

# Lab 01 - Salut R!

L'objectif principal de ce laboratoire est de vous présenter R et RStudio, que nous utiliserons tout au long du cours à la fois pour apprendre les concepts statistiques abordés dans le cours et pour analyser des données réelles et tirer des conclusions éclairées.

Un objectif supplémentaire est de vous présenter Git et GitHub, qui est le système de collaboration et de contrôle de version que nous utiliserons tout au long du cours.

Au fur et à mesure que les laboratoires progressent, vous êtes encouragé à explorer au-delà de ce que les laboratoires dictent ; une volonté d'expérimenter fera de vous un bien meilleur scientifique de données. Avant d'en arriver à cette étape, cependant, vous devez développer une maîtrise de base de R. Aujourd'hui, nous commençons par les éléments de base de R et RStudio : l'interface, la lecture des données et les commandes de base.

Et pour simplifier la gestion des versions, il s'agit d'un laboratoire solo. De plus, je veux m'assurer que tout le monde passe beaucoup de temps à pratiquer. Dans les

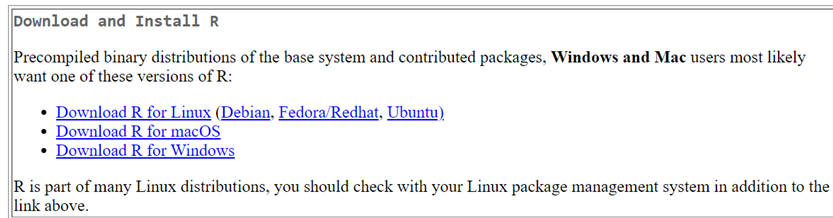
R est le nom du langage de programmation lui-même et RStudio est une interface pratique.

git est un système de contrôle de version (comme les fonctionnalités "Suivi des modifications" de Microsoft Word sur les stéroïdes) et GitHub est l'environnement pour vos projets basés sur Git sur Internet (comme DropBox mais beaucoup, beaucoup mieux).

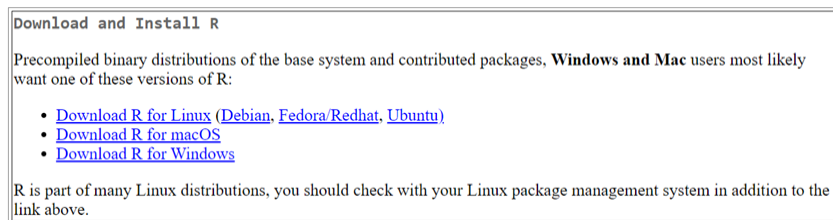
prochains ateliers, vous apprendrez à collaborer sur GitHub et à produire un seul rapport d'atelier pour votre équipe.

## Installer tout ce que vous avez besoin

- Installer R <https://cran.r-project.org/>



- Installer Rstudio  
<https://www.rstudio.com/products/rstudio/download/#download>



- Installer Git : o Windows :  
<https://gitforwindows.org/> o Mac : Installez les outils de ligne de commande Xcode (pas tous Xcode), qui incluent Git.

## Connecter GitHub et RStudio

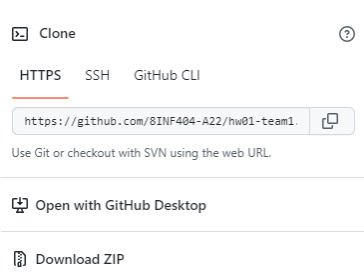
Vous devriez déjà avoir reçu une invitation à rejoindre l'organisation GitHub pour ce cours. Vous devez accepter l'invitation avant de passer à l'étape suivante.

Chacune de vos missions (laboratoire, devoir ou projet) commencera par les étapes suivantes. Ce sont les étapes des notes de cours, elles sont à nouveau décrites en détail ici. À l'avenir, chaque laboratoire commencera par une section “Commençons”, mais les détails ne seront pas aussi clairs que ce premier laboratoire! Vous pouvez toujours vous reporter à ce laboratoire pour obtenir une liste détaillée des étapes nécessaires à la mise en route d'un devoir.

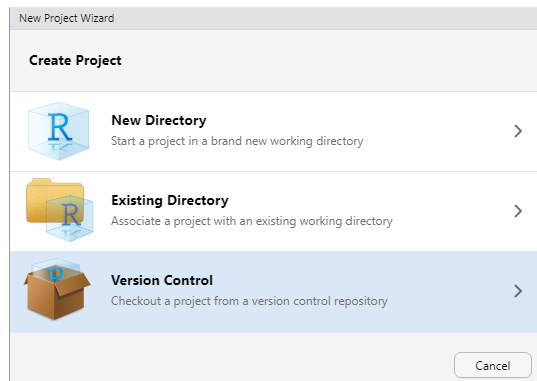
## Commençons

- Cliquez sur le lien du devoir que vous avez dû recevoir dans votre e-mail pour créer votre dépôt GitHub (que nous appellerons désormais “dépôt”) pour le devoir. Ce dépôt contient un modèle sur lequel vous pouvez vous appuyer pour terminer votre devoir.
- Sur GitHub, cliquez sur le bouton vert **Cloner ou télécharger**, sélectionnez **Utiliser HTTPS** (cela peut déjà être sélectionné par défaut, et si c'est le

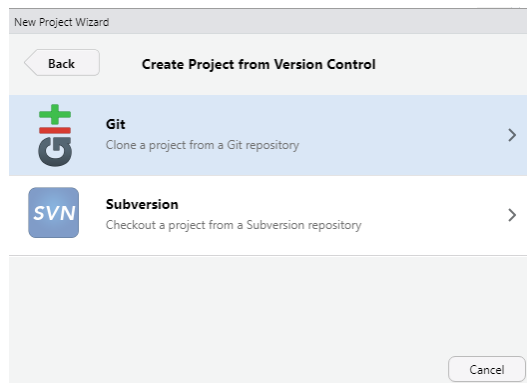
cas, vous verrez le texte **Cloner avec HTTPS** comme dans l'image ci-dessous). Cliquez sur l'icône du presse-papiers pour copier l'URL du référentiel.



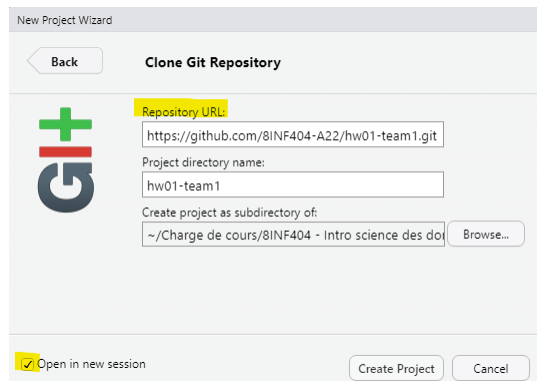
- Accédez à RStudio et dans file -> New Projet. Créez un nouveau projet **version control**



- Choisir l'option Git:



- Copiez et collez l'URL de votre devoir dans la boîte de dialogue :



- Cliquez sur OK, vous êtes maintenant prêt à partir !

## On s'échauffe

Avant de vous présentez les données, échauffons-nous avec quelques exercices simples.

## YAML

Ouvrez le fichier R Markdown (Rmd) dans votre projet, remplacez le nom de l'auteur par votre nom et assemblez (nouton *knit*) le document.

La partie supérieure de votre fichier R Markdown (entre les trois lignes pointillées) s'appelle YAML. Il s'agit d'une norme de sérialisation de données conviviale pour tous les langages de programmation. Tout ce que vous devez savoir, c'est que cette zone s'appelle le YAML (nous l'appellerons ainsi) et qu'elle contient des méta-informations sur votre document.

```
1 ---
2 title: "Lab 01 - Bonjour R"
3 author: "Insere ton nom ici"
4 date: "Insere la date ici"
5 output: github_document
6 ---
```

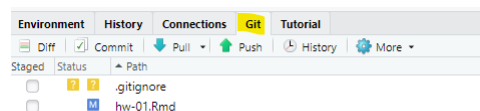


## Lab 01 - Bonjour R

Insere ton nom ici Insere la date ici

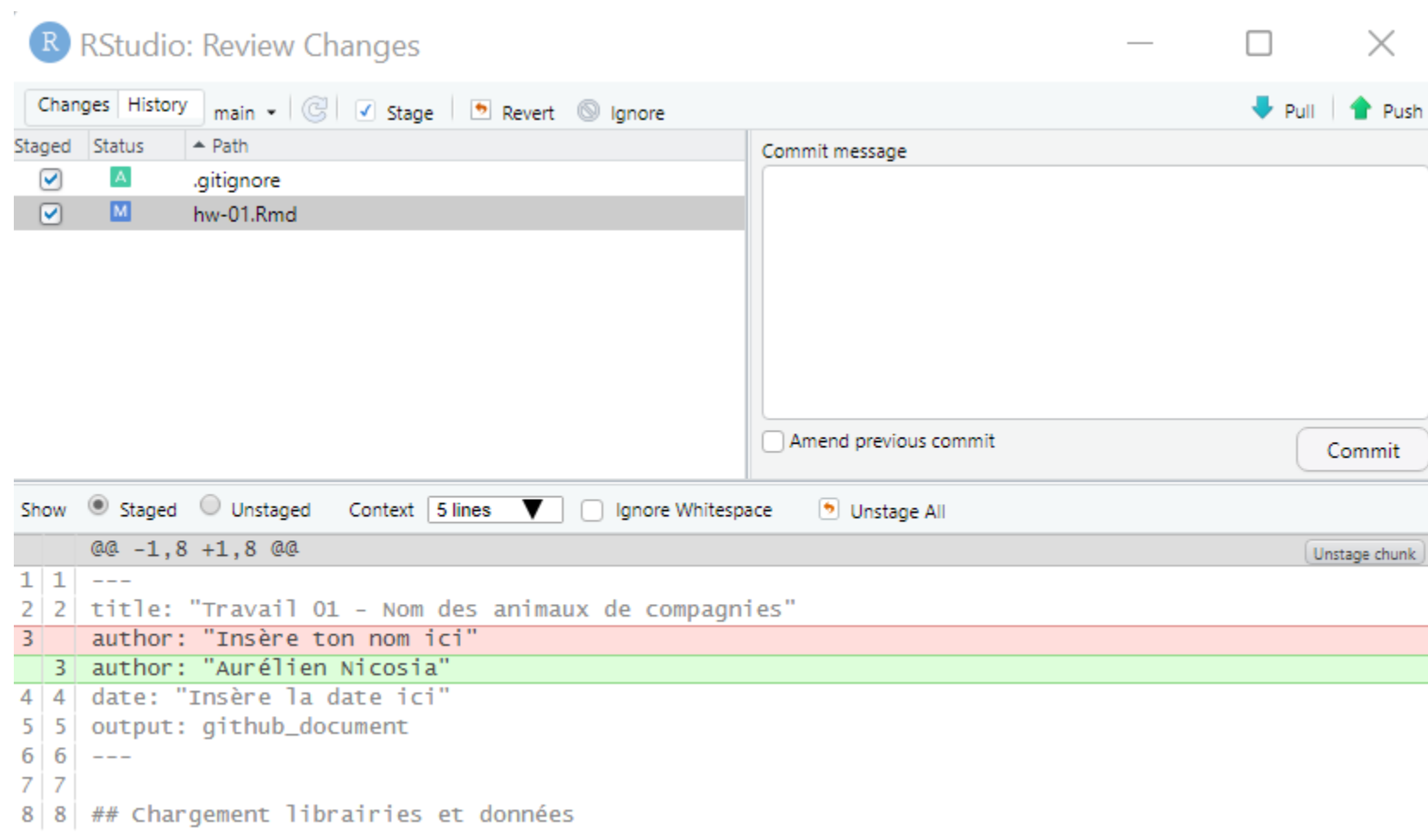
### Validation des modifications

Accédez ensuite au volet Git de votre RStudio (en haut à droite)



Si vous avez apporté des modifications à votre fichier Rmd, vous devriez le voir répertorié ici. Cliquez dessus pour le sélectionner dans cette liste puis cliquez sur **Diff**. Cela vous montre la *différence* entre le dernier état validé du

document et son état actuel qui inclut vos modifications. Si vous êtes satisfait de ces changements, écrivez “Mettre à jour le nom de l’auteur” dans la case **Commit message (Message de validation)** et appuyez sur **Commit (Valider)**.



Vous n’avez pas valider (commit) après chaque modification, cela deviendrait assez fastidieux. Vous devriez envisager de valider des nouveautés de votre fichier .Rmd

qui sont \* significatifs pour vous \*. Dans les premières missions, je vous dirais exactement quand valider et, dans certains cas, quel message de validation utiliser. Au fur et à mesure que le semestre avance, je vous laisserai prendre ces décisions.

### **Modifications poussées**

Maintenant que vous avez effectué une mise à jour et validé cette modification, il est temps de diffuser ces modifications sur le Web ! Ou plus précisément, à votre repo sur GitHub.

Pourquoi?

Pour que les autres puissent voir vos modifications. Et par d'autres, j'entends moi (vos dépôts dans ce cours sont privés pour vous et moi).

Pour pousser vos modifications sur GitHub, cliquez sur **Push**. Cela ouvrira une boîte de dialogue dans laquelle vous devrez d'abord entrer votre nom d'utilisateur, puis votre mot de passe. Cela peut sembler fastidieux. Soyez indulgent avec moi... Nous *allons* vous apprendre à enregistrer votre mot de passe pour que vous n'ayez pas à le saisir à chaque fois. Mais pour ce laboratoire, vous devrez entrer manuellement chaque fois que vous appuyez sur le bouton *push* afin d'acquérir une certaine expérience.



## Packages

Dans cet atelier, nous travaillerons avec deux packages :

**datasauRus** qui contient l'ensemble de données que nous allons utiliser et **tidyverse** qui est une collection de packages permettant d'analyser les données de manière "propre". Si vous ne les avez jamais installés, vous devez le faire, par les lignes de commandes suivantes à inscrire dans la console:

```
install.packages("tidyverse")
```

```
## Error in install.packages : Updating loaded packages
```

```
install.packages("datasauRus")
```

```
## le package 'datasauRus' a été décompressé et les somn  
##  
## Les packages binaires téléchargés sont dans  
## C:\Users\unic3\AppData\Local\Temp\RtmpwBSqmN\downlo
```

Vous pouvez charger les packages en exécutant ce qui suit dans la console.

```
library(tidyverse)
library(datasauRus)
```

Notez que les packages sont également chargés avec les mêmes commandes dans votre document R Markdown.

## Data

Le jeu de données avec laquelle nous allons travailler aujourd'hui s'appelle **datasauros\_dozen** et se trouve dans le paquage **datasauRus**. En fait, ce jeu de données contient 13 ensembles de données, conçus pour nous montrer pourquoi la visualisation des données est importante et comment les statistiques descriptives seules peuvent être trompeuses. Les différents jeux de données sont marqués par la variable **dataset**.

Pour en savoir plus sur l'ensemble de données, saisissez ce qui suit dans votre console : **?datasauros\_dozen**. Un point d'interrogation devant le nom d'un objet fera toujours apparaître son fichier d'aide. Cette commande doit être exécutée dans la console.

Si vous trouvez bizarre que le jeu de données s'appelle **datasauros\_dozen** alors qu'il contient 13 ensembles de données, vous n'êtes pas seul ! Avez-vous déjà entendu parler d'un baker's dozen?

## Exercices

**Exercise 1.** D'après le fichier d'aide, combien de lignes et combien de colonnes le fichier **datasaurus\_dozen** a-t-il ? Quelles sont les variables incluses dans la base de données ? Ajoutez vos réponses à votre rapport de laboratoire.




Voyons quels sont ces ensembles de données. Pour ce faire, nous pouvons créer un *tableau de fréquence* de la variable de l'ensemble de données :

```
datasaurus_dozen %>%  
  count(dataset) %>%  
  print(13)
```

```
## # A tibble:  
## #   13 x 2  
##   dataset  
##   <chr>  
## 1 away  
## 2 bullseye  
## 3 circle  
## 4 dino  
## 5 dots  
## 6 h_lines  
## # ... with 7  
## #   more  
## #   rows,  
## #   and 1
```

```
## # more
## # variable:
## # n <int>
## # i Use `print(n = ...)` to see more rows, and `colna
```

Le Datasaurus original (**dino**) a été créé par Alberto Cairo dans [ce super article de blog](#). Les autres douzaines ont été générées à l'aide d'un recuit simulé (simulated annealing) et le processus est décrit dans l'article *Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing* de Justin Matejka et George Fitzmaurice. Dans l'article, les auteurs simulent une variété d'ensembles de données qui ont les mêmes statistiques descriptives que le Datasaurus mais qui ont des distributions très différentes.

   *Knit, ccommit, et poussez vos modifications vers GitHub avec le message de commit "ajout de la réponse pour l'ex 1". Assurez-vous de valider et de pousser tous les fichiers modifiés afin que votre volet Git soit effacé par la suite.*

Matejka, Justin et George Fitzmaurice. "Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing." Actes de la conférence CHI 2017 sur les facteurs humains dans les systèmes informatiques. AMC, 2017.

**Exercice 2.** Tracez  $y$  vs.  $x$  pour le jeu de données **dino**. Ensuite, calculez le coefficient de corrélation entre  $x$  et  $y$  pour cet ensemble de données.

Vous trouverez ci-dessous le code dont vous aurez besoin pour effectuer cet exercice. Fondamentalement, la réponse est déjà donnée, mais vous devez inclure les éléments pertinents dans votre document Rmd, le générer (knit) avec succès et afficher les résultats.

Commencez par `datasaurus_dozen` et utiliser la fonction `filter` pour filtrer les observations où `dataset == "dino"`. Stockez le jeu de données filtré résultant en tant que nouvelle base de données appelée `dino_data`.

```
dino_data <- datasaurus_dozen %>%  
  filter(dataset == "dino")
```

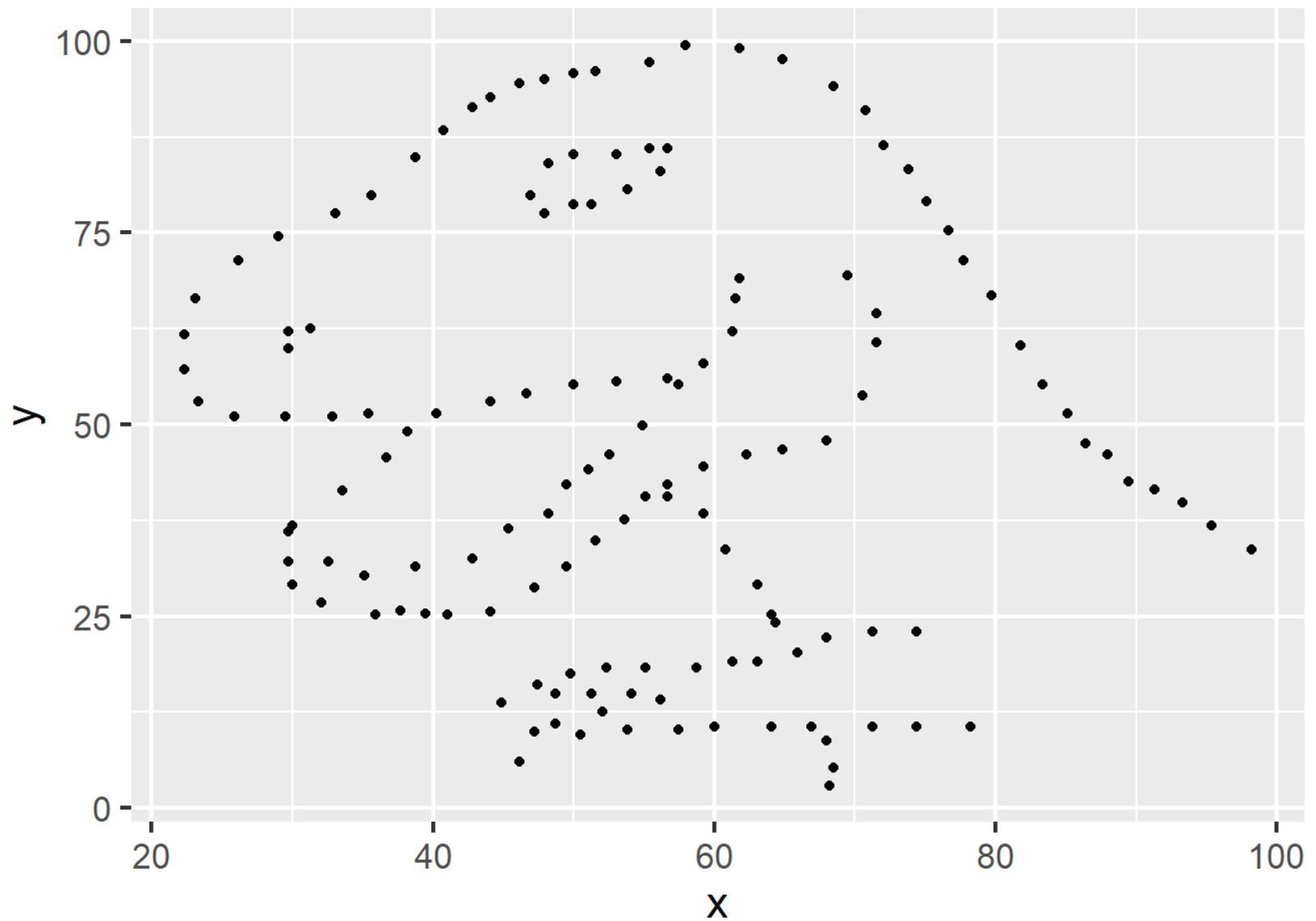
Il se passe beaucoup de choses ici, alors ralentissons et décompressons un peu.

Tout d'abord, l'opérateur pipe : `%>%`, prend ce qui vient avant et l'envoie comme premier argument à ce qui vient après. Donc `x %>% f(y)` est la version raccourci pour `f(x,y)`. Vous allez comprendre l'utilité de l'opérateur pipe au fur et à mesure. Donc ici, nous disons `filter` le jeu de données `datasaurus_dozen` pour les observations où `dataset == "dino"`.

Deuxièmement, l'opérateur d'affectation : `<-`, attribue le nom `dino_data` à la base de données filtrée.

Ensuite, nous devons visualiser ces données. Nous utiliserons la fonction `ggplot` pour cela. Son premier argument est le nom du jeu de données que vous visualisez. Ensuite, nous définissons les mappages « esthétiques ». En d'autres termes, les colonnes des données qui sont mappées à certaines caractéristiques esthétiques du graphique, par ex. l'axe 'x' représentera la variable appelée 'x' et l'axe 'y' représentera la variable appelée 'y'. Ensuite, nous ajoutons une autre couche à ce tracé où nous définissons les formes "géométriques" que nous voulons utiliser pour représenter chaque observation dans les données. Dans ce cas, nous voulons que ce soient des points, d'où `geom_point`.

```
ggplot(data = dino_data, mapping = aes(x = x, y = y))  
  +  
  geom_point()
```



Si cela semble beaucoup, ça l'est! Ne paniquez pas! Vous découvrirez en détail la philosophie de la création de visualisations de données en couches la semaine prochaine. Pour l'instant, suivez le code fourni.




Pour la deuxième partie de ces exercices, nous devons calculer une statistique descriptive : le coefficient de corrélation. Le coefficient de corrélation, souvent appelé  $r$  en statistique, mesure l'association linéaire entre deux variables. Vous verrez que certaines des paires de variables que nous traçons n'ont pas de relation linéaire entre elles. C'est exactement pourquoi nous voulons d'abord visualiser : visualiser pour évaluer la forme de la relation, et calculer  $r$  uniquement si cela est pertinent. Dans ce cas, calculer un coefficient de corrélation n'a vraiment aucun sens puisque la relation entre  $x$  et  $y$  n'est certainement pas linéaire!

Mais, à des fins d'illustration, calculons le coefficient de corrélation entre  $x$  et  $y$ .

```
dino_data %>%  
  summarize(r = cor(x, y))
```

```
## # A tibble: 1 x 1  
##       r  
##   <dbl>  
## 1 -0.0645
```




Commencez par `dino_data` et calculez une statistique descriptive que nous appellerons  $r$  en tant que corrélation entre  $x$  et  $y$ .

   *Knit, commit, et poussez vos modifications vers GitHub avec le message de commit "Ajout de la réponse pour Ex 2". Assurez-vous de valider et de pousser tous les*






*fichiers modifiés afin que votre volet Git soit effacé par la suite.*

**Exercise 3.** Tracez  $y$  par rapport à  $x$  pour l'ensemble de données **star**. Vous pouvez (et devriez) réutiliser le code que nous avons présenté ci-dessus, remplacez simplement le nom du jeu de données par le jeu de données souhaité. Ensuite, calculez le coefficient de corrélation entre  $x$  et  $y$  pour cet ensemble de données. Comment cette valeur se compare-t-elle au  $r$  de **dino** ?

   *C'est un autre bon endroit pour faire une pause, générer, valider les modifications avec le message de validation "Ajout de la réponse Ex 3" et pousser. Assurez-vous de valider et de pousser tous les fichiers modifiés afin que votre volet Git soit effacé par la suite.*

**Exercise 4.** Tracez  $y$  par rapport à  $x$  pour l'ensemble de données **circle**. Vous pouvez (et devriez) réutiliser le code que nous avons présenté ci-dessus, remplacez simplement le nom du jeu de données par le jeu de données souhaité. Ensuite, calculez le coefficient de corrélation entre  $x$  et  $y$  pour cet ensemble

de données. Comment cette valeur se compare-t-elle au `r` de `dino` ?

   vous devez faire une pause à nouveau, valider les modifications avec le message de validation “Ajout réponse Ex 4” et pousser. Assurez-vous de valider et de pousser tous les fichiers modifiés afin que votre volet Git soit effacé par la suite.

**Exercice 5.** Enfin, traçons tous les ensembles de données à la fois. Pour ce faire, nous utiliserons le facettage.

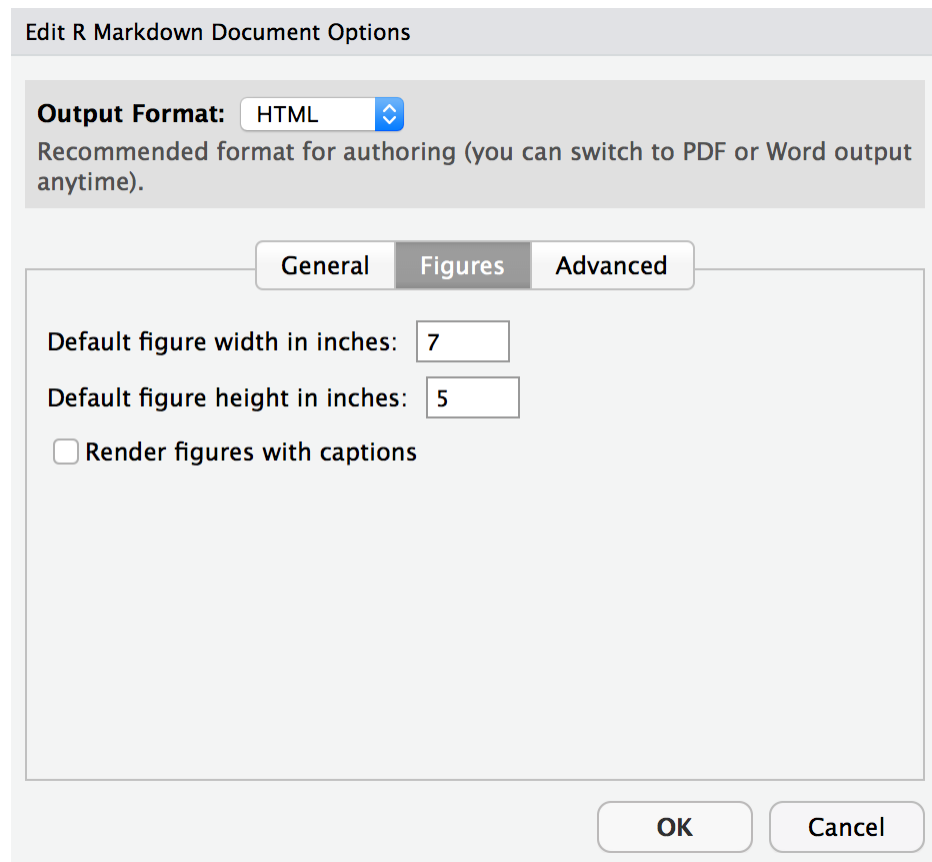
Faire des facettes (facet) par la variable `dataset`, en plaçant les tracés dans une grille à 3 colonnes, et n’ajoutez pas de légende.

```
ggplot(datasaurus_dozen, aes(x = x, y = y, color =  
  dataset)) +  
  geom_point() +  
  facet_wrap(~ dataset, ncol = 3) +  
  theme(legend.position = "none")
```

Et nous pouvons utiliser la fonction `group_by` pour générer tous les coefficients de corrélation:

```
datasaurus_dozen %>%  
  group_by(dataset) %>%  
  summarize(r = cor(x, y)) %>%  
  print(13)
```

Vous avez terminé les exercices d'analyse de données, mais nous aimerions que vous fassiez encore deux choses :

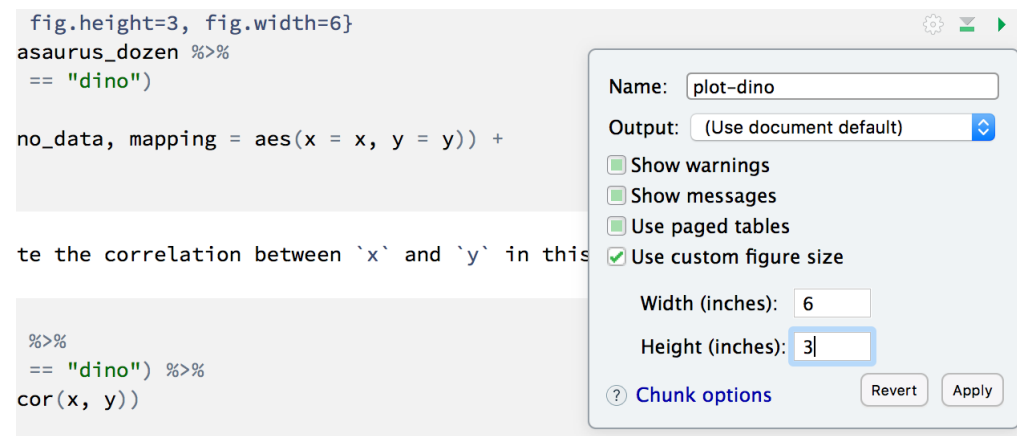


The screenshot shows a dialog box titled "Edit R Markdown Document Options". At the top, there is a section for "Output Format" with a dropdown menu set to "HTML" and a note: "Recommended format for authoring (you can switch to PDF or Word output anytime)". Below this are three tabs: "General", "Figures" (which is selected), and "Advanced". In the "Figures" tab, there are two input fields: "Default figure width in inches:" with the value "7", and "Default figure height in inches:" with the value "5". There is also a checkbox labeled "Render figures with captions" which is currently unchecked. At the bottom right of the dialog are "OK" and "Cancel" buttons.

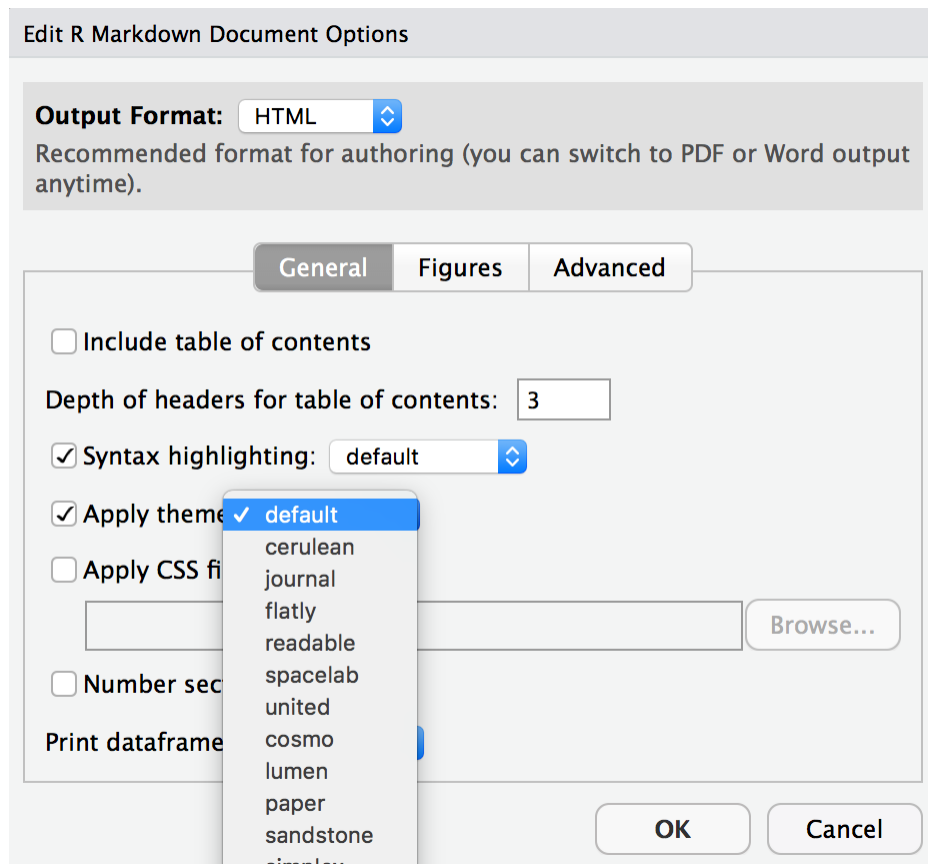
- **Redimensionnez vos figures :**

Cliquez sur l'icône d'engrenage (à droite du bouton knit) en haut du document R Markdown et sélectionnez "Options de sortie..." dans le menu déroulant. Dans la boîte de dialogue contextuelle, accédez à l'onglet Chiffres et modifiez la hauteur et la largeur des chiffres, puis appuyez sur OK

lorsque vous avez terminé. Ensuite, généré votre document et voyez comment vous aimez les nouvelles tailles. Changez et généré encore et encore jusqu'à ce que vous soyez satisfait des tailles de figurines. Notez que ces valeurs sont enregistrées dans le YAML.






Vous pouvez également utiliser différentes tailles de figurines pour différentes figurines. Pour ce faire, cliquez sur l'icône d'engrenage dans le morceau où vous souhaitez apporter une modification. La modification des tailles de figurines a ajouté de nouvelles options à ces morceaux : **fig.width** et **fig.height**. Vous pouvez également les modifier en définissant différentes valeurs directement dans votre document R Markdown.



- **Modifiez l'apparence de votre rapport :**

Cliquez à nouveau sur l'icône d'engrenage en haut du document R Markdown et sélectionnez "Options de sortie..." dans le menu déroulant. Dans l'onglet Général de la boîte de dialogue contextuelle, essayez différentes options de coloration syntaxique et de thème. Appuyez sur OK et générez votre document pour voir à quoi il ressemble. Jouez avec ceux-ci jusqu'à ce que vous soyez satisfait du look.

   *Youpi, vous avez terminé ! Validez toutes les modifications restantes, utilisez le message de validation “Lab 1 terminé! 💪 “. Assurez-vous de valider et de pousser tous les fichiers modifiés afin que votre volet Git soit effacé par la suite. Avant de conclure la mission, assurez-vous que tous les documents sont mis à jour sur votre dépôt GitHub.*

Vous ne savez pas comment utiliser les emojis sur votre ordinateur ? Peut-être qu’un coéquipier peut vous aider ? Ou vous pouvez aussi me demander sur le forum !