Visualisation personnalisée de données génomiques à l'aide du package `grid` dans R

Mardi 7 novembre 2017

Marc-André Lemay

Qu'est-ce que `grid`?

- Un système graphique inclus sous forme de package dans l'installation de base de R
- Un ensemble d'outils pour créer des représentations graphiques arbitrairement complexes
- La base sur laquelle sont construits `ggplot2` et `lattice`

`grid` : un système de bas niveau

- `grid` ne fournit pas de fonctions graphiques « toutes faites » comme c'est le cas de `graphics`, `ggplot2` ou `lattice`.

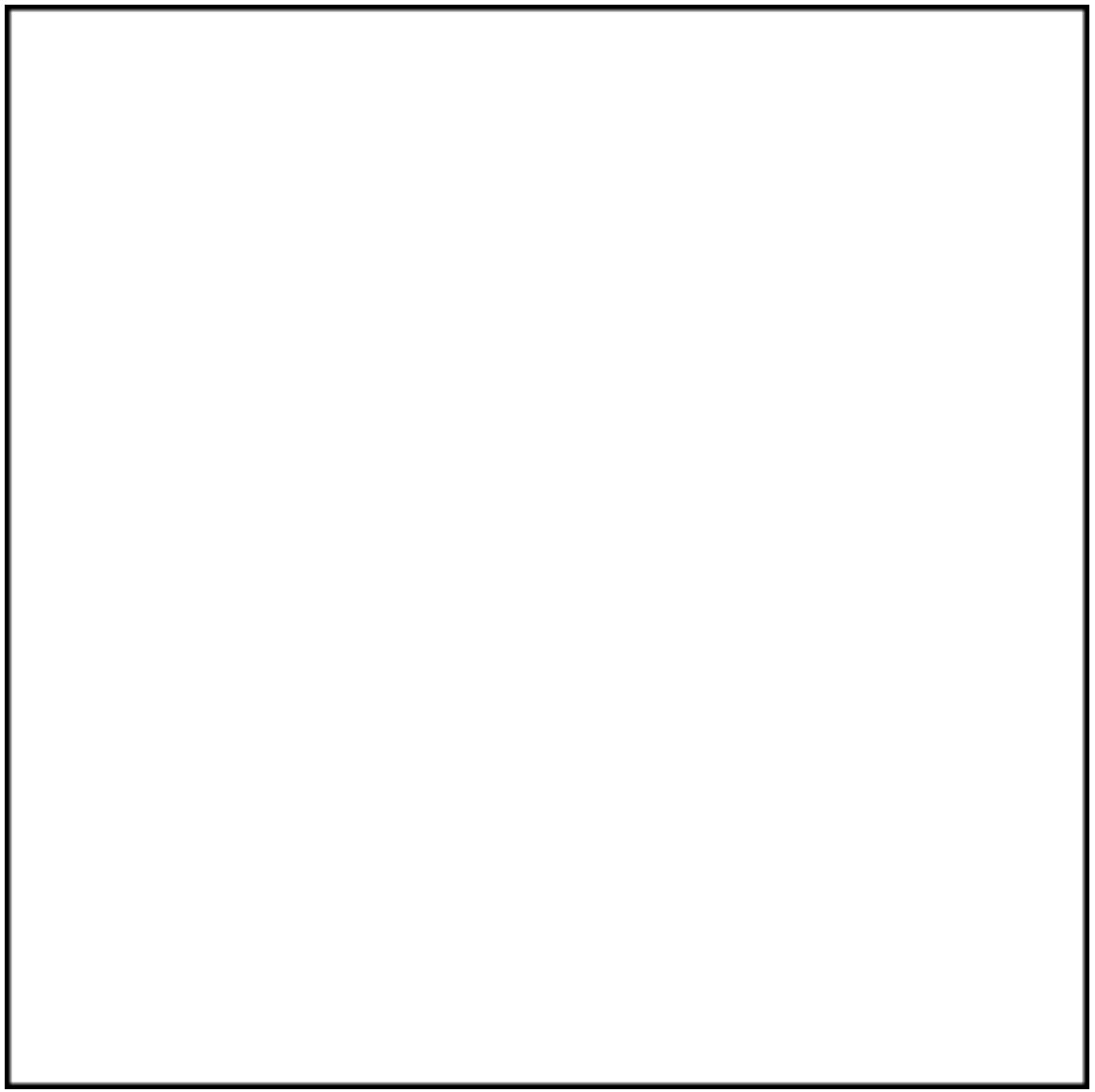
- L'utilisation de `grid` nécessite l'apprentissage de quelques nouveaux concepts.

- L'avantage : `grid` permet une très grande flexibilité à l'utilisateur.

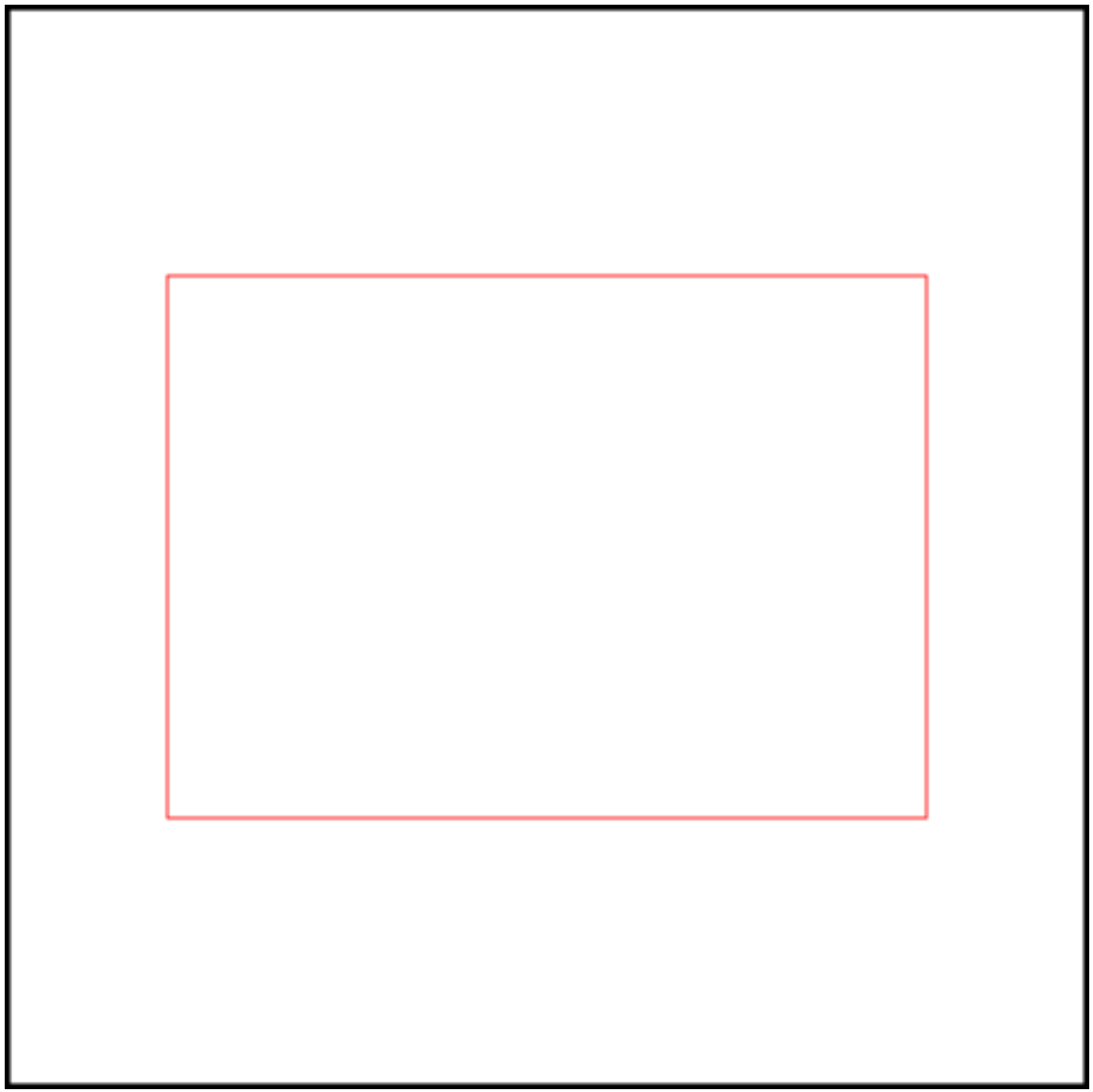
`grid` : quelques concepts

- `grid` implique la manipulation de fenêtres graphiques appelées **viewports**.
- En tout temps, `grid` sait dans quel **viewport** il se trouve.
- Le traçage (**drawing**) d'objets graphiques (**grobs**) prend place dans ces fenêtres.
- La disposition et l'apparence des **viewports** et des **grobs** sont définis par l'utilisateur.

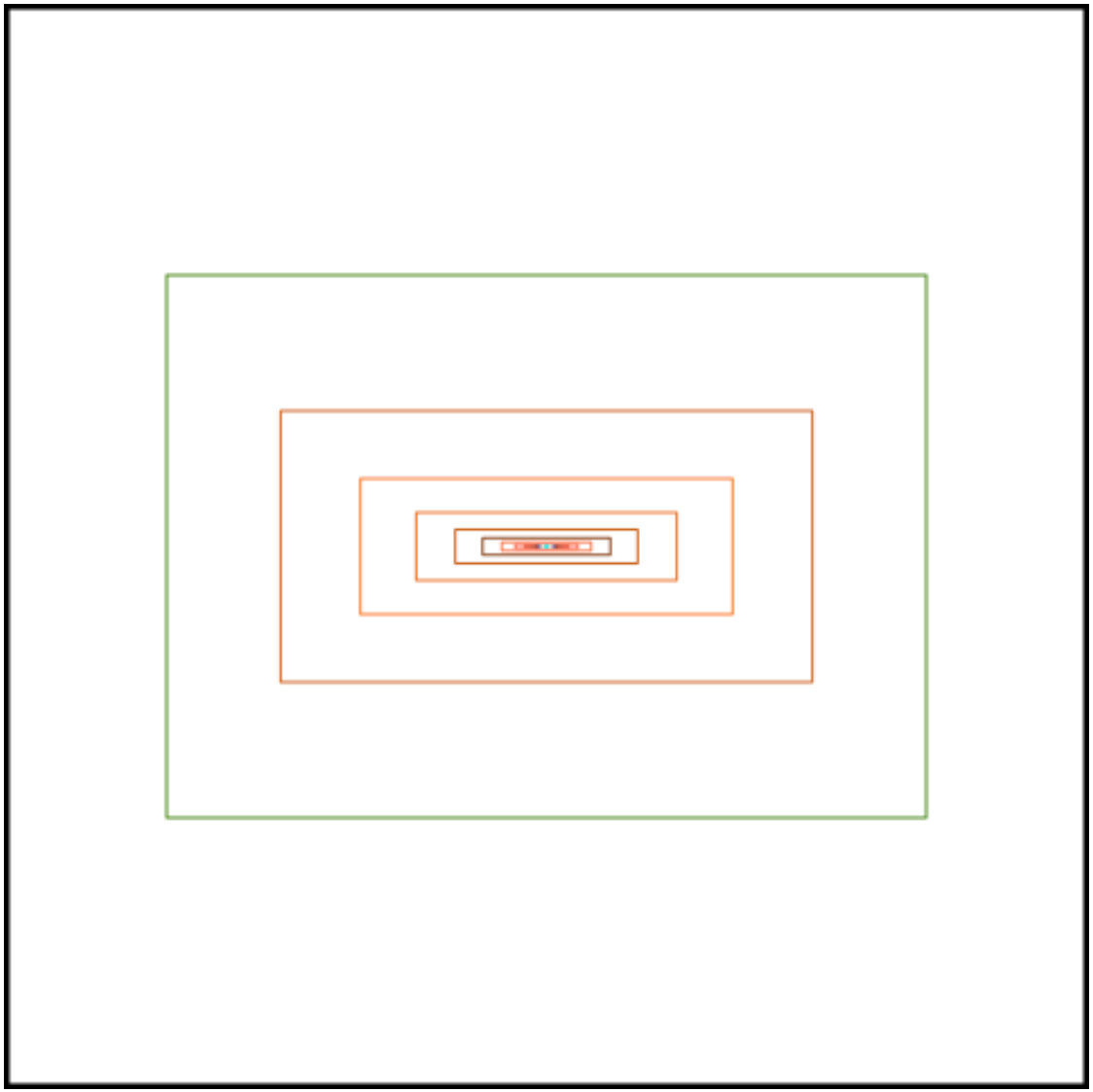
```
> grid.newpage()
> current.viewport()
## viewport[ROOT]
> grid.rect(gp = gpar(lwd = 5))
```



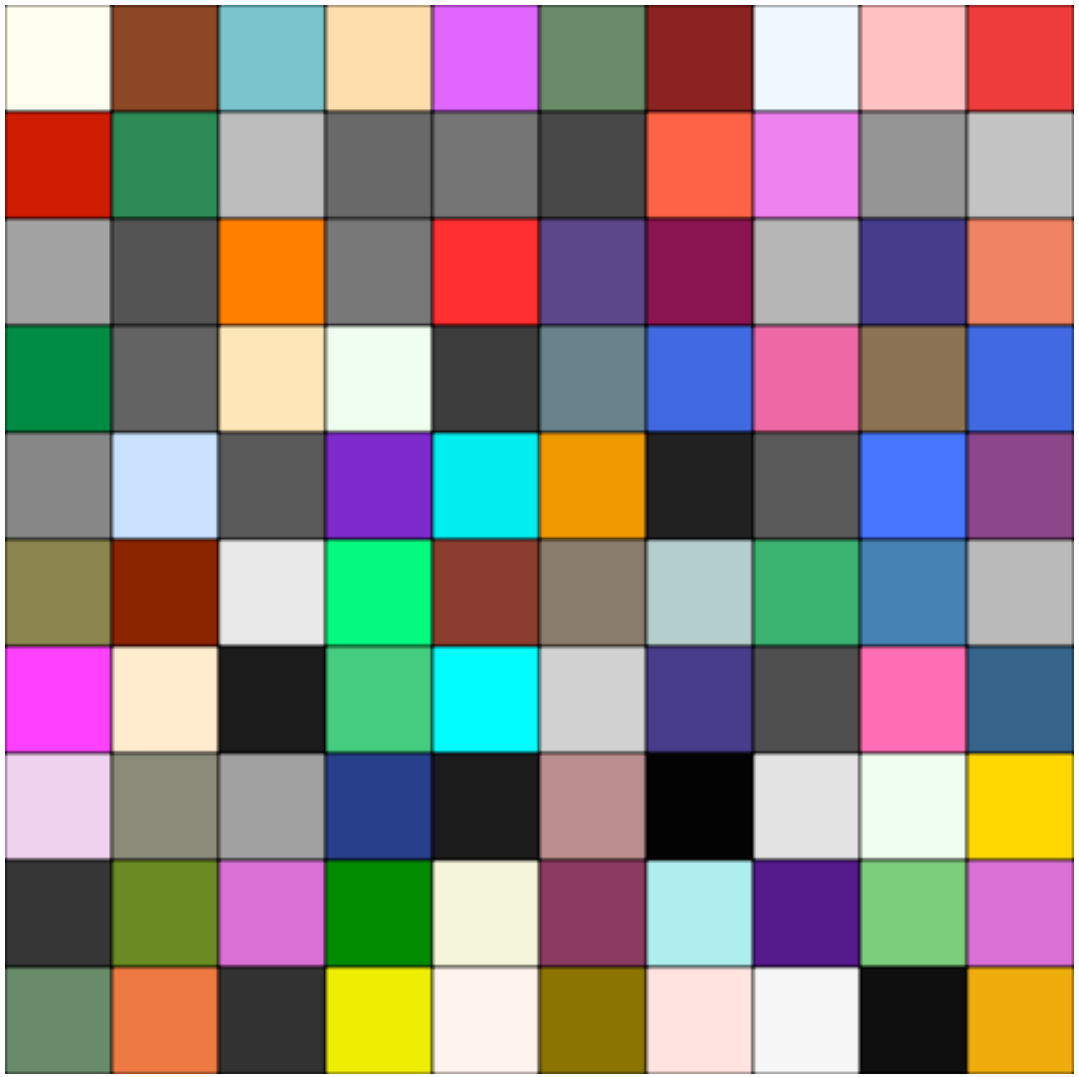
```
> vp <- viewport(height = 0.5, width = 0.7)
> pushViewport(vp)
> grid.rect(gp = gpar(col = "red"))
```



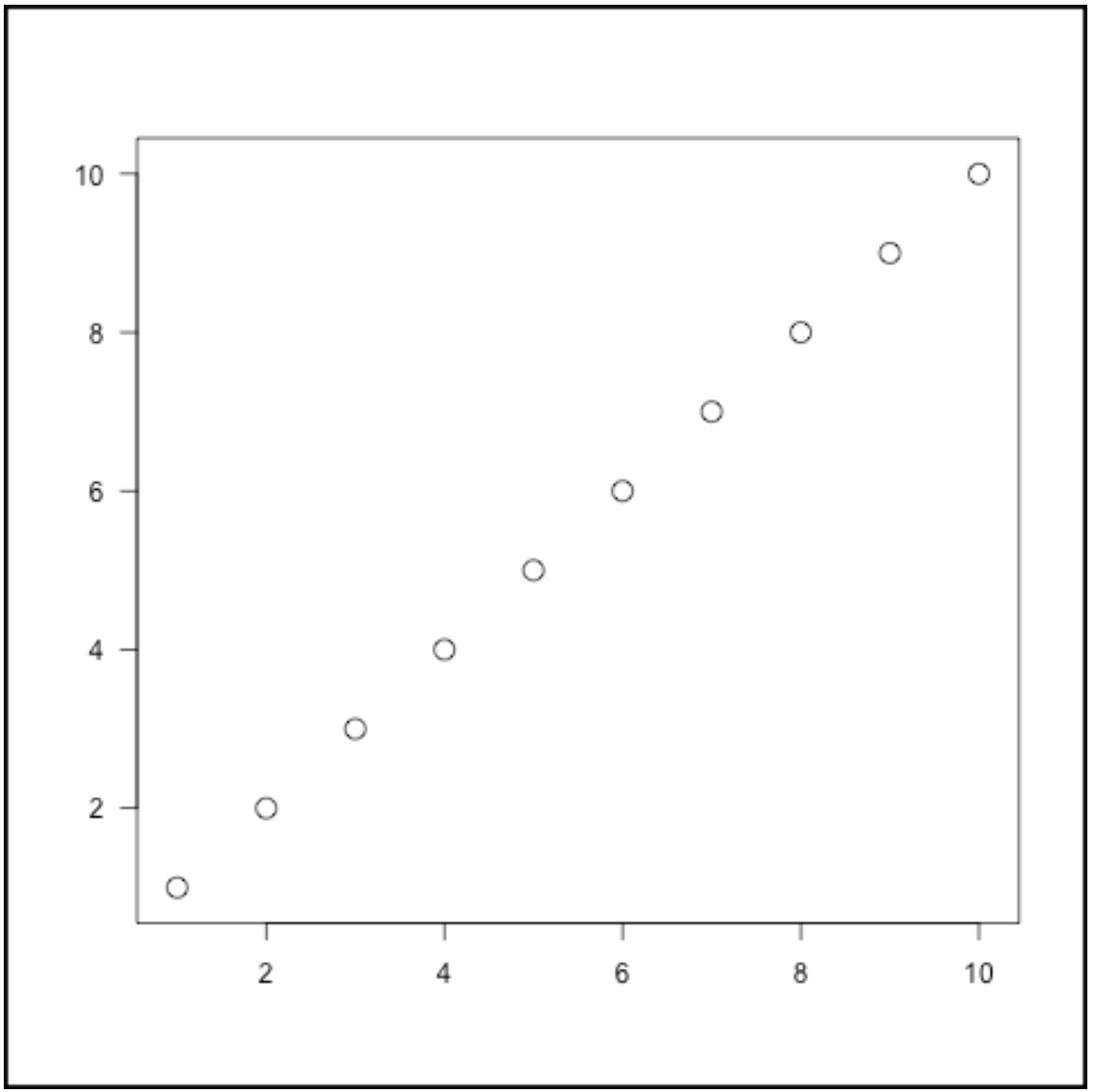
```
> vp <- viewport(height = 0.5, width = 0.7)
> for(i in 1:20) {
    pushViewport(vp)
    grid.rect(gp = gpar(col = colors()[i + 50]))
  }
```



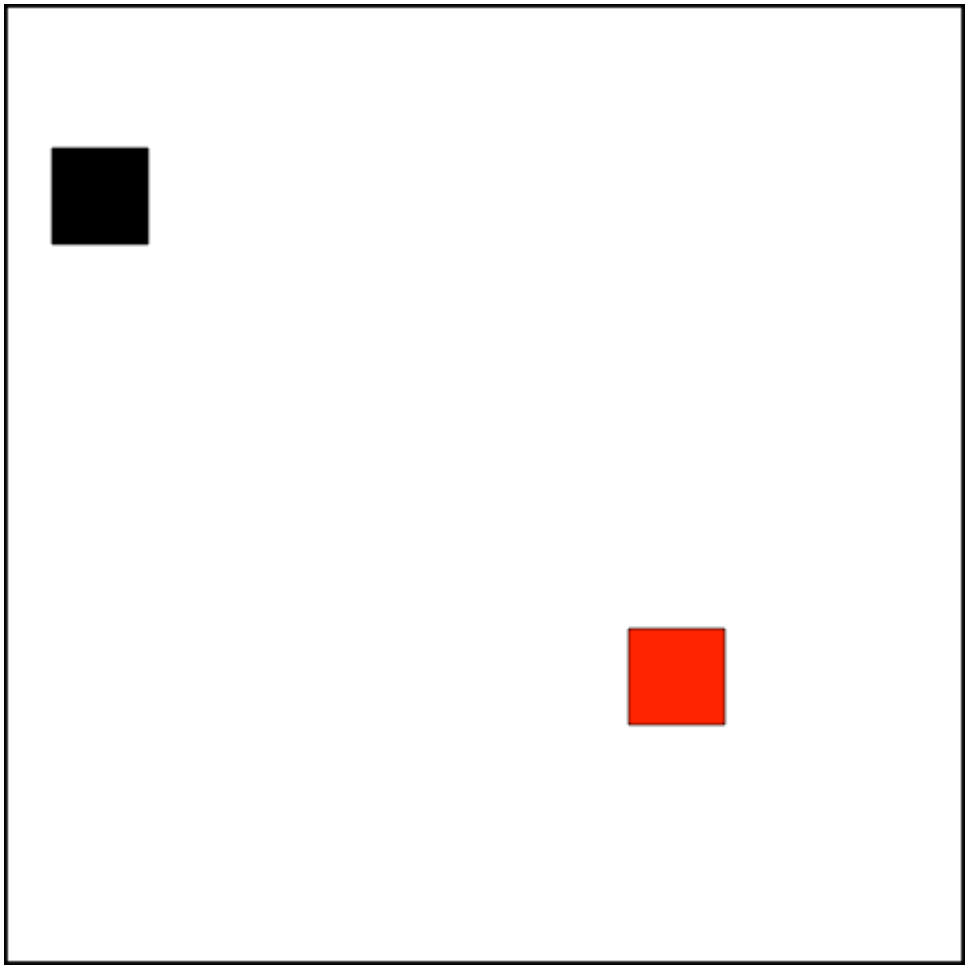
```
> pushViewport(viewport(layout = grid.layout(10, 10)))
> for(i in 1:10) {
    for(j in 1:10) {
    grid.rect(gp = gpar(fill = sample(colors(), 1)),
              vp = viewport(layout.pos.row = i,
                            layout.pos.col = j))
  }
}
```



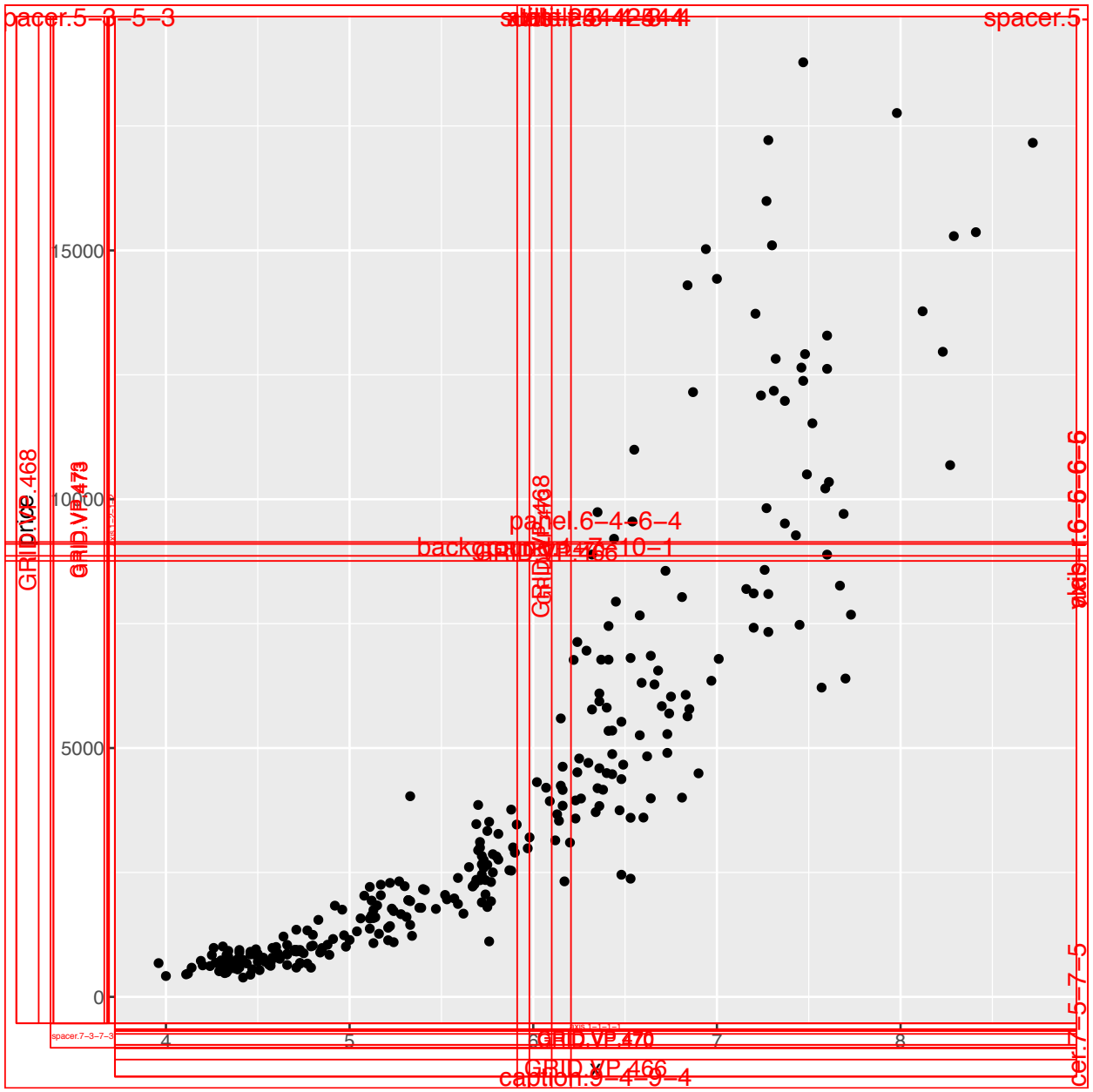
```
> x <- y <- 1:10
> pushViewport(plotViewport())
> pushViewport(dataViewport(x, y))
> grid.rect(); grid.xaxis(); grid.yaxis(); grid.points(x, y)
```



```
> pushViewport(viewport(x = 0.1, y = 0.8,
                        w = 0.1, h = 0.1, name = "a"))
> grid.rect(gp = gpar(fill = "black")) ; upViewport()
> pushViewport(viewport(x = 0.7, y = 0.3,
                        w = 0.1, h = 0.1, name = "b"))
> grid.rect(gp = gpar(fill = "red")) ; upViewport()
> current.vpTree()
## viewport[ROOT]->(viewport[a], viewport[b])
```



```
> ggplot(diamonds[sample(nrow(diamonds), 300), ],
         aes(x = x, y = price)) + geom_point()
> showViewport(fill = NULL, col = "red", newpage = FALSE)
```

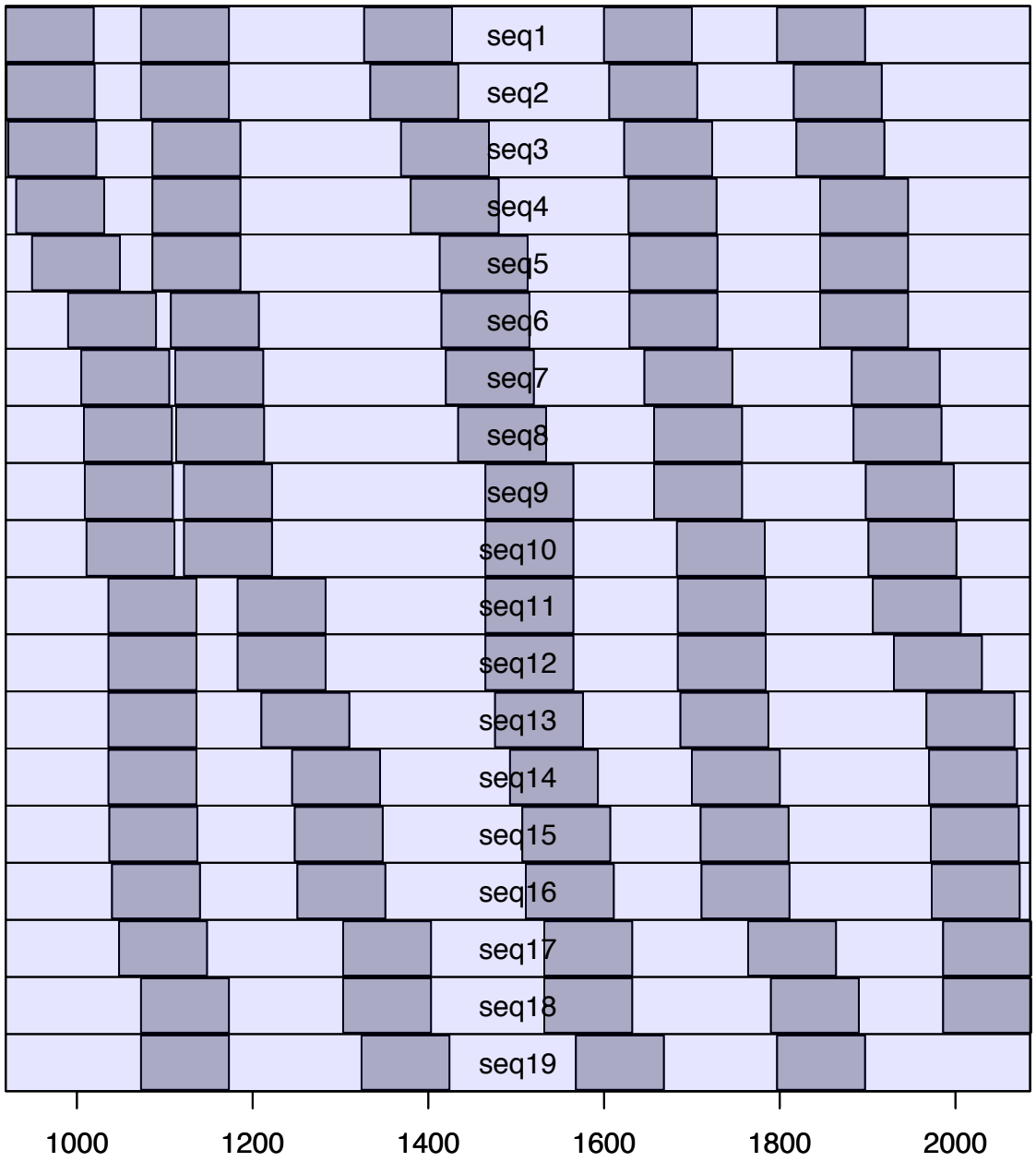


Pourquoi `grid` en génomique?

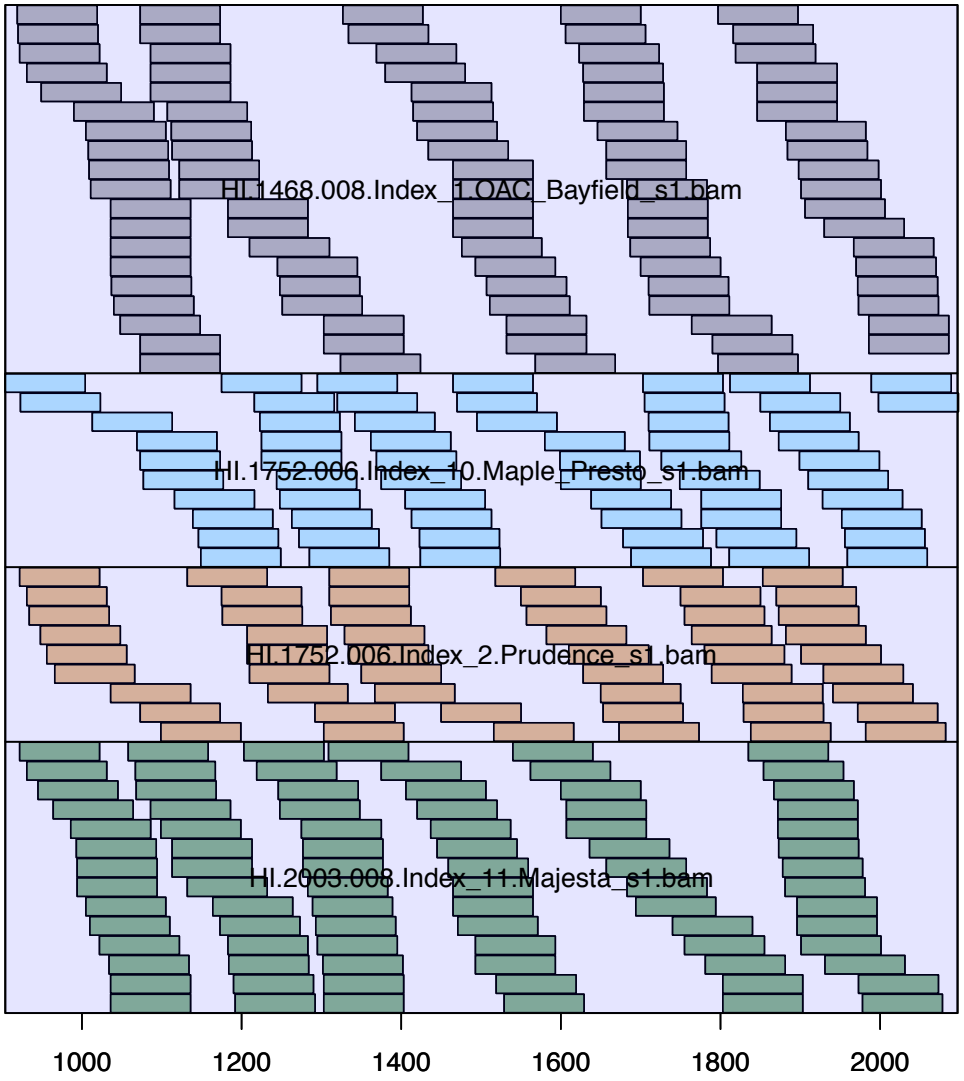
- Pour plus de liberté dans la représentation graphique des données
- Pour éviter le va-et-vient entre R et d'autres logiciels de visualisation
- Pour éviter les problèmes de mémoire qui surviennent parfois avec les *genome browsers*
- Pour générer des images de qualité pour la publication
- Pour créer des fonctions graphiques pour vos propres packages

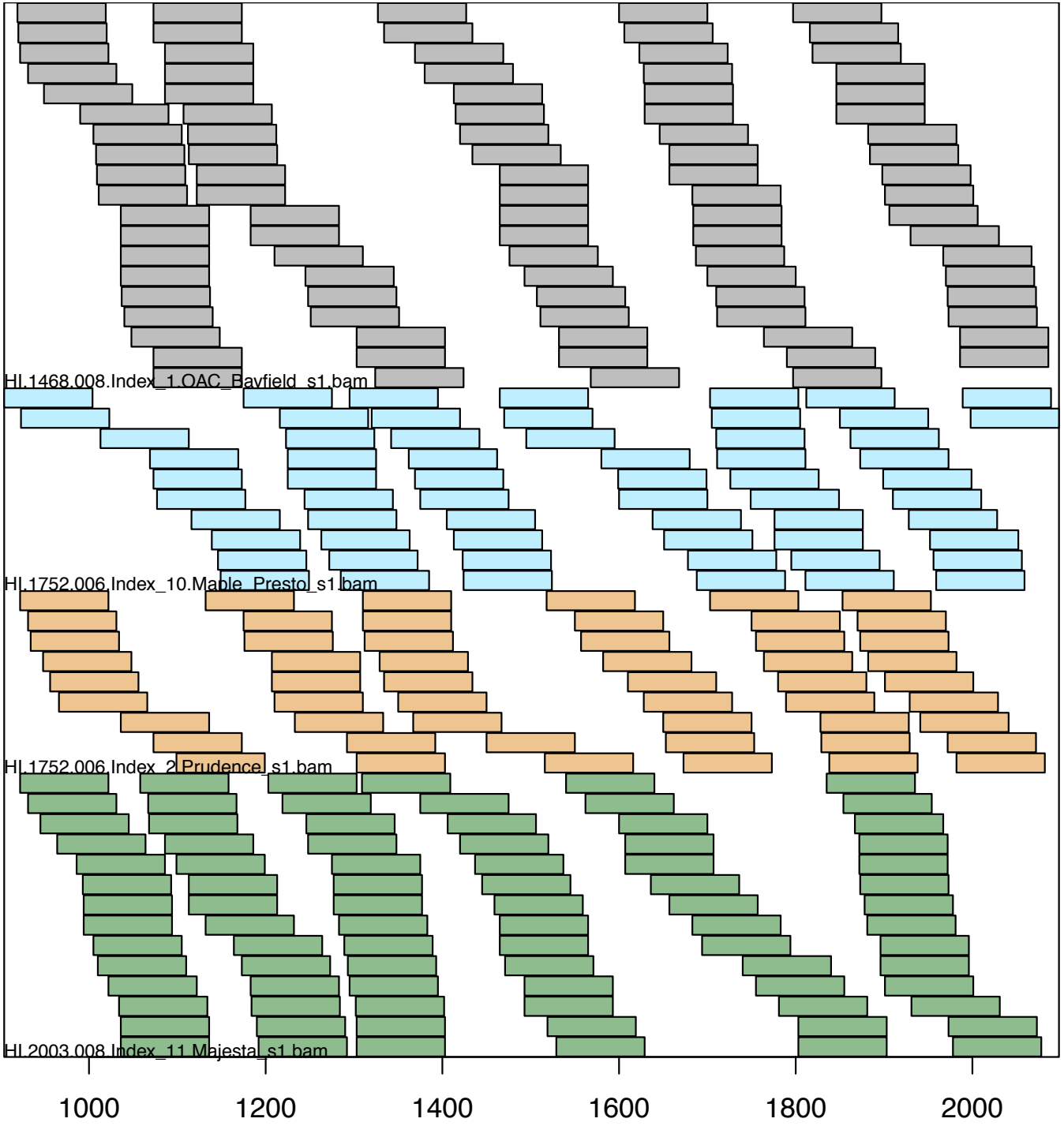

Espace pour les alignements
Espace pour l'axe

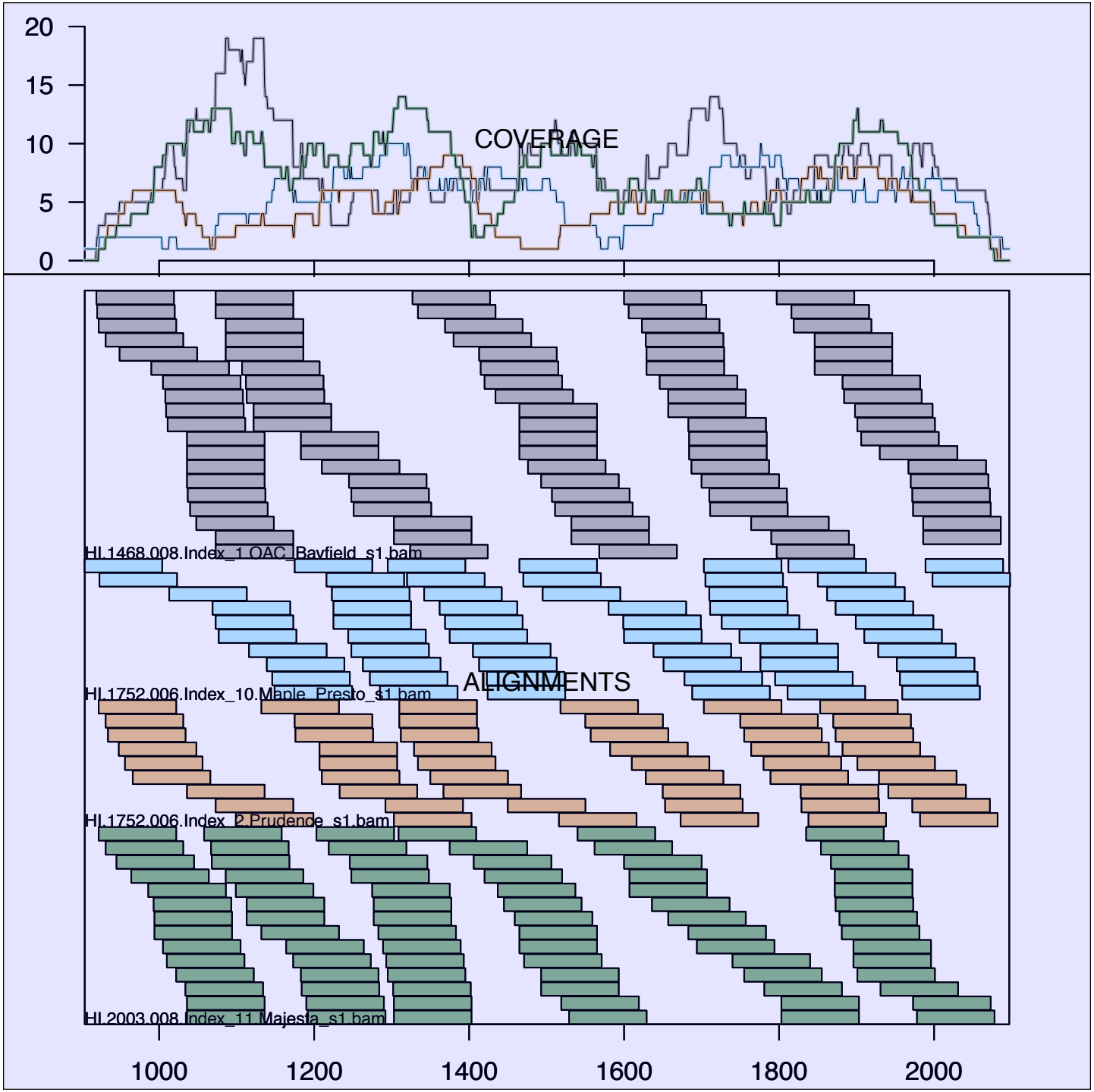
```
> samview("HI.1468.008.Index_1.OAC_Bayfield_s1.bam",
          "Chr01", c(1000, 2000))
```



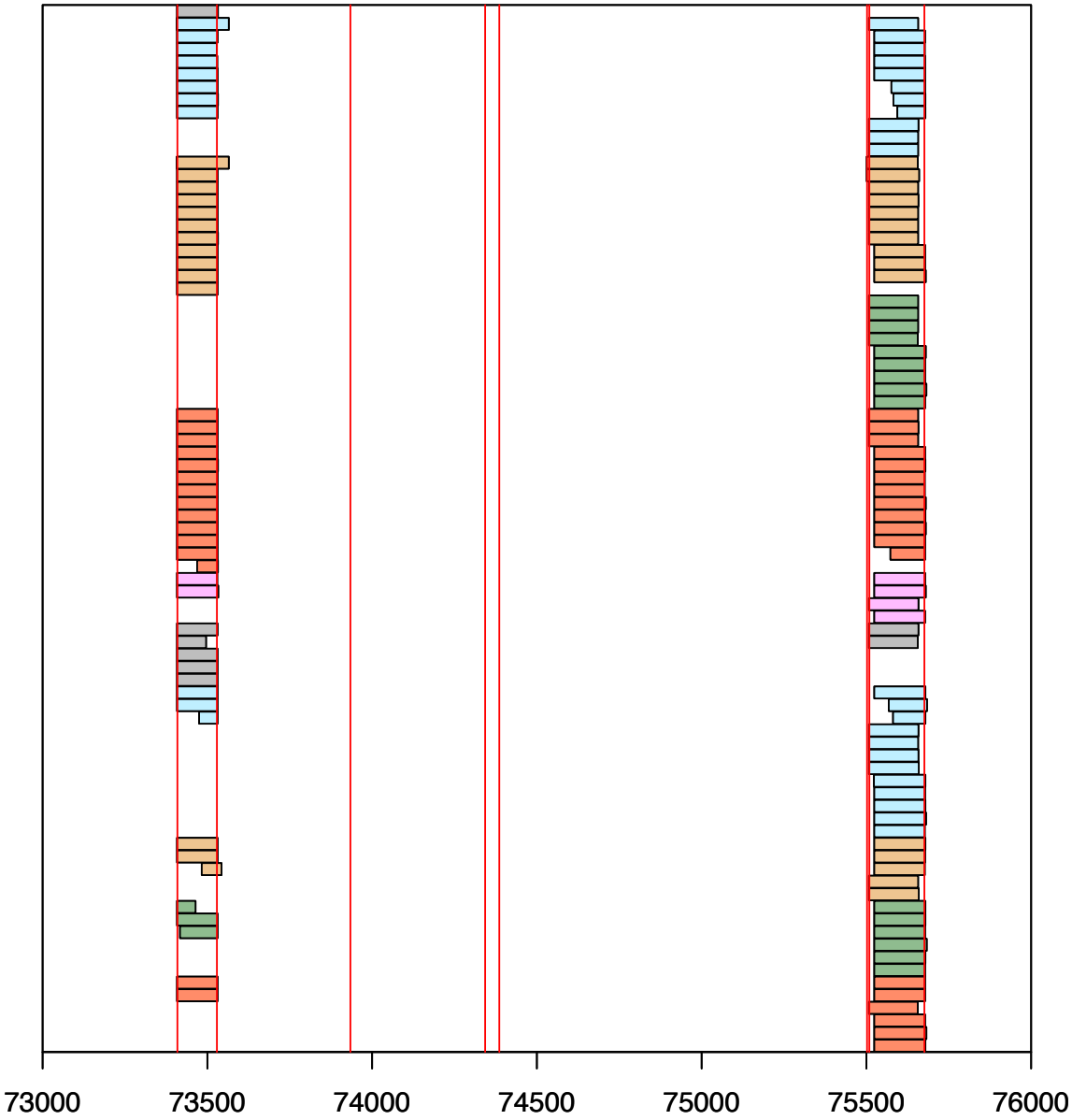
```
> samview(c("HI.1468.008.Index_1.OAC_Bayfield_s1.bam",
            "HI.1752.006.Index_10.Maple_Presto_s1.bam",
            "HI.1752.006.Index_2.Prudence_s1.bam",
            "HI.2003.008.Index_11.Majesta_s1.bam"),
            "Chr01", c(1000, 2000))
```



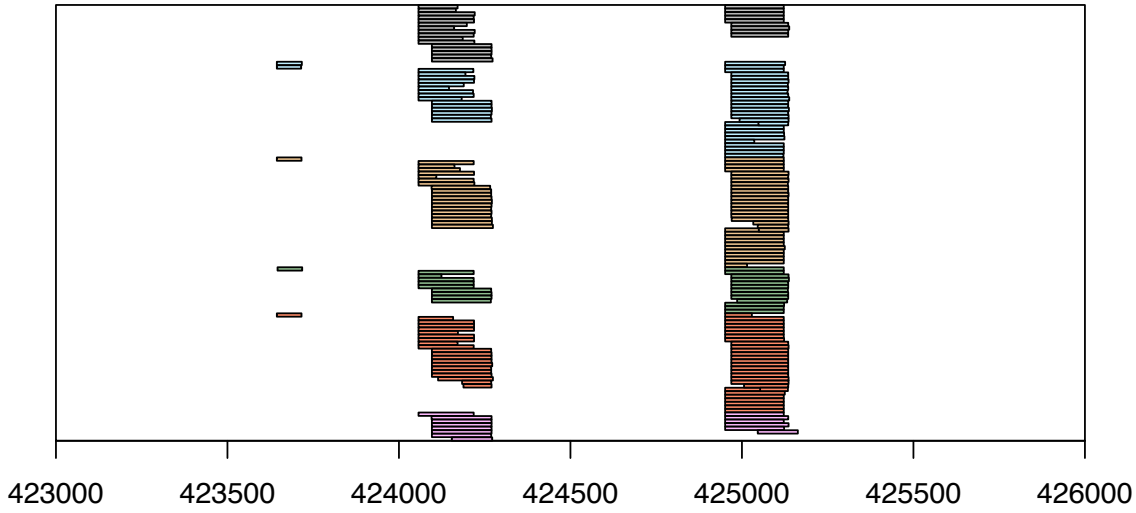

HI.1468.008.Index_1.OAC_Bayfield_s1.bam
HI.1752.006.Index_10.Mable_Presto_s1.bam
HI.1752.006.Index_2.Prudence_s1.bam
HI.2003.008.Index_11.Majesta_s1.bam
1000
1200
1400
1600
1800
2000

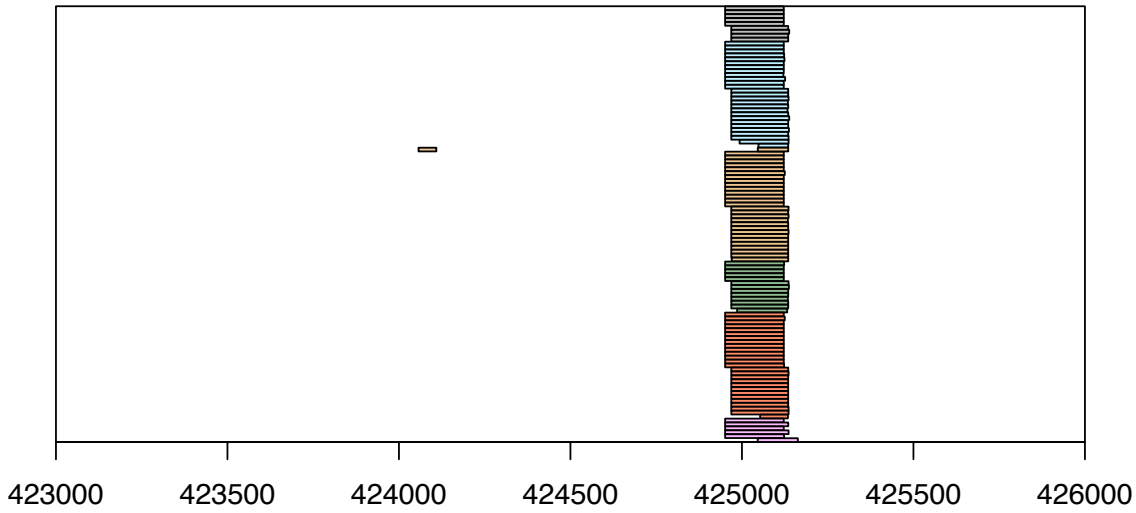

20
15
10
5
0
COVERAGE
HI.1468.008.Index_1.OAC_Bayfield_s1.bam
HI.1752.006.Index_10.Maple_Presto_s1.bam
HI.1752.006.Index_2.Prudence_s1.bam
HI.2003.008.Index_11.Majesta_s1.bam
ALIGNMENTS
1000
1200
1400
1600
1800
2000

```
> samview(FN_bam[100:110], "Chr04", c(73000, 76000))
> downViewport("alignments")
> rsite_plot(apeki_fragments, "Chr04", c(73000, 76000))
```



```
> samview(FN_bam[100:105], "Chr04", c(423000, 426000),
          newpage = FALSE)
> samview(FN_bam[100:105], "Chr04", c(423000, 426000),
          newpage = FALSE, mapqFilter = 50)
```





Pour plus d'infos

- Les vignettes du package `grid`
- R Graphics par Paul Murrell
- ggplot2 par Hadley Wickham, p. 151-155 et p. 199-202
- M'écrire à marc-andre.lemay.2@ulaval.ca