TP2 Statistique

January 26, 2024

0.1 TP2 - Estimation par intervalles

0.2 Question 6

[41]:
```r
# Définition de la fonction pour calculer les intervalles de confiance
calculate_confidence_intervals <- function(sample, alpha) {
  n <- length(sample)
  x_bar <- mean(sample)
  s <- sd(sample)
  z <- qnorm(1 - alpha/2)
  t <- qt(1 - alpha/2, df = n - 1)
  chi2_lower <- qchisq(1 - alpha/2, df = n - 1)
  chi2_upper <- qchisq(alpha/2, df = n - 1)

  interval_1 <- c(x_bar - t * s/sqrt(n), x_bar + t * s/sqrt(n))
  interval_2 <- c(x_bar - z * s/sqrt(n), x_bar + z * s/sqrt(n))
  interval_3 <- c(exp(log(x_bar) - z * s/sqrt(n)), exp(log(x_bar) + z * s/
 ↪sqrt(n)))
  interval_4 <- c((n - 1) * s^2 / chi2_lower, (n - 1) * s^2 / chi2_upper)
  interval_5 <- c(x_bar/(1 + z/sqrt(n)), x_bar/(1 - z/sqrt(n)))

  matrix(c(interval_1, interval_2, interval_3, interval_4, interval_5), nrow =␣
 ↪5, byrow = TRUE)
}

# Exemple d'utilisation avec un échantillon de données et une valeur alpha
set.seed(123)  # Pour la reproductibilité des résultats
sample <- rexp(100, rate = 0.5)  # Génération d'un échantillon de taille 100 à␣
 ↪partir d'une loi exponentielle
alpha <- 0.05  # Valeur spécifiée pour alpha

# Calcul des intervalles de confiance
intervals <- calculate_confidence_intervals(sample, alpha)
print(intervals)
```

```
         [,1]     [,2]
[1,] 1.679269 2.503606
[2,] 1.684307 2.498568
[3,] 1.391971 3.142386
```

```
[4,] 3.326349 5.822928
[5,] 1.748699 2.601279
```

0.3 Question 7

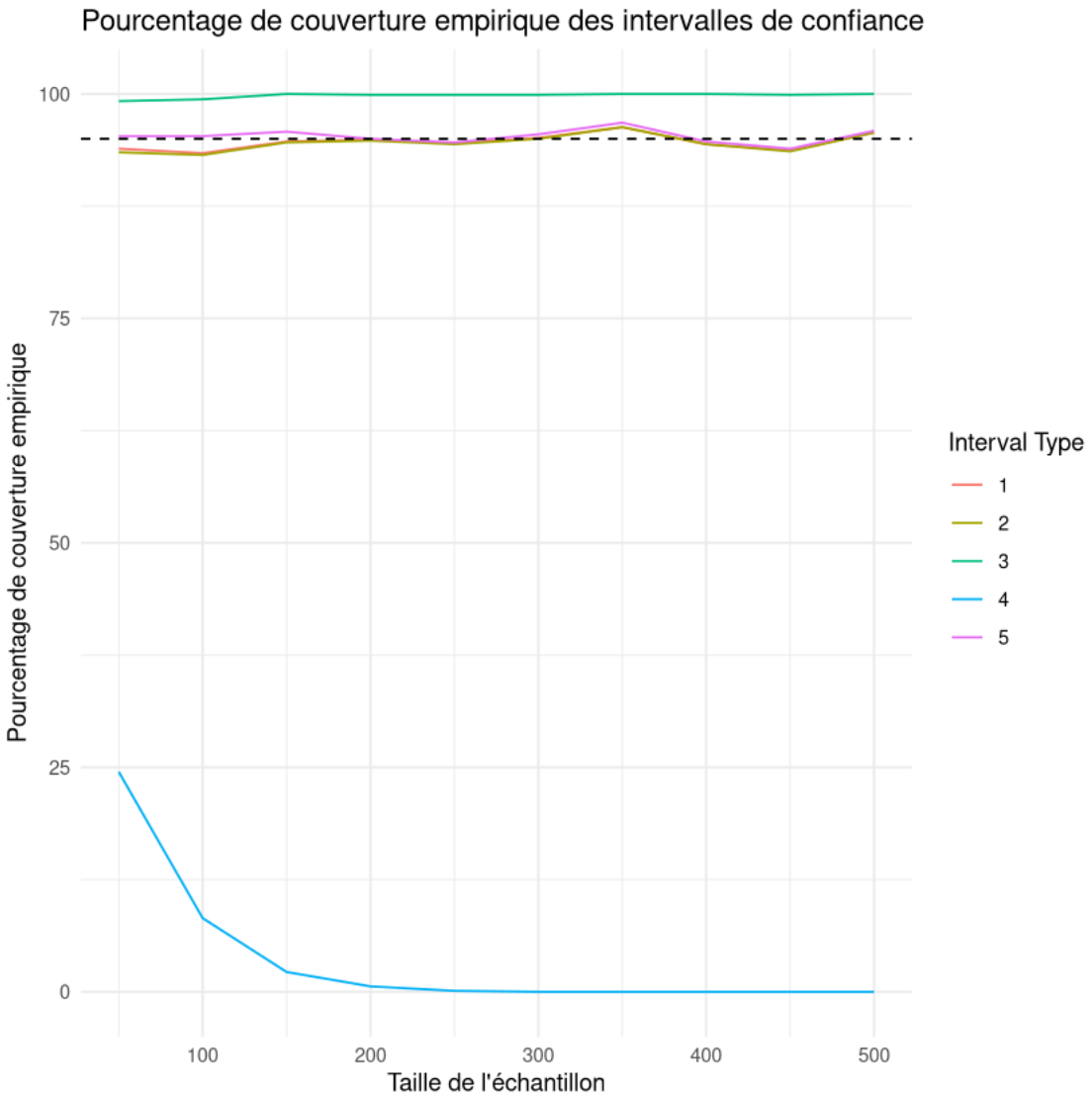
[42]:
```r
# Function to calculate empirical coverage percentages and median lengths
calculate_empirical_coverage <- function(n, nrep, mu, alpha) {
  coverage <- numeric(5)
  lengths <- numeric(5)

  for (i in 1:nrep) {
    # Generate random samples from exponential distribution
    sample_data <- rexp(n, rate = 1/mu)

    # Calculate confidence intervals
    intervals <- calculate_confidence_intervals(sample_data, alpha)

    # Check if the true mean (mu) is within each interval
    for (j in 1:5) {
      if (intervals[j, 1] <= mu & mu <= intervals[j, 2]) {
        coverage[j] <- coverage[j] + 1
      }
      lengths[j] <- lengths[j] + (intervals[j, 2] - intervals[j, 1])
    }
  }

  # Calculate empirical coverage percentages
  coverage_percentages <- coverage / nrep * 100

  # Calculate median lengths of intervals
  median_lengths <- lengths / nrep

  # Return the results as vectors
  return(list(couv = coverage_percentages, longueur = median_lengths))
}

# Example of usage
set.seed(123)  # For reproducibility
n <- 100  # Sample size
nrep <- 1000  # Number of samples to generate
mu <- 2  # True mean of the exponential distribution
alpha <- 0.05  # Significance level

# Calculate empirical coverage percentages and median lengths
results <- calculate_empirical_coverage(n, nrep, mu, alpha)
print(results$couv)
print(results$longueur)
```

```
[1] 94.0 93.8 99.7  8.1 95.8
[1] 0.7827503 0.7731828 1.5988512 2.2919384 0.8132359
```

0.4 Question 8

[43]:
```r
library(ggplot2)

# Fonction pour calculer les pourcentages de couverture empiriques
calculate_empirical_coverage <- function(n, nrep, mu, alpha) {
  coverage <- numeric(5)

  for (i in 1:nrep) {
    # Générer des échantillons aléatoires de la distribution exponentielle
    sample_data <- rexp(n, rate = 1/mu)

    # Calculer les intervalles de confiance
    intervals <- calculate_confidence_intervals(sample_data, alpha)

    # Vérifier si la vraie moyenne (mu) est dans chaque intervalle
    for (j in 1:5) {
      if (intervals[j, 1] <= mu & mu <= intervals[j, 2]) {
        coverage[j] <- coverage[j] + 1
      }
    }
  }

  # Calculer les pourcentages de couverture empiriques
  coverage_percentages <- coverage / nrep * 100

  # Retourner les résultats comme un vecteur
  return(coverage_percentages)
}

# Paramètres
set.seed(123)
nrep <- 1000
mu <- 2
alpha <- 0.05

# Tailles d'échantillons à considérer
sample_sizes <- seq(50, 500, by = 50)

# Calculer les pourcentages de couverture empiriques pour chaque taille␣
 ↪d'échantillon
empirical_coverages <- sapply(sample_sizes, function(n) {
  calculate_empirical_coverage(n, nrep, mu, alpha)
})
```

```r
# Calculer les pourcentages théoriques attendus (100 * (1 - alpha))
theoretical_coverage <- rep(100 * (1 - alpha), length(sample_sizes))

# Créer un dataframe pour ggplot
df <- data.frame(
  Sample_Size = rep(sample_sizes, each = 5),
  Empirical_Coverage = c(empirical_coverages),
  Interval_Type = rep(1:5, times = length(sample_sizes))
)

# Tracer le graphique
ggplot(df, aes(x = Sample_Size, y = Empirical_Coverage, color =␣
 ↪factor(Interval_Type))) +
  geom_line() +
  geom_hline(yintercept = 100 * (1 - alpha), linetype = "dashed", color =␣
 ↪"black") +
  labs(
    title = "Pourcentage de couverture empirique des intervalles de confiance",
    x = "Taille de l'échantillon",
    y = "Pourcentage de couverture empirique"
  ) +
  scale_color_discrete(name = "Interval Type") +
  theme_minimal()
```



0.5 COMMENTAIRE

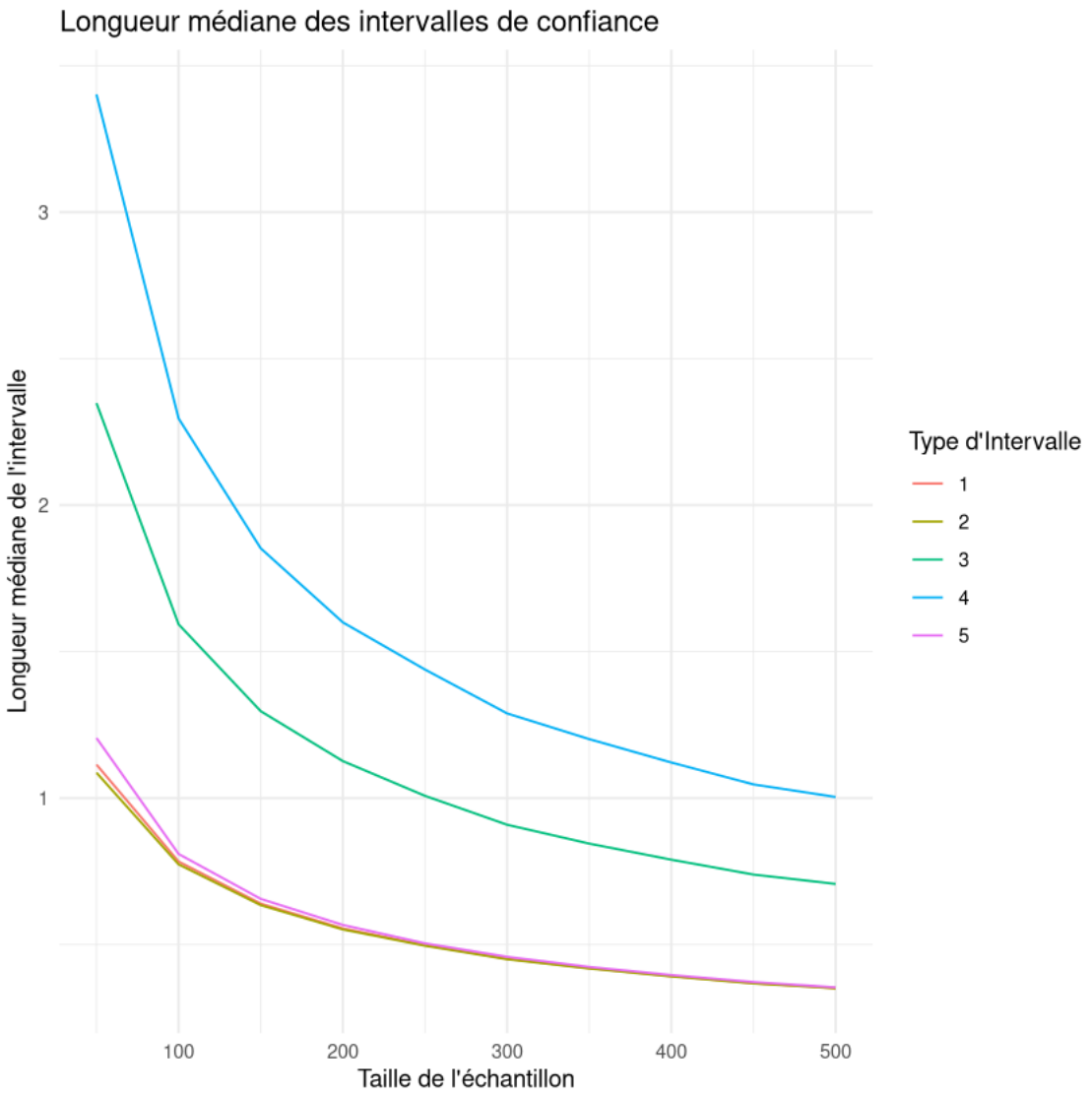
On observe que les pourcentages de couverture empiriques des intervalles I1, I2, I3 et I5 convergent vers la valeur théorique de 100(1 -) % lorsque la taille de l'échantillon augmente. Cela signifie que ces intervalles sont efficaces pour couvrir la vraie valeur du paramètre avec la probabilité donnée par .

0.6 Question 9

[44]:
```
library(ggplot2)

# Fonction pour calculer les longueurs médianes
calculate_median_lengths <- function(n, nrep, mu, alpha) {
```

```r
  lengths <- numeric(5)

  for (i in 1:nrep) {
    # Générer des échantillons aléatoires de la distribution exponentielle
    sample_data <- rexp(n, rate = 1/mu)

    # Calculer les intervalles de confiance
    intervals <- calculate_confidence_intervals(sample_data, alpha)

    # Ajouter les longueurs médianes
    for (j in 1:5) {
      lengths[j] <- lengths[j] + (intervals[j, 2] - intervals[j, 1])
    }
  }

  # Calculer les longueurs médianes
  median_lengths <- lengths / nrep

  # Retourner les résultats comme un vecteur
  return(median_lengths)
}

# Paramètres
set.seed(123)
nrep <- 1000
mu <- 2
alpha <- 0.05

# Tailles d'échantillons à considérer
sample_sizes <- seq(50, 500, by = 50)

# Calculer les longueurs médianes pour chaque taille d'échantillon
median_lengths <- sapply(sample_sizes, function(n) {
  calculate_median_lengths(n, nrep, mu, alpha)
})

# Créer un dataframe pour ggplot
df <- data.frame(
  Sample_Size = rep(sample_sizes, each = 5),
  Median_Length = c(median_lengths),
  Interval_Type = rep(1:5, times = length(sample_sizes))
)

# Tracer le graphique
ggplot(df, aes(x = Sample_Size, y = Median_Length, color =␣
 ↪factor(Interval_Type))) +
  geom_line() +
```

```r
labs(
  title = "Longueur médiane des intervalles de confiance",
  x = "Taille de l'échantillon",
  y = "Longueur médiane de l'intervalle"
) +
scale_color_discrete(name = "Type d'Intervalle") +
theme_minimal()
```



0.7 COMMENTAIRE

On observe que la longueur médiane des intervalles I1, I2, I3, I4 et I5 est plus ou moins constante à partir d'une taille d'échantillon de 100. Cela signifie que la précision de ces intervalles ne s'améliore plus de manière significative lorsque la taille de l'échantillon est supérieure à 100

[ ]: