Dérivée

Corentin BRETONNIERE

1/ Dériver par une fonction de dérivation

A. Obtenir l'expression d'une dérivée (méthode facile)

La fonction R "D" permet de dériver des *expressions* définies au préalable

```
f=expression(x^2+3*x)
D(f,'x')
```

```
## 2 * x + 3
```

C'est également possible de dériver avec avec deux inconnus

```
a<-expression(x^2+3*x+5*y^6)
D(a,'x')
```

```
## 2 * x + 3
```

```
D(a,'y')
```

```
## 5 * (6 * y^5)
```

A.bis. Obtenir l'expression d'une dérivée (méthode compliquée)

D'abord, il faut assigner à notre variable une fonction (maDerive <- function(x)), cette fonction est l'expression de la dérivée (monExpression <- deriv(~2*x^4 + sqrt(x), "x")). La "~" est essentielle et elle doit figurer avant l'équation à dériver. Ensuite, on assigne à une autre variale (r) la fonction R "eval", elle sert à évaluer une expression R. A cette variable r dans laquelle est stocké la valeur de la dérivée on lui ajoute un attribut avec la fonction R "attr", la dimension est 'gradient'. L'ajout de "; monExpression" à la fin permet d'afficher l'expression de la dérivée en "x".

```
monExpression <- deriv(~2*x^4 + sqrt(x), "x")
maDerive <- function(x){
    monExpression
    r <- eval(monExpression);
    r <- attr(r, 'gradient');
} ; monExpression
```

```
## expression({
##     .value <- 2 * x^4 + sqrt(x)
##     .grad <- array(0, c(length(.value), 1L), list(NULL, c("x")))
##     .grad[, "x"] <- 2 * (4 * x^3) + 0.5 * x^-0.5
##     attr(.value, "gradient") <- .grad
##     .value
## })
```

B. Obtenir la valeur en 1 point

Pour afficher la valeur en x=2 par exemple il faut utiliser cette notation.

```
valeurDunPoint <- maDerive(2); valeurDunPoint
```

```
##             x
## [1,] 64.35355
```

C. Dériver un vecteur ou une matrice

On peut également appliquer cette dérivé à toutes les valeurs d'un vecteur ... Pour ce faire il faut lui appliquer la fonction "Vectorize".

```
maDerive <- Vectorize(maDerive)

v <- c(1,2,3,4,5); maDerive(v)
```

```
## [1]    8.50000   64.35355  216.28868  512.25000 1000.22361
```

... ou d'une matrice !

```
m <- matrix(data = 1:12, nrow = 3, ncol = 4); m; maDerive(m)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

```
##  [1]     8.50000    64.35355   216.28868   512.25000  1000.22361  1728.20412
##  [7]  2744.18898  4096.17678  5832.16667  8000.15811 10648.15076 13824.14434
```

2/ Dériver par le calcul direct numérique de la dérivée

A. Avec le taux d'accroissement dx

```
derivee  <- function (f, dx){
            d <- function(x)(f(x + dx)-f(x))/dx
            }
```

Avec cette méthode, on exprime directement "l'essence même" de la dérivée avec son expression originel. Il faudra cependant préciser dx (valeur d'écart).

```
f <- function(x) x^2
d_ <- derivee(f,0.00000000001) #Valeur d'écart = 1 dix milliardième
d_(2)
```

```
## [1] 4
```

Si on augmente la valeur d'écart, ça modifie logiquement la pente de la dérivée

```
d_2 <- derivee(f,0.001)
d_2(2)
```

```
## [1] 4.001
```

B. Application à des vecteurs et des matrices

Avec un vecteur :

```r
d_vm <- Vectorize(d_)
v_ <- c(1,2,3,4); v_
```

```
## [1] 1 2 3 4
```

```r
d_vm(v_)
```

```
## [1] 2.000000 4.000000 6.000000 8.000001
```

Avec une matrice diagonale

```r
m_ <- diag(1:3,3); m_
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    2    0
## [3,]    0    0    3
```

```r
d_vm(m_)
```

```
## [1] 2e+00 1e-11 1e-11 1e-11 4e+00 1e-11 1e-11 1e-11 6e+00
```