_{n+m-2;\alpha/2} S_p\sqrt{\frac{1}{n}+\frac{1}{m}}, \overline{X} - \overline{Y} + t_{n+m-2;\alpha/2} S_p\sqrt{\frac{1}{n}+\frac{1}{m}}\right].$$

We can again verify this confidence interval calculation by use of R's built-in t.test function with the option var.equal = TRUE.

```
m = 100
Y = matrix(rnorm(m * nsim, mu, sigma), m)
MLE = colMeans(X) - colMeans(Y)
SX = apply(X, 2, var)
SY = apply(Y, 2, var)
Sp = sqrt(((n - 1) * SX + (m - 1) * SY)/(n + m - 2))
CI = cbind(MLE - qt(1 - alpha/2, n + m - 2) * Sp * sqrt((n + m)/(n * m)),
    MLE + qt(1 - alpha/2, n + m - 2) * Sp * sqrt((n + m)/(n * m)))
mean(CI[, 1] < 0 & 0 < CI[, 2])
```

```
## [1] 0.95077
```

```
print(CI[1, ])
```

```
## [1] -0.8387731  0.2451934
```

```
as.vector(t.test(X[, 1], Y[, 1], var.equal = TRUE)$conf.int)
```

```
## [1] -0.8387731  0.2451934
```

Additionally, we know that:

$$\mathcal{I}_{\sigma_1^2/\sigma_2^2;\,1-\alpha}(X,Y) = \left[F_{m-1,n-1;1-\alpha/2}\frac{\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i-\overline{X}\right)^2}{\frac{1}{n-1}\sum_{i=1}^{m}\left(Y_i-\overline{Y}\right)^2}, F_{m-1,n-1;\alpha/2}\frac{\frac{1}{n-1}\sum_{i=1}^{n}\left(X_i-\overline{X}\right)^2}{\frac{1}{n-1}\sum_{i=1}^{m}\left(Y_i-\overline{Y}\right)^2}\right].$$

We can verify this confidence interval calculation by use of R's built-in var.test function.

```r
CI = cbind(qf(alpha/2, m - 1, n - 1) * SX/SY, qf(1 - alpha/2, m - 1, n -
    1) * SX/SY)
mean(CI[, 1] < 1 & 1 < CI[, 2])
```

```
## [1] 0.94966
```

```r
print(CI[1, ])
```

```
## [1] 0.5775758 1.2758018
```

```r
as.vector(var.test(X[, 1], Y[, 1])$conf.int)
```

```
## [1] 0.5775758 1.2758018
```

As far as asymptotic confidence intervals are concerned, we might also be interested in ascertaining how their coverage rate and average length change as we generate more and more observations for our sample.

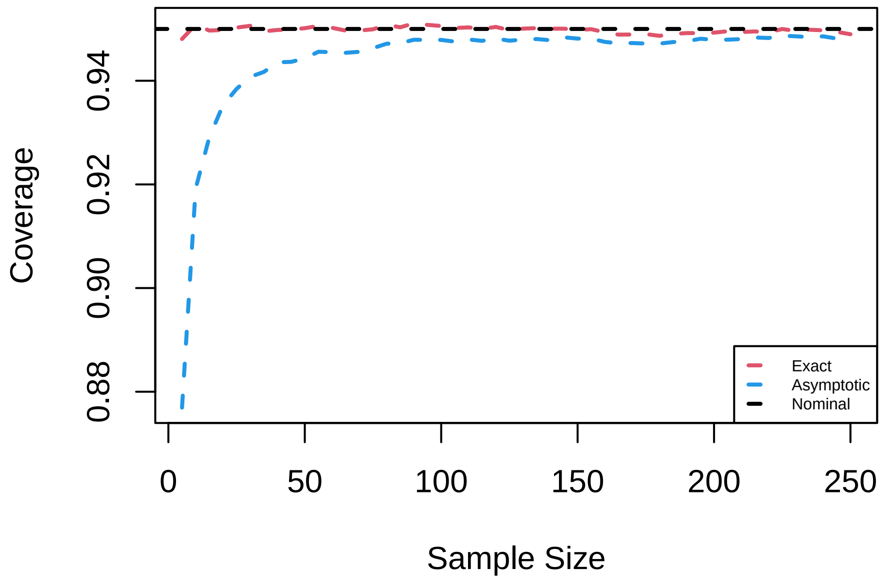
Example 2.3. Let $X_1, \ldots, X_n \sim \mathcal{N}\left(\mu, \sigma^2\right)$ be a random sample. Then, we know that:

$$\mathcal{I}^{(n)}_{\mu;\, 1-\alpha}(X) = \left[\overline{X}_n - Z_{\alpha/2}\frac{S_n}{\sqrt{n}}, \overline{X}_n + Z_{\alpha/2}\frac{S_n}{\sqrt{n}}\right].$$

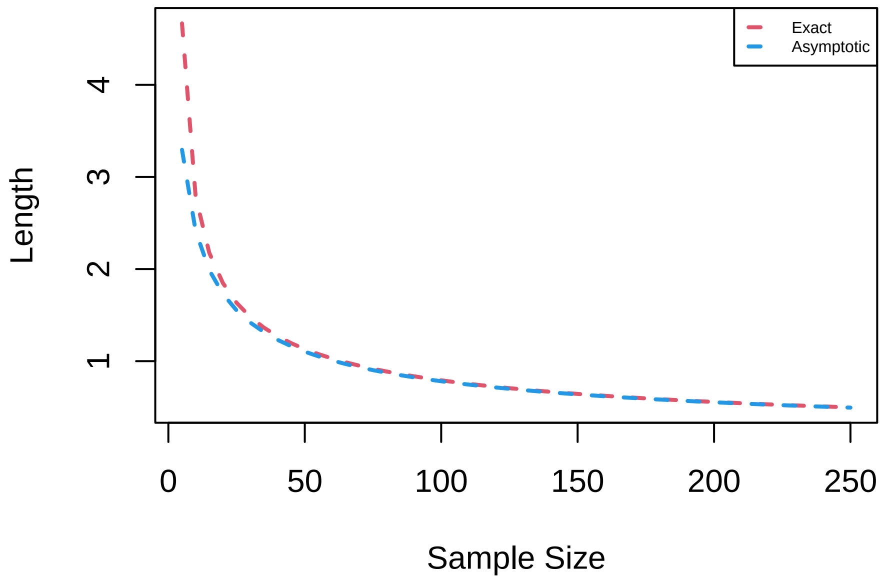
We observe that the asymptotic confidence interval for μ has significantly lower than nominal coverage rate for small sample sizes, but this empirical coverage rate quickly converges to the nominal coverage rate as the sample size increases. In contrast, the exact confidence interval for μ always has close to nominal coverage rate, even for a sample size of just $n = 5$ observations. Accordingly, the average length of the asymptotic confidence interval is initially shorter than that of the corresponding exact confidence interval, but this difference in lengths vanishes as the sample size increases, and both lengths become increasingly smaller.

```r
step = 5
n = seq(5, 250, step)
X = matrix(0, 0, nsim)
Coverage = matrix(0, 50, 2)
Length = matrix(0, 50, 2)
for (k in 1:50) {
    X = rbind(X, matrix(rnorm(step * nsim, mu, sigma), step))
    MLE = colMeans(X)
    S = apply(X, 2, sd)
    CIExact = cbind(MLE - qt(1 - alpha/2, n[k] - 1) * S/sqrt(n[k]), MLE +
        qt(1 - alpha/2, n[k] - 1) * S/sqrt(n[k]))
    CIAsymptotic = cbind(MLE - qnorm(1 - alpha/2) * S/sqrt(n[k]), MLE +
        qnorm(1 - alpha/2) * S/sqrt(n[k]))
    Coverage[k, 1] = mean(CIExact[, 1] < mu & mu < CIExact[, 2])
    Length[k, 1] = mean(CIExact[, 2] - CIExact[, 1])
    Coverage[k, 2] = mean(CIAsymptotic[, 1] < mu & mu < CIAsymptotic[,
        2])
    Length[k, 2] = mean(CIAsymptotic[, 2] - CIAsymptotic[, 1])
}
```

```
plot(n, Coverage[, 1], "l", ylim = range(Coverage), xlab = "Sample Size",
    ylab = "Coverage", col = 2, lty = 2, lwd = 2)
lines(n, Coverage[, 2], col = 4, lty = 2, lwd = 2)
abline(h = 0.95, lty = 2, lwd = 2)
legend("bottomright", c("Exact", "Asymptotic", "Nominal"), col = c(2, 4,
    1), lty = rep(2, 3), lwd = rep(2, 3), cex = 0.5)
```



```
plot(n, Length[, 1], "l", xlab = "Sample Size", ylab = "Length", col = 2,
    lty = 2, lwd = 2)
lines(n, Length[, 2], col = 4, lty = 2, lwd = 2)
legend("topright", c("Exact", "Asymptotic"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```



Example 2.4. Let $X_1, \dots, X_n \sim \text{Exp}(\lambda)$ and $Y_1, \dots, Y_n \sim \text{Exp}(1/\lambda)$ be 2 independent random samples. Then,

we know that:

$$\mathcal{I}_{\lambda;\ 1-\alpha}(X,Y) = \left[\sqrt{F_{2n,2n;1-\alpha/2}\frac{\overline{Y}}{\overline{X}}}, \sqrt{F_{2n,2n;\alpha/2}\frac{\overline{Y}}{\overline{X}}}\right].$$

```
n = 100
lambda = 2
X = matrix(rexp(n * nsim, lambda), n)
Y = matrix(rexp(n * nsim, lambda^(-1)), n)
MLE = sqrt(colSums(Y)/colSums(X))
CI = cbind(sqrt(qf(alpha/2, 2 * n, 2 * n)) * MLE, sqrt(qf(1 - alpha/2,
    2 * n, 2 * n)) * MLE)
mean(CI[, 1] < lambda & lambda < CI[, 2])
```

```
## [1] 0.94879
```

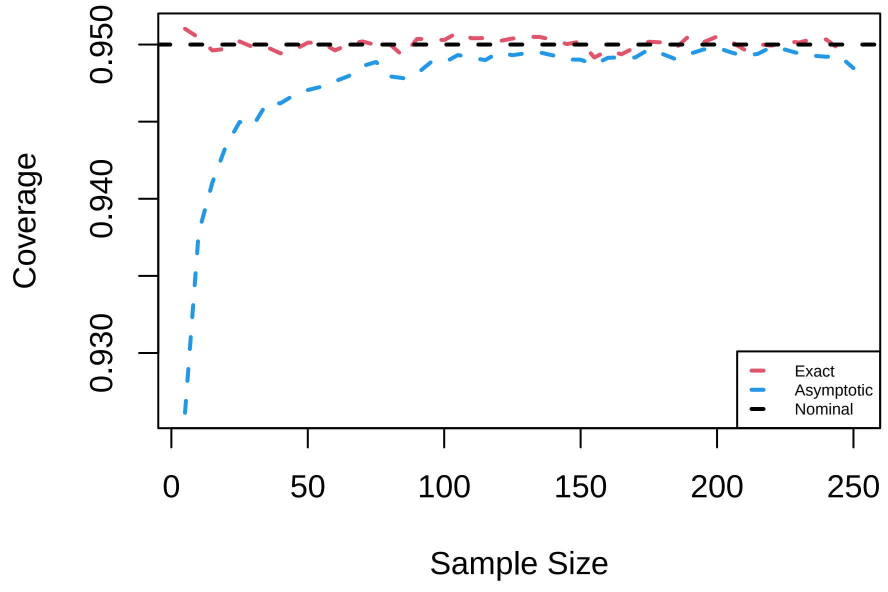
Furthermore, we know that:

$$\mathcal{I}^{(n)}_{\lambda;\ 1-\alpha}(X,Y) = \left[\frac{1}{1+Z_{\alpha/2}/\sqrt{2n}}\sqrt{\frac{\overline{Y}_n}{\overline{X}_n}}, \frac{1}{1-Z_{\alpha/2}/\sqrt{2n}}\sqrt{\frac{\overline{Y}_n}{\overline{X}_n}}\right].$$

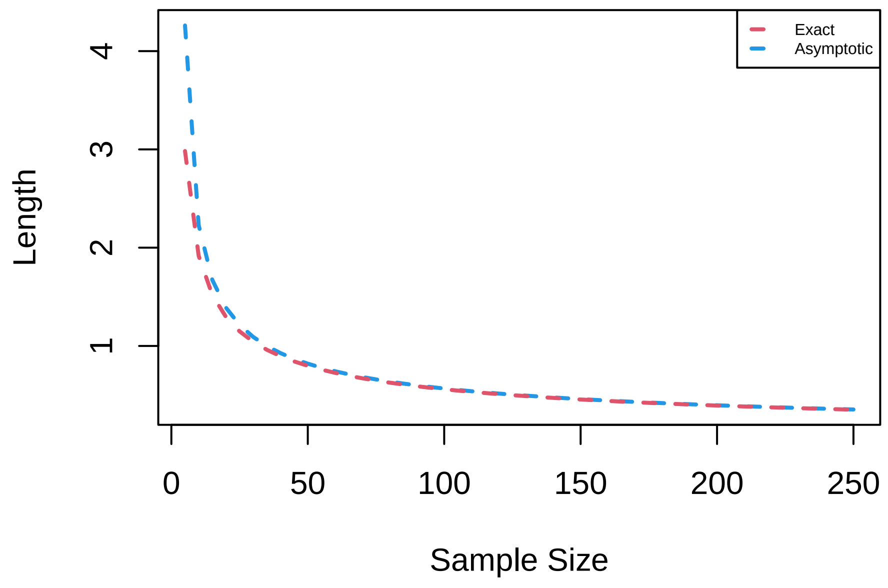
The average length of the asymptotic confidence interval is initially greater than that of the corresponding exact confidence interval, but this difference in lengths vanishes as the sample size increases.

```
n = seq(5, 250, step)
X = matrix(0, 0, nsim)
Y = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, matrix(rexp(step * nsim, lambda), step))
    Y = rbind(Y, matrix(rexp(step * nsim, lambda^(-1)), step))
    MLE = sqrt(colSums(Y)/colSums(X))
    CIExact = cbind(sqrt(qf(alpha/2, 2 * n[k], 2 * n[k])) * MLE, sqrt(qf(1 -
        alpha/2, 2 * n[k], 2 * n[k])) * MLE)
    CIAsymptotic = cbind(sqrt(colSums(Y)/colSums(X))/(1 + qnorm(1 - alpha/2)/sqrt(2 *
        n[k])), sqrt(colSums(Y)/colSums(X))/(1 - qnorm(1 - alpha/2)/sqrt(2 *
        n[k])))
    Coverage[k, 1] = mean(CIExact[, 1] < lambda & lambda < CIExact[, 2])
    Length[k, 1] = mean(CIExact[, 2] - CIExact[, 1])
    Coverage[k, 2] = mean(CIAsymptotic[, 1] < lambda & lambda < CIAsymptotic[,
        2])
    Length[k, 2] = mean(CIAsymptotic[, 2] - CIAsymptotic[, 1])
}
plot(n, Coverage[, 1], "l", ylim = range(Coverage), xlab = "Sample Size",
    ylab = "Coverage", col = 2, lty = 2, lwd = 2)
lines(n, Coverage[, 2], col = 4, lty = 2, lwd = 2)
abline(h = 0.95, lty = 2, lwd = 2)
```

```
legend("bottomright", c("Exact", "Asymptotic", "Nominal"), col = c(2, 4,
    1), lty = rep(2, 3), lwd = rep(2, 3), cex = 0.5)
```



```
plot(n, Length[, 2], "l", xlab = "Sample Size", ylab = "Length", col = 4,
    lty = 2, lwd = 2)
lines(n, Length[, 1], col = 2, lty = 2, lwd = 2)
legend("topright", c("Exact", "Asymptotic"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```



Example 2.5. Let $X_1, X_2, \ldots, X_n$ be a random sample with $f(x;\lambda,\vartheta) = \lambda e^{-\lambda(x-\vartheta)}$ for $x \geqslant \vartheta$, $\lambda > 0$ and $\vartheta \in \mathbb{R}$. Then, we know that:

$$\mathcal{I}_{\lambda;\,1-\alpha}(X;\vartheta) = \left[\frac{\chi^2_{2n;1-\alpha/2}}{2n\left(\overline{X}-\vartheta\right)}, \frac{\chi^2_{2n;\alpha/2}}{2n\left(\overline{X}-\vartheta\right)}\right].$$

```
n = 100
theta = -1
X = matrix(rexp(n * nsim, lambda), n) + theta
CI = cbind(qchisq(alpha/2, 2 * n)/(2 * (colSums(X) - n * theta)), qchisq(1 -
    alpha/2, 2 * n)/(2 * (colSums(X) - n * theta)))
mean(CI[, 1] < lambda & lambda < CI[, 2])
```

```
## [1] 0.95061
```

Furthermore, we know that:

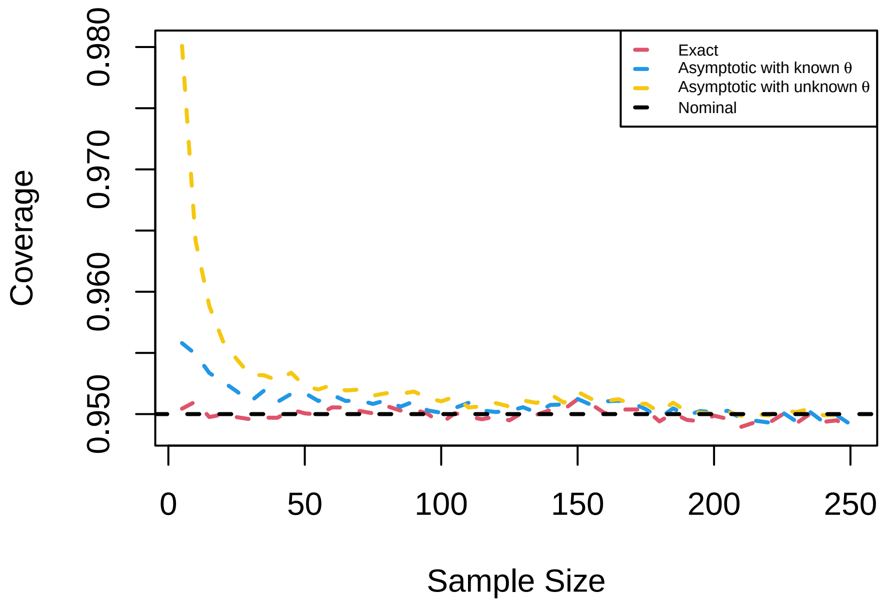
$$\mathcal{I}^{(n)}_{\lambda;\,1-\alpha}(X;\vartheta) = \left[\frac{1}{\overline{X}_n - \vartheta}\left(1 - \frac{1}{\sqrt{n}}Z_{\alpha/2}\right), \frac{1}{\overline{X}_n - \vartheta}\left(1 + \frac{1}{\sqrt{n}}Z_{\alpha/2}\right)\right],$$

$$\mathcal{I}^{(n)}_{\lambda;\,1-\alpha}(X) = \left[\frac{1}{\overline{X}_n - X_{(1)}}\left(1 - \frac{1}{\sqrt{n}}Z_{\alpha/2}\right), \frac{1}{\overline{X}_n - X_{(1)}}\left(1 + \frac{1}{\sqrt{n}}Z_{\alpha/2}\right)\right].$$

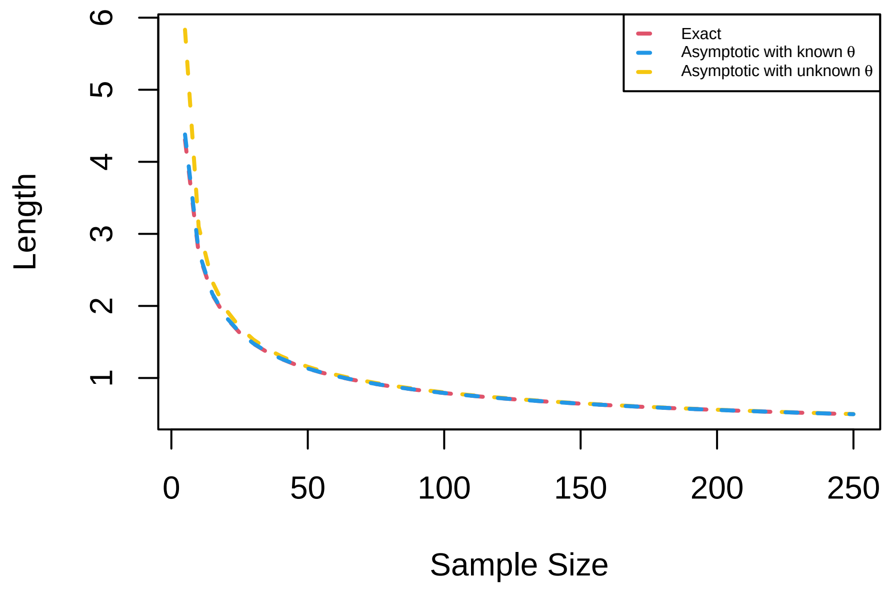
We observe that the asymptotic confidence interval for λ with unknown ϑ has significantly higher than nominal coverage rate for small sample sizes, whereas the asymptotic confidence interval for λ with known ϑ only has slightly higher than nominal coverage rate for small sample sizes. Accordingly, the average length of the asymptotic confidence interval with unknown ϑ is initially greater than that of the corresponding exact confidence interval, whereas the average length of the asymptotic confidence interval with known ϑ almost always coincides with that of the corresponding exact confidence interval.

```
n = seq(5, 250, step)
X = matrix(0, 0, nsim)
Coverage = matrix(0, 50, 3)
Length = matrix(0, 50, 3)
for (k in 1:50) {
    X = rbind(X, matrix(rexp(step * nsim, lambda), step) + theta)
    CIExact = cbind(qchisq(alpha/2, 2 * n[k])/(2 * (colSums(X) - n[k] *
        theta)), qchisq(1 - alpha/2, 2 * n[k])/(2 * (colSums(X) - n[k] *
        theta)))
    CIKnownAsymptotic = cbind((1 - qnorm(1 - alpha/2)/sqrt(n[k]))/(colMeans(X) -
        theta), (1 + qnorm(1 - alpha/2)/sqrt(n[k]))/(colMeans(X) - theta))
    CIUnknownAsymptotic = cbind((1 - qnorm(1 - alpha/2)/sqrt(n[k]))/(colMeans(X) -
        apply(X, 2, min)), (1 + qnorm(1 - alpha/2)/sqrt(n[k]))/(colMeans(X) -
        apply(X, 2, min)))
    Coverage[k, 1] = mean(CIExact[, 1] < lambda & lambda < CIExact[, 2])
    Length[k, 1] = mean(CIExact[, 2] - CIExact[, 1])
    Coverage[k, 2] = mean(CIKnownAsymptotic[, 1] < lambda & lambda < CIKnownAsymptotic[,
        2])
    Length[k, 2] = mean(CIKnownAsymptotic[, 2] - CIKnownAsymptotic[, 1])
    Coverage[k, 3] = mean(CIUnknownAsymptotic[, 1] < lambda & lambda <
        CIUnknownAsymptotic[, 2])
    Length[k, 3] = mean(CIUnknownAsymptotic[, 2] - CIUnknownAsymptotic[,
```

```r
        1])
}
plot(n, Coverage[, 1], "l", ylim = range(Coverage), xlab = "Sample Size",
    ylab = "Coverage", col = 2, lty = 2, lwd = 2)
lines(n, Coverage[, 2], col = 4, lty = 2, lwd = 2)
lines(n, Coverage[, 3], col = 7, lty = 2, lwd = 2)
abline(h = 0.95, lty = 2, lwd = 2)
legend("topright", c("Exact", expression("Asymptotic with known" ~ theta),
    expression("Asymptotic with unknown" ~ theta), "Nominal"), col = c(2,
    4, 7, 1), lty = rep(2, 4), lwd = rep(2, 4), cex = 0.5)
```



```r
plot(n, Length[, 3], "l", xlab = "Sample Size", ylab = "Length", col = 7,
    lty = 2, lwd = 2)
lines(n, Length[, 1], col = 2, lty = 2, lwd = 2)
lines(n, Length[, 2], col = 4, lty = 2, lwd = 2)
legend("topright", c("Exact", expression("Asymptotic with known" ~ theta),
    expression("Asymptotic with unknown" ~ theta)), col = c(2, 4, 7), lty = rep(2,
    3), lwd = rep(2, 3), cex = 0.5)
```



Additionally, we know that:

$$\mathcal{I}^{\mathrm{ET}}_{\vartheta;\,1-\alpha}(X;\lambda) = \left[X_{(1)} + \frac{1}{n\lambda}\log\frac{\alpha}{2}, X_{(1)} + \frac{1}{n\lambda}\log\left(1-\frac{\alpha}{2}\right)\right],$$

$$\mathcal{I}^{\mathrm{ML}}_{\vartheta;\,1-\alpha}(X;\lambda) = \left[X_{(1)} + \frac{1}{n\lambda}\log\alpha, X_{(1)}\right].$$

The minimum length confidence interval obviously displays a shorter average length than the corresponding equal-tailed confidence interval for ϑ, while both confidence intervals boast empirical coverage rates which are really close to nominal.

```
n = 100
X = matrix(rexp(n * nsim, lambda), n) + theta
MLE = apply(X, 2, min)
CIET = cbind(MLE + log(alpha/2)/(n * lambda), MLE + log(1 - alpha/2)/(n *
    lambda))
CIML = cbind(MLE + log(alpha)/(n * lambda), MLE)
mean(CIET[, 1] < theta & theta < CIET[, 2])
```

```
## [1] 0.94999
```

```
mean(CIML[, 1] < theta & theta < CIML[, 2])
```

```
## [1] 0.94945
```

```
mean(CIET[, 2] - CIET[, 1])
```

```
## [1] 0.01831781
```

```
mean(CIML[, 2] - CIML[, 1])
```

```
## [1] 0.01497866
```

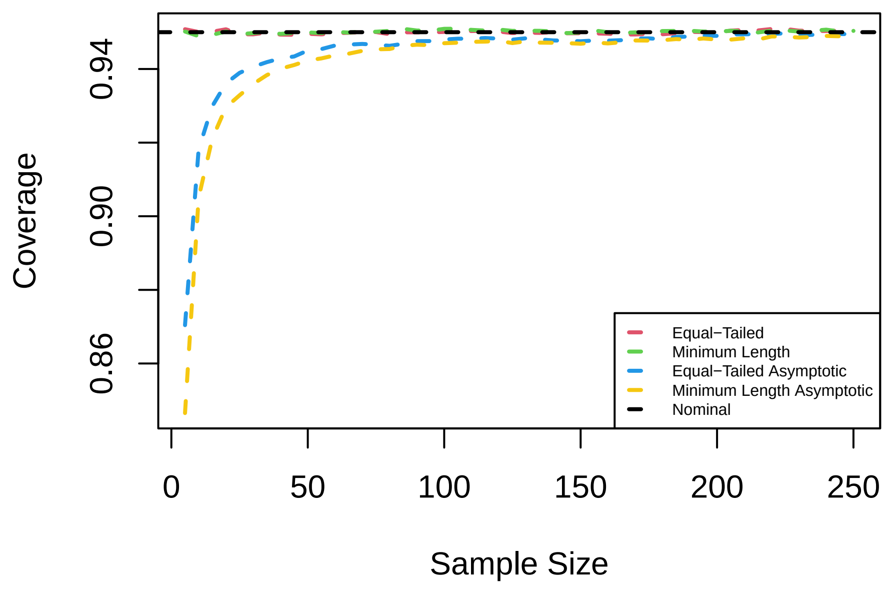
Lastly, we know that:

$$\mathcal{I}^{\mathrm{ET}_n}_{\vartheta;\,1-\alpha}(X) = \left[X_{(1)} + \frac{\overline{X}_n - X_{(1)}}{n}\log\frac{\alpha}{2}, X_{(1)} + \frac{\overline{X}_n - X_{(1)}}{n}\log\left(1-\frac{\alpha}{2}\right)\right],$$

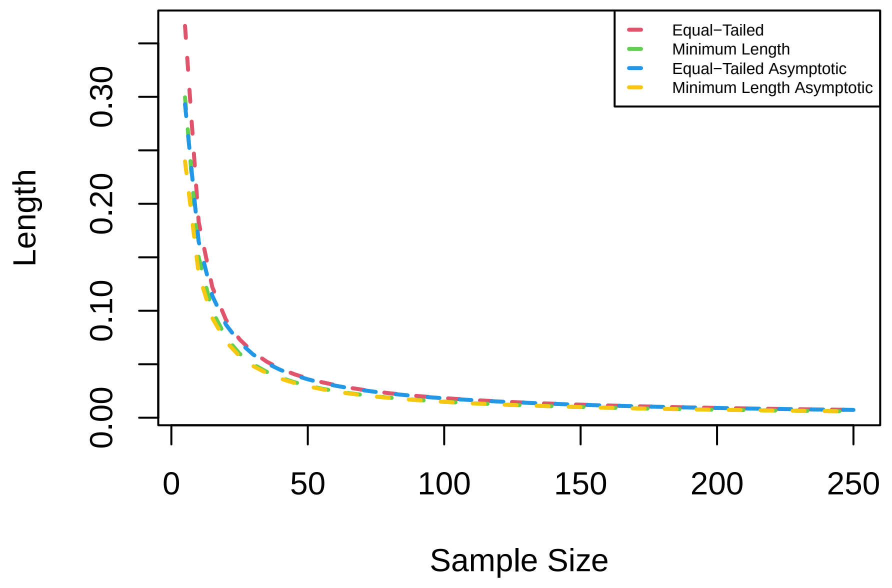
$$\mathcal{I}^{\mathrm{ML}_n}_{\vartheta;\,1-\alpha}(X) = \left[X_{(1)} + \frac{\overline{X}_n - X_{(1)}}{n}\log\alpha, X_{(1)}\right].$$

The minimum length asymptotic confidence interval for ϑ has lower coverage rate than the corresponding equal-tailed asymptotic confidence interval, which in turn has lower than nominal coverage rate, for small sample sizes.

```r
n = seq(5, 250, step)
X = matrix(0, 0, nsim)
Coverage = matrix(0, 50, 4)
Length = matrix(0, 50, 4)
for (k in 1:50) {
    X = rbind(X, matrix(rexp(step * nsim, lambda), step) + theta)
    MLE = apply(X, 2, min)
    CIET = cbind(MLE + log(alpha/2)/(n[k] * lambda), MLE + log(1 - alpha/2)/(n[k] *
        lambda))
    CIML = cbind(MLE + log(alpha)/(n[k] * lambda), MLE)
    CIETAsymptotic = cbind(MLE + log(alpha/2) * (colMeans(X) - MLE)/n[k],
        MLE + log(1 - alpha/2) * (colMeans(X) - MLE)/n[k])
    CIMLAsymptotic = cbind(MLE + log(alpha) * (colMeans(X) - MLE)/n[k],
        MLE)
    Coverage[k, 1] = mean(CIET[, 1] < theta & theta < CIET[, 2])
    Length[k, 1] = mean(CIET[, 2] - CIET[, 1])
    Coverage[k, 2] = mean(CIML[, 1] < theta & theta < CIML[, 2])
    Length[k, 2] = mean(CIML[, 2] - CIML[, 1])
    Coverage[k, 3] = mean(CIETAsymptotic[, 1] < theta & theta < CIETAsymptotic[,
        2])
    Length[k, 3] = mean(CIETAsymptotic[, 2] - CIETAsymptotic[, 1])
    Coverage[k, 4] = mean(CIMLAsymptotic[, 1] < theta & theta < CIMLAsymptotic[,
        2])
    Length[k, 4] = mean(CIMLAsymptotic[, 2] - CIMLAsymptotic[, 1])
}
plot(n, Coverage[, 1], "l", ylim = range(Coverage), xlab = "Sample Size",
    ylab = "Coverage", col = 2, lty = 2, lwd = 2)
lines(n, Coverage[, 2], col = 3, lty = 2, lwd = 2)
lines(n, Coverage[, 3], col = 4, lty = 2, lwd = 2)
lines(n, Coverage[, 4], col = 7, lty = 2, lwd = 2)
abline(h = 0.95, lty = 2, lwd = 2)
legend("bottomright", c("Equal-Tailed", "Minimum Length", "Equal-Tailed Asymptotic",
    "Minimum Length Asymptotic", "Nominal"), col = c(2, 3, 4, 7, 1), lty = rep(2,
    5), lwd = rep(2, 5), cex = 0.5)
```



```r
plot(n, Length[, 1], "l", xlab = "Sample Size", ylab = "Length", col = 2,
    lty = 2, lwd = 2)
lines(n, Length[, 2], col = 3, lty = 2, lwd = 2)
lines(n, Length[, 3], col = 4, lty = 2, lwd = 2)
lines(n, Length[, 4], col = 7, lty = 2, lwd = 2)
legend("topright", c("Equal-Tailed", "Minimum Length", "Equal-Tailed Asymptotic",
    "Minimum Length Asymptotic"), col = c(2, 3, 4, 7), lty = rep(2, 4),
    lwd = rep(2, 4), cex = 0.5)
```



Example 2.6. Let $X_1, X_2, \ldots, X_n$ be a random sample with $f(x;\vartheta) = e^{-(x-\vartheta)}$ for $x \geqslant \vartheta$ and $\vartheta < 0$. We want to estimate the parametric function $g(\vartheta) = \mathbb{P}_\vartheta(X_1 < 0) = 1 - e^{\vartheta}$. Then, we know that:

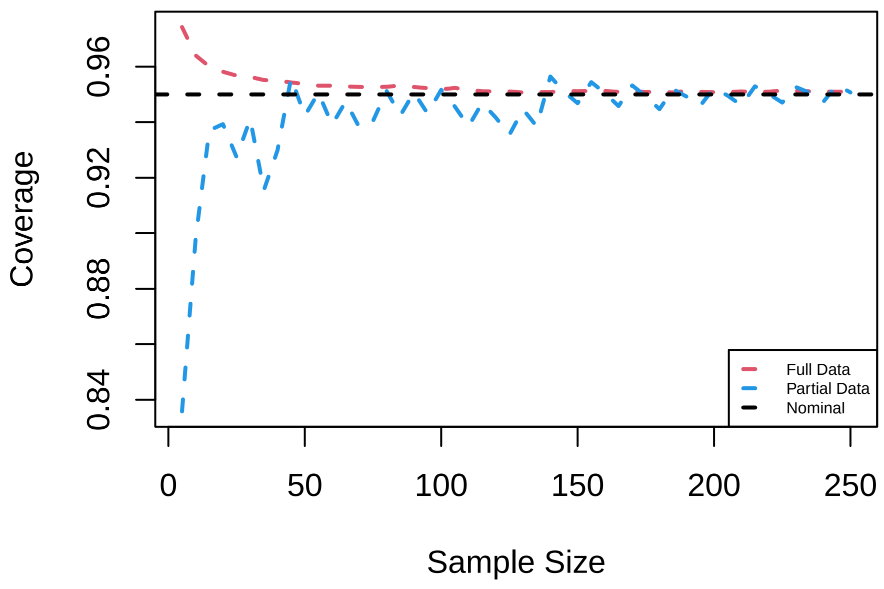
$$\widehat{g(\vartheta)} = 1 - e^{\min\{X_{(1)},0\}}, \quad \mathcal{I}^{\mathrm{FD}_n}_{1-e^{\vartheta};\,1-\alpha}(X) = \left[\widehat{g(\vartheta)} - \frac{1-\widehat{g(\vartheta)}}{n}\log\left(1-\frac{\alpha}{2}\right), \widehat{g(\vartheta)} - \frac{1-\widehat{g(\vartheta)}}{n}\log\frac{\alpha}{2}\right].$$

Suppose that we instead only observe the values of the random variable $W = \sum_{i=1}^n \mathbb{1}_{[\vartheta,0)}(X_i)$ and those of the random variables $X_1, X_2, \ldots, X_n$ which are positive. Then, we know that:

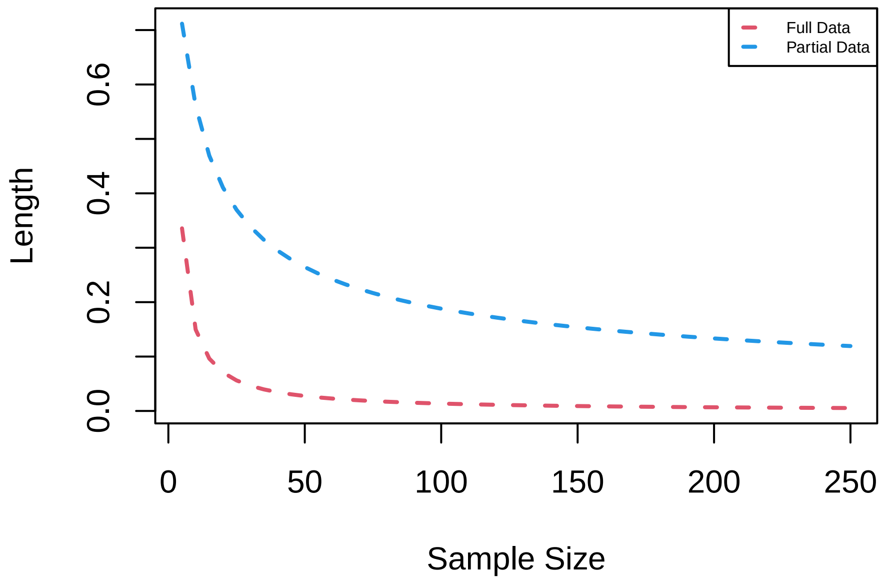
$$\widehat{g(\vartheta)} = \frac{1}{n}W, \quad \mathcal{I}^{\mathrm{PD}_n}_{1-e^{\vartheta};\,1-\alpha}(W, X) = \left[\widehat{g(\vartheta)} - Z_{\alpha/2}\sqrt{\frac{1}{n}\widehat{g(\vartheta)}\left(1-\widehat{g(\vartheta)}\right)}, \widehat{g(\vartheta)} + Z_{\alpha/2}\sqrt{\frac{1}{n}\widehat{g(\vartheta)}\left(1-\widehat{g(\vartheta)}\right)}\right].$$

For $\vartheta = -1$, the full-data asymptotic confidence interval for ϑ has higher than nominal coverage rate, whereas the partial-data asymptotic confidence interval has lower than nominal coverage rate for small sample sizes. Additionally, the full-data asymptotic confidence interval always has shorter length than the corresponding partial-data asymptotic confidence interval.

```r
X = matrix(0, 0, nsim)
Coverage = matrix(0, 50, 2)
Length = matrix(0, 50, 2)
for (k in 1:50) {
    X = rbind(X, matrix(rexp(step * nsim, 1), step) + theta)
    W = colSums(X < 0)
    MLE = 1 - exp(pmin(apply(X, 2, min), 0))
    CIFull = cbind(MLE - log(1 - alpha/2) * (1 - MLE)/n[k], MLE - log(alpha/2) *
        (1 - MLE)/n[k])
    CIPartial = cbind(W/n[k] - qnorm(1 - alpha/2) * sqrt(W * (n[k] - W)/n[k]^3),
        W/n[k] + qnorm(1 - alpha/2) * sqrt(W * (n[k] - W)/n[k]^3))
    Coverage[k, 1] = mean(CIFull[, 1] < 1 - exp(theta) & 1 - exp(theta) <
        CIFull[, 2])
    Length[k, 1] = mean(CIFull[, 2] - CIFull[, 1])
    Coverage[k, 2] = mean(CIPartial[, 1] < 1 - exp(theta) & 1 - exp(theta) <
        CIPartial[, 2])
    Length[k, 2] = mean(CIPartial[, 2] - CIPartial[, 1])
}
plot(n, Coverage[, 1], "l", ylim = range(Coverage), xlab = "Sample Size",
    ylab = "Coverage", col = 2, lty = 2, lwd = 2)
lines(n, Coverage[, 2], col = 4, lty = 2, lwd = 2)
abline(h = 0.95, lty = 2, lwd = 2)
legend("bottomright", c("Full Data", "Partial Data", "Nominal"), col = c(2,
    4, 1), lty = rep(2, 3), lwd = rep(2, 3), cex = 0.5)
```

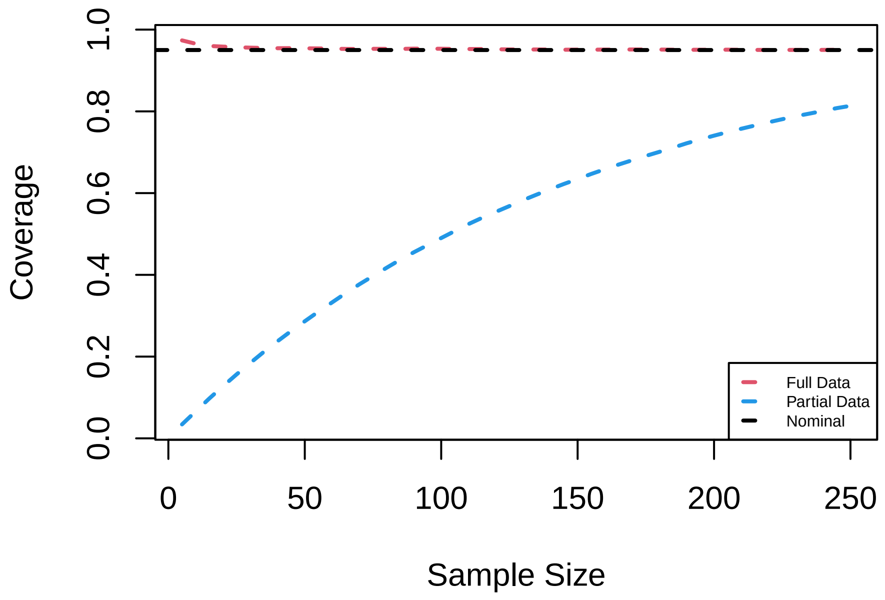


```
plot(n, Length[, 1], "l", ylim = range(Length), xlab = "Sample Size", ylab = "Length",
    col = 2, lty = 2, lwd = 2)
lines(n, Length[, 2], col = 4, lty = 2, lwd = 2)
legend("topright", c("Full Data", "Partial Data"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```

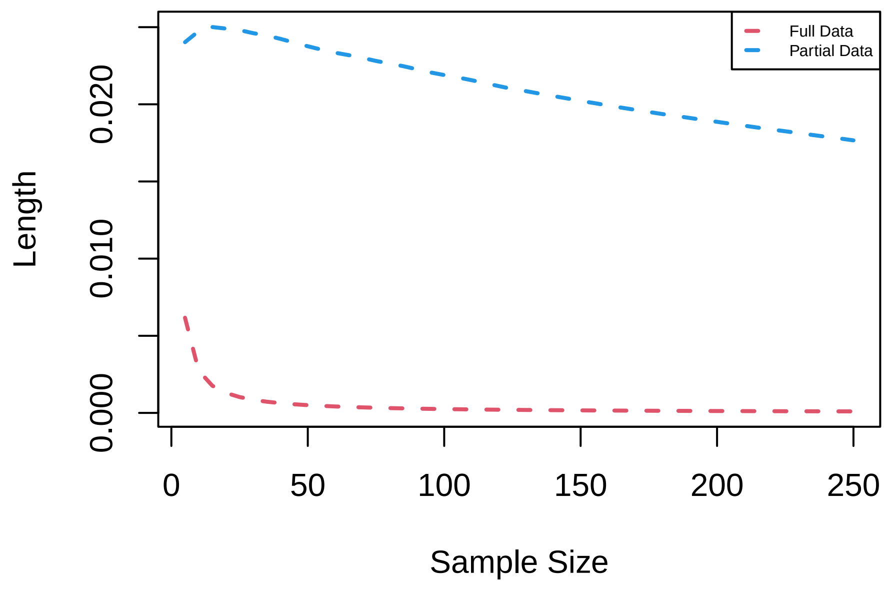


For $\vartheta = -5$, the full-data asymptotic confidence interval for ϑ still behaves the same way, but the partial-data asymptotic confidence interval always has lower than nominal coverage rate, even for a large sample size of $n = 250$ observations, and considerably greater length. That's because the true probability to be estimated is very close to 1, i.e. $g(\vartheta) \approx 0.99$.

```
theta = -5
X = matrix(0, 0, nsim)
for (k in 1:50) {
    X = rbind(X, matrix(rexp(step * nsim, 1), step) + theta)
    W = colSums(X < 0)
    MLE = 1 - exp(pmin(apply(X, 2, min), 0))
    CIFull = cbind(MLE - log(1 - alpha/2) * (1 - MLE)/n[k], MLE - log(alpha/2) *
        (1 - MLE)/n[k])
    CIPartial = cbind(W/n[k] - qnorm(1 - alpha/2) * sqrt(W * (n[k] - W)/n[k]^3),
        W/n[k] + qnorm(1 - alpha/2) * sqrt(W * (n[k] - W)/n[k]^3))
    Coverage[k, 1] = mean(CIFull[, 1] < 1 - exp(theta) & 1 - exp(theta) <
        CIFull[, 2])
    Length[k, 1] = mean(CIFull[, 2] - CIFull[, 1])
    Coverage[k, 2] = mean(CIPartial[, 1] < 1 - exp(theta) & 1 - exp(theta) <
        CIPartial[, 2])
    Length[k, 2] = mean(CIPartial[, 2] - CIPartial[, 1])
}
plot(n, Coverage[, 1], "l", ylim = range(Coverage), xlab = "Sample Size",
    ylab = "Coverage", col = 2, lty = 2, lwd = 2)
lines(n, Coverage[, 2], col = 4, lty = 2, lwd = 2)
abline(h = 0.95, lty = 2, lwd = 2)
legend("bottomright", c("Full Data", "Partial Data", "Nominal"), col = c(2,
    4, 1), lty = rep(2, 3), lwd = rep(2, 3), cex = 0.5)
```



```
plot(n, Length[, 1], "l", ylim = range(Length), xlab = "Sample Size", ylab = "Length",
    col = 2, lty = 2, lwd = 2)
lines(n, Length[, 2], col = 4, lty = 2, lwd = 2)
legend("topright", c("Full Data", "Partial Data"), col = c(2, 4), lty = c(2,
    2), lwd = c(2, 2), cex = 0.5)
```



Example 2.7. Let $X_1, \ldots, X_n \sim \mathcal{U}(0, \vartheta)$ be a random sample. Then, we know that:

$$\mathcal{I}^{\text{ET}}_{\vartheta;\, 1-\alpha}(X) = \left[X_{(n)}\left(1 - \frac{\alpha}{2}\right)^{-1/n}, X_{(n)}\left(\frac{\alpha}{2}\right)^{-1/n}\right],$$

$$\mathcal{I}^{\text{ML}}_{\vartheta;\, 1-\alpha}(X) = \left[X_{(n)}, X_{(n)}\alpha^{-1/n}\right].$$

```
n = 100
theta = 2
X = matrix(runif(n * nsim, max = theta), n)
MLE = apply(X, 2, max)
CIET = cbind(MLE * (1 - alpha/2)^(-n^(-1)), MLE * (alpha/2)^(-n^(-1)))
CIML = cbind(MLE, MLE * alpha^(-n^(-1)))
mean(CIET[, 1] < theta & theta < CIET[, 2])
```

```
## [1] 0.95018
```

```
mean(CIML[, 1] < theta & theta < CIML[, 2])
```

```
## [1] 0.95028
```

```
mean(CIET[, 2] - CIET[, 1])
```

```
## [1] 0.0739118
```

```
mean(CIML[, 2] - CIML[, 1])
```

```
## [1] 0.0602206
```

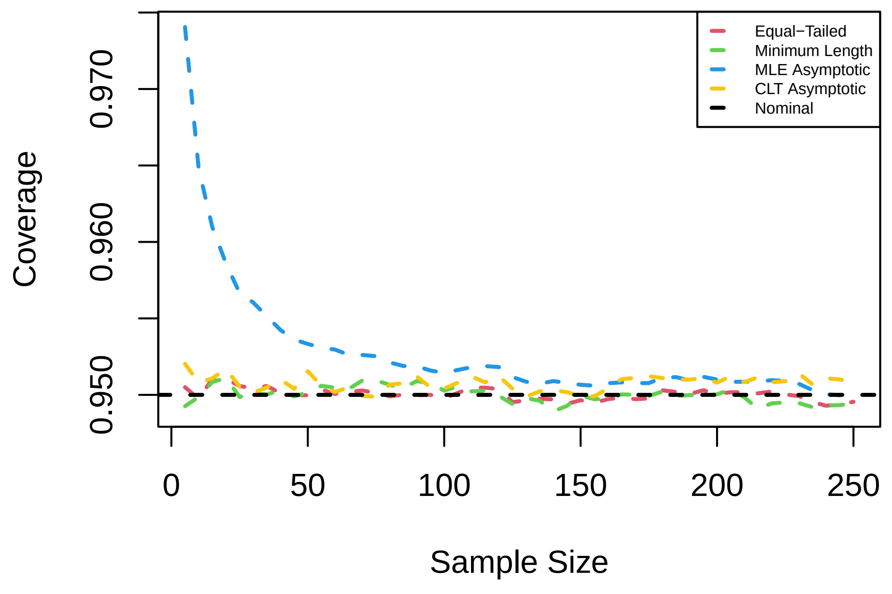
Furthermore, we know that:

$$\mathcal{I}^{\text{MLE}_n}_{\vartheta;\, 1-\alpha}(X) = \left[\frac{X_{(n)}}{1 + \log(1 - \alpha/2)/n}, \frac{X_{(n)}}{1 + \log(\alpha/2)/n}\right],$$

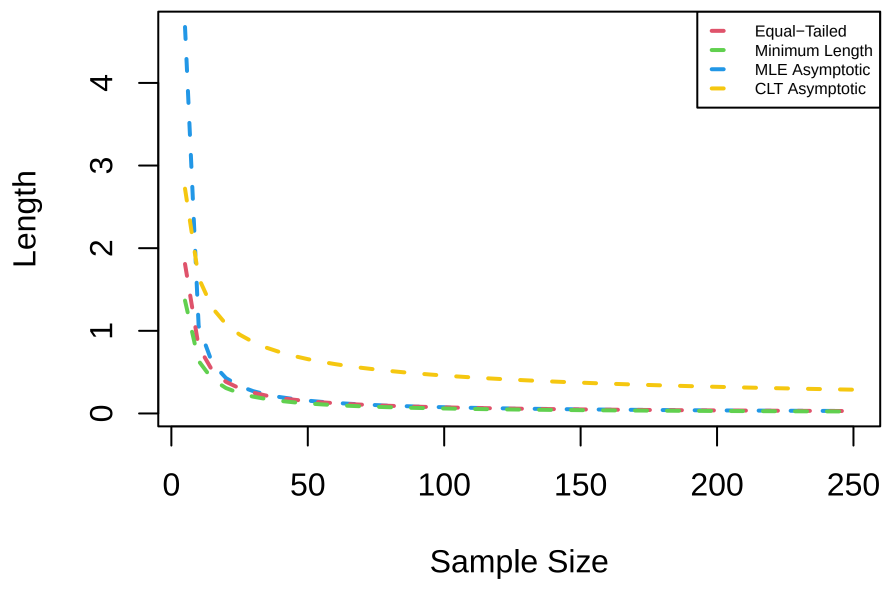
$$\mathcal{I}_{\vartheta;\,1-\alpha}^{\mathrm{CLT}_n}(X) = \left[\frac{2\overline{X}_n}{1+Z_{\alpha/2}/\sqrt{3n}}, \frac{2\overline{X}_n}{1-Z_{\alpha/2}/\sqrt{3n}}\right].$$

The asymptotic confidence interval based on the MLE of ϑ initially displays significantly greater length and higher than nominal coverage rate for small sample sizes, but the difference in lengths with the exact confidence intervals quickly vanishes as the sample size increases. In contrast, the asymptotic confidence interval based on the central limit theorem always has close to nominal coverage rate, but also always displays greater length than the corresponding exact confidence intervals.

```
n = seq(5, 250, step)
X = matrix(0, 0, nsim)
Coverage = matrix(0, 50, 4)
Length = matrix(0, 50, 4)
for (k in 1:50) {
    X = rbind(X, matrix(runif(step * nsim, max = theta), step))
    MLE = apply(X, 2, max)
    CIET = cbind(MLE * (1 - alpha/2)^(-n[k]^(-1)), MLE * (alpha/2)^(-n[k]^(-1)))
    CIML = cbind(MLE, MLE * alpha^(-n[k]^(-1)))
    CIMLEAsymptotic = cbind(MLE/(1 - qchisq(alpha/2, 2)/(2 * n[k])), MLE/(1 -
        qchisq(1 - alpha/2, 2)/(2 * n[k])))
    CICLTAsymptotic = cbind(2 * colMeans(X)/(1 + qnorm(1 - alpha/2)/sqrt(3 *
        n[k])), 2 * colMeans(X)/(1 - qnorm(1 - alpha/2)/sqrt(3 * n[k])))
    Coverage[k, 1] = mean(CIET[, 1] < theta & theta < CIET[, 2])
    Length[k, 1] = mean(CIET[, 2] - CIET[, 1])
    Coverage[k, 2] = mean(CIML[, 1] < theta & theta < CIML[, 2])
    Length[k, 2] = mean(CIML[, 2] - CIML[, 1])
    Coverage[k, 3] = mean(CIMLEAsymptotic[, 1] < theta & theta < CIMLEAsymptotic[,
        2])
    Length[k, 3] = mean(CIMLEAsymptotic[, 2] - CIMLEAsymptotic[, 1])
    Coverage[k, 4] = mean(CICLTAsymptotic[, 1] < theta & theta < CICLTAsymptotic[,
        2])
    Length[k, 4] = mean(CICLTAsymptotic[, 2] - CICLTAsymptotic[, 1])
}
plot(n, Coverage[, 1], "l", ylim = range(Coverage), xlab = "Sample Size",
    ylab = "Coverage", col = 2, lty = 2, lwd = 2)
lines(n, Coverage[, 2], col = 3, lty = 2, lwd = 2)
lines(n, Coverage[, 3], col = 4, lty = 2, lwd = 2)
lines(n, Coverage[, 4], col = 7, lty = 2, lwd = 2)
abline(h = 0.95, lty = 2, lwd = 2)
legend("topright", c("Equal-Tailed", "Minimum Length", "MLE Asymptotic",
    "CLT Asymptotic", "Nominal"), col = c(2, 3, 4, 7, 1), lty = rep(2,
    5), lwd = rep(2, 5), cex = 0.5)
```



```r
plot(n, Length[, 3], "l", xlab = "Sample Size", ylab = "Length", col = 4,
    lty = 2, lwd = 2)
lines(n, Length[, 1], col = 2, lty = 2, lwd = 2)
lines(n, Length[, 2], col = 3, lty = 2, lwd = 2)
lines(n, Length[, 4], col = 7, lty = 2, lwd = 2)
legend("topright", c("Equal-Tailed", "Minimum Length", "MLE Asymptotic",
    "CLT Asymptotic"), col = c(2, 3, 4, 7), lty = rep(2, 4), lwd = rep(2,
    4), cex = 0.5)
```



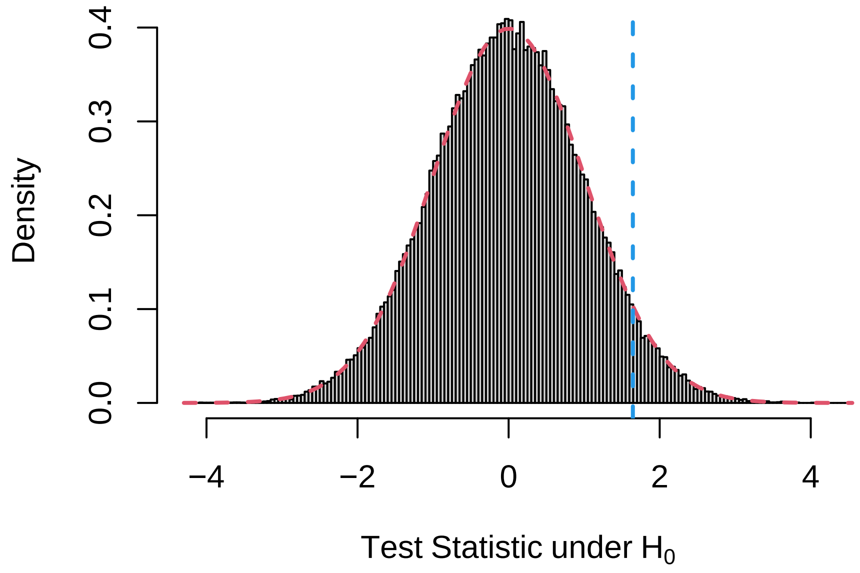
3 Statistical Hypothesis Testing

We want to verify that the hypothesis tests which we're conducting have accurate type I error control and to compare the power of different tests for the same hypotheses. In order to achieve that, we first have to generate n_{sim} independent random samples under the null hypothesis and compute the observed value of the test statistic for each of the generated samples. Then, we can calculate the empirical type I error rate of the hypothesis test as the percentage of the observed test statistics which lead to the rejection of the null hypothesis. Similarly, we can generate independent random samples under the alternative hypothesis and estimate the power of the test as the percentage of samples which lead to the rejection of the null hypothesis. Lastly, we might be interested in exploring what the distribution of the test statistic and the p-value of the test look like under the null vs. under the alternative hypothesis.

```
n = 100
nsim = 1e+05
mu0 = 1
sigma = 6
alpha = 0.05
X = matrix(rnorm(n * nsim, mu0, sigma), n)
MLE = colMeans(X)
z = (MLE - mu0) * sqrt(n)/sigma
mean(z > qnorm(1 - alpha))
```

```
## [1] 0.04881
```

```
hist(z, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[0]))
curve(dnorm(x), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qnorm(1 - alpha), col = 4, lty = 2, lwd = 2)
```

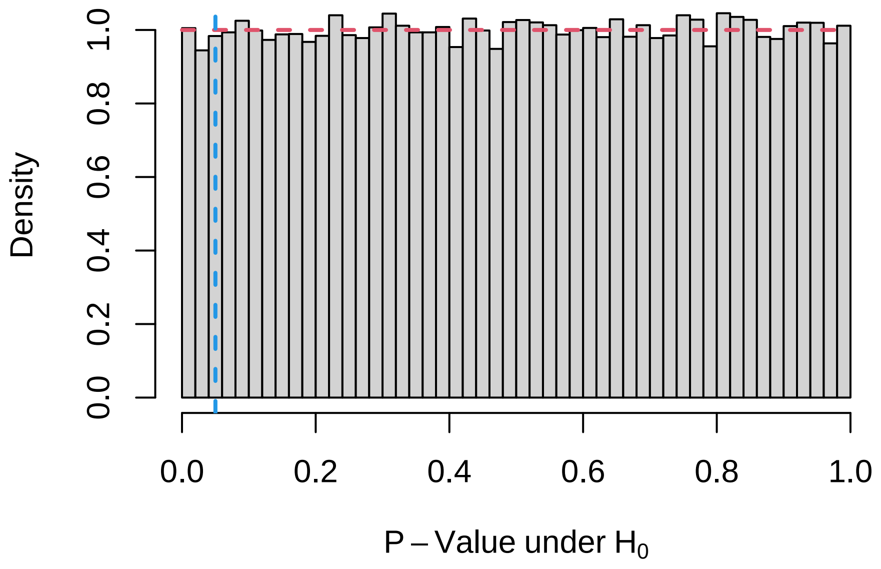


```
pvalue = pnorm(z, lower.tail = FALSE)
mean(pvalue < alpha)
```

```
## [1] 0.04881
```

```
hist(pvalue, "FD", freq = FALSE, main = NA, xlim = c(0, 1), xlab = expression(P -
    Value ~ under ~ H[0]))
curve(dunif(x), add = TRUE, col = 2, lty = 2, lwd = 2)
```

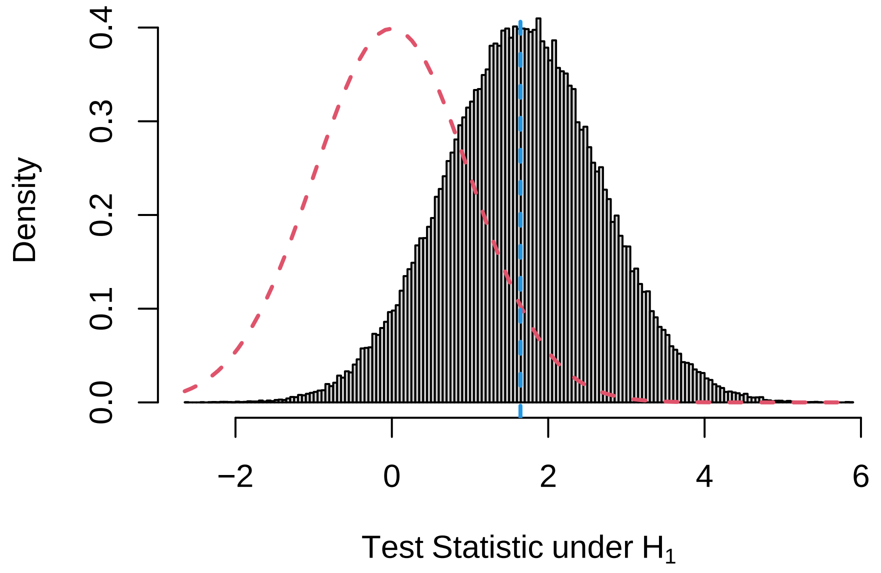
```
abline(v = alpha, col = 4, lty = 2, lwd = 2)
```



```
mu = 2
Y = matrix(rnorm(n * nsim, mu, sigma), n)
MLE = colMeans(Y)
z = (MLE - mu0) * sqrt(n)/sigma
mean(z > qnorm(1 - alpha))
```

```
## [1] 0.51014
```

```
hist(z, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[1]))
curve(dnorm(x), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qnorm(1 - alpha), col = 4, lty = 2, lwd = 2)
```



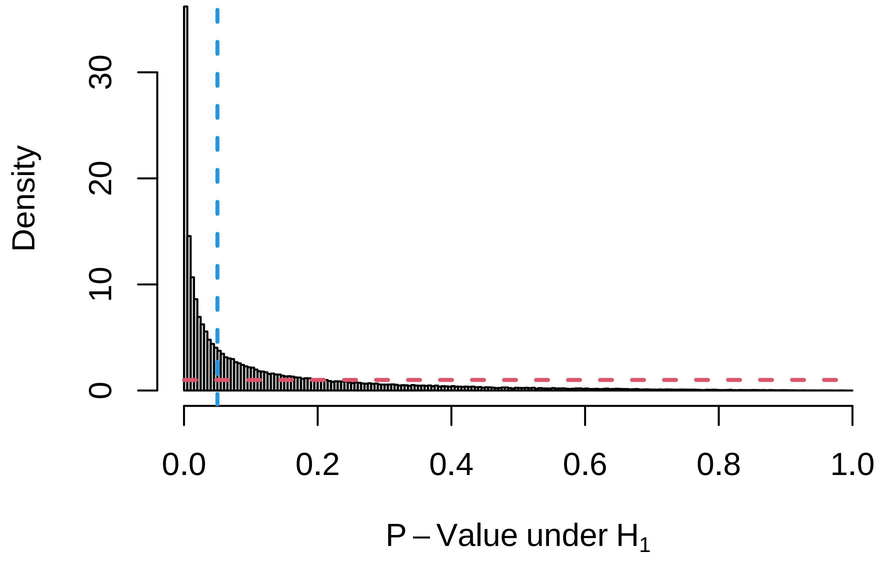
```
pvalue = pnorm(z, lower.tail = FALSE)
mean(pvalue < alpha)
```

```
## [1] 0.51014
```

```
pnorm(qnorm(1 - alpha) - (mu - mu0) * sqrt(n)/sigma, lower.tail = FALSE)
```

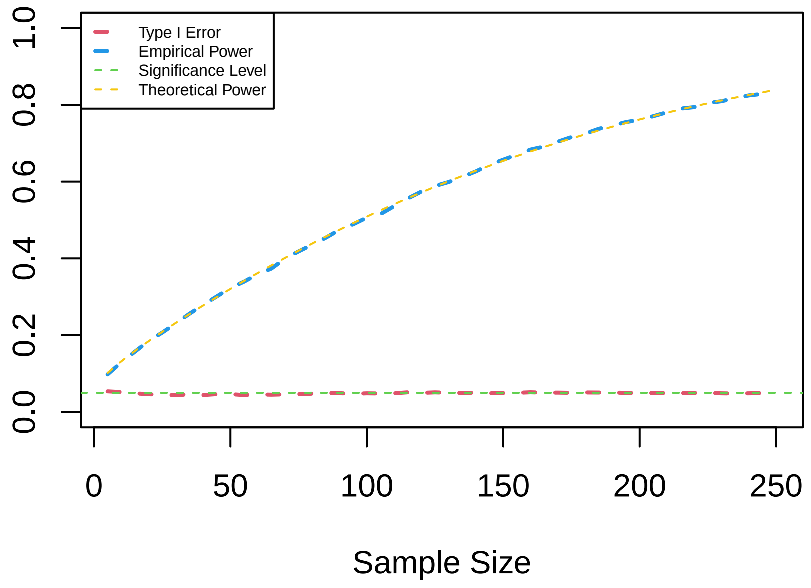
```
## [1] 0.5087015
```

```
hist(pvalue, "FD", freq = FALSE, main = NA, xlab = expression(P - Value ~
    under ~ H[1]))
curve(dunif(x), add = TRUE, xlim = c(0, 1), col = 2, lty = 2, lwd = 2)
abline(v = alpha, col = 4, lty = 2, lwd = 2)
```

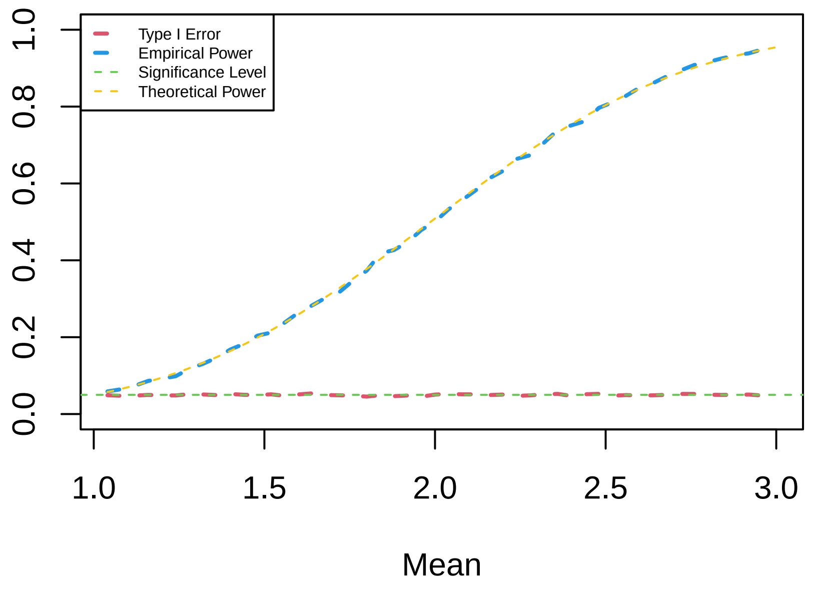


```
step = 5
n = seq(5, 250, step)
nsim = 10000
X = matrix(0, 0, nsim)
Y = matrix(0, 0, nsim)
Error = numeric(50)
Power = numeric(50)
for (k in 1:50) {
    X = rbind(X, matrix(rnorm(step * nsim, mu0, sigma), step))
    MLE = colMeans(X)
    z = (MLE - mu0) * sqrt(n[k])/sigma
    Error[k] = mean(z > qnorm(1 - alpha))
    Y = rbind(Y, matrix(rnorm(step * nsim, mu, sigma), step))
    MLE = colMeans(Y)
    z = (MLE - mu0) * sqrt(n[k])/sigma
    Power[k] = mean(z > qnorm(1 - alpha))
}
plot(n, Power, "l", ylim = c(0, 1), xlab = "Sample Size", ylab = NA, col = 4,
    lty = 2, lwd = 2)
lines(n, Error, col = 2, lty = 2, lwd = 2)
```

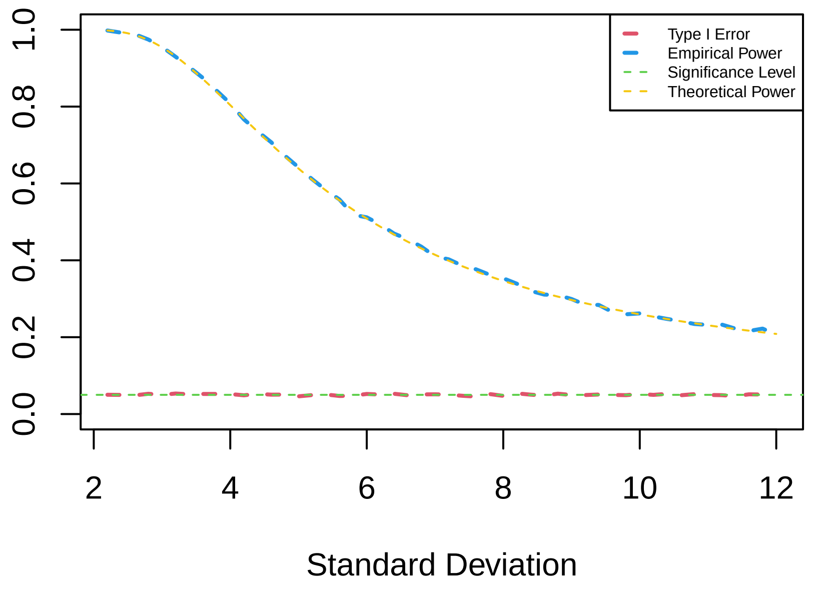
```r
curve(pnorm(qnorm(1 - alpha) - (mu - mu0) * sqrt(x)/sigma, lower.tail = FALSE),
    add = TRUE, col = 7, lty = 2)
abline(h = 0.05, col = 3, lty = 2)
legend("topleft", c("Type I Error", "Empirical Power", "Significance Level",
    "Theoretical Power"), col = c(2, 4, 3, 7), lty = rep(2, 4), lwd = c(2,
    2, 1, 1), cex = 0.5)
```



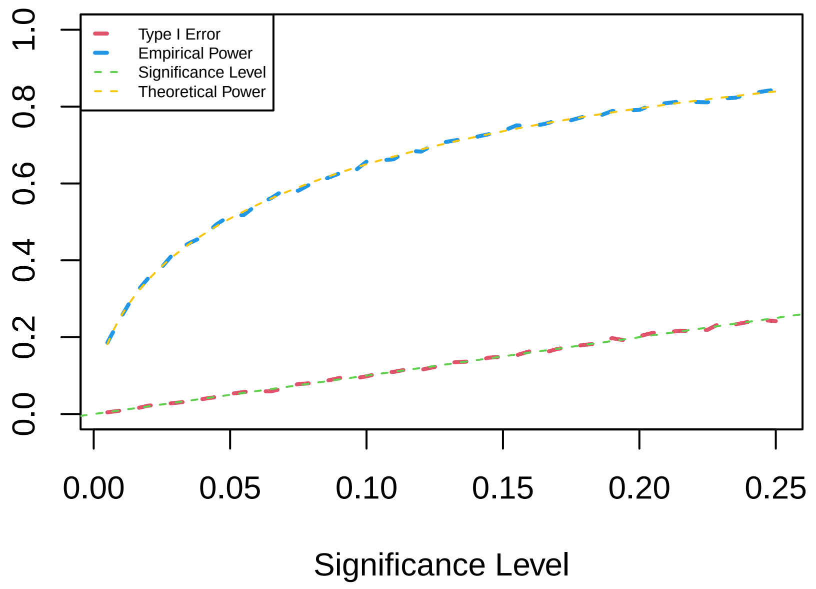
```r
n = 100
mu = seq(1.04, 3, 0.04)
for (k in 1:50) {
    X = matrix(rnorm(n * nsim, mu0, sigma), n)
    MLE = colMeans(X)
    z = (MLE - mu0) * sqrt(n)/sigma
    Error[k] = mean(z > qnorm(1 - alpha))
    Y = matrix(rnorm(n * nsim, mu[k], sigma), n)
    MLE = colMeans(Y)
    z = (MLE - mu0) * sqrt(n)/sigma
    Power[k] = mean(z > qnorm(1 - alpha))
}
plot(mu, Power, "l", ylim = c(0, 1), xlab = "Mean", ylab = NA, col = 4,
    lty = 2, lwd = 2)
lines(mu, Error, col = 2, lty = 2, lwd = 2)
curve(pnorm(qnorm(1 - alpha) + (mu0 - x) * sqrt(n)/sigma, lower.tail = FALSE),
    add = TRUE, col = 7, lty = 2)
abline(h = 0.05, col = 3, lty = 2)
legend("topleft", c("Type I Error", "Empirical Power", "Significance Level",
    "Theoretical Power"), col = c(2, 4, 3, 7), lty = rep(2, 4), lwd = c(2,
    2, 1, 1), cex = 0.5)
```



```r
mu = 2
sigma = seq(2.2, 12, 0.2)
for (k in 1:50) {
    X = matrix(rnorm(n * nsim, mu0, sigma[k]), n)
    MLE = colMeans(X)
    z = (MLE - mu0) * sqrt(n)/sigma[k]
    Error[k] = mean(z > qnorm(1 - alpha))
    Y = matrix(rnorm(n * nsim, mu, sigma[k]), n)
    MLE = colMeans(Y)
    z = (MLE - mu0) * sqrt(n)/sigma[k]
    Power[k] = mean(z > qnorm(1 - alpha))
}
plot(sigma, Power, "l", ylim = c(0, 1), xlab = "Standard Deviation", ylab = NA,
    col = 4, lty = 2, lwd = 2)
lines(sigma, Error, col = 2, lty = 2, lwd = 2)
curve(pnorm(qnorm(1 - alpha) + (mu0 - mu) * sqrt(n)/x, lower.tail = FALSE),
    add = TRUE, col = 7, lty = 2)
abline(h = 0.05, col = 3, lty = 2)
legend("topright", c("Type I Error", "Empirical Power", "Significance Level",
    "Theoretical Power"), col = c(2, 4, 3, 7), lty = rep(2, 4), lwd = c(2,
    2, 1, 1), cex = 0.5)
```



```
sigma = 6
alpha = seq(0.005, 0.25, 0.005)
for (k in 1:50) {
    X = matrix(rnorm(n * nsim, mu0, sigma), n)
    MLE = colMeans(X)
    z = (MLE - mu0) * sqrt(n)/sigma
    Error[k] = mean(z > qnorm(1 - alpha[k]))
    Y = matrix(rnorm(n * nsim, mu, sigma), n)
    MLE = colMeans(Y)
    z = (MLE - mu0) * sqrt(n)/sigma
    Power[k] = mean(z > qnorm(1 - alpha[k]))
}
plot(alpha, Power, "l", ylim = c(0, 1), xlab = "Significance Level", ylab = NA,
    col = 4, lty = 2, lwd = 2)
lines(alpha, Error, col = 2, lty = 2, lwd = 2)
curve(pnorm(qnorm(1 - x) + (mu0 - mu) * sqrt(n)/sigma, lower.tail = FALSE),
    add = TRUE, col = 7, lty = 2)
abline(0, 1, col = 3, lty = 2)
legend("topleft", c("Type I Error", "Empirical Power", "Significance Level",
    "Theoretical Power"), col = c(2, 4, 3, 7), lty = rep(2, 4), lwd = c(2,
    2, 1, 1), cex = 0.5)
```



```
nsim = 1e+05
alpha = 0.05
X = matrix(rnorm(n * nsim, mu0, sigma), n)
MLE = colMeans(X)
z = (MLE - mu0) * sqrt(n)/sigma
pvalue = 2 * pnorm(abs(z), lower.tail = FALSE)
mean(abs(z) > qnorm(1 - alpha/2))
```

```
## [1] 0.05062
```

```
mean(pvalue < alpha)
```

```
## [1] 0.05062
```

```
Y = matrix(rnorm(n * nsim, mu, sigma), n)
MLE = colMeans(Y)
z = (MLE - mu0) * sqrt(n)/sigma
pvalue = 2 * pnorm(abs(z), lower.tail = FALSE)
mean(abs(z) > qnorm(1 - alpha/2))
```

```
## [1] 0.38206
```

```
mean(pvalue < alpha)
```

```
## [1] 0.38206
```

```
pnorm((mu0 - mu) * sqrt(n)/sigma - qnorm(1 - alpha/2)) + pnorm((mu0 - mu) *
    sqrt(n)/sigma + qnorm(1 - alpha/2), lower.tail = FALSE)
```

```
## [1] 0.384791
```

```
X = matrix(rnorm(n * nsim, mu0, sigma), n)
t = (colMeans(X) - mu0) * sqrt(n)/apply(X, 2, sd)
pvalue = 2 * pt(abs(t), n - 1, lower.tail = FALSE)
```

```
mean(abs(t) > qt(1 - alpha/2, n - 1))
```

```
## [1] 0.05009
```

```
mean(pvalue < alpha)
```

```
## [1] 0.05009
```

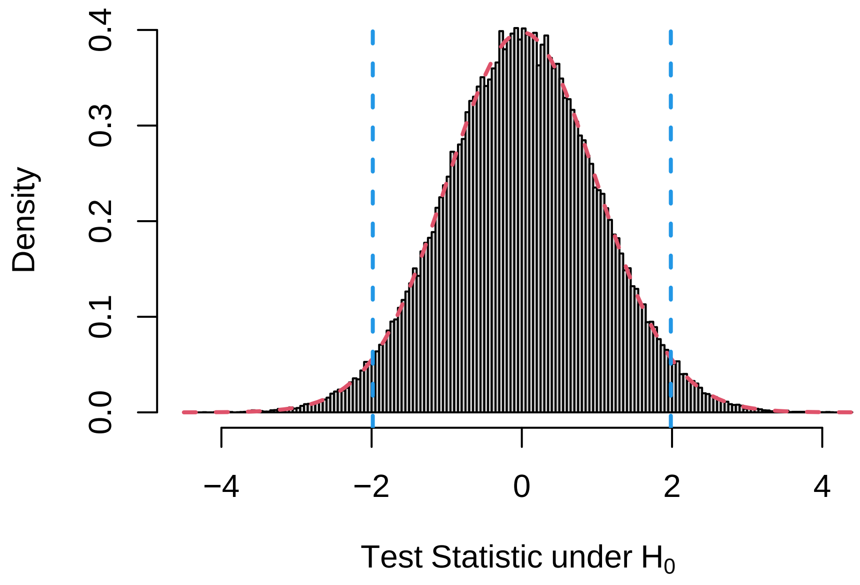
```
print(t[1])
```

```
## [1] 0.8045512
```

```
t.test(X[, 1], mu = mu0)$statistic
```

```
##         t
## 0.8045512
```

```
hist(t, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[0]))
curve(dt(x, n - 1), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qt(c(alpha/2, 1 - alpha/2), n - 1), col = 4, lty = 2, lwd = 2)
```



```
Y = matrix(rnorm(n * nsim, mu, sigma), n)
t = (colMeans(Y) - mu0) * sqrt(n)/apply(Y, 2, sd)
pvalue = 2 * pt(abs(t), n - 1, lower.tail = FALSE)
mean(abs(t) > qt(1 - alpha/2, n - 1))
```

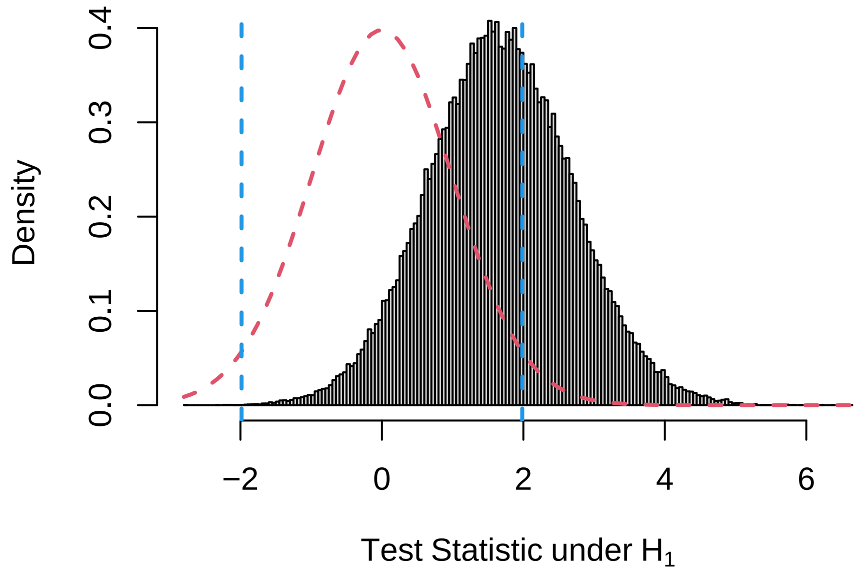
```
## [1] 0.37689
```

```
mean(pvalue < alpha)
```

```
## [1] 0.37689
```

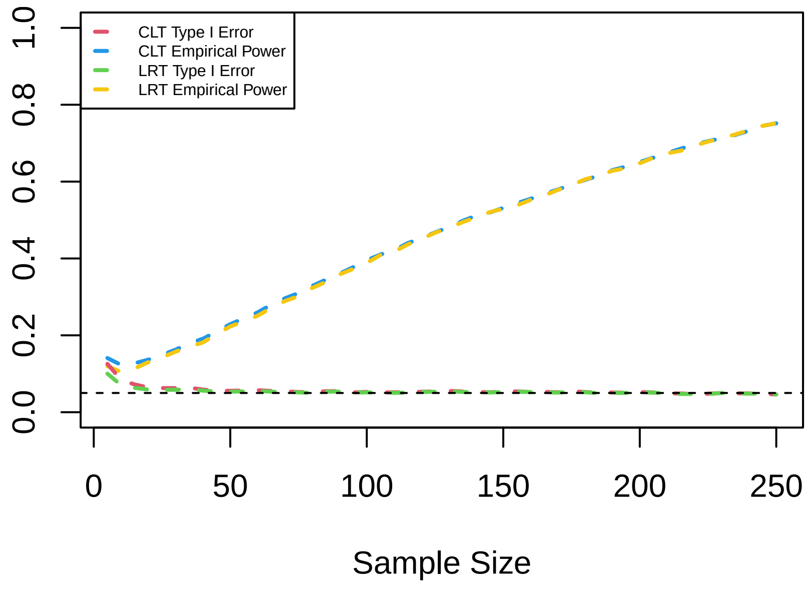
```
hist(t, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[1]))
```

```r
curve(dt(x, n - 1), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qt(c(alpha/2, 1 - alpha/2), n - 1), col = 4, lty = 2, lwd = 2)
```



```r
step = 5
n = seq(5, 250, step)
nsim = 10000
X = matrix(0, 0, nsim)
Y = matrix(0, 0, nsim)
Error = matrix(0, 50, 2)
Power = matrix(0, 50, 2)
for (k in 1:50) {
    X = rbind(X, matrix(rnorm(step * nsim, mu0, sigma), step))
    MLE = colMeans(X)
    z = (MLE - mu0) * sqrt(n[k])/apply(X, 2, sd)
    Error[k, 1] = mean(abs(z) > qnorm(1 - alpha/2))
    Error[k, 2] = mean(n[k] * log(1 + (MLE - mu0)^2/colMeans(t(t(X) - colMeans(X))^2)) >
        qchisq(1 - alpha, 1))
    Y = rbind(Y, matrix(rnorm(step * nsim, mu, sigma), step))
    MLE = colMeans(Y)
    z = (MLE - mu0) * sqrt(n[k])/apply(X, 2, sd)
    Power[k, 1] = mean(abs(z) > qnorm(1 - alpha/2))
    Power[k, 2] = mean(n[k] * log(1 + (MLE - mu0)^2/colMeans(t(t(Y) - colMeans(Y))^2)) >
        qchisq(1 - alpha, 1))
}
plot(n, Power[, 1], "l", ylim = c(0, 1), xlab = "Sample Size", ylab = NA,
    col = 4, lty = 2, lwd = 2)
lines(n, Power[, 2], col = 7, lty = 2, lwd = 2)
lines(n, Error[, 1], col = 2, lty = 2, lwd = 2)
lines(n, Error[, 2], col = 3, lty = 2, lwd = 2)
```

```r
abline(h = 0.05, lty = 2)
legend("topleft", c("CLT Type I Error", "CLT Empirical Power", "LRT Type I Error",
    "LRT Empirical Power"), col = c(2, 4, 3, 7), lty = rep(2, 4), lwd = rep(2,
    4), cex = 0.5)
```



```r
n = 100
m = 100
mu1 = 1
mu2 = 1
X = matrix(rnorm(n * nsim, mu1, sigma), n)
Y = matrix(rnorm(m * nsim, mu2, sigma), m)
Sp = sqrt(((n - 1) * apply(X, 2, var) + (m - 1) * apply(Y, 2, var))/(n +
    m - 2))
t = (colMeans(X) - colMeans(Y)) * sqrt(n * m/(n + m))/Sp
pvalue = 2 * pt(abs(t), n + m - 2, lower.tail = FALSE)
mean(abs(t) > qt(1 - alpha/2, n + m - 2))
```

```
## [1] 0.0508
```

```r
mean(pvalue < alpha)
```

```
## [1] 0.0508
```

```r
print(t[1])
```

```
## [1] -0.7560073
```

```r
t.test(X[, 1], Y[, 1])$statistic
```

```
##          t
## -0.7560073
```

```r
mu2 = 2
Y = matrix(rnorm(m * nsim, mu2, sigma), m)
Sp = sqrt(((n - 1) * apply(X, 2, var) + (m - 1) * apply(Y, 2, var))/(n +
    m - 2))
t = (colMeans(X) - colMeans(Y)) * sqrt(n * m/(n + m))/Sp
pvalue = 2 * pt(abs(t), n + m - 2, lower.tail = FALSE)
mean(abs(t) > qt(1 - alpha/2, n + m - 2))
```

```
## [1] 0.2156
```

```r
mean(pvalue < alpha)
```

```
## [1] 0.2156
```

```r
sigma1 = 2
sigma2 = 2
X = matrix(rnorm(n * nsim, mu, sigma1), n)
Y = matrix(rnorm(m * nsim, mu, sigma2), m)
f = apply(X, 2, var)/apply(Y, 2, var)
pvalue = 2 * pmin(pf(f, n - 1, m - 1), pf(f, n - 1, m - 1, lower.tail = FALSE))
mean(f < qf(alpha/2, n - 1, m - 1) | f > qf(1 - alpha/2, n - 1, m - 1))
```

```
## [1] 0.0533
```

```r
mean(pvalue < alpha)
```

```
## [1] 0.0533
```

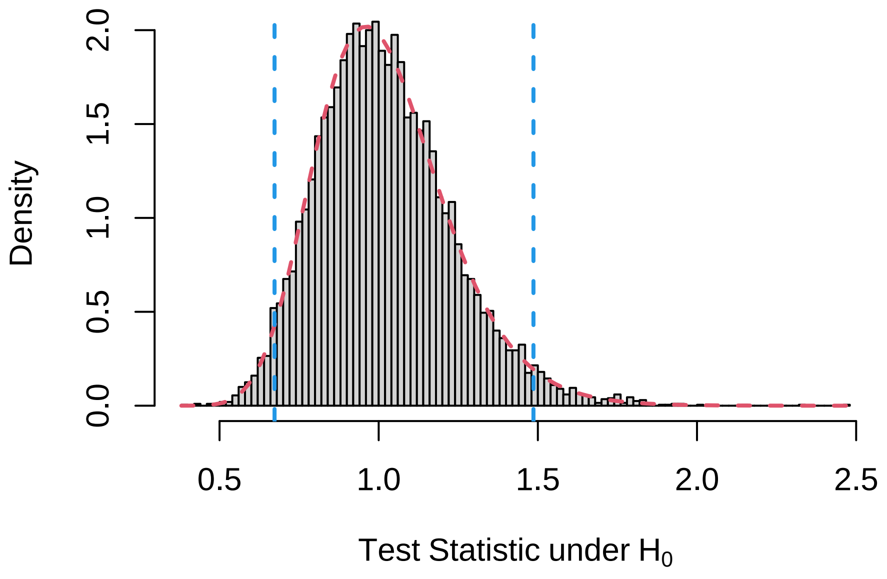
```r
print(f[1])
```

```
## [1] 0.8416738
```

```r
var.test(X[, 1], Y[, 1])$statistic
```

```
##         F
## 0.8416738
```

```r
hist(f, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[0]))
curve(df(x, n - 1, m - 1), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qf(c(alpha/2, 1 - alpha/2), n - 1, m - 1), col = 4, lty = 2,
    lwd = 2)
```



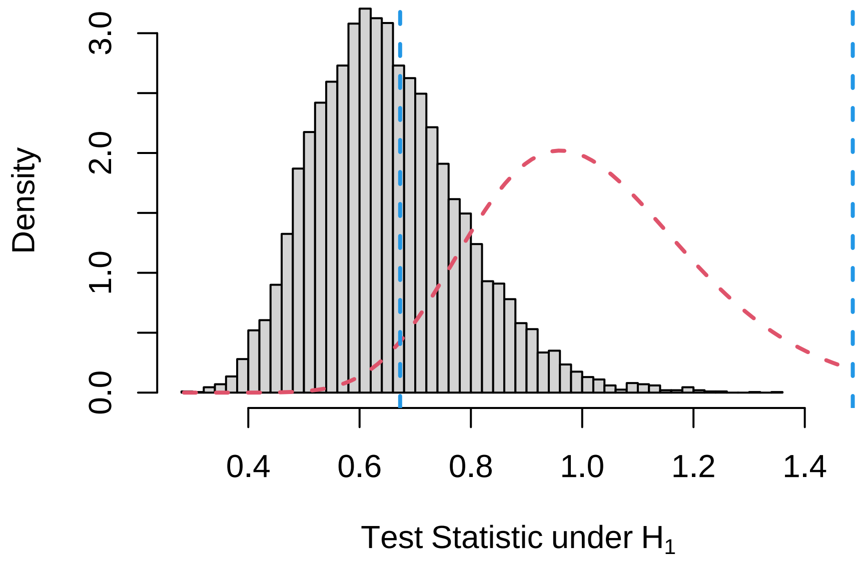
```
sigma2 = 2.5
Y = matrix(rnorm(m * nsim, mu, sigma2), m)
f = apply(X, 2, var)/apply(Y, 2, var)
pvalue = 2 * pmin(pf(f, n - 1, m - 1), pf(f, n - 1, m - 1, lower.tail = FALSE))
mean(f < qf(alpha/2, n - 1, m - 1) | f > qf(1 - alpha/2, n - 1, m - 1))
```

```
## [1] 0.598
```

```
mean(pvalue < alpha)
```

```
## [1] 0.598
```

```
hist(f, "FD", freq = FALSE, main = NA, xlim = c(min(f), qf(1 - alpha/2,
    n - 1, m - 1)), xlab = expression(Test ~ Statistic ~ under ~ H[1]))
curve(df(x, n - 1, m - 1), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qf(c(alpha/2, 1 - alpha/2), n - 1, m - 1), col = 4, lty = 2,
    lwd = 2)
```



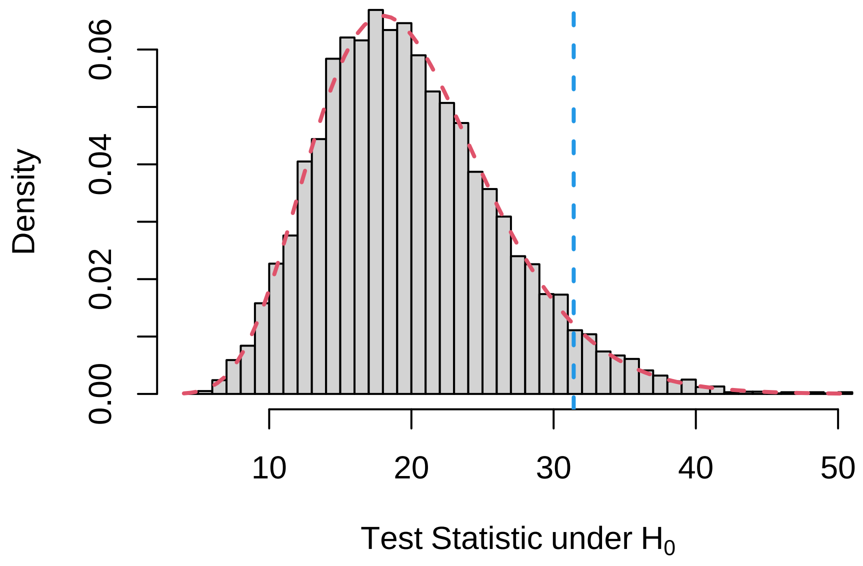
```r
n = 10
lambda0 = 1
X = matrix(rexp(n * nsim, lambda0), n)
s = 2 * lambda0 * colSums(X)
pvalue = pchisq(s, 2 * n, lower.tail = FALSE)
mean(s > qchisq(1 - alpha, 2 * n))
```

```
## [1] 0.0537
```

```r
mean(pvalue < alpha)
```

```
## [1] 0.0537
```

```r
hist(s, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[0]))
curve(dchisq(x, 2 * n), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qchisq(1 - alpha, 2 * n), col = 4, lty = 2, lwd = 2)
```



```r
lambda = 1.25
Y = matrix(rexp(n * nsim, lambda), n)
s = 2 * lambda0 * colSums(Y)
pvalue = pchisq(s, 2 * n, lower.tail = FALSE)
mean(s > qchisq(1 - alpha, 2 * n))
```

```
## [1] 0.0058
```

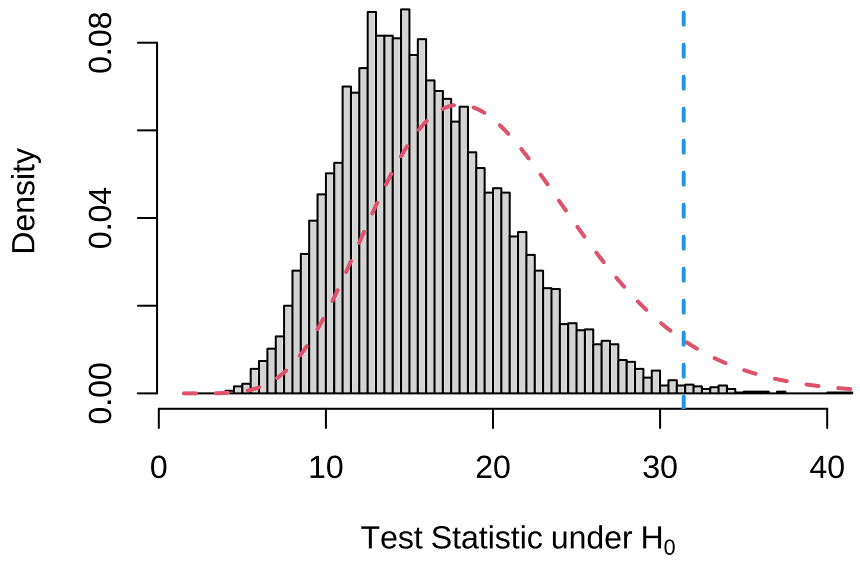
```r
mean(pvalue < alpha)
```

```
## [1] 0.0058
```

```r
pchisq(qchisq(1 - alpha, 2 * n) * lambda/lambda0, 2 * n, lower.tail = FALSE)
```

```
## [1] 0.006182675
```

```
hist(s, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[0]))
curve(dchisq(x, 2 * n), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qchisq(1 - alpha, 2 * n), col = 4, lty = 2, lwd = 2)
```



```
lambda = 0.75
Y = matrix(rexp(n * nsim, lambda), n)
s = 2 * lambda0 * colSums(Y)
pvalue = pchisq(s, 2 * n, lower.tail = FALSE)
mean(s > qchisq(1 - alpha, 2 * n))
```

```
## [1] 0.2583
```

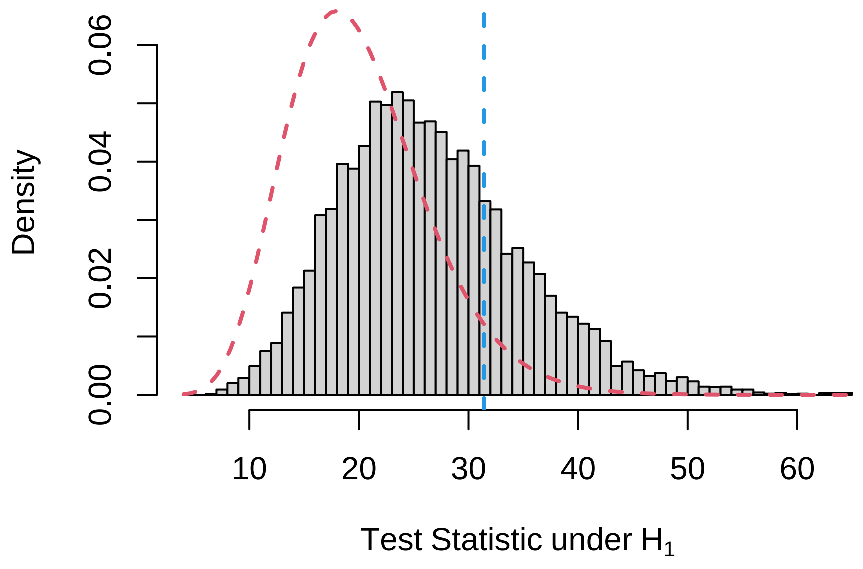
```
mean(pvalue < alpha)
```

```
## [1] 0.2583
```

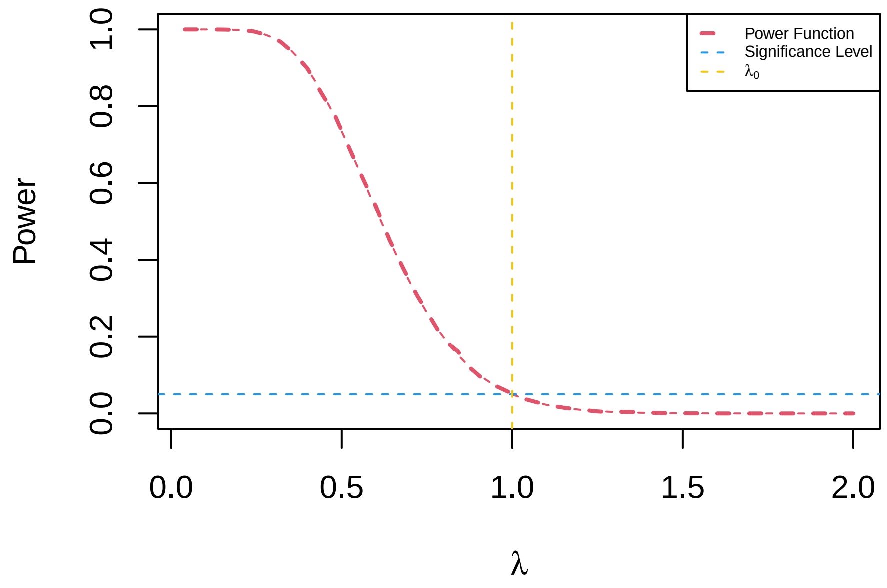
```
pchisq(qchisq(1 - alpha, 2 * n) * lambda/lambda0, 2 * n, lower.tail = FALSE)
```

```
## [1] 0.2622408
```

```
hist(s, "FD", freq = FALSE, main = NA, ylim = c(0, max(dchisq(s, 2 * n))),
    xlab = expression(Test ~ Statistic ~ under ~ H[1]))
curve(dchisq(x, 2 * n), add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = qchisq(1 - alpha, 2 * n), col = 4, lty = 2, lwd = 2)
```



```
lambda = seq(0.04, 2, 0.04)
Power = numeric(50)
for (k in 1:50) {
    X = matrix(rexp(n * nsim, lambda[k]), n)
    s = 2 * lambda0 * colSums(X)
    Power[k] = mean(s > qchisq(1 - alpha, 2 * n))
}
plot(lambda, Power, "l", ylim = c(0, 1), xlab = expression(lambda), ylab = "Power",
    col = 2, lty = 2, lwd = 2)
curve(pchisq(qchisq(1 - alpha, 2 * n) * x/lambda0, 2 * n, lower.tail = FALSE),
    add = TRUE, col = 2, lty = 2)
abline(h = alpha, col = 4, lty = 2)
abline(v = lambda0, col = 7, lty = 2)
legend("topright", c("Power Function", "Significance Level", expression(lambda[0])),
    col = c(2, 4, 7), lty = rep(2, 3), lwd = c(2, 1, 1), cex = 0.5)
```



```
theta0 = 2
X = matrix(runif(n * nsim, max = theta0), n)
MLE = apply(X, 2, max)
pvalue = ifelse(MLE < theta0, (MLE/theta0)^n, 0)
mean(MLE < theta0 * alpha^(n^(-1)) | MLE > theta0)
```

```
## [1] 0.0557
```

```
mean(pvalue < alpha)
```

```
## [1] 0.0557
```

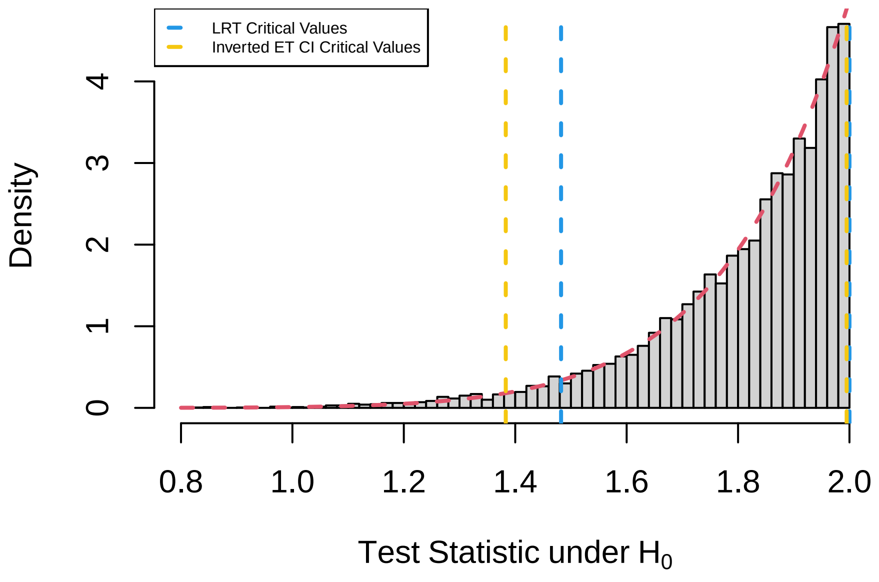
```
pvalue = ifelse(MLE < theta0, 2 * pmin((MLE/theta0)^n, 1 - (MLE/theta0)^n),
    0)
mean(MLE < theta0 * (alpha/2)^(n^(-1)) | MLE > theta0 * (1 - alpha/2)^(n^(-1)))
```

```
## [1] 0.0508
```

```
mean(pvalue < alpha)
```

```
## [1] 0.0508
```

```
hist(MLE, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[0]))
curve(n * x^(n - 1)/theta0^n, add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = c(theta0 * alpha^(n^(-1)), theta0), col = 4, lty = 2, lwd = 2)
abline(v = c(theta0 * (alpha/2)^(n^(-1)), theta0 * (1 - alpha/2)^(n^(-1))),
    col = 7, lty = 2, lwd = 2)
legend("topleft", c("LRT Critical Values", "Inverted ET CI Critical Values"),
    col = c(4, 7), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```



```
theta = 2.25
Y = matrix(runif(n * nsim, max = theta), n)
```

```
MLE = apply(Y, 2, max)
pvalue = ifelse(MLE < theta0, (MLE/theta0)^n, 0)
mean(MLE < theta0 * alpha^(n^(-1)) | MLE > theta0)
```

```
## [1] 0.711
```

```
mean(pvalue < alpha)
```

```
## [1] 0.711
```

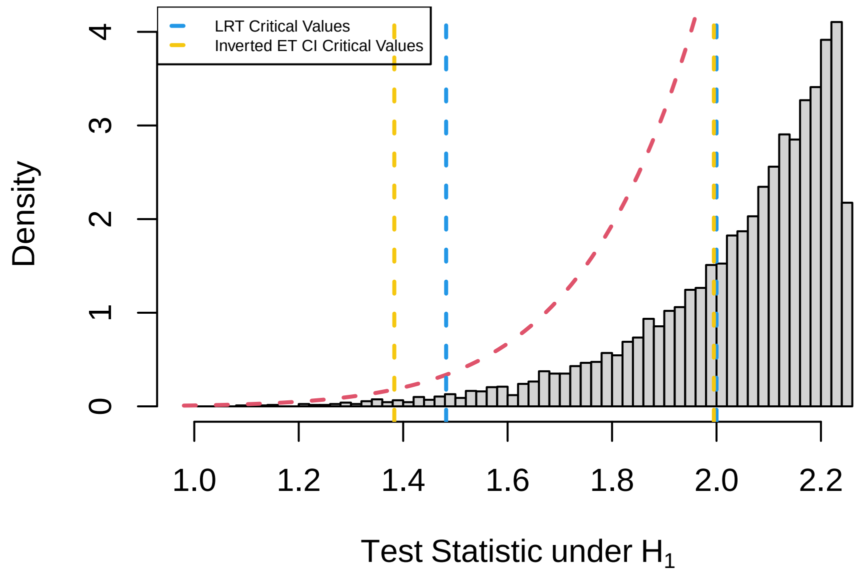
```
pvalue = ifelse(MLE < theta0, 2 * pmin((MLE/theta0)^n, 1 - (MLE/theta0)^n),
    0)
mean(MLE < theta0 * (alpha/2)^(n^(-1)) | MLE > theta0 * (1 - alpha/2)^(n^(-1)))
```

```
## [1] 0.712
```

```
mean(pvalue < alpha)
```

```
## [1] 0.712
```

```
hist(MLE, "FD", freq = FALSE, main = NA, xlab = expression(Test ~ Statistic ~
    under ~ H[1]))
curve(n * x^(n - 1)/theta0^n, add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = c(theta0 * alpha^(n^(-1)), theta0), col = 4, lty = 2, lwd = 2)
abline(v = c(theta0 * (alpha/2)^(n^(-1)), theta0 * (1 - alpha/2)^(n^(-1))),
    col = 7, lty = 2, lwd = 2)
legend("topleft", c("LRT Critical Values", "Inverted ET CI Critical Values"),
    col = c(4, 7), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```



```
theta = 1.75
Y = matrix(runif(n * nsim, max = theta), n)
MLE = apply(Y, 2, max)
pvalue = ifelse(MLE < theta0, (MLE/theta0)^n, 0)
```

```
mean(MLE < theta0 * alpha^(n^(-1)) | MLE > theta0)
```

```
## [1] 0.1866
```

```
mean(pvalue < alpha)
```

```
## [1] 0.1866
```

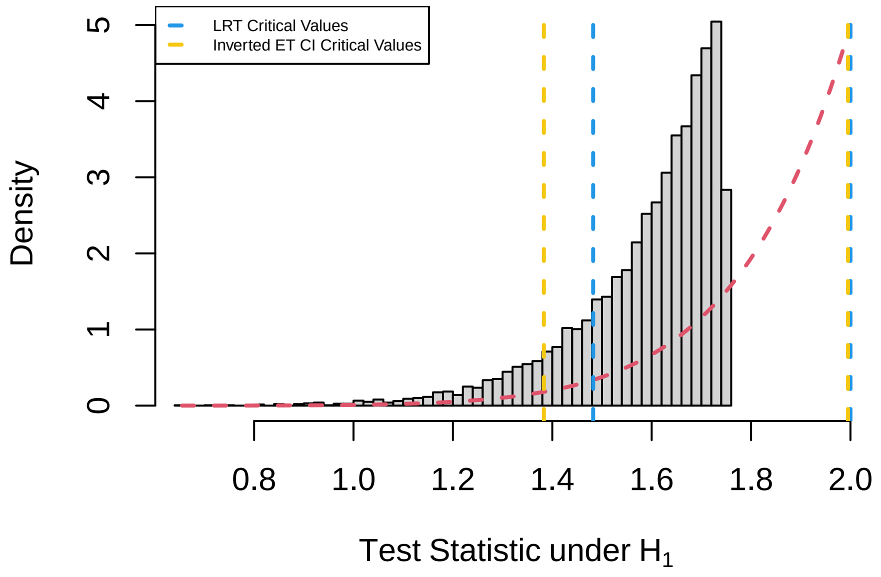
```
pvalue = ifelse(MLE < theta0, 2 * pmin((MLE/theta0)^n, 1 - (MLE/theta0)^n),
    0)
mean(MLE < theta0 * (alpha/2)^(n^(-1)) | MLE > theta0 * (1 - alpha/2)^(n^(-1)))
```

```
## [1] 0.0929
```

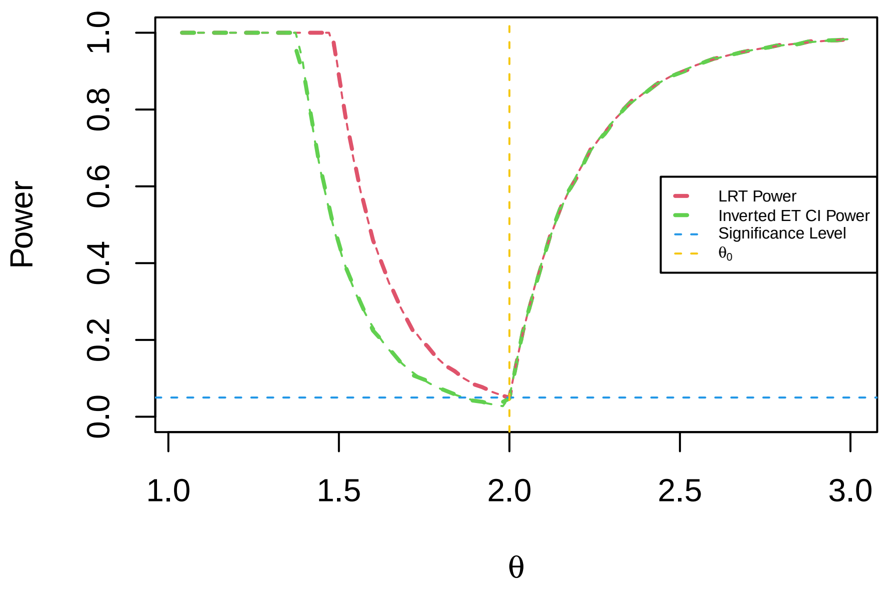
```
mean(pvalue < alpha)
```

```
## [1] 0.0929
```

```
hist(MLE, "FD", freq = FALSE, main = NA, xlim = c(min(MLE), theta0), xlab = expression(Test ~
    Statistic ~ under ~ H[1]))
curve(n * x^(n - 1)/theta0^n, add = TRUE, col = 2, lty = 2, lwd = 2)
abline(v = c(theta0 * alpha^(n^(-1)), theta0), col = 4, lty = 2, lwd = 2)
abline(v = c(theta0 * (alpha/2)^(n^(-1)), theta0 * (1 - alpha/2)^(n^(-1))),
    col = 7, lty = 2, lwd = 2)
legend("topleft", c("LRT Critical Values", "Inverted ET CI Critical Values"),
    col = c(4, 7), lty = c(2, 2), lwd = c(2, 2), cex = 0.5)
```



```
LRTPower = function(theta, theta0, alpha, n) {
    ifelse(theta < theta0 * alpha^(n^(-1)), 1, ifelse(theta > theta0, 1 -
        (1 - alpha) * (theta0/theta)^n, (theta0/theta)^n * alpha))
}

CIPower = function(theta, theta0, alpha, n) {
    ifelse(theta < theta0 * (alpha/2)^(n^(-1)), 1, ifelse(theta > theta0 *
        (1 - alpha/2)^(n^(-1)), 1 - (1 - alpha) * (theta0/theta)^n, (theta0/theta)^n *
        alpha/2))
}

theta = seq(1.04, 3, 0.04)
Power = matrix(0, 50, 2)
for (k in 1:50) {
    X = matrix(runif(n * nsim, max = theta[k]), n)
    MLE = apply(X, 2, max)
    Power[k, 1] = mean(MLE < theta0 * alpha^(n^(-1)) | MLE > theta0)
    Power[k, 2] = mean(MLE < theta0 * (alpha/2)^(n^(-1)) | MLE > theta0 *
        (1 - alpha/2)^(n^(-1)))
}
plot(theta, Power[, 1], "l", ylim = c(0, 1), xlab = expression(theta),
    ylab = "Power", col = 2, lty = 2, lwd = 2)
lines(theta, Power[, 2], col = 3, lty = 2, lwd = 2)
curve(LRTPower(x, theta0, alpha, n), add = TRUE, col = 2, lty = 2)
curve(CIPower(x, theta0, alpha, n), add = TRUE, col = 3, lty = 2)
abline(h = alpha, col = 4, lty = 2)
abline(v = theta0, col = 7, lty = 2)
legend("right", c("LRT Power", "Inverted ET CI Power", "Significance Level",
    expression(theta[0])), col = c(2, 3, 4, 7), lty = rep(2, 4), lwd = c(2,
    2, 1, 1), cex = 0.5)
```

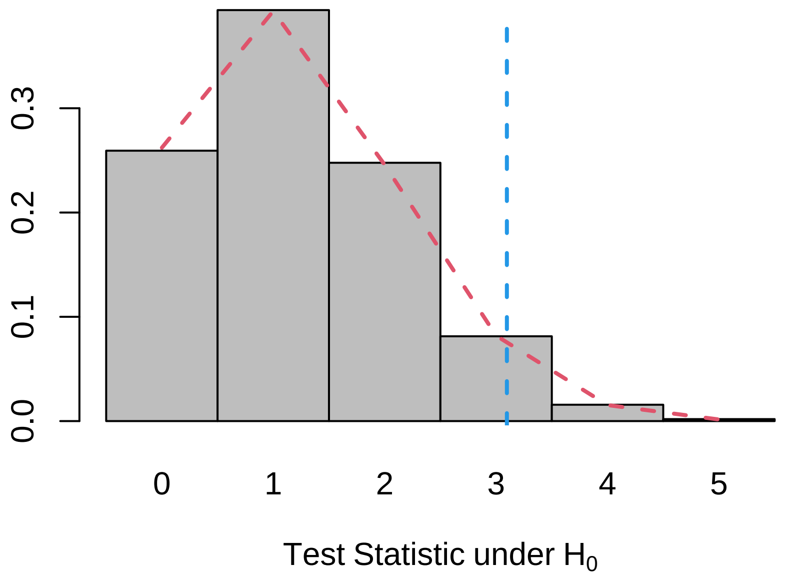


```
n = 6
p0 = 0.2
c = 3
gamma = (pbinom(c, n, p0) - 1 + alpha)/dbinom(c, n, p0)
```

```r
X = matrix(rbinom(n * nsim, 1, p0), n)
s = colSums(X)
mean((s > c) + gamma * (s == c))
```

```
## [1] 0.05038994
```

```r
barplot(table(factor(s, levels = 0:max(s)))/nsim, space = 0, xlab = expression(Test ~
    Statistic ~ under ~ H[0]))
lines(0:max(s) + 0.5, dbinom(0:max(s), n, p0), col = 2, lty = 2, lwd = 2)
abline(v = c + 1 - gamma, col = 4, lty = 2, lwd = 2)
```



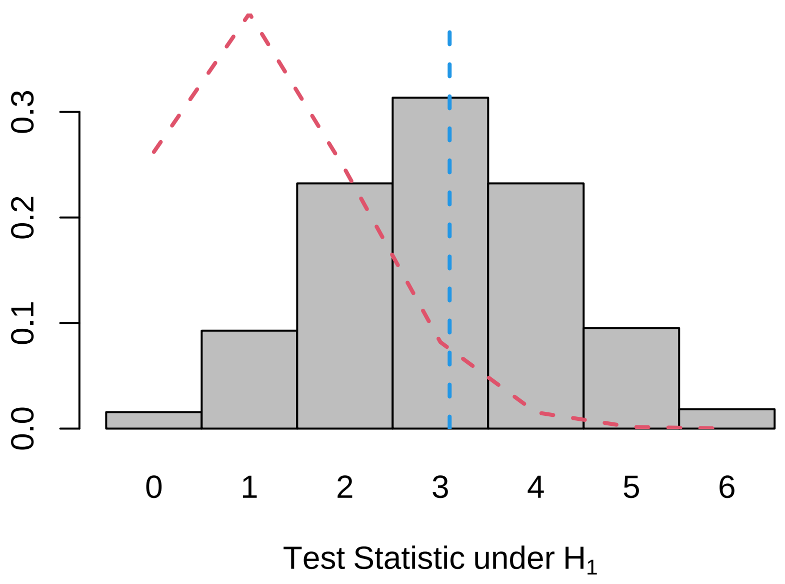
```r
p = 0.5
Y = matrix(rbinom(n * nsim, 1, p), n)
s = colSums(Y)
mean((s > c) + gamma * (s == c))
```

```
## [1] 0.4722409
```

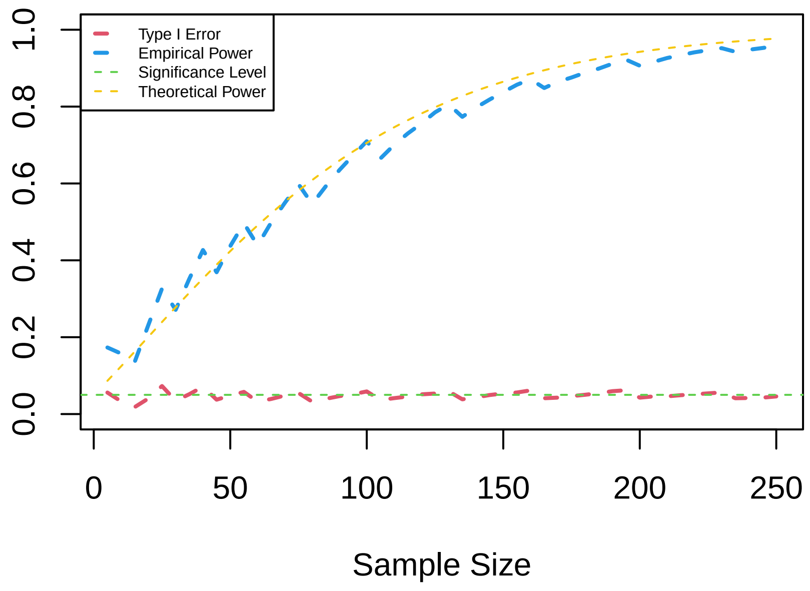
```r
pbinom(c, n, p, lower.tail = FALSE) + gamma * dbinom(c, n, p)
```

```
## [1] 0.4697876
```

```r
barplot(table(factor(s, levels = 0:max(s)))/nsim, space = 0, ylim = c(0,
    max(dbinom(0:max(s), n, p0))), xlab = expression(Test ~ Statistic ~
    under ~ H[1]))
lines(0:max(s) + 0.5, dbinom(0:max(s), n, p0), col = 2, lty = 2, lwd = 2)
abline(v = c + 1 - gamma, col = 4, lty = 2, lwd = 2)
```



```
n = seq(5, 250, step)
nsim = 10000
p = 0.3
X = matrix(0, 0, nsim)
Y = matrix(0, 0, nsim)
Error = numeric(50)
Power = numeric(50)
for (k in 1:50) {
    X = rbind(X, matrix(rbinom(step * nsim, 1, p0), step))
    MLE = colMeans(X)
    z = (MLE - p0) * sqrt(n[k]/(p0 * (1 - p0)))
    Error[k] = mean(abs(z) > qnorm(1 - alpha/2))
    Y = rbind(Y, matrix(rbinom(step * nsim, 1, p), step))
    MLE = colMeans(Y)
    z = (MLE - p0) * sqrt(n[k]/(p0 * (1 - p0)))
    Power[k] = mean(abs(z) > qnorm(1 - alpha/2))
}
plot(n, Power, "l", ylim = c(0, 1), xlab = "Sample Size", ylab = NA, col = 4,
    lty = 2, lwd = 2)
lines(n, Error, col = 2, lty = 2, lwd = 2)
curve(pnorm((p0 - p) * sqrt(x/(p0 * (1 - p0))) - qnorm(1 - alpha/2)) +
    pnorm((p0 - p) * sqrt(x/(p0 * (1 - p0))) + qnorm(1 - alpha/2), lower.tail = FALSE),
    add = TRUE, col = 7, lty = 2)
abline(h = 0.05, col = 3, lty = 2)
legend("topleft", c("Type I Error", "Empirical Power", "Significance Level",
    "Theoretical Power"), col = c(2, 4, 3, 7), lty = rep(2, 4), lwd = c(2,
    2, 1, 1), cex = 0.5)
```



4 Linear Regression

5 Analysis of Variance

6 Nonparametric Methods