# Efecte de l'aspirina sobre les hemorràgies

Ocaña, J & Civit, S

21 de febrer de 2018

## 1 Enunciat

En un estudi sobre l'efecte de l'aspirina en les hemorràgies i l'adherència de plaquetes, uns investigadors (Bick, R.L., Adams, T. and Schmalhorts, W.R. "Bleeding times, platelet adhesion and aspirin". *J. Clin. Path.* **65**, 69-72, 1976) van estudiar les reaccions de subjectes sans a l'aspirina. Les dades de la taula mostren la durada d'hemorràgia, en segons, abans i després (passades 2 hores) de l'ingestió de 600mg d'àcid acetilsalicílic.

| **subjecte** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **abans** | 270 | 150 | 270 | 420 | 202 | 255 | 165 | 220 | 305 | 210 | 240 |
| **després** | 525 | 570 | 190 | 395 | 370 | 210 | 490 | 250 | 360 | 285 | 630 |
| **subjecte** | 12 | 13 | 14 | | | | | | | | |
| **abans** | 300 | 300 | 70 | | | | | | | | |
| **després** | 385 | 195 | 295 | | | | | | | | |

La intenció d'aquests investigadors era demostrar que la ingestió d'aspirina incrementa la durada de les hemorràgies. Intenta provar aquesta hipòtesi mitjançant una prova de permutacions, i determina un interval de confiança al 95% per a la durada mitjana de les hemorràgies.

## 2 Solució

Es tracta clarament d'una situació de dades aparellades. En primer lloc obtindrem les diferències entre les dades d'abans i després.

```
> abans = c(270, 150, 270, 420, 202, 255, 165, 220, 305, 210, 240, 300, 300, 70)
> despres = c(525, 570, 190, 395, 370, 210, 490, 250, 360, 285, 630, 385, 195, 295)
> dif = despres - abans
> dif
```

```
 [1]  255  420  -80  -25  168  -45  325   30   55   75  390   85 -105  225
```

```
> # Valor de la mitjana sobre les diferències originals:
> meanDif = mean(dif)
> meanDif
```

```
[1] 126.6429
```

```
> # vector amb els valors absoluts de les diferències:
> absDif = abs(dif)
> # Aquesta funció calcula la mitjana de les diferències sobre una mostra
> # permutada per files:
> meanPerm = function(signs, absDiff) mean(signs * absDiff)
> # 'signs' ha de ser un vector de valors -1 o +1 de la mateixa llargada que 'absDiff'
```

## 2.1 Test de permutacions exacte

```
> # Enumeració de totes les permutacions possibles:
> n = length(dif)
> sgn = c(-1, +1)
> signsTab = expand.grid(as.data.frame(matrix(rep(sgn, n), ncol = n)))
> # Les primeres 10 files de 'signsTab':
> signsTab[1:10,]
```

```
   V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14
1  -1 -1 -1 -1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
2   1 -1 -1 -1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
3  -1  1 -1 -1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
4   1  1 -1 -1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
5  -1 -1  1 -1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
6   1 -1  1 -1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
7  -1  1  1 -1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
8   1  1  1 -1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
9  -1 -1 -1  1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
10  1 -1 -1  1 -1 -1 -1 -1 -1  -1  -1  -1  -1  -1
```

```
> # Càlcul de la diferència de mitjanes per cada permutació possible:
> dMeansPerm = apply(signsTab, 1, meanPerm, absDiff = absDif)
> # p-valor exacte del test unilateral "després > abans":
> sum(dMeansPerm >= meanDif) / nrow(signsTab)
```

```
[1] 0.009155273
```

## 2.2 Test de permutacions de Monte Carlo

Tal com hem vist, amb 14 dades aparellades podem realitzar perfectament un test de permutacions exacte. Si tinguèssim més dades possiblement necessitaríem fer un test de Monte Carlo.

El codi d'un test de permutacions aproximat de Monte Carlo podria ser quelcom així:

```
> # Generació d'una permutació (segons la convenció amb els signes):
> sample(sgn, size = n, replace = TRUE)
```

```
 [1]  1  1 -1  1 -1  1  1 -1 -1 -1  1  1 -1  1
```

```
> # Avaluació de la mitjana de les diferències sobre una permutació aleatòria:
> meanPerm(sample(sgn, size = n, replace = TRUE), absDif)
```

```
[1] -93.07143
```

```
> # Generació de nPerms permutacions aleatòries i càlcul de la mitjana de les
> # diferències sobre cadascuna d'elles:
> nPerms = 9999
> dMeansPerm = replicate(nPerms, meanPerm(sample(sgn, size = n, replace = TRUE), absDif))
> # Càlcul del p-valor aproximat pel test unilateral "després > abans":
> (sum(dMeansPerm >= meanDif) + 1) / (nPerms + 1)
```

```
[1] 0.0089
```

## 2.3 Interval de confiança per a la diferència de mitjanes

Suposem que $\delta$ és la diferència poblacional de mitjanes "després - abans". Donades unes dades $\mathbf{x}$ com ara les d'aquest exemple, si per qualsevol valor $\delta_0$ existeix un test $\phi(\delta_0, \mathbf{x})$ amb nivell de significació $\alpha$ pel contrast unilateral $H_0 : \delta = \delta_0$ vs $H_1 : \delta > \delta_0$ i $p(\phi(\delta_0, \mathbf{x}))$ representa el p-valor del test per les dades $\mathbf{x}$, l'interval $\{\delta_0 : p(\phi(\delta_0, \mathbf{x})) > \alpha\}$ és un interval de confiança de la forma $(\hat{\delta}_L, +\infty]$ amb nivell de confiança $1 - \alpha$.

```
> # Funció que calcula el p-valor (exacte):
> # (per raons de brevetat i senzillesa, solament vàlida pel contrast unilateral
> # anterior)
> pairedP.val = function(delta0, x, y, signsTable) {
+   n = length(x)
+   if (missing(signsTable))
+     signsTable = expand.grid(as.data.frame(matrix(rep(c(-1,+1), n), ncol = n)))
+   y = y - delta0
+   dif = y - x
+   meanDif = mean(dif)
+   dMeansPerm = apply(signsTable, 1, meanPerm, absDiff = abs(dif))
+   return(sum(dMeansPerm >= meanDif) / nrow(signsTab))
+ }
> # Successives aproximacions:
> deltas = seq(from = 0, to = 100, by = 10)
> pvals = sapply(deltas, pairedP.val, x = abans, y = despres, signsTable = signsTab)
> matrix(c(deltas, pvals), nrow = 2, byrow = TRUE)
```

```
            [,1]       [,2]        [,3]        [,4]        [,5]        [,6]
[1,] 0.000000000 10.0000000 20.00000000 30.00000000 40.00000000 50.00000000
[2,] 0.009155273  0.0135498  0.02032471  0.02966309  0.04327393  0.06164551
            [,7]       [,8]       [,9]       [,10]       [,11]
[1,] 60.00000000 70.0000000 80.0000000 90.0000000 100.0000000
[2,]  0.08776855  0.1217651  0.1661987  0.2210083   0.2867432
```

```
> deltas = seq(from = 40, to = 50, by = 1)
> pvals = sapply(deltas, pairedP.val, x = abans, y = despres, signsTable = signsTab)
> matrix(c(deltas, pvals), nrow = 2, byrow = TRUE)

            [,1]        [,2]        [,3]       [,4]        [,5]        [,6]
[1,] 40.00000000 41.00000000 42.00000000 43.0000000 44.00000000 45.00000000
[2,]  0.04327393  0.04467773  0.04595947  0.0480957  0.04949951  0.05187988
            [,7]        [,8]        [,9]       [,10]       [,11]
[1,] 46.00000000 47.00000000 48.00000000 49.00000000 50.00000000
[2,]  0.05334473  0.05541992  0.05743408  0.05944824  0.06164551

> deltas = seq(from = 44, to = 45, by = 0.1)
> pvals = sapply(deltas, pairedP.val, x = abans, y = despres, signsTable = signsTab)
> matrix(c(deltas, pvals), nrow = 2, byrow = TRUE)

            [,1]        [,2]       [,3]        [,4]        [,5]        [,6]
[1,] 44.00000000 44.10000000 44.2000000 44.30000000 44.40000000 44.50000000
[2,]  0.04949951  0.04949951  0.0501709  0.05041504  0.05053711  0.05059814
            [,7]        [,8]        [,9]       [,10]       [,11]
[1,] 44.60000000 44.70000000 44.80000000 44.90000000 45.00000000
[2,]  0.05072021  0.05096436  0.05108643  0.05108643  0.05187988

> deltas = seq(from = 44.1, to = 44.2, by = 0.01)
> pvals = sapply(deltas, pairedP.val, x = abans, y = despres, signsTable = signsTab)
> matrix(c(deltas, pvals), nrow = 2, byrow = TRUE)

            [,1]        [,2]        [,3]        [,4]        [,5]        [,6]
[1,] 44.10000000 44.11000000 44.12000000 44.13000000 44.14000000 44.15000000
[2,]  0.04949951  0.04949951  0.04949951  0.04974365  0.04974365  0.04974365
            [,7]       [,8]       [,9]      [,10]      [,11]
[1,] 44.16000000 44.1700000 44.1800000 44.1900000 44.2000000
[2,]  0.04974365  0.0501709  0.0501709  0.0501709  0.0501709

> # Extrem inferior de l'interval, finalment aproximat per interpolació:
> approx(pvals, deltas, 0.05)$y

[1] 44.169
```