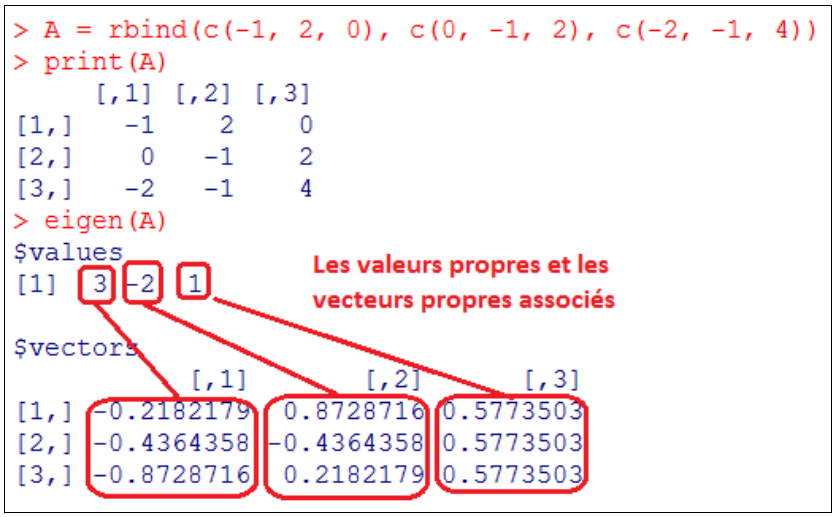
R et calcul matriciel

http://revue.sesamath.net/spip.php?article881
patrick.raffinat@univ-pau.fr

A) Exemple 1

Comme je l'ai dit dans l'article, j'illustre quelques fonctionnalités de R en calcul matriciel avec un premier exemple portant sur la diagonalisation d'une matrice :



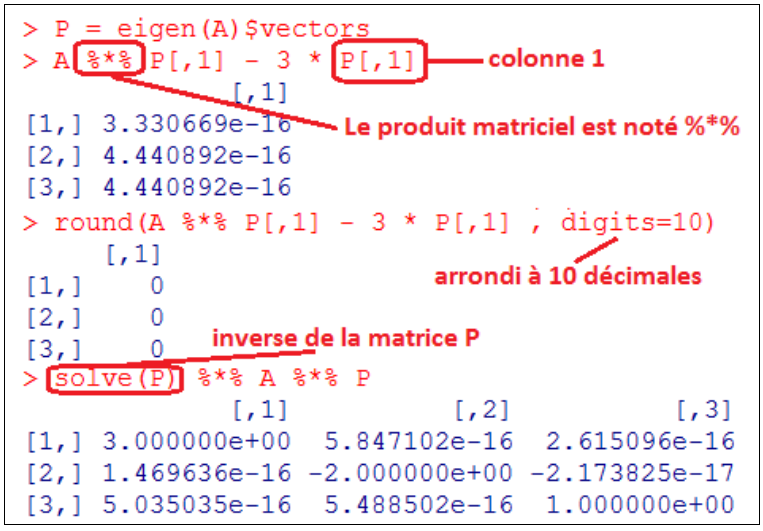

```
> A = rbind(c(-1, 2, 0), c(0, -1, 2), c(-2, -1, 4))
> print(A)
     [,1] [,2] [,3]
[1,]   -1    2    0
[2,]    0   -1    2
[3,]   -2   -1    4
> eigen(A)
$values
[1]  3 -2  1

$vectors
           [,1]       [,2]      [,3]
[1,] -0.2182179  0.8728716 0.5773503
[2,] -0.4364358 -0.4364358 0.5773503
[3,] -0.8728716  0.2182179 0.5773503
```

La matrice, définie ligne par ligne avec la fonction rbind, peut aussi être définie colonne par colonne avec la fonction cbind. La fonction eigen renvoie une liste composées de 2 éléments :

- eigen(A)$values est un vecteur composé de 3 valeurs propres
- eigen(A)$vectors est une matrice composée de 3 vecteurs propres

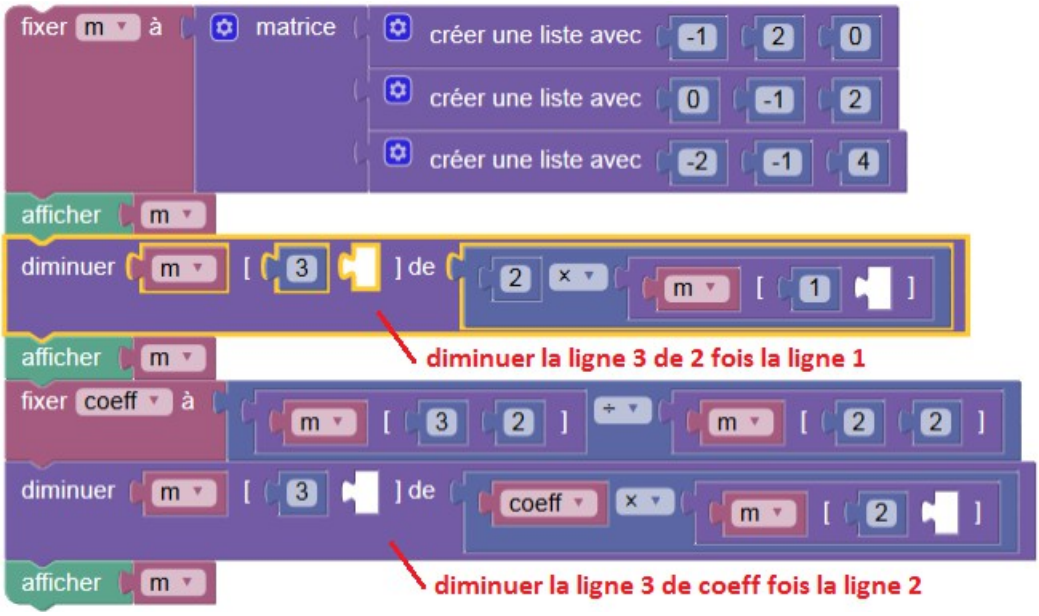
Le programme ci-dessous vérifie que 3 est une valeur propre et que la première colonne de la matrice eigen(A)$vectors est un vecteur propre associé :




```
> P = eigen(A)$vectors
> A %*% P[,1] - 3 * P[,1]
              [,1]
[1,] 3.330669e-16
[2,] 4.440892e-16
[3,] 4.440892e-16
> round(A %*% P[,1] - 3 * P[,1] , digits=10)
     [,1]
[1,]    0
[2,]    0
[3,]    0
> solve(P) %*% A %*% P
              [,1]          [,2]          [,3]
[1,] 3.000000e+00  5.847102e-16  2.615096e-16
[2,] 1.469636e-16 -2.000000e+00 -2.173825e-17
[3,] 5.035035e-16  5.488502e-16  1.000000e+00
```

B) Exemple 2

Ne trouvant pas pertinent de développer des blocs Blockly pour faciliter l'écriture de programmes R tels que celui de l'exemple 1, je me suis épargné ce travail. Néanmoins, j'ai ajouté quelques blocs permettant de faire de la programmation visuelle avec des matrices :



Cet exemple montre comment calculer le déterminant d'une matrice en se ramenant à une matrice triangulaire grâce à des combinaisons linéaires entre lignes. A l'exécution, on obtient ceci :

```
> m = rbind(c(-1, 2, 0), c(0, -1, 2), c(-2, -1, 4))
> print(m)
     [,1] [,2] [,3]
[1,]   -1    2    0
[2,]    0   -1    2
[3,]   -2   -1    4
> m[3,] = m[3,] - (2 * m[1,])
> print(m)
     [,1] [,2] [,3]
[1,]   -1    2    0
[2,]    0   -1    2
[3,]    0   -5    4
> coeff = m[3,2] / m[2,2]
> m[3,] = m[3,] - (coeff * m[2,])
> print(m)
     [,1] [,2] [,3]
[1,]   -1    2    0
[2,]    0   -1    2
[3,]    0    0   -6
```