Fiche TD avec le logiciel : tdr18

(Ne pas) utiliser des couleurs

J.R. Lobry

L'utilisation de la couleur n'est pas un problème simple. Ce que voient les personnes atteintes de déficiences oculaires. Comment peut-on fausser un message ? Les niveaux de gris. Superpositions des couleurs transparentes.

Table des matières

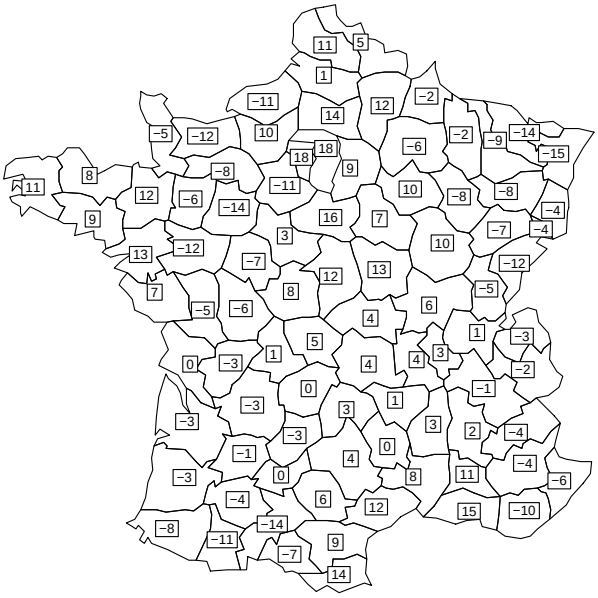
1 Introduction

Il est indispensable ici de citer Jacques Bertin [1] :

> On fait à l'auteur la réputation d'être contre la couleur. Je suis contre la couleur qui camoufle l'incompétence et je reste contre tant que l'on croira qu'elle suffit pour représenter un ordre, qu'elle permet de superposer des caractères jusqu'à la limite de l'absurde, tant qu'elle coûtera inutilement des fortunes aux deniers publics dans une confusion entre connaissance et publicité, tant que, par son truchement, les responsables et la télévision diffuseront des images fausses et illisibles.

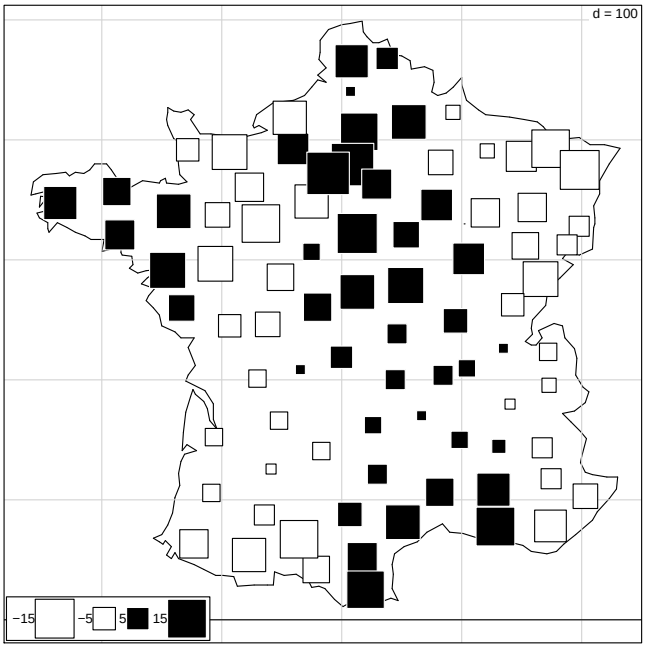
Reprenons l'exemple donné page 220 dans [1], l'information est la suivante :

```
bertin <- read.table("http://pbil.univ-lyon1.fr/R/donnees/bertin.txt",
    sep = "\t")
library(ade4)
data(elec88)
area.plot(elec88$area[1:1845, ], clab = 0.75, lab = bertin[, 2])
```



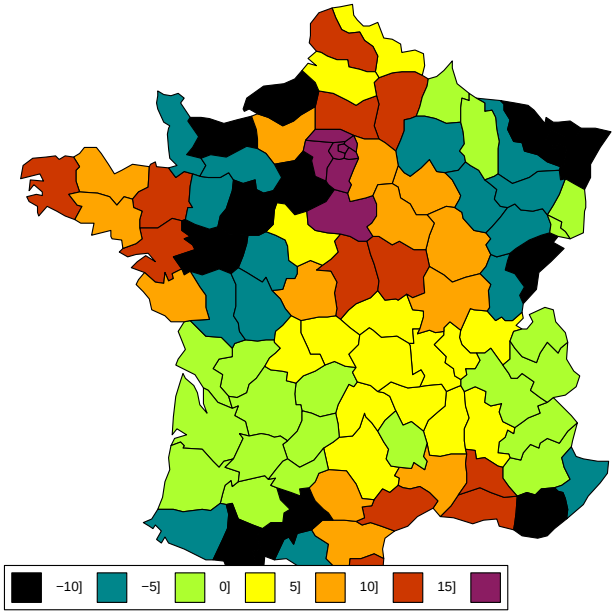
La structuration selon un axe nord-sud est évidente sur la représentation suivante :

```
s.value(elec88$xy[1:89, ], bertin[1:89, 2], contour = elec88$contour,
    meth = "squaresize", incl = F)
```

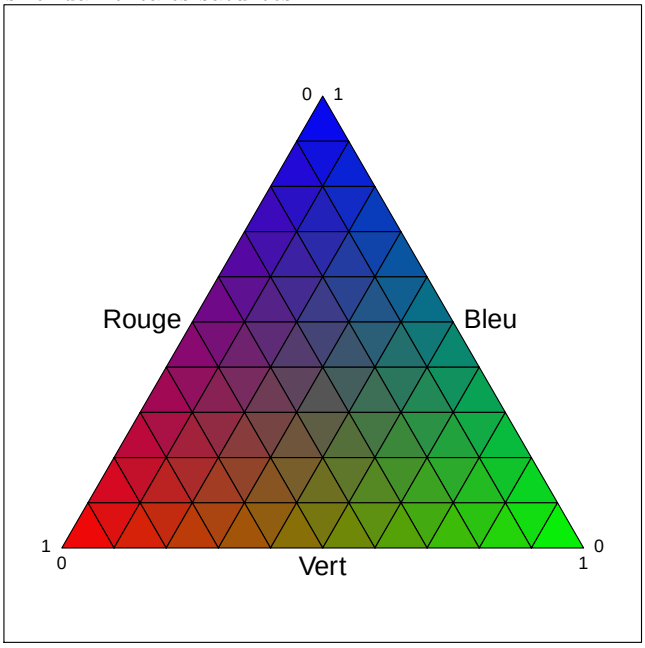


Essayons maintenant de colorier sauvagement cette carte :

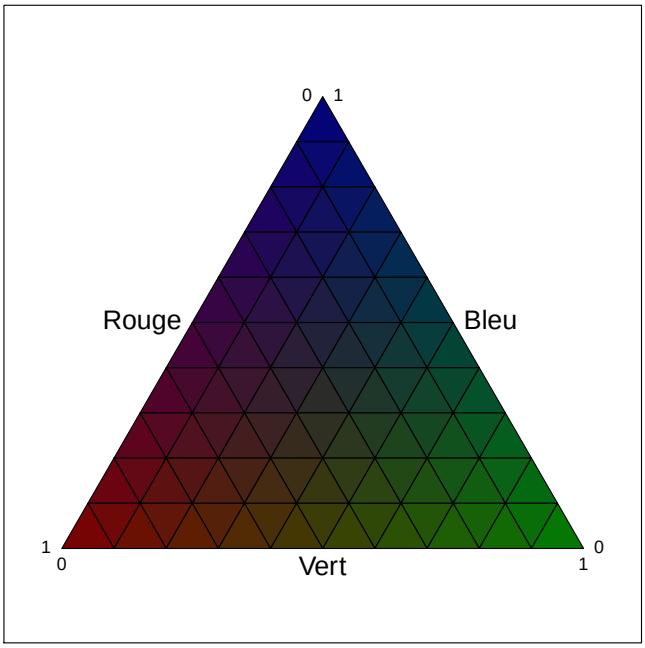
```
grey.bck <- grey
gray <- grey <- function(level) {
    pal <- rev(c("black", "turquoise4", "greenyellow", "yellow",
        "orange", "orangered3", "maroon4"))
    pal[round(1 + 6 * level)]
}
area.plot(elec88$area, val = c(bertin[, 2], rep(18, 5)), clab = 0,
    sub = "", csub = 3)
grey <- gray <- grey.bck
```



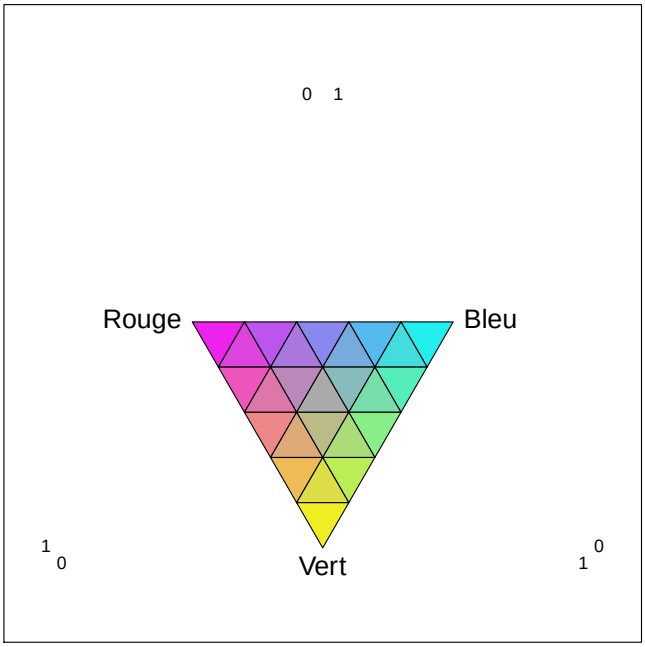
On obtient une structuration selon un axe est-ouest, et il est impossible d'abstraire cette orientation. La perception des valeurs domine la perception des teintes. Les extrémités du spectre sont foncées alors que les couleurs centrales sont claires, nous avons tendance à assimiler les deux extrémités dans une même unité perceptive. *À égalité de valeur, les couleurs ne sont pas visuellement ordonnées, elles ne peuvent donc pas représenter une information ordonnée*. Voici la palette des couleurs disponibles de même valeur que les couleurs fondamentales saturées :



Voici la palette des couleurs disponibles de même valeur que les couleurs fondamentales à 50 % de saturation, on assombrit globalement tout :



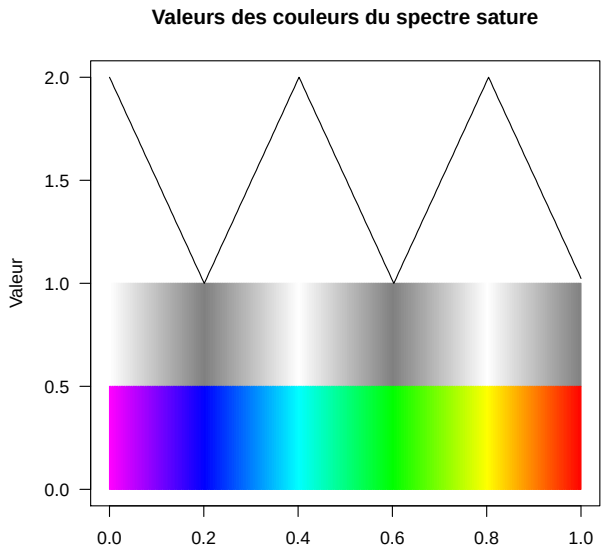
Voici la palette des couleurs disponibles de même valeur que les couleurs primaires à 100 % de saturation, les couleurs fondamentales ne sont alors plus disponibles :



Le problème de l'utilisation des couleurs est résumé dans le graphique suivant. Les couleurs pures n'ont pas toutes la même valeur : pour toute couleur pure il existe au moins deux autre couleurs pures que nous percevrons semblables avant de voir les différences.

```
n <- 201
plot(0, 0, xlim = c(0, 1), ylim = c(0, 2), type = "n", las = 1,
```

```
    main = "Valeurs des couleurs du spectre sature", xlab = "",
    ylab = "Valeur")
rb <- rev(rainbow(n, end = 5/6))
y <- numeric(n - 1)
for (i in seq(0, 1, length = n)[-n]) {
    ix <- n * i + 1
    col <- rb[ix]
    val <- sum(col2rgb(col))/255
    rect(i, 0.5, i + 1/n, 1, col = grey(val/2), border = grey(val/2))
    rect(i, 0, i + 1/n, 0.5, col = col, border = col)
    y[ix] <- val
}
lines(seq(0, 1, length = n - 1), y)
```



Le problème est que les couleurs pures sont les plus sélectives, ce sont celles que l'on aimerait bien utiliser pour faire "joli". Mais vouloir utiliser les spectre des couleurs pures pour représenter une variable ordonnée est une grave erreur car elle impose un ordre complètement artificiel, détruisant la perception de l'ordre naturel de la variable. Cela peut coûter très cher que de vouloir faire "joli". Êtes-vous bien certain de toujours vouloir utiliser des couleurs ?

2 Dyschromatopsies

L'utilisation de couleurs dans des documents destinés à être diffusés est assez délicate. Une forte proportion (de l'ordre de 5 %) des individus a des problèmes plus ou moins importants pour distinguer certaines couleurs.

Exercice : un scientifique envoie un article avec des figures en couleur à un journal pour publication. Cet article va être évalué par trois experts. Quelle est la probabilité pour qu'au moins un expert ait des problèmes pour lire les figures en couleur ?

```
[1] 0.142625
```

Exercice : les gènes codants pour les pigments rouges et verts étant à localisation hétrochromosomique, les dyschromatopsies héréditaires ne concernent,

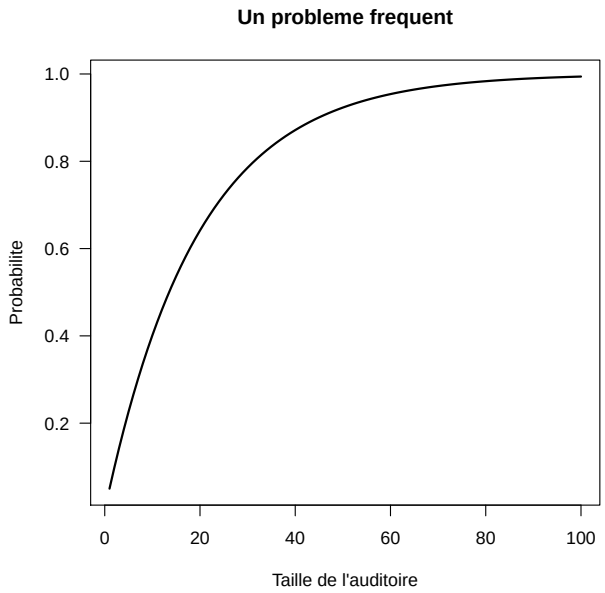
en première approximation, que les mâles. Sachant que la sexe-ratio des experts est de 0.9, quelle est la probabilité pour qu'au moins un expert ait des problèmes pour lire les figures en couleur ?

```
[1] 0.246429
```

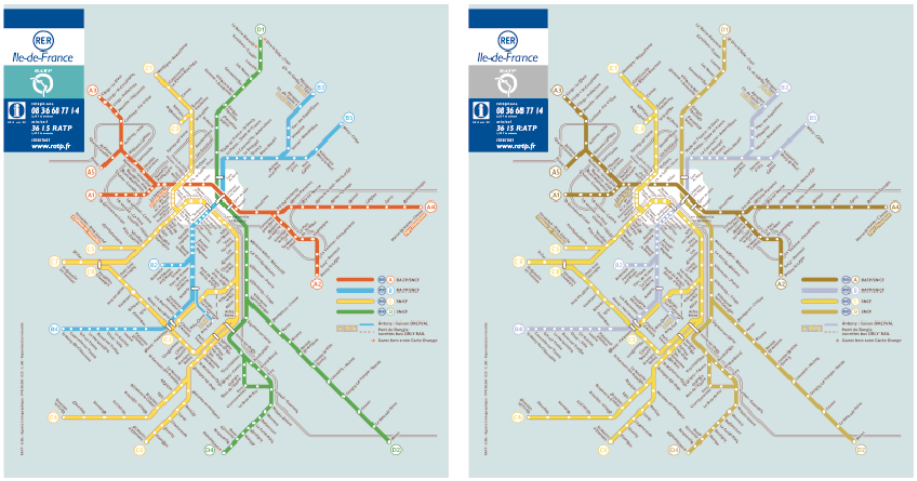
Exercice : un enseignant donne un cours dans un amphi de 180 étudiants avec des transparents en couleur. Quelle est la probabilité pour qu'au moins un étudiant ait des problèmes pour lire les transparents en couleur.

```
[1] 0.9999022
```

Exercice : représentez graphiquement la probabilité d'avoir au moins une personne ayant des problèmes pour voir les couleurs en fonction de la taille de l'auditoire.

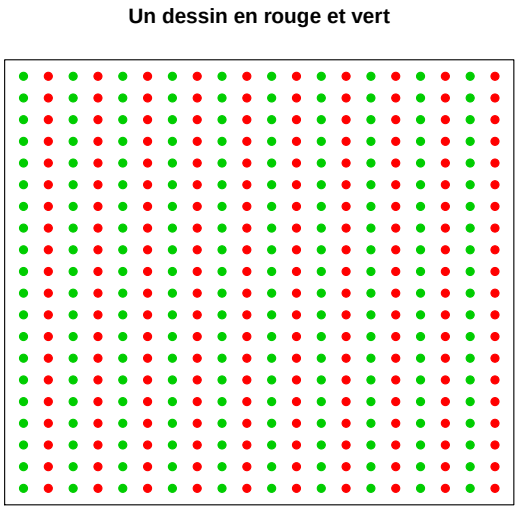


La bibliothèque `dichromat` vous permet de tester si votre choix de couleurs est acceptable. Elle est basée sur les travaux de Françoise Viénot et collaborateurs [2]. Ces travaux cherchent à aider les concepteurs de documents à large diffusion en leur permettant de vérifier qu'ils sont toujours intelligibles pour les protanopes et deutéranopes. L'idée est simplement de simuler ce que voient ces personnes. L'exemple ci-dessous est tiré de [2], il montre que la carte des transports en commun dans la région Île-de-France est toujours intelligible pour les protanopes (à droite) :



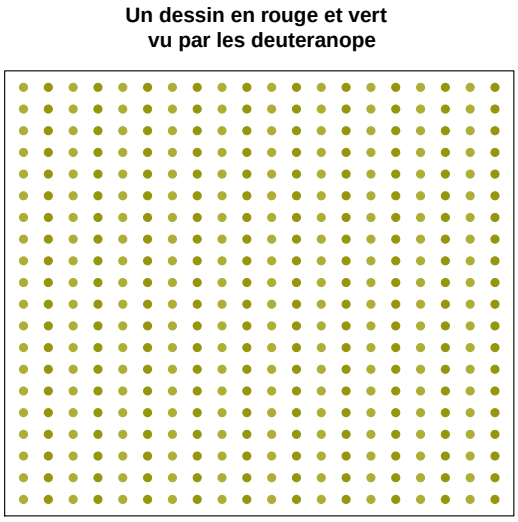
La bibliothèque `dichromat` vous permet très simplement d'atteindre le même niveau de professionnalisme. Commencez par faire un graphique en couleur, par exemple :

```
n <- 20
x <- rep(1:n, n)
y <- rep(1:n, each = n)
couleur <- c("green3", "red")
plot(x, y, col = couleur, pch = 19, main = "Un dessin en rouge et vert",
    xaxt = "n", yaxt = "n", xlab = "", ylab = "")
```

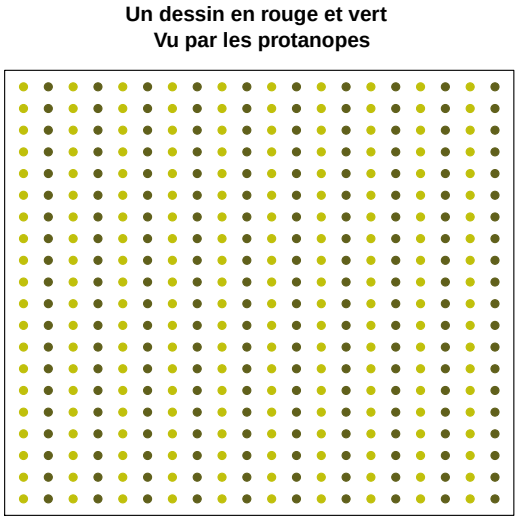


Les formes les plus communes de déficience dans la perception des couleurs sont les dyschromatopsies héréditaires de type protan et deutan correspondant a une difficulté à distinguer le rouge du vert :

```
library(dichromat)
plot(x, y, col = dichromat(couleur, "deutan"), pch = 19, main = "Un dessin en rouge et vert \n vu par les deut
    xaxt = "n", yaxt = "n", xlab = "", ylab = "")
```




```
plot(x, y, col = dichromat(couleur, "protan"), pch = 19, main = "Un dessin en rouge et vert \n Vu par les prota
    xaxt = "n", yaxt = "n", xlab = "", ylab = "")
```



L'utilisation des couleurs est donc extrêmement délicate car elle risque de fausser la perception du message pour une proportion non négligeable des individus. Pour bien s'en convaincre il suffit de considérer la version colorisée de la célèbre lettre dite de George Sand à Alfred de Musset[1] :

[1] George Sand n'a jamais écrit de textes érotiques, ceux qui sont publiés sur internet sont des faux fabriqués au XIX^{e} siècle (`http://www.george-sand.info/`).

original

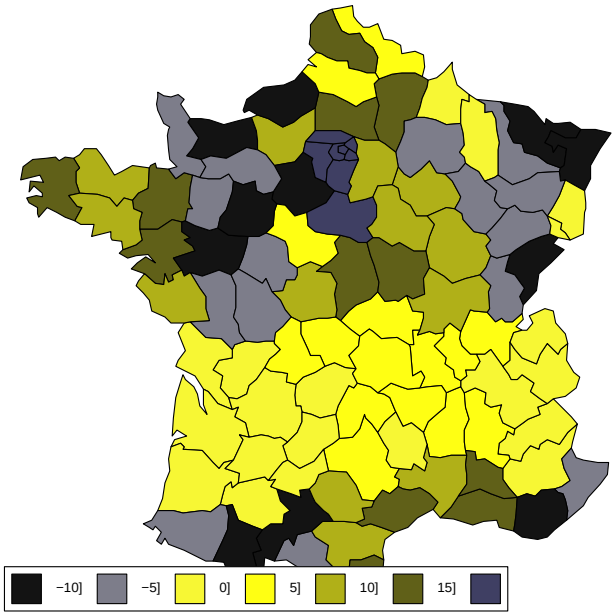
Je suis émue de vous dire que j'ai
bien compris l'autre soir que vous aviez
toujours une envie folle de me faire
danser. Je garde le souvenir de votre
baiser et je voudrais bien que ce soit
là une preuve que je puisse être aimée
par vous. Je suis prête à vous montrer mon
affection toute désintéressée et sans cal−
cul, et si vous voulez me voir aussi
vous dévoiler sans artifice mon âme
toute nue, daignez me faire une visite.
Nous causerons en amis, franchement.
Je vous prouverai que je suis la femme
sincère et capable de vous offrir l'affection
la plus profonde comme la plus étroite
en amitié, en un mot, la meilleure épouse
que vous puissiez rêver puisque votre
âme est libre. Pensez que la solitude où j'ha−
bite est bien longue, bien dure et souvent
difficile. Ainsi en y songeant j'ai l'âme
grosse. Accourrez donc vite et venez me la
faire oublier. A l'amour que je veux vous sou−
mettre.

vu par deuteranope

Je suis émue de vous dire que j'ai
bien compris l'autre soir que vous aviez
toujours une envie folle de me faire
danser. Je garde le souvenir de votre
baiser et je voudrais bien que ce soit
là une preuve que je puisse être aimée
par vous. Je suis prête à vous montrer mon
affection toute désintéressée et sans cal−
cul, et si vous voulez me voir aussi
vous dévoiler sans artifice mon âme
toute nue, daignez me faire une visite.
Nous causerons en amis, franchement.
Je vous prouverai que je suis la femme
sincère et capable de vous offrir l'affection
la plus profonde comme la plus étroite
en amitié, en un mot, la meilleure épouse
que vous puissiez rêver puisque votre
âme est libre. Pensez que la solitude où j'ha−
bite est bien longue, bien dure et souvent
difficile. Ainsi en y songeant j'ai l'âme
grosse. Accourrez donc vite et venez me la
faire oublier. A l'amour que je veux vous sou−
mettre.

C'est un problème très sérieux, si nous reprenons l'exemple de la carte de Bertin :

```
grey.bck <- grey
gray <- grey <- function(level) {
    pal <- rev(dichromat(c("black", "turquoise4", "greenyellow",
        "yellow", "orange", "orangered3", "maroon4"), type = "protan"))
    pal[round(1 + 6 * level)]
}
area.plot(elec88$area, val = c(bertin[, 2], rep(18, 5)), clab = 0,
    sub = "", csub = 3)
grey <- gray <- grey.bck
```



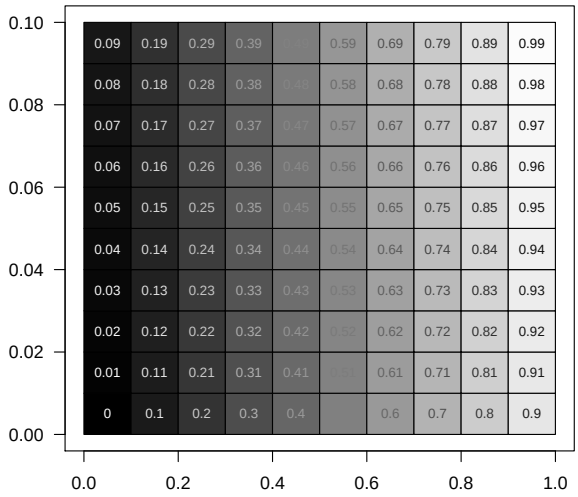


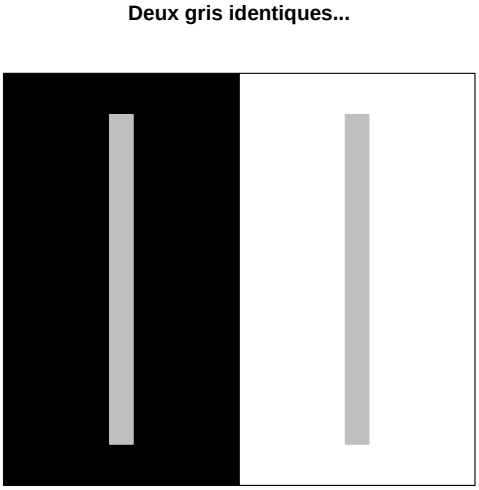
FIG. 1 – Nuancier des gris avec `grey()` ou `gray()`.

Nous voyons qu'un protanope verra très clairement une structure complètement orthogonale à la véritable structure des données. Êtes-vous bien certain de toujours vouloir utiliser des couleurs ?

3 Niveaux de gris

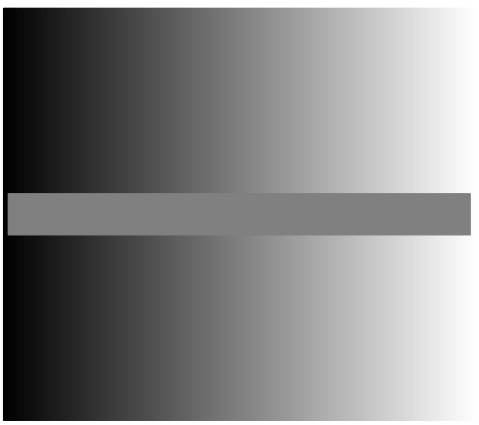
La fonction `grey()`, que l'on peut également orthographier à l'américaine `gray()`, permet de générer très facilement toutes les nuances de gris entre le noir (codé 0) et le blanc (codé 1). Un nuancier est donné dans la figure 1. Les problèmes de vision des couleurs ne se posent plus avec des niveaux de gris, mais notre perception des gris n'est pas très bonne dans l'absolu parce que trop dépendante du contexte. Le code ci-dessous montre bien que les deux rectangles gris sont exactement du même gris, pourtant celui qui est sur un fond clair semble être plus sombre :

```
plot(0, 0, type = "n", xlim = c(0, 2), ylim = 0:1, axes = FALSE,
    xlab = "", ylab = "", main = "Deux gris identiques...")
rect(0, 0, 1, 1, col = grey(0))
rect(1, 0, 2, 1, col = grey(1))
rect(0.45, 0.1, 0.55, 0.9, col = grey(0.75), border = grey(0.75))
rect(1.45, 0.1, 1.55, 0.9, col = grey(0.75), border = grey(0.75))
```



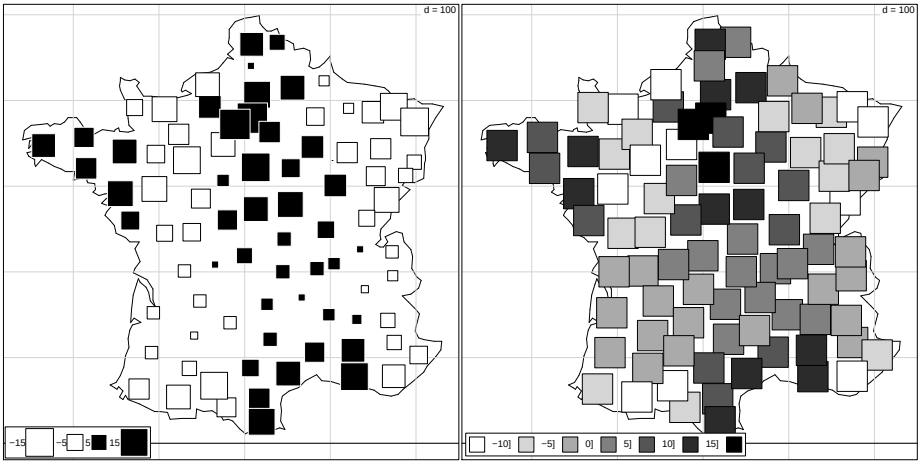
De même, si nous n'avions pas sous les yeux le code R ayant généré la figure suivante, il nous serait difficile de croire que le rectangle central est du même gris :

```
plot(0, 0, type = "n", xlim = 0:1, ylim = 0:1, axes = FALSE, xlab = "",
    ylab = "", main = "Un seul gris...")
n <- 200
for (i in seq(0, 1, length = n)) rect(i, 0, i + 1/n, 1, col = grey(i),
    border = grey(i))
rect(0.01, 0.45, 0.99, 0.55, col = grey(0.5), border = grey(0.5))
```



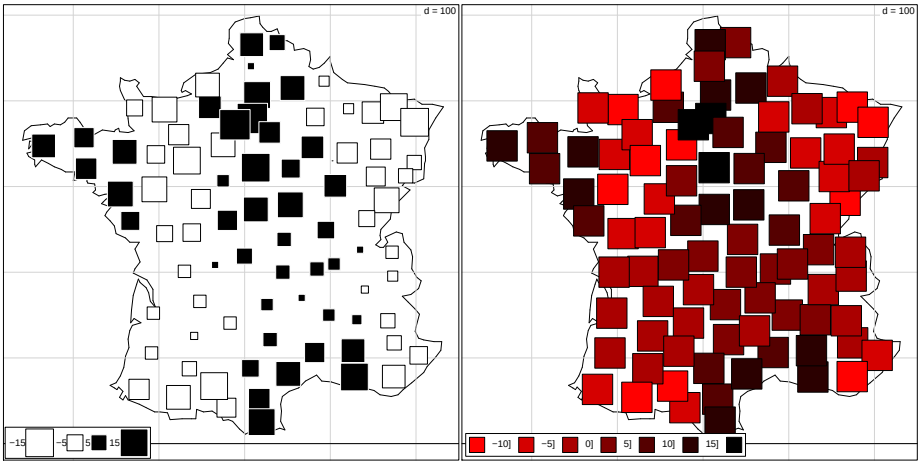
Reprendre l'exemple de Bertin pour comparer avec une représentation en niveaux de gris :

```
opar <- par(no.readonly = TRUE)
par(mfrow = c(1, 2))
s.value(elec88$xy[1:89, ], bertin[1:89, 2], contour = elec88$contour,
    meth = "squaresize", incl = F)
s.value(elec88$xy[1:89, ], bertin[1:89, 2], contour = elec88$contour,
    meth = "greylevel", incl = F)
par(opar)
```



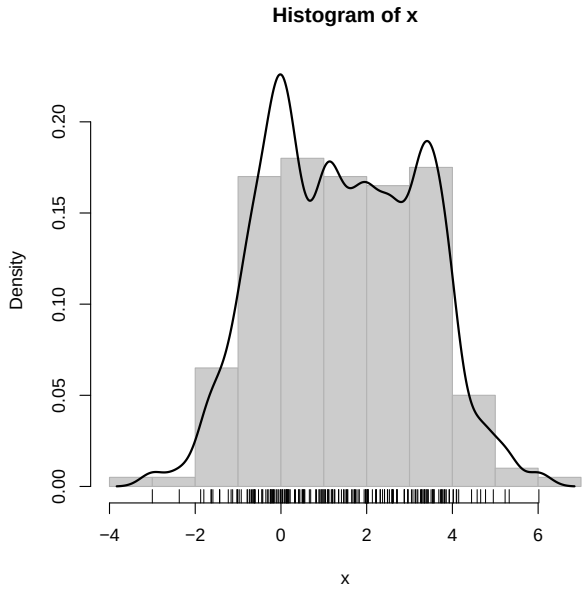
On peut utiliser des couleurs de valeur variable, mais quel est l'intérêt par rapport à un simple niveau de gris ?

```
opar <- par(no.readonly = TRUE)
par(mfrow = c(1, 2))
grey.bck <- grey
gray <- grey <- function(level) hsv(1, 1, level)
s.value(elec88$xy[1:89, ], bertin[1:89, 2], contour = elec88$contour,
    meth = "squaresize", incl = F)
s.value(elec88$xy[1:89, ], bertin[1:89, 2], contour = elec88$contour,
    meth = "greylevel", incl = F)
grey <- gray <- grey.bck
par(opar)
```



Exercice : définir une fonction pour faire le dessin ci-après. Vous aurez besoin des fonctions `grey()`, `hist()`, `density()`, `lines()`, `rug()`, `jitter()` et d'utiliser l'argument point-point-point.

```
x <- c(rnorm(100), rnorm(100, 3))
mon.hist(x, adjust = 0.5)
```



4 Les couleurs

Il y a 657 couleurs prédéfinies dans R. Leur liste est donnée par la fonction `colors()` :

```
colors()[1:10]
```

```
[1] "white"         "aliceblue"     "antiquewhite"  "antiquewhite1" "antiquewhite2"
[6] "antiquewhite3" "antiquewhite4" "aquamarine"    "aquamarine1"   "aquamarine2"
```

Pour choisir une couleur rapidement, les représenter dans une grille (figure 2) :

```
couleurs <- colors()
couleurs <- couleurs[-(grep("gray", couleurs))]
couleurs <- couleurs[-(grep("grey", couleurs))]
length(couleurs)
```

```
[1] 433
```

```
n <- ceiling(sqrt(length(couleurs)))
w <- matrix(0, n, n)
opar <- par(no.readonly = TRUE)
par(mar = c(0.1, 0.1, 0.1, 0.1))
plot(c(1, n + 1), c(1, n + 1), type = "n")
rect(col(w), row(w), col(w) + 1, row(w) + 1, col = couleurs)
editcolor <- function() {
    w <- as.numeric(locator(1))
    w <- floor(w)
    num <- (w[1] - 1) * n + w[2]
    return(couleurs[num])
}
par(opar)
```

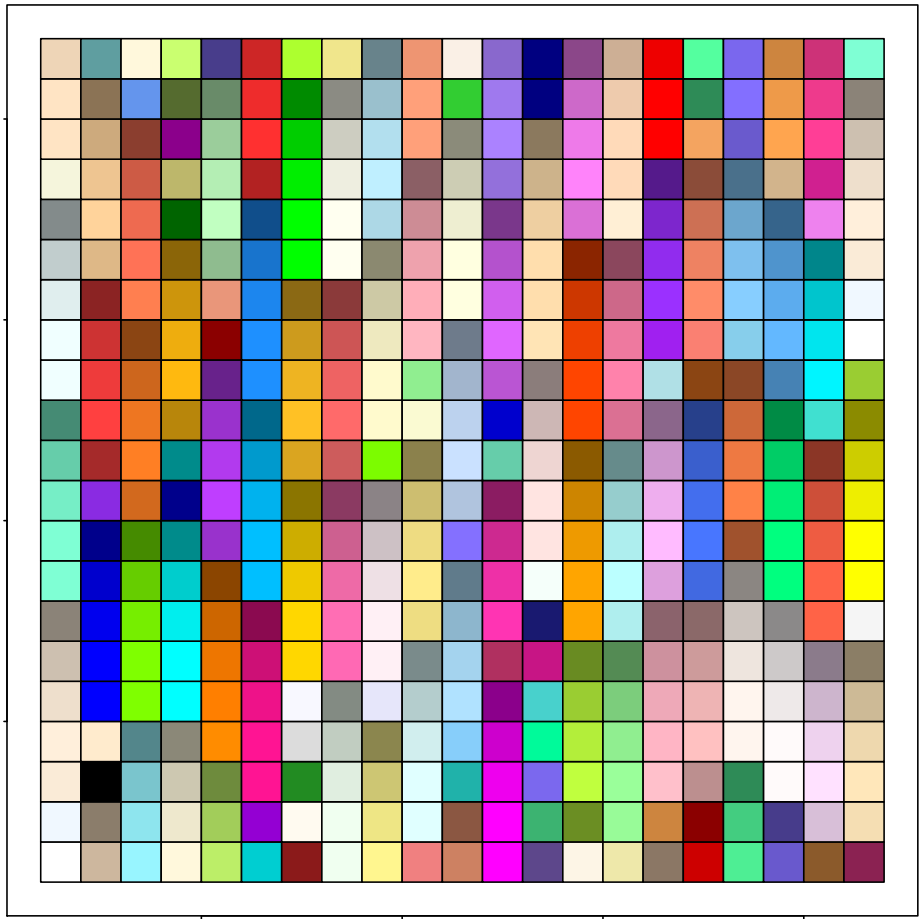
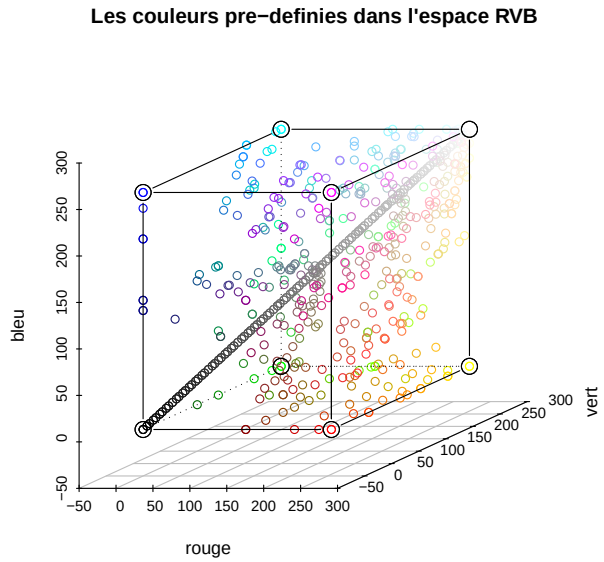


FIG. 2 – Grille des couleurs accessibles par leur nom. Utiliser `editcolor` pour avoir le nom en cliquant dans un carré

.

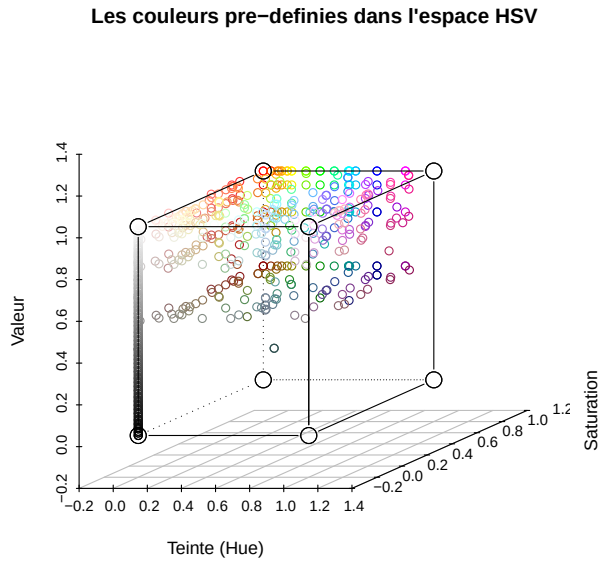
La fonction `col2rgb()` permet de récupérer le code RGB (Red Green Bleu) des couleurs :

```
library(scatterplot3d)
cubedraw <- function(res3d, min = 0, max = 255, cex = 2, text. = FALSE) {
    cube01 <- rbind(c(0, 0, 1), 0, c(1, 0, 0), c(1, 1, 0), 1, c(0,
        1, 1), c(1, 0, 1), c(0, 1, 0))
    cub <- min + (max - min) * cube01
    res3d$points3d(cub[c(1:6, 1, 7, 3, 7, 5), ], cex = cex, type = "b",
        lty = 1)
    res3d$points3d(cub[c(2, 8, 4, 8, 6), ], cex = cex, type = "b",
        lty = 3)
    if (text.)
        text(res3d$xyz.convert(cub), labels = 1:nrow(cub), col = "tomato",
            cex = 2)
}
cc <- colors()
crgb <- t(col2rgb(cc))
par(xpd = TRUE)
rr <- scatterplot3d(crgb, color = cc, box = FALSE, angle = 24, main = "Les couleurs pre-definies dans l'espace
    xlab = "rouge", ylab = "vert", zlab = "bleu", xlim = c(-50,
        300), ylim = c(-50, 300), zlim = c(-50, 300))
cubedraw(rr)
```



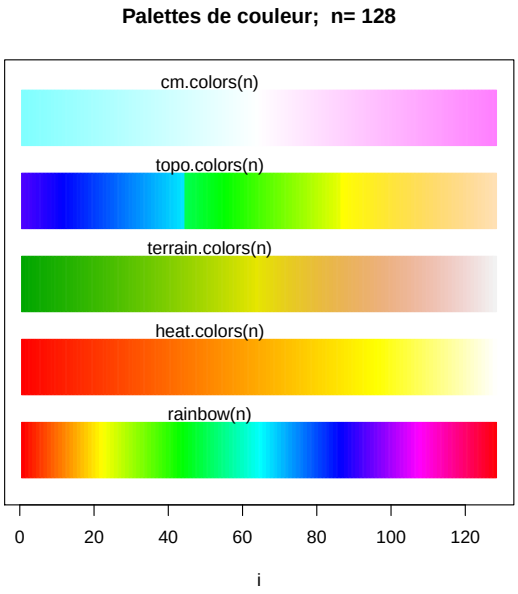
La fonction `rgb2hsv()` permet de passer à la représentation HSV (teinte, saturation, valeur) des couleurs :

```
chsv <- t(rgb2hsv(col2rgb(cc)))
rr <- scatterplot3d(chsv, color = cc, box = FALSE, angle = 24, main = "Les couleurs pre-definies dans l'espace
    xlab = "Teinte (Hue)", ylab = "Saturation", zlab = "Valeur",
    xlim = c(-0.1, 1.1), ylim = c(-0.1, 1.1), zlim = c(-0.1, 1.1))
cubedraw(rr, min = 0, max = 1)
```



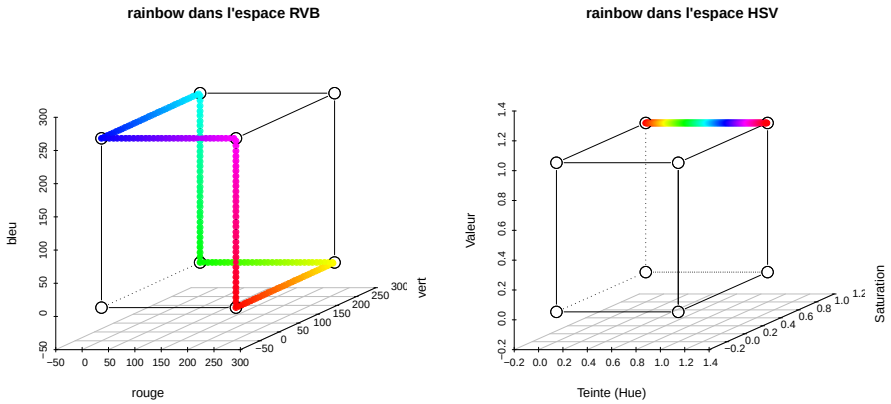
Il existe plusieurs fonctions utilitaires permettant de choisir n couleurs dans une palette pré-définie :

```
demo.pal <- function(n, border = if (n < 32) "light gray" else NA,
    main = paste("Palettes de couleur;  n=", n), ch.col = c("rainbow(n)",
        "heat.colors(n)", "terrain.colors(n)", "topo.colors(n)",
        "cm.colors(n)")) {
    nt <- length(ch.col)
    i <- 1:n
    j <- n/nt
    d <- j/6
    dy <- 2 * d
    plot(i, i + d, type = "n", yaxt = "n", ylab = "", main = main)
    for (k in 1:nt) {
        rect(i - 0.5, (k - 1) * j + dy, i + 0.4, k * j, col = eval(parse(text = ch.col[k])),
            border = eval(parse(text = ch.col[k])))
        text(2 * j, k * j + dy/4, ch.col[k])
    }
}
demo.pal(128)
```

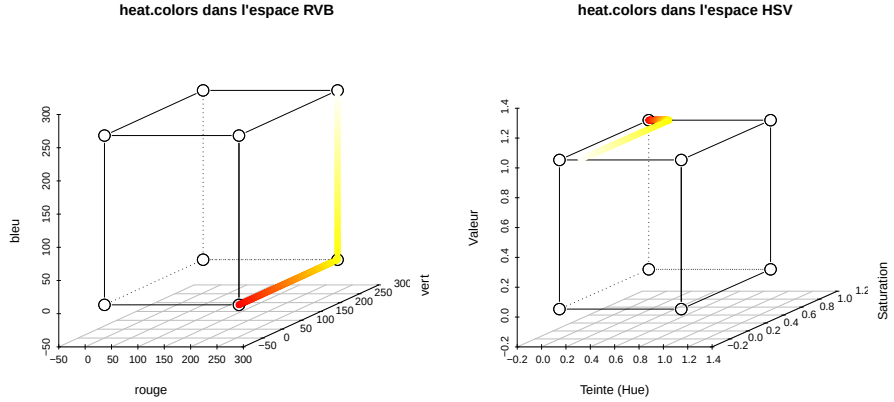


Exercice : écrire une fonction pour représenter la trajectoire des palettes prédéfinies dans les espaces RGB et HSV.

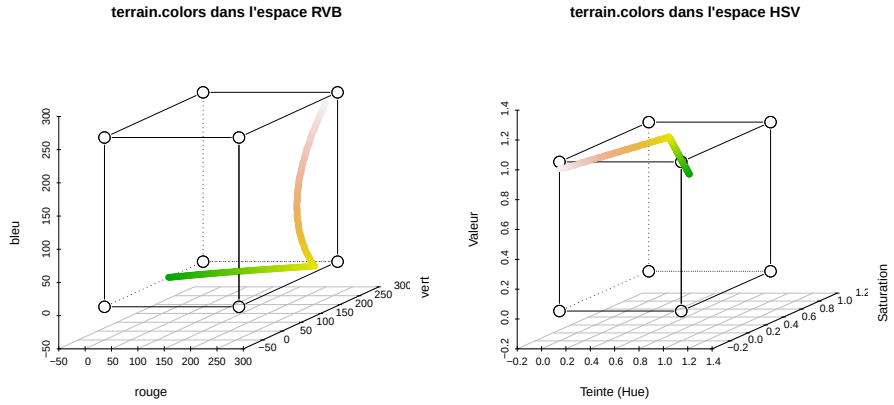
```
show.palette(rainbow)
```



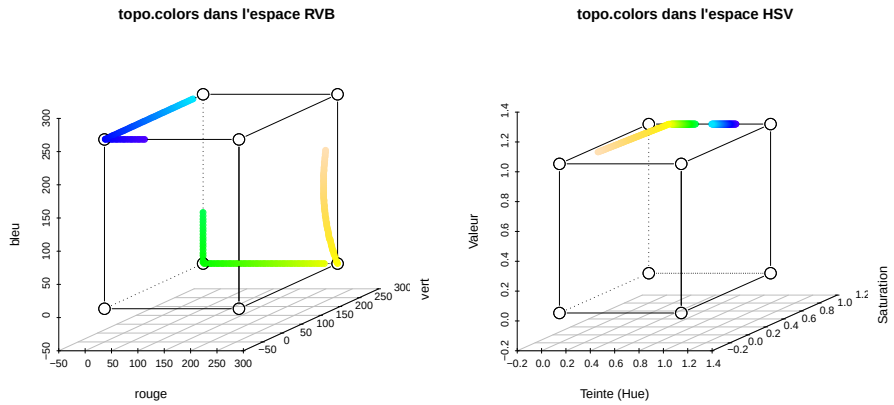

```
show.palette(heat.colors)
```



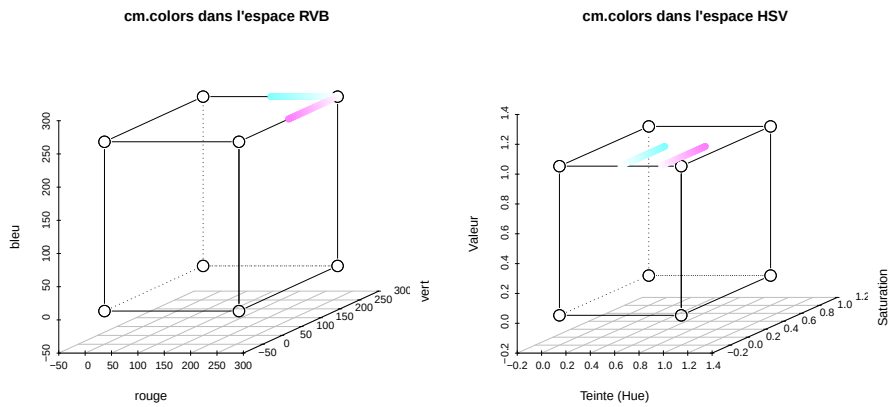

```
show.palette(terrain.colors)
```




```
show.palette(topo.colors)
```



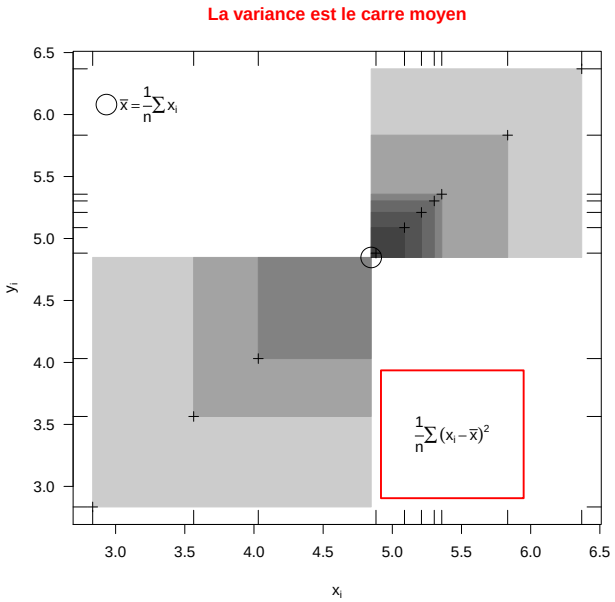
```
show.palette(cm.colors)
```



5 Couleurs transparentes

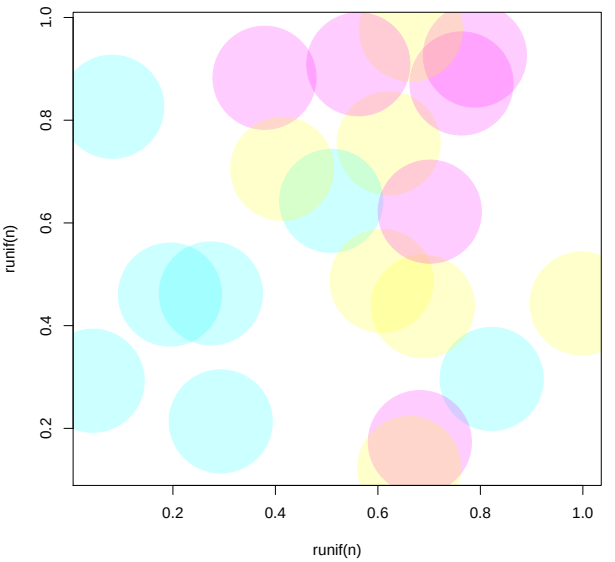
Les couleurs transparentes ne sont pas toujours disponibles (seuls `quartz()` et `pdf()`).

```
pdf("figs/essai.pdf", version = "1.4")
set.seed(1071966)
n <- 10
x <- rnorm(n = n, mean = 5)
plot(x = x, y = x, main = "La variance est le carre moyen", col.main = "red",
    las = 1, pch = 3, xlab = expression(x[i]), ylab = expression(y[i]))
points(mean(x), mean(x), cex = 3)
sapply(1:4, function(side) rug(x = x, side = side))
rect(xleft = x, ybottom = x, xright = mean(x), ytop = mean(x), col = rgb(0,
    0, 0, 2/n), border = rgb(0, 0, 0, 2/n))
var.n <- function(x) sum((x - mean(x))^2)/length(x)
offset <- diff(range(x))/50
xleft <- offset + mean(x)
xright <- xleft + var.n(x)
ybottom <- offset + min(x)
ytop <- ybottom + var.n(x)
rect(xleft, ybottom, xright, ytop, lwd = 2, border = "red")
text((xleft + xright)/2, (ytop + ybottom)/2, expression(frac(1,
    n) * sum((x[i] - bar(x))^2)))
legend(x = min(x), y = max(x), expression(bar(x) == frac(1, n) *
    sum(x[i])), pt.cex = 3, pch = 1, bty = "n")
dev.off()
```

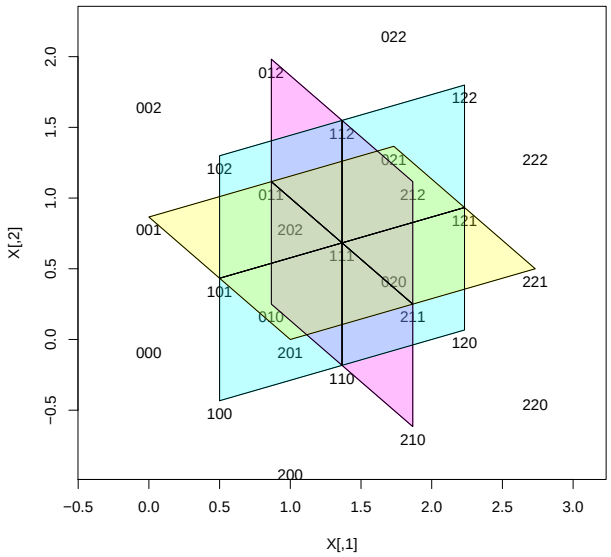


Les couleurs transparentes peuvent être intéressantes pour visualiser les superpositions de points :

```
pdf("figs/ronds.pdf", version = "1.4")
n <- 20
plot(runif(n), runif(n), pch = 19, col = c(rgb(1, 1, 0, 0.2), rgb(0,
    1, 1, 0.2), rgb(1, 0, 1, 0.2)), cex = 15)
dev.off()
```



Les couleurs transparentes peuvent être intéressantes dans certains cas très particuliers, comme la représentation de plans dans des vues en perspective :



6 Couleurs définies *a priori*

Pour un domaine d'application bien défini, il peut exister un code des couleurs standard, par exemple en géologie (*cf* figure 3), ou pour les cartes topographiques :

```
data(volcano)
x <- 10 * (1:nrow(volcano))
y <- 10 * (1:ncol(volcano))
image(x, y, volcano, col = terrain.colors(100), axes = FALSE)
contour(x, y, volcano, levels = seq(90, 200, by = 5), add = TRUE,
    col = "peru")
axis(1, at = seq(100, 800, by = 100))
axis(2, at = seq(100, 600, by = 100))
box()
title(main = "Maunga Whau Volcano", font.main = 4)
```

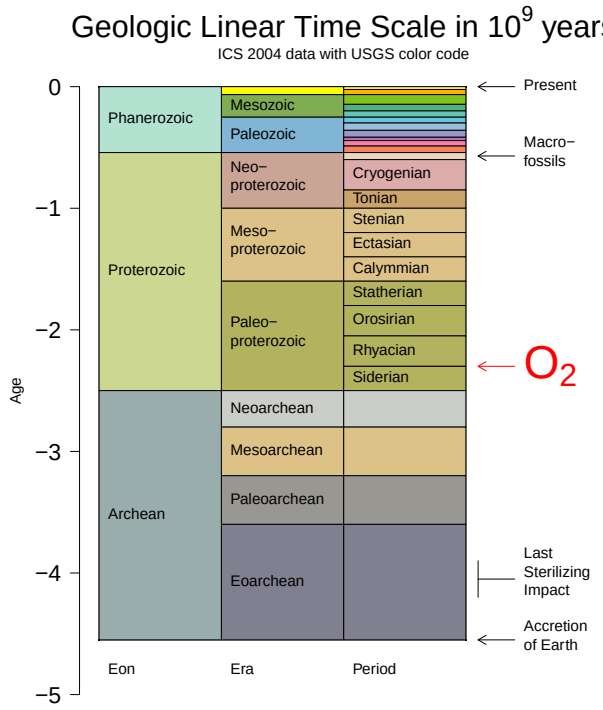
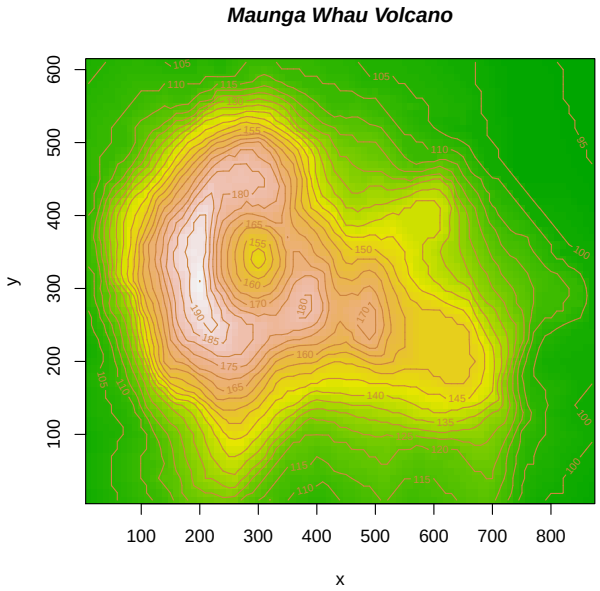


FIG. 3 – Les couleurs des cartes géologiques sont définies dans des standards internationaux (il existe en fait deux standards en compétition)



Références

[1] J. Bertin. *La graphique et le traitement graphique de l'information.* Flammarion, Paris, 1973.

[2] F. Viénot, H. Brettel, and J.D. Mollon. Digital video colourmaps for checking the legibility of displays by dichromats. *Color Research and Application*, 24 :243–252, 1999.