Utiliser l'environnement R pour créer ses visualisations de données issues d'Analytics (2ème partie)

Par Guillaume et Sylvain Peyronnet

Domaine :	**Recherche**	**Référencement**
Niveau :	Pour tous	**Avancé**

Connaissez-vous R ? Il s'agit d'un langage de programmation et un environnement de travail largement utilisé lorsqu'il s'agit d'analyser des données statistiques issues de sources diverses. Dans cet ensemble de deux articles, nous allons vous présenter cet outil et son utilisation pour proposer des graphiques et tableaux de bord plus ou moins simples ou sophistiqués en partant des données extraites de Google Analytics. Après avoir appris, le mois dernier, à récupérer les informations d'Analytics et à créer un graphique simple affichant le nombre de visites sur un site web, nous poursuivons notre exploration avec 3 exemples de graphiques réalisés avec R autour du taux de rebond d'un site, du trafic par région et du « not provided ».

Ce mois-ci nous poursuivons l'article portant sur l'utilisation de R pour créer des visualisations de données issues de Google Analytics. Maintenant que nous avons vu les basiques, nous allons pouvoir nous attaquer à quelques représentations plus fouillées, qui pourraient par exemple être utiles pour la création d'un dashboard (tableau de bord) hebdomadaire.

La suite de l'article sera donc présentée sous forme de tutoriels pour trois visualisations différentes. La première va permettre de réfléchir au taux de rebond d'un site. La deuxième offrira une vision géolocalisée des sessions des visiteurs, du temps passé par ces derniers sur le site et du taux de rebond associé. Enfin, la troisième se consacre à la représentation du fameux « not provided ».

Une réflexion sur le taux de rebond, grâce à une belle visualisation

Le taux de rebond est une métrique très ambiguë. La définition en est simple : il s'agit du pourcentage de visiteurs qui, après avoir vu la page par laquelle ils sont entrés sur le site, vont le quitter aussitôt (et ne verrons donc qu'une seule page).

L'interprétation du taux de rebond est en revanche plus complexe. Un taux de rebond très fort peut-être vu comme un signal d'insatisfaction des visiteurs du site (qui sont déçus et donc s'en vont rapidement) ou au contraire comme un signal de qualité (l'information

recherchée a été délivrée efficacement, et donc le visiteur peut aller visiter un autre site rapidement).

Nous allons voir comment réaliser une visualisation permettant une vraie réflexion sur le taux de rebond d'un site donné.

Nous commençons par charger les packages indispensables et réactiver le *token* Google Analytics que nous avons pris soin de stocker le mois dernier.

```
>library(RGoogleAnalytics)
>library(ggplot2)
>library(reshape2)
>library(plyr)
>library(scales)
>library(gridExtra)

>load("./token_file")
>ValidateToken(token)
```

Nous choisissons ensuite les dimensions et métriques qui vont nous être utiles. Ici on s'intéresse aux sessions, à leurs dates, au rebond mais aussi aux nombres de sessions issues du trafic organique. Notre objectif va être de comparer le taux de rebond et le taux de sessions issues de l'organique, pour être capable de comprendre si le taux de rebond évolue d'une manière satisfaisante ou non par rapport au trafic moteur.

```
>ga_dimensions <- 'ga:date'
>ga_metrics <- "ga:sessions,ga:bounces,ga:organicSearches"
>sort <- 'ga:date'
```

On va ensuite faire un appel à l'API de Google Analytics. Nous avons déjà vu le code pour cela le mois dernier, le voici à nouveau.

```
>query.list <- Init(start.date = startdate,
                    end.date = enddate,
                    dimensions = ga_dimensions,
                    metrics = ga_metrics,
                    max.results = 10000,
                    sort = sort,
                    table.id = profileid)

>ga.query <- QueryBuilder(query.list)
>ga.data <- GetReportData(ga.query, token)
```

Une fois l'acquisition des données effectuée, on peut les travailler un peu. La première étape est de mettre la date à un format acceptable pour créer la visualisation.

```
> ga.data$date <- as.Date(ga.data$date, "%Y%m%d")
```

Ensuite, on calcule le taux de rebond et le taux de sessions « organiques » (notez que j'aurais pu demander directement à l'API le taux de rebond, ce que nous ferons dans l'exemple suivant), et on ordonne les données par date.

```
> ga.data$bounce_rate <- round(100 * ga.data$bounces / ga.data$sessions,0)
> ga.data$organic_rate <- round(100 * ga.data$organicSearches / ga.data$sessions,0)
> df <- ga.data[order(ga.data$date), ]
```

On obtient un *dataset* du type suivant (la fonction head permet de visualiser les premières lignes d'un dataset) :

```
> head(df)
      date sessions bounces organicSearches bounce_rate organic_rate
1 2015-01-01      106      74             58          70          55
2 2015-01-02      182     111             95          61          52
3 2015-01-03      153     108            103          71          67
4 2015-01-04      158     108             96          68          61
```

La dernière étape est de générer la figure représentant ce que l'on souhaite visualiser.

```
> ggplot(df[305:365,], aes(date))
+ geom_line(aes(y = bounce_rate, colour = "Taux de rebond"))
+ geom_line(aes(y = organic_rate, colour = "Sessions organiques"))
+ coord_cartesian(ylim=c(0,100)) + labs(y="Pourcentage",x = "Novembre et décembre 2015")
+ theme(legend.title=element_blank())
+ theme(legend.background = element_rect(fill="pink", size=.5, linetype="dotted"))
```

Ici, on a créé un graphique qui met en abscisse la date pour les jours 302 à 365 de 2015 (il s'agit de novembre et décembre). On a deux instructions geom_line car on trace deux quantités différentes sur le même graphique. Les autres commandes permettent d'avoir une belle visualisation (des axes qui vont de 0 à 100 pour les pourcentages et une boite de légende en rose avec les noms des métriques).

Visuellement, le résultat est représenté sur la fig.1.

Il ne reste plus qu'à interpréter le graphique. On voit qu'il existe une corrélation graphique entre le taux de rebond et le taux de sessions organiques. Cela signifie que lorsque l'on augmente le trafic moteur, on génère des visiteurs qui quittent le site immédiatement. Selon les requêtes utilisées par les visiteurs, il devient facile de savoir si le site a un problème de rétention ou non. Ici il y a fort à parier qu'il y a ici un problème, car les utilisateurs qui viennent des moteurs ont un taux de rebond plus fort que les visiteurs habituels et issus des réseaux sociaux.

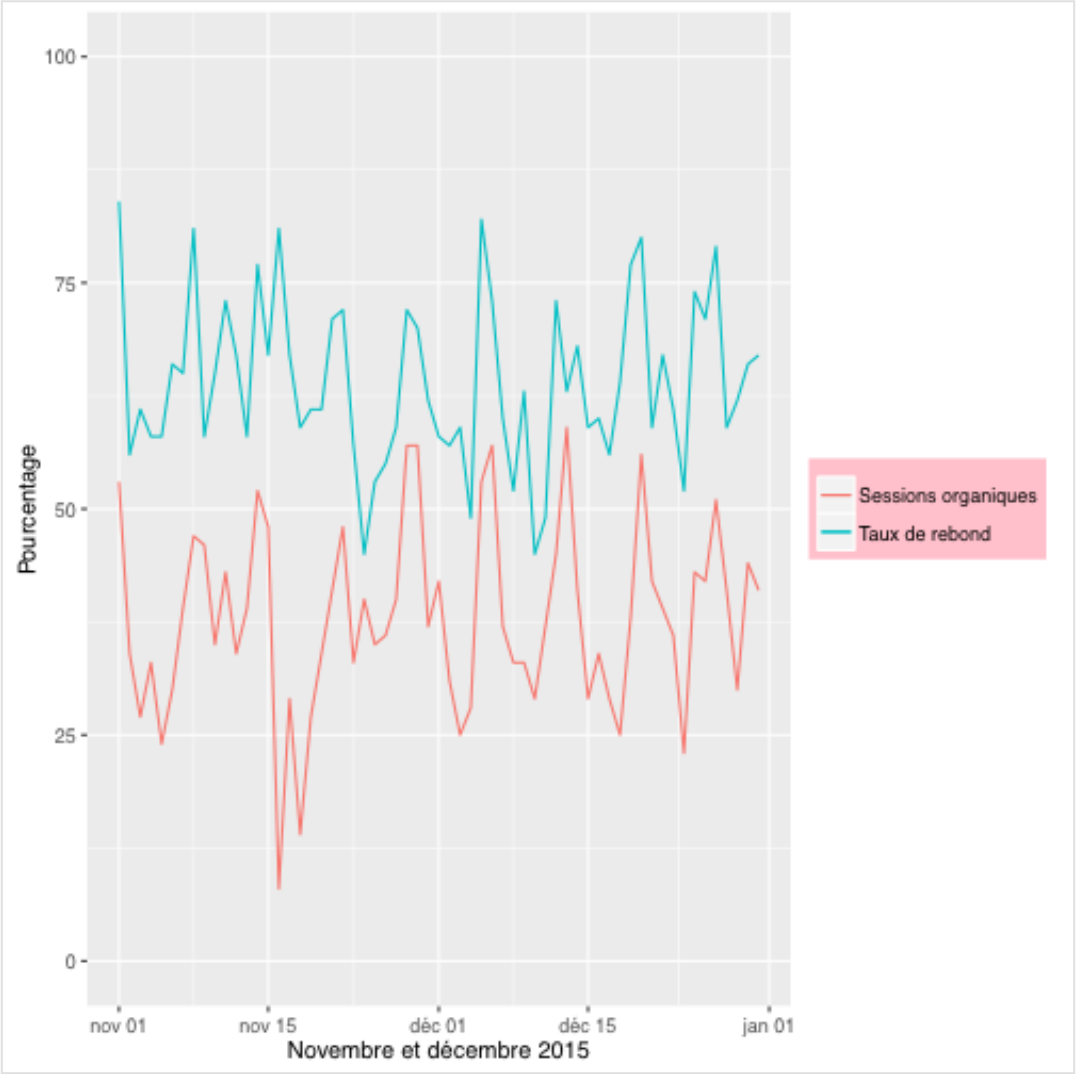

Fig.1. Visualisation du taux de rebond sur un site

Une visualisation unifiée du trafic régional

Nous allons maintenant créer une vue unifiée du trafic régional français pour un site donné. Avec ce type de graphique, on se situe résolument dans la création de tableaux de pilotage.

On commence par définir les données que l'on souhaite récupérer.

```
> ga_dimensions <- 'ga:date,ga:region'
> ga_metrics <- "ga:visits, ga:avgTimeOnSite,ga:bounceRate"
> sort <- 'ga:date, -ga:visits'
> ga_filter <- 'ga:country==France'
```

La nouveauté ici est le ga_filter qui va nous permettre de ne conserver que les sessions venues de France (car on s'intéresse aux régions françaises).

On va ensuite exécuter la requête à l'API de Google Analytics, avec le rajout d'une ligne pour le filtre « filters = ga_filter, ». Nous ne remettons pas le code permettant de faire la requête, il s'agit bien évidemment du même qu'auparavant.

L'étape suivante est, vous l'aurez deviné, le nettoyage et le formatage des données.

```
> ga.clean <- ga.data[!ga.data$region == "(not set)", ]

> totvis <-
ddply(ga.clean,.(region),summarize,totvis=sum(visits),avgvis=mean(avgTimeOnSite),bounc
er=mean(bounceRate))

> ordered_totvis <- totvis[order(totvis$totvis,decreasing=TRUE),]
```

La première ligne du code source visible ci-dessus permet d'enlever du *dataset* les sessions françaises ne déclarant aucune région. totvis correspond au dataset formaté pour faire apparaître le nombre total de sessions par régions, ainsi que le temps moyen par sessions pour chaque région, et le taux de rebond moyen pour chaque région. Tout ceci est calculé grâce au très bon package plyr qui a été conçu pour faciliter le travail sur les *datasets* dans R. On annonce ici que l'on crée un résumé par région (.(region),summarize), qui comporte la somme du nombre de sessions sur la période, et les moyennes décrites auparavant.

Enfin, on va trier ce nouveau dataset par ordre décroissant du nombre de sessions, ce qui va donner le résultat suivant (seules les premières lignes sont affichées ici).

```
> head(ordered_totvis)
                  region totvis    avgvis  bouncer
10            Ile-de-France   7770 141.41000 72.39343
17         Pays de la Loire   2488 299.08515 60.14177
21             Rhone-Alpes   2457  84.05730 67.47071
20 Provence-Alpes-Cote d'Azur   2212 152.02636 57.68094
16       Nord-Pas-de-Calais   1190  53.21502 76.60129
15           Midi-Pyrenees   1043  91.35912 63.43083
```

On peut maintenant créer une visualisation résumant toutes ces informations.

```
> ggplot(ordered_totvis, aes(x=region, y=totvis))
+ geom_point(aes(size=avgvis, colour=bouncer))
+ theme(axis.text.x = element_text(angle = 45, hjust = 1))
+ scale_colour_gradient(low = "green",high="red")
+ labs(y="Nombre total de sessions",x = "Région",size="temps moyen\npar
session",colour="Taux de \n rebond")
+ theme(legend.background = element_rect(fill="pink", size=.5, linetype="dotted")
```

Cette (longue) ligne de code va tracer le contenu du dataset reformaté et trié pour afficher un point par région, l'ordonnée du point représentant le nombre de sessions pour la région. Ensuite, la taille du point (size=avgvis) représentera le temps moyen par session pour la région, et la couleur du point (colour=bouncer) représentera le taux de rebond. Plus ce dernier sera haut plus le point sera rouge (scale_colour_gradient(low = "green",high="red")). Les dernières lignes servent uniquement à rendre le graphique encore plus beau en choisissant les titres et les couleurs des légendes.

Au final, on obtient la visualisation de la fig.2.

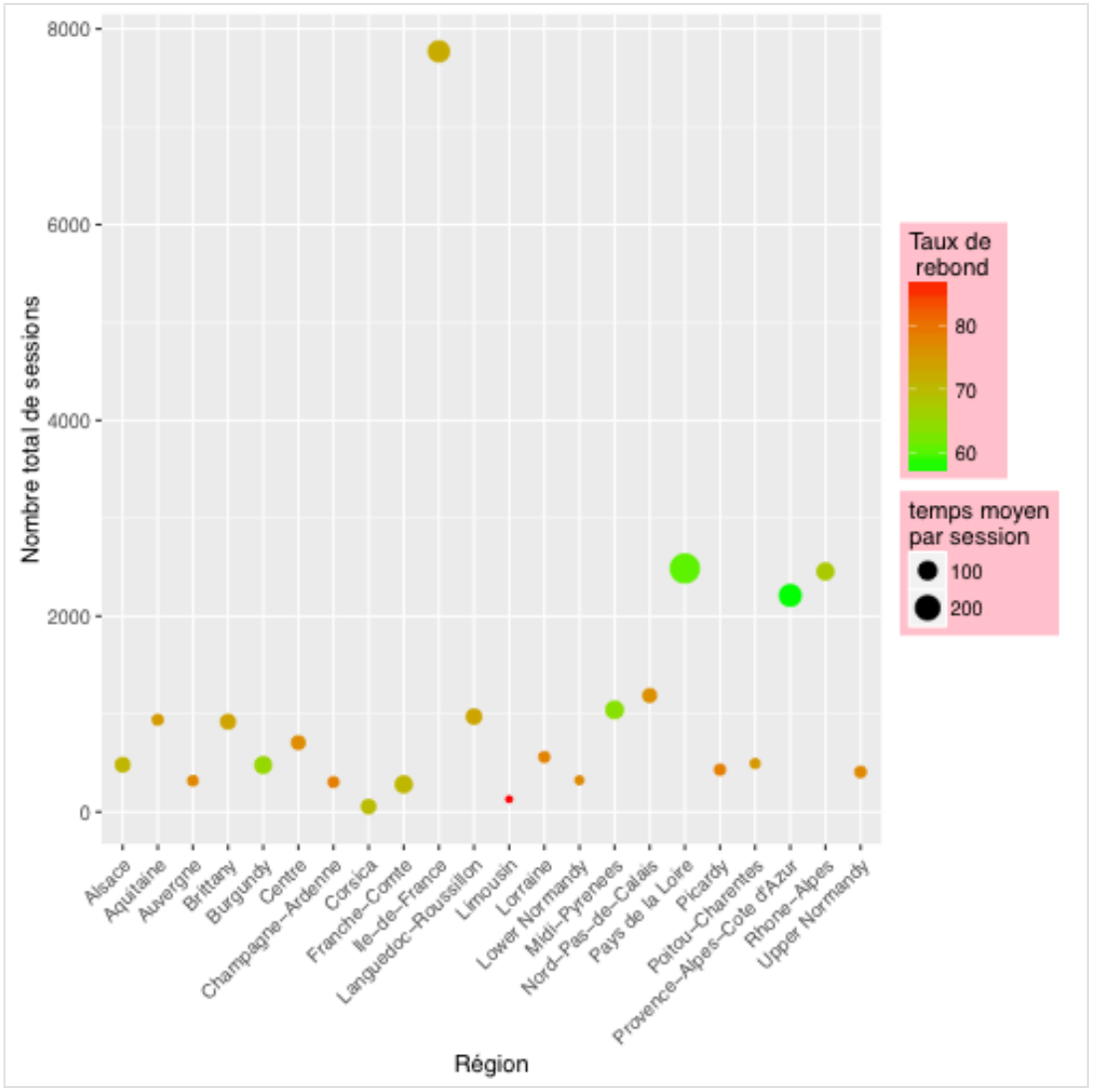

Fig.2. Visualisation d'un trafic régional

La lecture de ce type de visualisation est très simple : plus un point est haut sur le graphique, et plus il est gros et vert, plus votre site est performant pour la région.

Une visualisation simple : la proportion de not provided

Une petite dernière visualisation, vraiment très facile à réaliser : un histogramme permettant de mettre en évidence la proportion de « not provided » sur une période de temps donnée. Parce que c'est une visualisation plus simple, nous ne mettons pas tout le code dans cet article, ce qui vous permettra de faire un petit exercice d'utilisation de R.

Comme toujours, on va définir les métriques qui nous intéressent.

```
> ga_dimensions <- 'ga:date,ga:keyword'
> ga_metrics <- "ga:visits"
> sort <- 'ga:date'
> ga_filter <- 'ga:source==google,ga:medium==organic'
```

On s'intéresse aux visites issues du moteur de recherche Google, par date, avec comme information supplémentaire le mot-clé qui a permis au visiteur d'arriver sur le site. Quand le mot-clé n'est pas donné, l'API renvoie « (not provided) » comme vous le savez.

La requête associée à ces métriques et dimensions renvoie des données qui ressemblent à cela :

```
> head(ga.data)
     date                              keyword visits
1 20150101                        (not provided)     95
2 20150101         adele blanc sec scene execution      1
3 20150101                 bande dessinee erotique      1
4 20150101 bande dessinée gothique erotique photo      1
5 20150101                      bd hot charlotte      1
6 20150101                        bibifricotin      1
```

Contrairement à ce que l'on pourrait croire, le site en question est tous publics, mais les demandes des internautes le sont moins. Bref, une fois les données acquises, il faut les nettoyer et les formater pour obtenir un *dataset* facilement utilisable pour construire une visualisation.

Pour cela on va faire plusieurs opérations que nous vous laissons trouver vous-même, mais parmi celles-ci se trouve l'utilisation d'une boucle sur le *dataset* fourni par l'API.

```
> for( i in 1:4123){
ifelse(ga.data$keyword[i] != "(not provided)" , ga.data$keyword[i] <- "(provided)" ,
ga.data$keyword[i] <- "(not provided)" )
}
```

Une fois ces multiples opérations effectuées, on obtient un dataset qui est sous la forme suivante :

```
> head(ga.data)
        date        keyword visits
1 2015-01-01 (not provided)     95
2 2015-01-01     (provided)      1
3 2015-01-01     (provided)      1
```

Il ne reste qu'à coder la visualisation.

```
>  ggplot(ga.data[3848:4123,],aes(x=date,y=visits,fill=keyword)) +
geom_bar(stat='sum',position='fill') + labs(y="Sessions",x = "Date") +
theme(legend.background = element_rect(fill="pink", size=.5, linetype="dotted")) +
theme(legend.position = "none")
```

Le code est ici un peu brutal avec un travail minimal sur le dataset, mais il permet d'obtenir la visualisation de la fig.3.

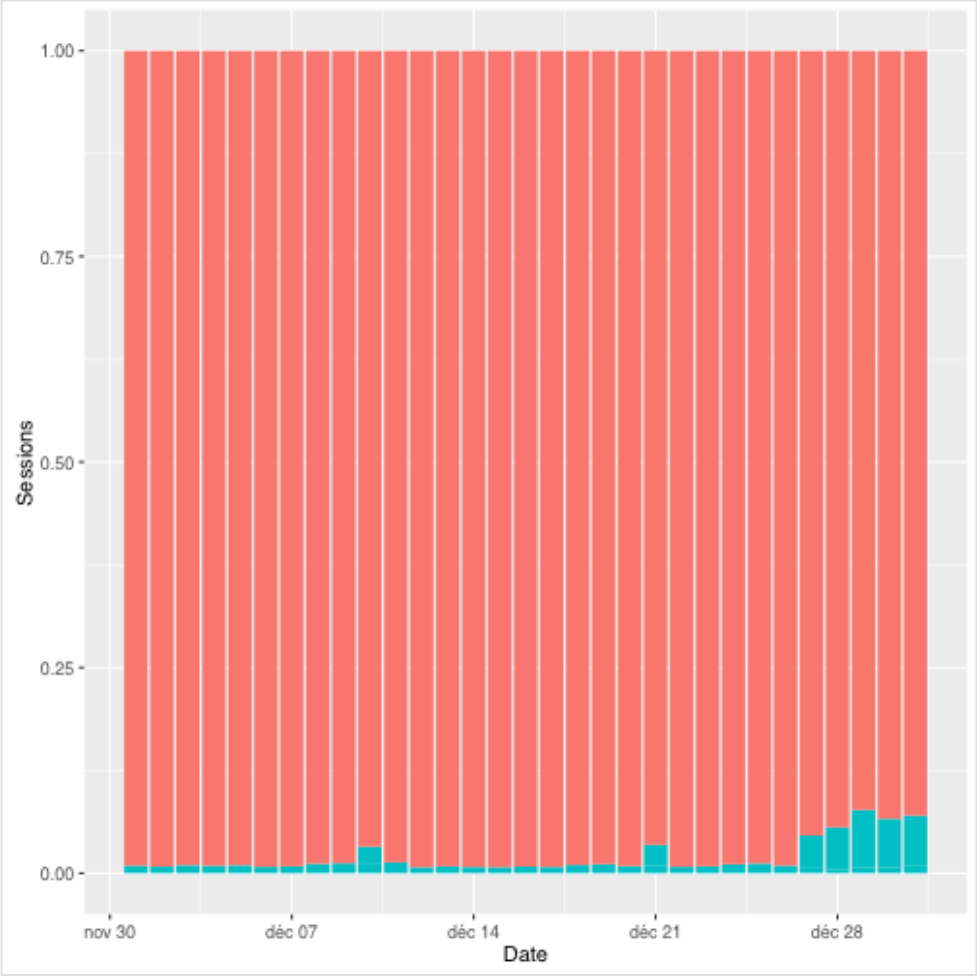

Fig.3. Représentation du « not provided » dans les requêtes d'accès à un site.

La proportion de « not provided » est indiquée en rouge dans ce graphique. On voit donc que pour ce site web particulier, la quasi-totalité des accès *via* Google ne transmet aucune information de mot-clé.

Conclusion

La conclusion de cet article en deux parties est assez simple : il est possible en utilisant l'API de Google Analytics de créer des visualisations personnalisées très utiles. En utilisant R en mode scripté plutôt qu'en mode interactif comme nous l'avons fait ici, et en utilisant des outils de planification de tâches comme CRON par exemple, il est possible d'automatiser totalement la création de tableau de bord journalier, hebdomadaire, etc.

La réalisation d'un fichier PDF contenant toutes les visualisations et l'envoi par mail de ce document étant également faisable à l'aide de R, c'est maintenant à vous de jouer !

Références

[1] https://www.r-project.org/

[2] https://developers.google.com/analytics/devguides/reporting/core/dimsmets

Guillaume Peyronnet *est gérant de Nalrem Médias.* Sylvain Peyronnet *est co-fondateur et responsable des ix-labs, un laboratoire de recherche privé. Ensemble, ils font des formations, pour en savoir plus : http://www.peyronnet.eu/blog/*