Confidence interval for differences in performance of two algorithms

Felipe Campelo

September 30th, 2016

Let the following situation be defined:

- 2 algorithms θ_1, θ_2
- N instances $\gamma_1, \ldots, \gamma_N \in \Gamma$
- $n_{1j} = n_{2j} = n_j$ runs of each algorithm on instance γ_j

Let the performance of algorithm θ_i on instance γ_j be expressed as an unknown probability density function with expected value μ_{ij} and variance σ_{ij}:

$$X_{ij} \sim \mathcal{P}_{ij}\left(\mu_{ij}, \sigma_{ij}^2\right)$$

Let $\bar{X}_{ij}$ denote the sample mean of X_{ij}. Assuming that n_j is large enough[1]:

$$\bar{X}_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma_{ij}^2/n_j)$$

1: Using Percent Differences

Assume now that the researcher is interested in comparing the performance of the two algorithms on problem class Γ, and that she is interested in testing hypotheses regarding the *mean percent improvement* of θ_2 in relation to θ_1. Assuming that $\mu_1 > 0$, the percent improvement of θ_2 over θ_1 on instance γ_j can be defined as:

$$\Delta_j = \frac{\mu_{2j} - \mu_{1j}}{\mu_{1j}}$$

In what follows we will assume the aditional assumption that $P(X_{1j} \leq 0) \to 0$ (which is guaranteed, for instance, in problem classes for which the objective function is always strictly positive, which is common in the operations research literature)[2]. If this assumption cannot be guaranteed, the use of percent differences is not advisable, and the researcher should perform her comparisons focusing on the simple differences $\mu_{2j} - \mu_{1j}$. The derivations for this case are given in Section .

The percent improvement can be estimated by the random variable:

[1]*Large enough* is a somewhat fuzzy concept, but the gist of this statement is that the sampling distribution of the means tends to a normal distribution as the sample size increases. If X_{ij} is normally distributed then it should be obvious that this result holds even for $n_j = 1$. Simulation results (see, e.g., http://orcslab.cpdee.ufmg.br:3838/CLT/) suggest that for symmetrical distributions this result holds even for very modest sample sizes ($n_j < 10$), and that for typical asymmetrical distributions of the data more observations are needed, typically $n_j \approx 20$ or 30. Pathologically asymmetrical distributions, however, may need thousands of observations for the sampling distribution of the means to converge to a normal shape. In these cases (which are relatively common in algorithmic research) a rank transformation can be performed to bring the (transformed) data to a more well-behaved distribution.

[2]The derivations that follow are also valid (except for a multiplication by -1) for cases where $P(X_{1j} \geq 0) \to 0$. Problems will only arise when X_{1j} can assume very small values, in which case the confidence intervals can be unbounded. If that is the case, it should be recommended that the algorithms be compared in terms of their absolute difference in mean performance, by defining $\Delta_j = \mu_{2j} - \mu_{1j}$.

$$D_j = \frac{\bar{X}_{2j} - \bar{X}_{1j}}{\bar{X}_{1j}} = \frac{\bar{X}_{(21)j}}{\bar{X}_{1j}}$$

The distribution of $\bar{X}_{(21)j}$ can be easily obtained from the definitions of $\bar{X}_{1j}$ and $\bar{X}_{1j}$ as:

$$\bar{X}_{(21)j} \sim \mathcal{N}\left(\mu_{1j} - \mu_{2j}, \frac{\sigma_{1j}^2 + \sigma_{2j}^2}{n_j}\right)$$

D_j is therefore distributed as the ratio of two (approximately) normal variables. A confidence interval on Δ_j can be calculated using Fieller's Theorem[3]. Considering the assumption that $P(X_{1j} \leq 0) \to 0$, a simplified form of Fieller's interval can be used, which provides good coverage properties. This alternative interval is derived using the *delta method*, also known as the *Taylor method*, as described by Volker Franz[4].

The approximate confidence interval is calculated as:

$$\Delta_j \underset{(1-\alpha)}{\in} d_j \pm t_{(1-\alpha/2)}^{2n_j-2} |d_j| \left[\frac{1}{n_j}\left(\frac{s_{1j}^2}{\bar{x}_{1j}^2} + \frac{\left(s_{1j}^2 + s_{2j}^2\right)}{\bar{x}_{(21)j}^2} - 2\frac{cov\left(\mathbf{x}_{1j}, \mathbf{x}_{(21)j}\right)}{\bar{x}_{1j}\bar{x}_{(21)j}}\right)\right]^{\frac{1}{2}}$$

where $\mathbf{x}_{ij}$ representas a vector of observations sampled from X_{ij}, and $\mathbf{x}_{(21)j} = \mathbf{x}_{2j} - \mathbf{x}_{1j}$. Since X_{1j} and X_{2j} are independently distributed, it follows that $\bar{X}_{1j}$ and $\bar{X}_{(21)j}$ are also independent. This means that the covariance term can be removed from the equation, leading to a simplified form:

$$\Delta_j \underset{(1-\alpha)}{\in} d_j \pm t_{(1-\alpha/2)}^{2n_j-2} |d_j| \left[\frac{1}{n_j}\left(\frac{s_{1j}^2}{\bar{x}_{1j}^2} + \frac{\left(s_{1j}^2 + s_{2j}^2\right)}{\bar{x}_{(21)j}^2}\right)\right]^{\frac{1}{2}}$$

Simulation results show that this approximation provides intervals with good coverage properties, as long as the stated assumptions hold.

```r
# Define simulation values
mu1   <- 900
mu2   <- 450
var1  <- (50) ^ 2
var2  <- (100) ^ 2
Delta <- (mu2 - mu1) / mu1
alpha <- 0.05
N     <- 25
nsim  <- 10000

# Prepare dataframe for results
tmp   <- numeric(nsim)
CI    <- data.frame(M1  = tmp, M21 = tmp, D   = tmp,
                    V11 = tmp, V22 = tmp, V12 = tmp,
                    SE  = tmp, CIl = tmp, CIu = tmp,
                    CIW = tmp)
CI2 <- CI[, 8:10]
for (i in 1:nsim){
  # Simulate values returned by N runs of algorithms 1 and 2
```

[3]E.C. Fieller, *Some Problems in Interval Estimation.* Journal of the Royal Statistical Society. Series B (Methodological) 16(2):175-185, 1954. Available at http://www.jstor.org/stable/2984043

[4]V.H. Franz, *Ratios: A short guide to confidence limits and proper use.* arXiv:0710.2024v1 [stat.AP], 10 Oct 2007. Available at https://arxiv.org/pdf/0710.2024v1.pdf

```r
  x1        <- rnorm(N, mu1, sqrt(var1))
  x2        <- rnorm(N, mu2, sqrt(var2))
  x21       <- x2 - x1

  # Calculate sample statistics
  CI$M1[i]  <- mean(x1)                          # \bar{x}_{1j}
  CI$M21[i] <- mean(x21)                         # \bar{x}_{(21)j}
  CI$V11[i] <- var(x1)                         # \sigma_{1j}^2
  CI$V22[i] <- var(x2)                         # \sigma_{2j}^2
  CI$V12[i] <- cov(x1, x21)                  # cov(x_{1j},x_{(21)j})
  CI$D[i]   <- CI$M21[i] / CI$M1[i] # d_j

  # Calculate standard error (considering the covariance)
  CI$SE[i]  <- with(CI[i, ], abs(D) * sqrt(
    (V11 / (M1 ^ 2) +
       (V11 + V22) / (M21 ^ 2) -
       2 * V12 / (M1 * M21)) / N))

  # Confidence bounds (considering the covariance)
  CI$CIl[i] <- CI$D[i] + qt(alpha/2, 2*N-2) * CI$SE[i]
  CI$CIu[i] <- CI$D[i] + qt(1 - alpha/2, 2*N-2) * CI$SE[i]
  CI$CIW[i] <- 0.5 * (CI$CIu[i] - CI$CIl[i])

  # Standard error (without the covariance)
  se2  <- with(CI[i, ], abs(D) * sqrt(
    (V11 / (M1 ^ 2) +
       (V11 + V22) / (M21 ^ 2)) / N))

  # Confidence bounds (without the covariance)
  CI2$CIl[i] <- CI$D[i] + qt(alpha/2, 2*N-2) * se2
  CI2$CIu[i] <- CI$D[i] + qt(1 - alpha/2, 2*N-2) * se2
  CI2$CIW[i] <- 0.5 * (CI2$CIu[i] - CI2$CIl[i])

}

# Get the effective confidence level of the intervals
cat("Effective confidence level, with covariance: ",
    sum((Delta > CI$CIl) & (Delta < CI$CIu))/nsim, "\n")
```

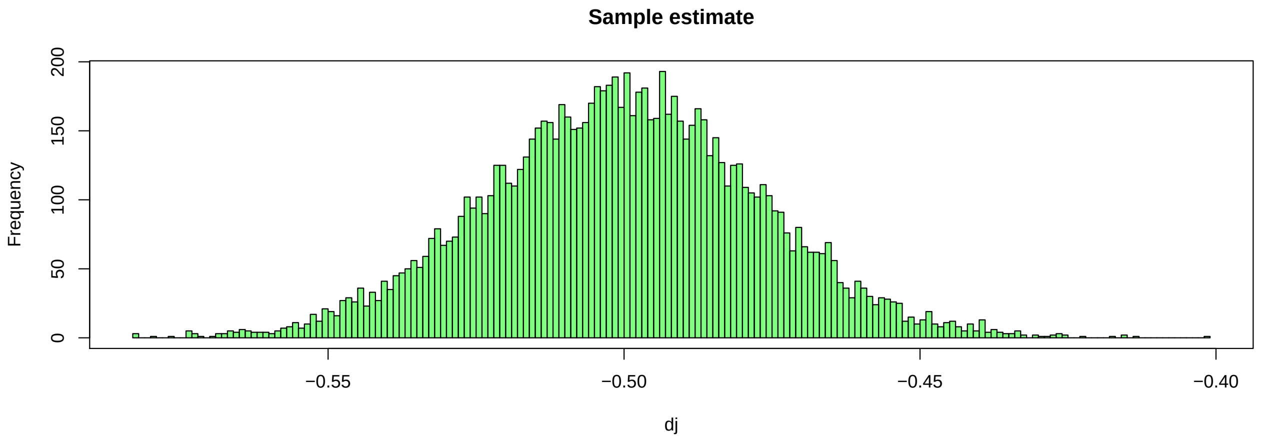
```
## Effective confidence level, with covariance:  0.9433
```

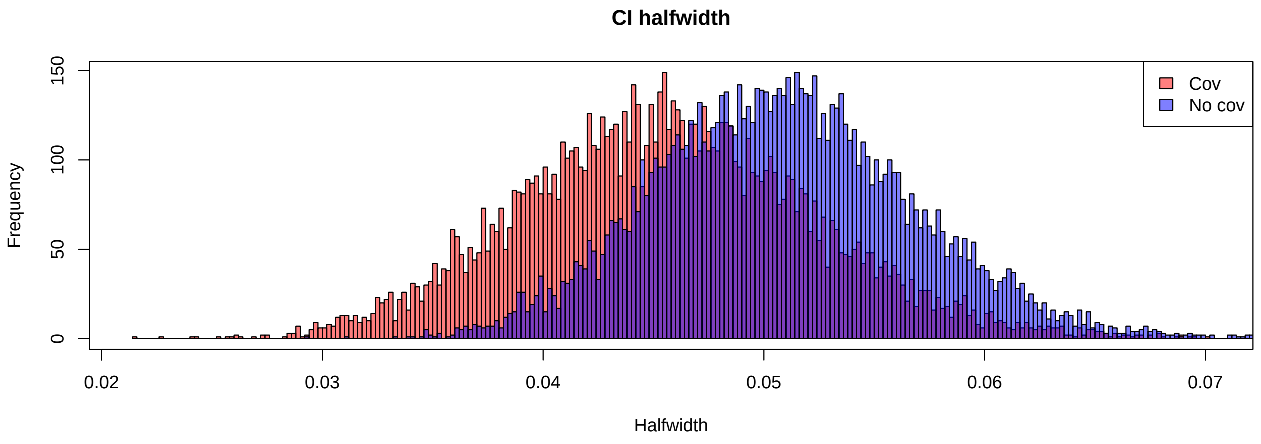
```r
cat("Effective confidence level, without covariance: ",
    sum((Delta > CI2$CIl) & (Delta < CI2$CIu))/nsim)
```

```
## Effective confidence level, without covariance:  0.9673
```

```r
# Plot the observed Delta values and CI half-widths
par(mfrow = c(2, 1))
hist(CI$D,    col = rgb(0,1,0,0.5), breaks = 200,
     main = "Sample estimate", xlab = "dj") # GREEN
box()
hist(CI$CIW,  col = rgb(1,0,0,0.5), breaks = 200,
     main = "CI halfwidth", xlab = "Halfwidth") # RED
hist(CI2$CIW, col = rgb(0,0,1,0.5), breaks = 200, add = TRUE) # BLUE
legend("topright", c("Cov", "No cov"), fill=c(rgb(1,0,0,0.5), rgb(0,0,1,0.5)))
```

```r
box()
```





It should be noted that the CI half-width increases when we do not consider the sample covariance, which in turn would mean that larger sample sizes would be required to control the half-width (or, equivalently, the standard error) at a predefined magnitude. On the other hand, it **may** be possible to compensate for this by considering different sample sizes in the no-covariance interval (this cannot be done with the interval that uses covariance, since covariances are calculated for equal-length vectors).

Before proceeding to the analysis of different-sized samples, a quick final note on sample size calculation for the balanced samples case: the standard error of the estimate, which is used in the calculation of the confidence interval, is given by:

$$s_{\Delta_j} = |d_j|\left[\frac{1}{n_j}\left(\frac{s_{1j}^2}{\bar{x}_{1j}^2} + \frac{\left(s_{1j}^2 + s_{2j}^2\right)}{\bar{x}_{(21)j}^2} - 2\frac{cov\left(\mathbf{x}_{1j}, \mathbf{x}_{(21)j}\right)}{\bar{x}_{1j}\bar{x}_{(21)j}}\right)\right]^{\frac{1}{2}}$$

for the expression considering the covariance, or:

$$s_{\Delta_j} = |d_j|\left[\frac{1}{n_j}\left(\frac{s_{1j}^2}{\bar{x}_{1j}^2} + \frac{\left(s_{1j}^2 + s_{2j}^2\right)}{\bar{x}_{(21)j}^2}\right)\right]^{\frac{1}{2}}$$

for the simplified standard error. In both cases, the standard error can be interpreted as the *measurement error* on the true value of Δ_j. For a desired upper limit on this measurement error ($s_{\Delta_j}^*$), the sample size n_j, corresponding to the number of runs to be performed be algorithms θ_1 and θ_2 on problem instance γ_j, can be determined iteratively. Solving for n_j in both cases yields the expressions:

$$n_j^{(cov)} = \frac{1}{\left(s^*_{\Delta_j}\bar{x}_{1j}\right)^2}\left[s_{1j}^2\left(d_j^2+1\right)+s_{2j}^2-2\left|d_j\right|cov\left(\mathbf{x}_{1j},\mathbf{x}_{(21)j}\right)\right]$$

$$n_j^{(no\ cov)} = \frac{1}{\left(s^*_{\Delta_j}\bar{x}_{1j}\right)^2}\left[s_{1j}^2\left(d_j^2+1\right)+s_{2j}^2\right]$$

Instead of using these formulas *a priori*, it is more practical to iterate the sampling until a predefined measurement error is achieved - particularly if we consider that this sampling is intended at providing an **approximately** upper-bounded measurement error, not an exact coverage for a confidence interval.

If we assume now that $n_{1j} \neq n_{2j}$, the formulas must be modified slightly. First, the distribution of the sample means must be updated to:

$$\bar{X}_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma_{ij}^2/n_{ij})$$

$$\bar{X}_{(21)j} \sim \mathcal{N}\left(\mu_{1j}-\mu_{2j}, \frac{\sigma_{1j}^2}{n_{1j}}+\frac{\sigma_{2j}^2}{n_{2j}}\right)$$

The simplified CI formula then becomes:

$$\Delta_j \underset{(1-\alpha)}{\in} d_j \pm t_{(1-\alpha/2)}^{n_{1j}+n_{2j}-2} s_{\Delta_j}$$

with:

$$s_{\Delta_j} = \left|d_j\right|\left[an_{1j}^{-1}+bn_{2j}^{-1}\right]^{1/2}$$

$$a = \left(\frac{s_{1j}}{\bar{x}_{1j}}\right)^2 + \left(\frac{s_{1j}}{\bar{x}_{(21)j}}\right)^2 \tag{1}$$

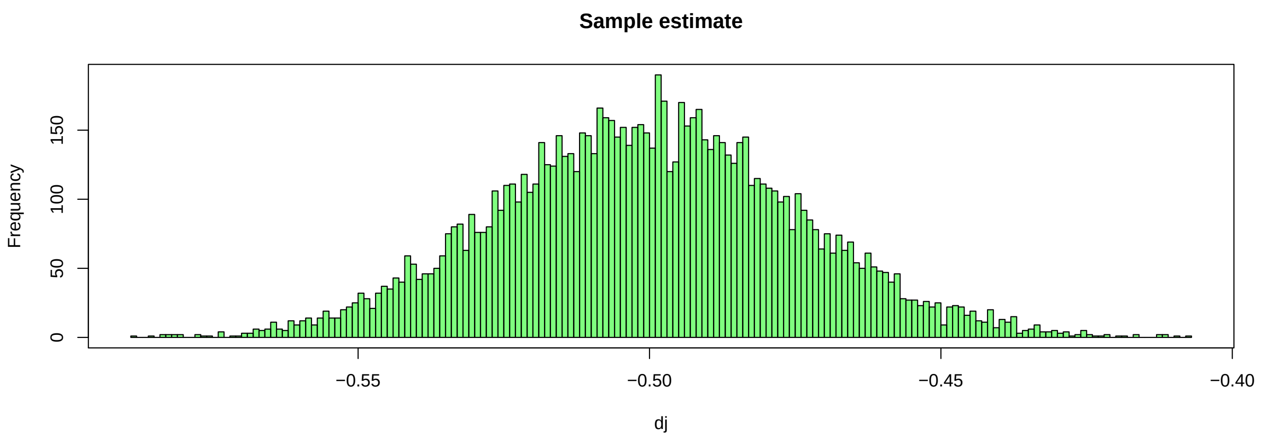
$$b = \left(\frac{s_{2j}}{\bar{x}_{(21)j}}\right)^2 \tag{2}$$

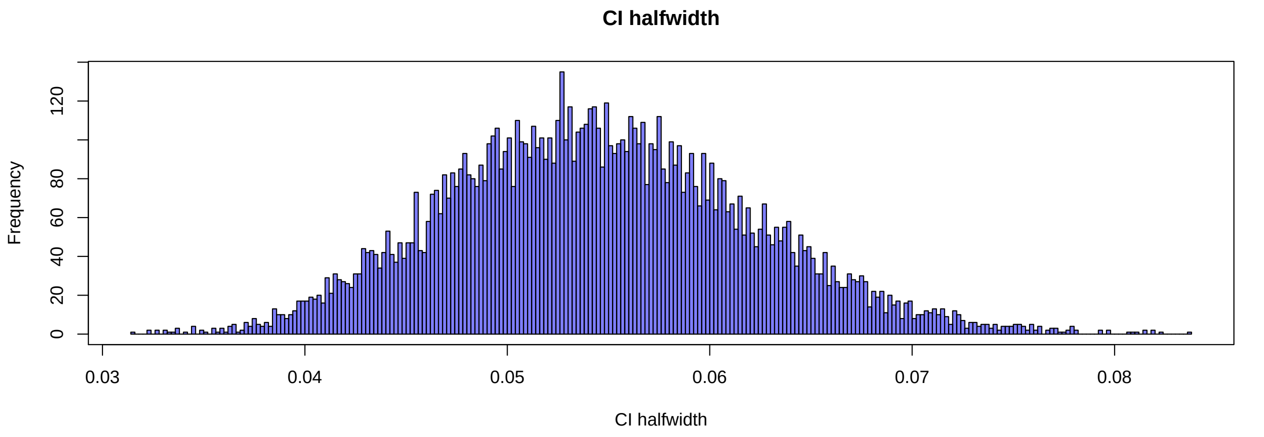
We can also validate this formula using simulation:

```r
# Different sample sizes
N1 <- 30
N2 <- 20
CI <- 0 * CI      # Reset values from the previous simulation

for (i in 1:nsim){
  # Simulate values returned by N runs of algorithms 1 and 2
  x1        <- rnorm(N1, mu1, sqrt(var1))
  x2        <- rnorm(N2, mu2, sqrt(var2))
  # Calculate sample statistics
  CI$M1[i]  <- mean(x1)               # \bar{x}_{1j}
  CI$M21[i] <- mean(x2) - CI$M1[i]  # \bar{x}_{(21)j}
  CI$V11[i] <- var(x1)            # \sigma_{1j}^2
  CI$V22[i] <- var(x2)            # \sigma_{2j}^2
  CI$D[i]   <- CI$M21[i] / CI$M1[i] # d_j

  # Calculate standard error
  CI$SE[i]  <- with(CI[i, ], abs(D) * sqrt(
    (V11/N1) / (M1 ^ 2) +
      (V11/N1 + V22/N2) / (M21 ^ 2)))

  # Confidence bounds
  CI$CIl[i] <- CI$D[i] + qt(alpha/2, N1+N2-2) * CI$SE[i]
  CI$CIu[i] <- CI$D[i] + qt(1 - alpha/2, N1+N2-2) * CI$SE[i]
  CI$CIW[i] <- 0.5 * (CI$CIu[i] - CI$CIl[i])
}

# Get the effective confidence level of the intervals
cat("Effective confidence level, different sample sizes: ",
    sum((Delta > CI$CIl) & (Delta < CI$CIu))/nsim, "\n")
```

```
## Effective confidence level, different sample sizes:  0.958
```

```
# Plot the observed Delta values and CI half-widths
par(mfrow = c(2, 1))
hist(CI$D,    col = rgb(0,1,0,0.5), breaks = 200,
     xlab = "dj", main = "Sample estimate") # GREEN
box()
hist(CI$CIW,  col = rgb(0,0,1,0.5), breaks = 200,
     xlab = "CI halfwidth", main = "CI halfwidth") # BLUE
box()
```





To find the smallest total sample size $n_{1j} + n_{2j}$ such that the *measurement error* becomes smaller than a predefined threshold (i.e., such that $s_{\Delta_j} \leq s^*_{\Delta_j}$), we can define the following optimization problem:

$$Minimize: \ f(\mathbf{n}_j) = n_{1j} + n_{2j} \tag{3}$$
$$Subject\ to: \ g(\mathbf{n}_j) = s_{\Delta_j} - s^*_{\Delta_j} \leq 0 \tag{4}$$

To find the values of n_{1j} and n_{2j} that satisfy the Karush-Kuhn-Tucker conditions for optimality, we need to solve:

$$\nabla f(\mathbf{n}^*_j) + \beta \nabla g(\mathbf{n}^*_j) = 0 \tag{5}$$
$$\beta g(\mathbf{n}^*_j) = 0 \tag{6}$$
$$\tag{7}$$

with $\beta \geq 0$. Taking the partial derivatives results in the following system:

$$n^2_{1j}\left[an^{-1}_{1j} + bn^{-1}_{2j}\right]^{1/2} = \frac{a\beta\,|d_j|}{2} \tag{8}$$
$$n^2_{2j}\left[an^{-1}_{1j} + bn^{-1}_{2j}\right]^{1/2} = \frac{b\beta\,|d_j|}{2} \tag{9}$$
$$an^{-1}_{1j} + bn^{-1}_{2j} = \left[\frac{s^*_{\Delta_j}}{d_j}\right]^2 \tag{10}$$

with $\beta > 0$ (since $\beta = 0$ would lead to inconsistencies in Eqs. (8)-(9)).
Isolating n_{1j}^{-1} in (10) we have:

$$n_{1j}^{-1} = \frac{1}{a}\left[\left(\frac{s^*_{\Delta_j}}{d_j}\right)^2 - bn_{2j}^{-1}\right] \tag{11}$$

Substituting (11) into (8) results in:

$$n_{2j}^{-1} = \frac{1}{b}\left[\left(\frac{s^*_{\Delta_j}}{d_j}\right)^2 - \left(\frac{2as^*_{\Delta_j}}{\beta d_j^2}\right)^{1/2}\right] \tag{12}$$

Replacing (12) back into (11):

$$n_{1j}^{-1} = \left[\frac{2s^*_{\Delta_j}}{a\beta d_j^2}\right]^{1/2} \tag{13}$$

Applying (12) and (13) into (9) and solving for β results in:

$$\beta = \frac{2d_j^2}{(s^*_{\Delta_j})^3}\left(\sqrt{a} + \sqrt{b}\right)^2 \tag{14}$$

Finally, applying this result back into (12) and (13) gives the expressions for n^*_{1j} and n^*_{2j}:

$$n^*_{1j} = \left(\frac{d_j}{s^*_{\Delta_j}}\right)^2\left(a + \sqrt{ab}\right) \tag{15}$$

$$n^*_{2j} = \left(\frac{d_j}{s^*_{\Delta_j}}\right)^2\left(b + \sqrt{ab}\right) \tag{16}$$

The optimal ratio of sample sizes can then be expressed as:

$$\left(\frac{n_{1j}}{n_{2j}}\right)^* = \frac{a + \sqrt{ab}}{b + \sqrt{ab}} \tag{17}$$

which, if we expand on a and b and simplify the resulting expression:

$$\left(\frac{n_{1j}}{n_{2j}}\right)^* = \frac{\left(s_{1j}/s_{2j}\right)^2\left(d_j^2 + 1\right) + \sqrt{d_j^2 + 1}}{1 + \sqrt{d_j^2 + 1}} \tag{18}$$

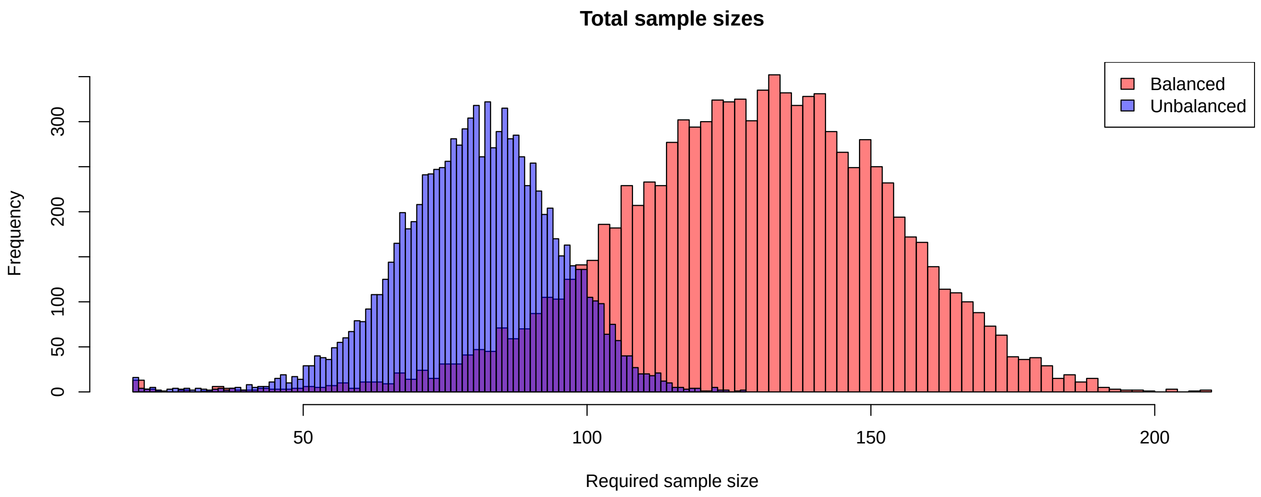
It is interesting to notice that the only situation in which a balanced sample (i.e., $n_{1j} = n_{2j}$) would be optimal is the particular case with:

$$\frac{s_{2j}^2}{s_{1j}^2} = d_j^2 + 1$$

Assuming that unequal sample sizes are to be used, n_{1j} and n_{2j} can be determined iteratively by monitoring the value of sample statistics and generating samples in proportion to the optimal ratio defined in (18).

The differences of total sample size required for achieving a predefined $s^*_{\Delta_j}$ can be verified using simulation.

```r
# Define simulation values
mu1   <- 50
mu2   <- 100
var1  <- (2) ^ 2
var2  <- (20) ^ 2
Delta <- (mu2 - mu1) / mu1
nsim  <- 10000

n0       <- 10   # initial number of samples to be obtained
s.target <- 0.05 # target standard error

N.equal   <- numeric(nsim)
N.unequal <- numeric(nsim)
for (i in 1:nsim){
  # simulate 10000 observations of algo1 and algo2
  X1 <- rnorm(10000, mu1, sqrt(var1))
  X2 <- rnorm(10000, mu2, sqrt(var2))

  # Balanced case
  nj <- n0 - 1
  SE <- Inf
  while ((SE > s.target) & (nj <= 10000)){
    # Cumulative samples of size nj
    nj  <- nj + 1
    x1  <- X1[1:nj]
    x2  <- X2[1:nj]
    x21 <- x2 - x1

    # Sample statistics
    M1  <- mean(x1)     # \bar{x}_{1j}
    M21 <- mean(x21)    # \bar{x}_{(21)j}
    V11 <- var(x1)      # \sigma_{1j}^2
    V22 <- var(x2)      # \sigma_{2j}^2
    V12 <- cov(x1, x21) # cov(x_{1j},x_{(21)j})
    D   <- M21 / M1     # d_j
    SE  <- abs(D) * sqrt((V11 / (M1 ^ 2) +
                          (V11 + V22) / (M21 ^ 2) -
                          2 * V12 / (M1 * M21)) / nj)
  }
  N.equal[i] <- 2 * nj # required total sample size, balanced case


  # Unbalanced case
  n1j       <- n0 - 1
  n2j       <- n0 - 1
  opt.ratio <- 1
  SE        <- Inf
  while ((SE > s.target) & (n1j <= 10000) & (n2j <= 10000)){
    # Update n1j, n2j according to the optimal ratio
    if (n1j/n2j <= opt.ratio) n1j <- n1j + 1
    if (n1j/n2j >= opt.ratio) n2j <- n2j + 1

    # Cumulative samples of size n1j, n2j
    x1 <- X1[1:n1j]
    x2 <- X2[1:n2j]

    # Sample statistics
    M1  <- mean(x1)              # \bar{x}_{1j}
    M21 <- mean(x2) - M1    # \bar{x}_{(21)j}
    V11 <- var(x1)             # \sigma_{1j}^2
    V22 <- var(x2)             # \sigma_{2j}^2
    D   <- M21 / M1         # d_j
    a   <- V11 / (M1 ^ 2) + V11 / (M21 ^ 2)
    b   <- V22 / (M21 ^ 2)
    SE  <- abs(D) * sqrt(a / n1j + b / n2j)

    opt.ratio <- (a + sqrt(a * b)) / (b + sqrt(a * b))
  }
  N.unequal[i] <- n1j + n2j # required total sample size, balanced case
}

# plot the total sample sizes for each case
hist(N.equal,  col = rgb(1,0,0,0.5), breaks = 100,
     main = "Total sample sizes", xlab = "Required sample size") # RED
hist(N.unequal, col = rgb(0,0,1,0.5), breaks = 100, add = TRUE) # BLUE
legend("topright", c("Balanced", "Unbalanced"), fill=c(rgb(1,0,0,0.5), rgb(0,0,1,0.5)))
```



Notice that in this case the expression used for the CI in the balanced case is the one in which the covariance estimate is included, since it requires smaller sample sizes for achieving the same upper bound for the standard error. Using the no-covariance formula would result in still larger sample sizes.

The difference between the total sample size for the balanced and the unbalanced cases can be derived analytically. This difference is given by:

$$\Delta_N = N_{balanced} - N_{unbalanced} = 2n_j^{(cov)} - (n_{1j}^* + n_{2j}^*)$$

Using the definition of $n_j^{(cov)}$ we have:

$$\begin{aligned} N_{balanced} &= \frac{2}{\left(s^*_{\Delta_j}\bar{x}_{1j}\right)^2}\left[s^2_{1j}\left(d^2_j+1\right)+s^2_{2j}-2\left|d_j\right| cov\left(\mathbf{x}_{1j},\mathbf{x}_{(21)j}\right)\right] \\ &= \frac{s^2_{1j}\left(d^2_j+1\right)+s^2_{2j}-2\left|d_j\right| cov\left(\mathbf{x}_{1j},\mathbf{x}_{(21)j}\right)}{\left(\bar{x}_{1j}s^*_{\Delta_j}\right)^2} \end{aligned} \tag{19}$$

Using (15) and (16) and expanding the values of a and b given in (1)-(2) yields:

$$\begin{aligned} N_{unbalanced} &= \left(\frac{d_j}{s^*_{\Delta_j}}\right)^2\left(a+b+\sqrt{ab}\right) \\ &= \left(\frac{d_j}{s^*_{\Delta_j}}\right)^2\left[\frac{s^2_{1j}}{\bar{x}^2_{1j}}+\frac{s^2_{1j}+s^2_{2j}}{\bar{x}^2_{(21)j}}+2\left(\frac{s^2_{1j}s^2_{2j}}{\bar{x}^2_{1j}\bar{x}^2_{(21)j}}+\frac{s^2_{1j}s^2_{2j}}{\bar{x}^4_{(21)j}}\right)^{1/2}\right] \\ &= \frac{s^2_{1j}\left(d^2_j+1\right)+s^2_{2j}+2s_{1j}s_{2j}\sqrt{d^2_j+1}}{\left(\bar{x}_{1j}s^*_{\Delta_j}\right)^2} \end{aligned} \tag{20}$$

Taking the difference results in:

$$\Delta_N = s^2_{1j}\left(d^2_j+1\right)+s^2_{2j}+2s_{1j}s_{2j}\left(\sqrt{d^2_j+1}-2r_{(12)j}\left|d_j\right|\right) \tag{21}$$

where

$$r_{(12)j} = \frac{cov\left(\mathbf{x}_{1j},\mathbf{x}_{(21)j}\right)}{s_{1j}s_{2j}}$$

is the Pearson's correlation coefficient of the two samples. Notice that the value of Δ_N defined in (21) for the typical case in which $r_{(12)j} \approxeq 0$ simplifies to:

$$\Delta_N \approxeq s^2_{1j}\left(d^2_j+1\right)+s^2_{2j}+2s_{1j}s_{2j}\sqrt{d^2_j+1}$$

Which is guaranteed to be positive, meaning that the total sample size needed to achieve a target standard error is always smaller in the unbalanced case than in the balanced one. In the more general situation, the unbalanced case will be more efficient than the balanced one if $\Delta_N > 0$, that is, if:

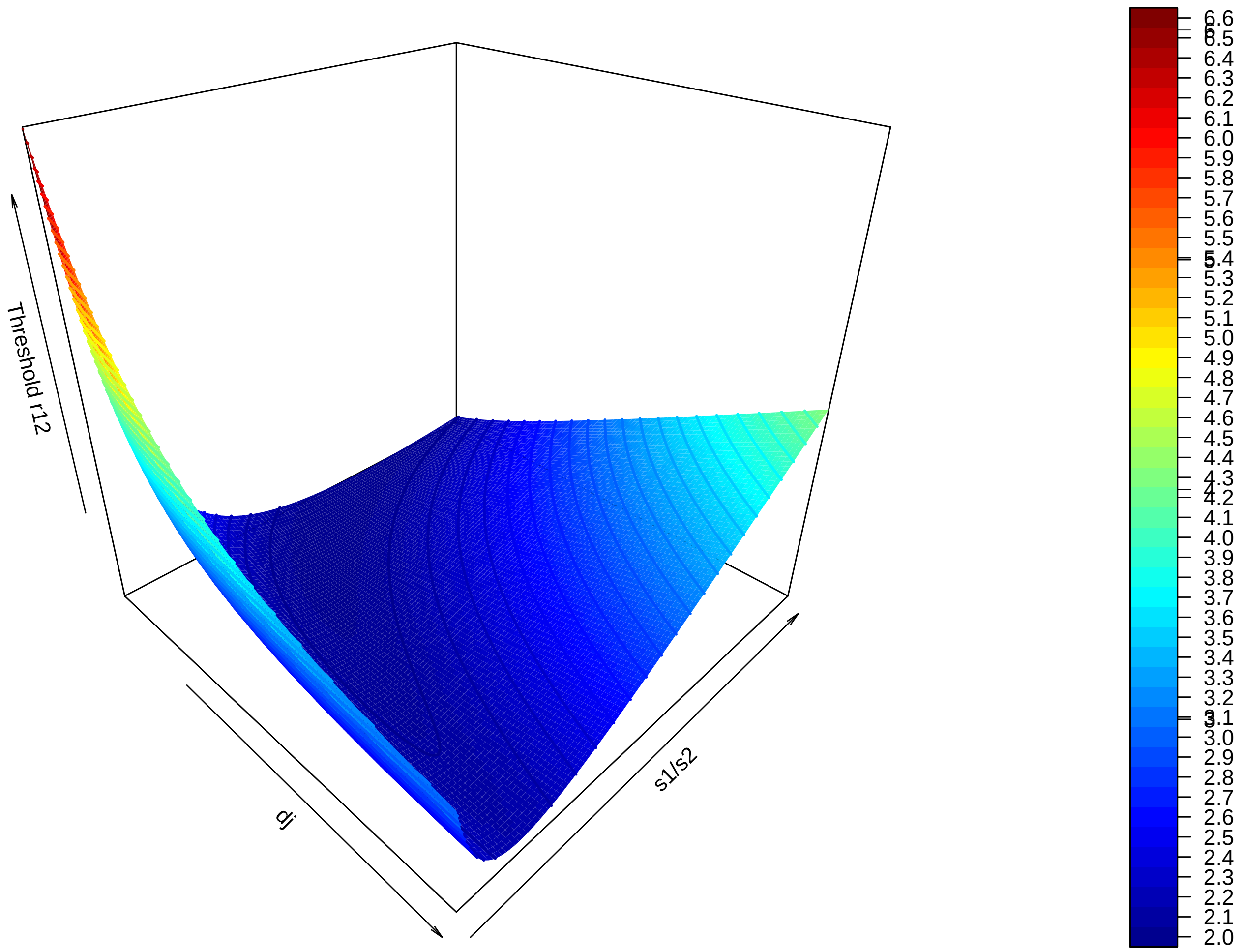
$$s^2_{1j}\left(d^2_j+1\right)+s^2_{2j}+2s_{1j}s_{2j}\left(\sqrt{d^2_j+1}-2r_{(12)j}\left|d_j\right|\right) > 0$$

Which can be solved for $r_{(12)j}$ as:

$$r_{(12)j} < \frac{1}{2\left|d_j\right|}\left[\left(\frac{s_{1j}}{s_{2j}}\right)\left(d^2_j+1\right)+\frac{s_{2j}}{s_{1j}}+2\sqrt{d^2_j+1}\right]$$

We can simulate threshold values for $r_{(12)j}$ using different values of $|d_j|$ and s_{1j}/s_{2j}:

```r
npts <- 100
dj   <- matrix(rep(seq(from = 1,  to = 15, length.out = npts), npts),
              nrow = npts, byrow = TRUE)
s1s2 <- matrix(rep(seq(from = .1, to = .4, length.out = npts), npts),
              nrow = npts, byrow = FALSE)
r12j <- (s1s2 * (dj^2 + 1) + (1 / s1s2) + 2 * sqrt(dj^2 + 1)) / (2 * dj)

library(plot3D)
persp3D(x = dj[1, ], y = s1s2[,1], z = r12j,
        plot = FALSE, phi = 30, theta = 45,
        xlab = "dj", ylab = "s1/s2", zlab = "Threshold r12")
contour3D(x = dj[1, ], y = s1s2[,1], z = r12j,
          colvar = r12j,
          lwd = 2, nlevels = 40,
          add = TRUE, phi = 30, theta = 45)
```



Notice that at no point in this simulation the threshold value for $r_{(12)j}$ falls below unity, which is the theoretical maximum value for a correlation coefficient. This strongly suggests that the conditions under which the unbalanced case is more effective than the balanced one are always (or, at least, almost always) met, and that we should proceed with the unbalanced sampling procedure outlined above.

2: Using Simple Differences

Let us now consider the situation in which the researcher is interested in comparing the performance of the two algorithms on problem class Γ by means of the simple difference,

$$\Delta'_j = \mu_{2j} - \mu_{1j}$$

This is useful in situations when the problem class of interest is composed of instances that are relatively homogeneous, so that defining the desired *measurement error* in absolute terms (instead of using a proportional error as was done in the previous section) makes intuitive sense. As mentioned at the beginning of Section , this approach is also useful in cases where the assumption $P(X_{1j} \leq 0) \rightarrow 0$ cannot be guaranteed.

The difference of means Δ_j can be estimated by the random variable:

$$D'_j = \bar{X}_{2j} - \bar{X}_{1j} \sim \mathcal{N}\left(\mu_{2j} - \mu_{1j}, \frac{\sigma^2_{1j}}{n_{1j}} + \frac{\sigma^2_{2j}}{n_{2j}}\right)$$

As in the previous section, we want to define the smallest total sample size $n_{1j} + n_{2j}$ such that the *measurement error* becomes smaller than a predefined threshold. We can achieve this by defining and solving the same optimization problem defined in (4),

$$\textit{Minimize}: \ f(\mathbf{n}_j) = n_{1j} + n_{2j} \tag{22}$$
$$\textit{Subject to}: \ g(\mathbf{n}_j) = s_{\Delta'_j} - s^*_{\Delta'_j} \leq 0 \tag{23}$$

but instead using the standard error of the sample difference of means defined by:

$$s_{\Delta'_j} = \frac{\sigma^2_{1j}}{n_{1j}} + \frac{\sigma^2_{2j}}{n_{2j}} \tag{24}$$

Taking the gradient of the objective and constrain functions and using the method of Lagrange multipliers to find the optimal sample size values, $\left[n^*_{1j}, n^*_{2j}\right]$, we obtain the system of equations:

$$\beta\frac{\sigma^2_{1j}}{n^2_{1j}} = 1 \tag{25}$$
$$\beta\frac{\sigma^2_{2j}}{n^2_{2j}} = 1 \tag{26}$$
$$\beta\left[\frac{\sigma^2_{1j}}{n_{1j}} + \frac{\sigma^2_{2j}}{n_{2j}} - s^*_{\Delta'_j}\right] = 0 \tag{27}$$

with $\beta > 0$ (simple inspection of (25)-(26) shows $\beta = 0$ to be an impossibility). Some simple manipulation of (25)-(26) yields:

$$n_{1j} = \sqrt{\beta}\sigma_{1j} \tag{28}$$
$$n_{2j} = \sqrt{\beta}\sigma_{2j} \tag{29}$$
$$\tag{30}$$

it is not even necessary to find the value of β to realize that the optimal ratio of sample sizes is given by

$$\left(\frac{n_{1j}}{n_{2j}}\right)^{*} = \frac{\sigma_{1j}}{\sigma_{2j}} \tag{31}$$

That is, that algorithms θ_1 and θ_2 must be sampled on instance γ_j in direct proportion to their standard deviations on that instance[5]. A good estimator for this optimal ratio of sample sizes can be obtained by using the sample standard deviations instead of the (unknown) population values.

3: Calculating the number of instances

---

[5]This result, by the way, is also known as the *optimal allocation of resources* in the literature on sample size calculations. See, for instance, Paul Matthew's *Sample Size Calculations: Practical Methods for Engineers and Scientists*, Chapter 2, MMB 2010.