main.R

s_ariel

Wed Oct 7 16:46:09 2015

```
# 1 ------------------------------------------------------------------------
```

derive the confidence interval of sigma^2 given i.i.d. observations

@param x vector of i.i.d. observations @param alph this is 1 - confidence level

@return a vector (left, right), (1 - alpha) confidence interval of sigma^2 @export

```
confint_sigma_sq = function(x, alpha = 0.05) {
    n = length(x)
    var_sample = var(x)
    left = (n - 1) * var_sample / qchisq(1 - alpha / 2, df = n - 1)
    right = (n - 1) * var_sample / qchisq(alpha / 2, df = n - 1)
    c(left, right)
}
```

simulate the confidence interval for i.i.d. normal random observations

@param reps number of replications for the simulation study @param n sample size of normal realization @param mu true mean of the normal distribution @param sigma true standard deviation of the normal distribution @param alpha alpha level used for constructing confidence interval of sigma^2 @param score_conf confidence level for constructing the confidence interval of the coverage probability

@return a vector (left, right) score_conf level confidence interval for the coverage probability @export

```
sim_confint_sigma_sq_norm = function(reps, n, mu, sigma, alpha, score_conf) {
    x_sim = rnorm(n = n * reps, mean = mu, sd = sigma)
    x_sim = matrix(x_sim, nrow = reps, ncol = n)
    int_sim = apply(x_sim, 1, confint_sigma_sq, alpha = alpha)
    indicator_sim = apply(int_sim, 2, function(x) {
        x[1] < sigma ^ 2 && x[2] > sigma ^ 2
    })
    prop.test(
        sum(indicator_sim), n = reps,
        p = 1 - alpha, conf.level = score_conf
    )
}
```

simulate the confidence interval for i.i.d. Exp(1) random observations

@param number of replications for the simulation study @param n sample size of normal realization @param alpha alpha level used for constructing confidence interval of sigma^2 @param score_conf confidence level for constructing the confidence interval of the coverage probability

@return @export

```
sim_confint_sigma_sq_exp = function(reps, n, alpha, score_conf) {
    x_sim = rexp(n = n * reps)
    x_sim = matrix(x_sim, nrow = reps, ncol = n)
```

```
    int_sim = apply(x_sim, 1, confint_sigma_sq, alpha = alpha)
    indicator_sim = apply(int_sim, 2, function(x) {
        x[1] < sigma ^ 2 && x[2] > sigma ^ 2
    })
    prop.test(
        sum(indicator_sim), n = reps,
        p = 1 - alpha, conf.level = score_conf
    )
}


reps = 10000
n = 10
mu = 68
sigma = 3
alpha = 0.05
score_conf = 0.99

sim_confint_sigma_sq_norm(reps, n, mu, sigma, alpha, score_conf)
```

```
##
##  1-sample proportions test with continuity correction
##
## data:  sum(indicator_sim) out of reps, null probability 1 - alpha
## X-squared = 1.2637, df = 1, p-value = 0.261
## alternative hypothesis: true p is not equal to 0.95
## 99 percent confidence interval:
##  0.9414000 0.9530014
## sample estimates:
##      p
## 0.9475
```