Les graphiques de dispersion et lignes de lissage

February 2024

Introduction

Les **diagrammes de dispersion** ou **nuages de points** (en anglais, **scatter plots**) sont des outils visuels puissants permettant d'explorer la **relation** entre deux variables quantitatives.

Ils sont très populaires dans le domaine de la visualisation de données, car ils offrent une vue instantanée de la manière dont une variable quantitative évolue par rapport à une autre.

Les diagrammes de dispersion permettent aussi de représenter plusieurs relations en associant une variable supplémentaire à des propriétés esthétiques, comme la couleur des points.

De plus, les tendances et les relations sur un diagramme de dispersion peuvent être rendues plus claires en ajoutant une ligne de lissage.

Nous allons utiliser ggplot pour faire tout cela et plus encore. C'est parti !

Objectifs d'apprentissage

1. Visualiser les relations entre des variables quantitatives en utilisant des **diagrammes de dispersion** avec `geom_point()`.
2. Utiliser `color` comme argument esthétique pour mapper les variables d'un dataset sur les points.
3. Changer la taille, la forme, la couleur, le remplissage et l'opacité des objets géométriques en définissant des **esthétiques fixes**.
4. Ajouter une **ligne de tendance** à un diagramme de dispersion avec `geom_smooth()`.

Maladies diarrhéiques chez les enfants au Mali

Nous allons utiliser les données recueillies lors d'une étude observationnelle prospective sur la **diarrhée aiguë chez les enfants** âgés de 0 à 59 mois. L'étude a été menée au Mali au début de l'année 2020.

Le dataset complet est disponible sur Dryad, et l'article peut être consulté ici.

Une étude prospective suit l'évolution d'un évènement (par exemple, le développement d'une maladie ou d'autres problèmes de santé) pendant la période de l'étude, et examine son lien avec d'éventuels facteurs de risque ou de protection.

Explorons ce dataset. Il comprend des variables démographiques, physiologiques, cliniques, socioéconomiques et géographiques. Chaque ligne correspond à un patient interrogé.

Nous allons commencer par visualiser la relation entre les deux variables quantitatives suivantes :

1. `age_months` : l'**âge** du patient en mois sur l'axe des **x** et
2. `viral_load` : la **charge virale** du patient sur l'axe des **y**.

Graphiques de dispersion avec `geom_point()`

Nous allons explorer la relations entre des variables quantitatives du dataframe **`malidd`**.

Commençons par exécuter le code pour créer le graphique de dispersion souhaité, tout en gardant à l'esprit le cadre GG. Examinons le code et décomposons-le couche par couche.

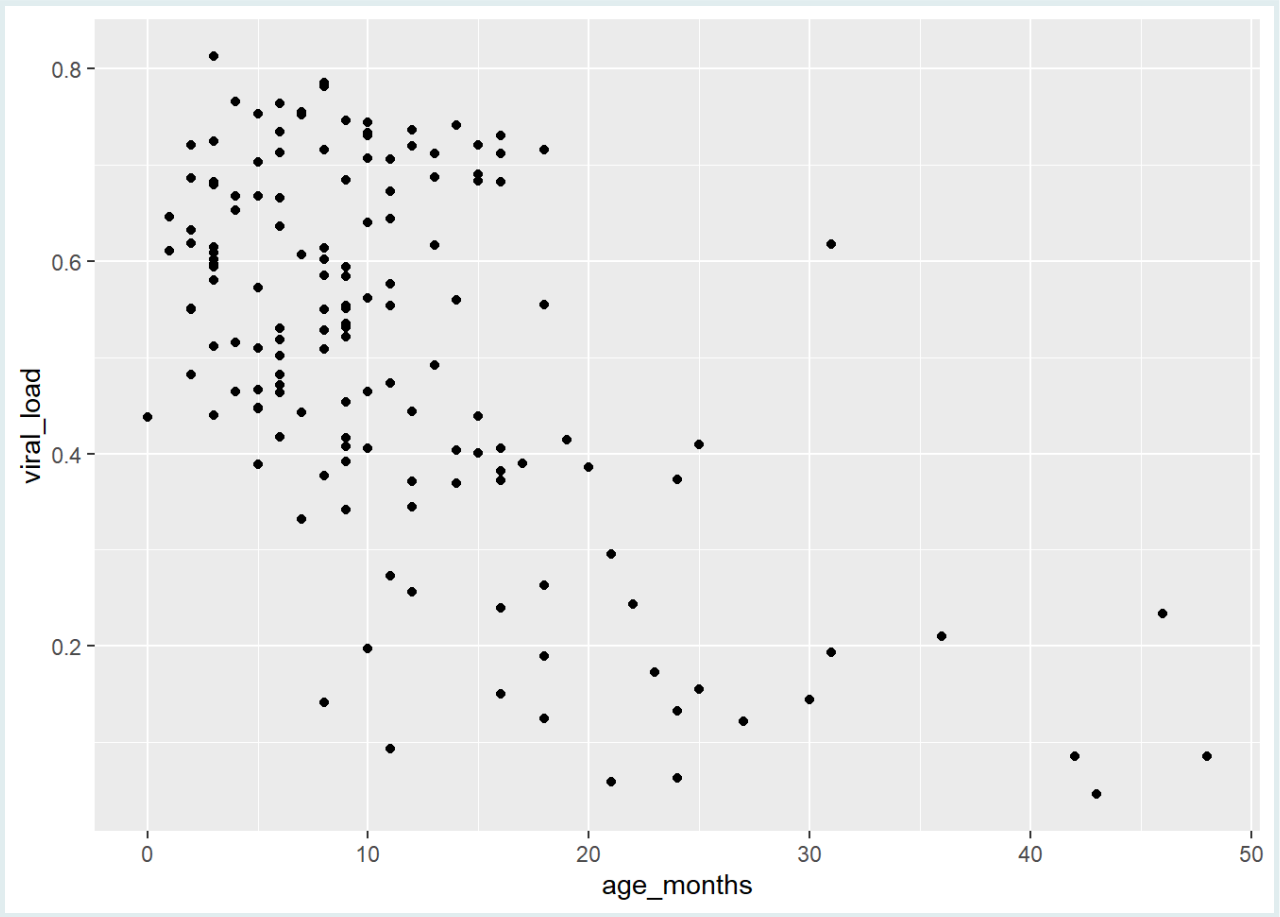
N'oubliez pas d'inclure les deux premières couches à l'intérieur de la fonction `ggplot()` :

1. Nous utilisons le dataframe `malidd` comme argument **`data`**, en spécifiant **`data = malidd`**.
2. Dans la fonction `aes`thetics de l'argument **`mapping`**, nous définissons les variables à représenter en utilisant **`mapping = aes(x = age_months, y = viral_load)`**. Plus précisément, la variable `age_months` est associée à l'axe des `x`, tandis que la variable `viral_load` est associée à l'axe des `y`.

Ensuite, nous ajoutons une nouvelle couche à l'aide de l'opérateur `+` et **la fonction `geom_*()`**. Pour un graphique de dispersion, les objets géométriques nécessaires sont des points, donc nous ajoutons **`geom_point()`**.

En exécutant le code suivant, vous obtiendrez le graphique de dispersion illustré ci-dessous :

```
# Graphique de dispersion basique de la charge virale en fonction de l'âge
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point()
```



Cela suggère que la charge virale **diminue** généralement avec l'âge.

- En utilisant le dataframe `malidd`, créez un graphique de dispersion montrant la relation entre l'âge et la taille (`height_cm`).

Modifier les esthétiques

Une esthétique est une propriété visuelle des objets géométriques (`geom`) de votre graphique. Les esthétiques incluent entre autres la taille, la forme ou la couleur de vos points. Vous pouvez présenter un point de différentes manières en modifiant les valeurs de ses propriétés esthétiques.

Rappelez-vous, il y a deux manières de modifier les propriétés esthétiques de vos éléments géométriques `geom` (ici, des points) :

1. **Mapper les esthétiques aux variables de vos données**, en utilisant la fonction `aes()` dans l'argument `mapping` pour associer le nom de l'esthétique à une variable à afficher.

2. **Définir manuellement les esthétiques** en spécifiant l'esthétique en tant qu'argument de la fonction `geom_*()`, *en dehors* de `aes()`. Dans ce cas, l'esthétique ne véhicule pas d'information sur une variable, mais sert uniquement à modifier l'apparence du graphique.

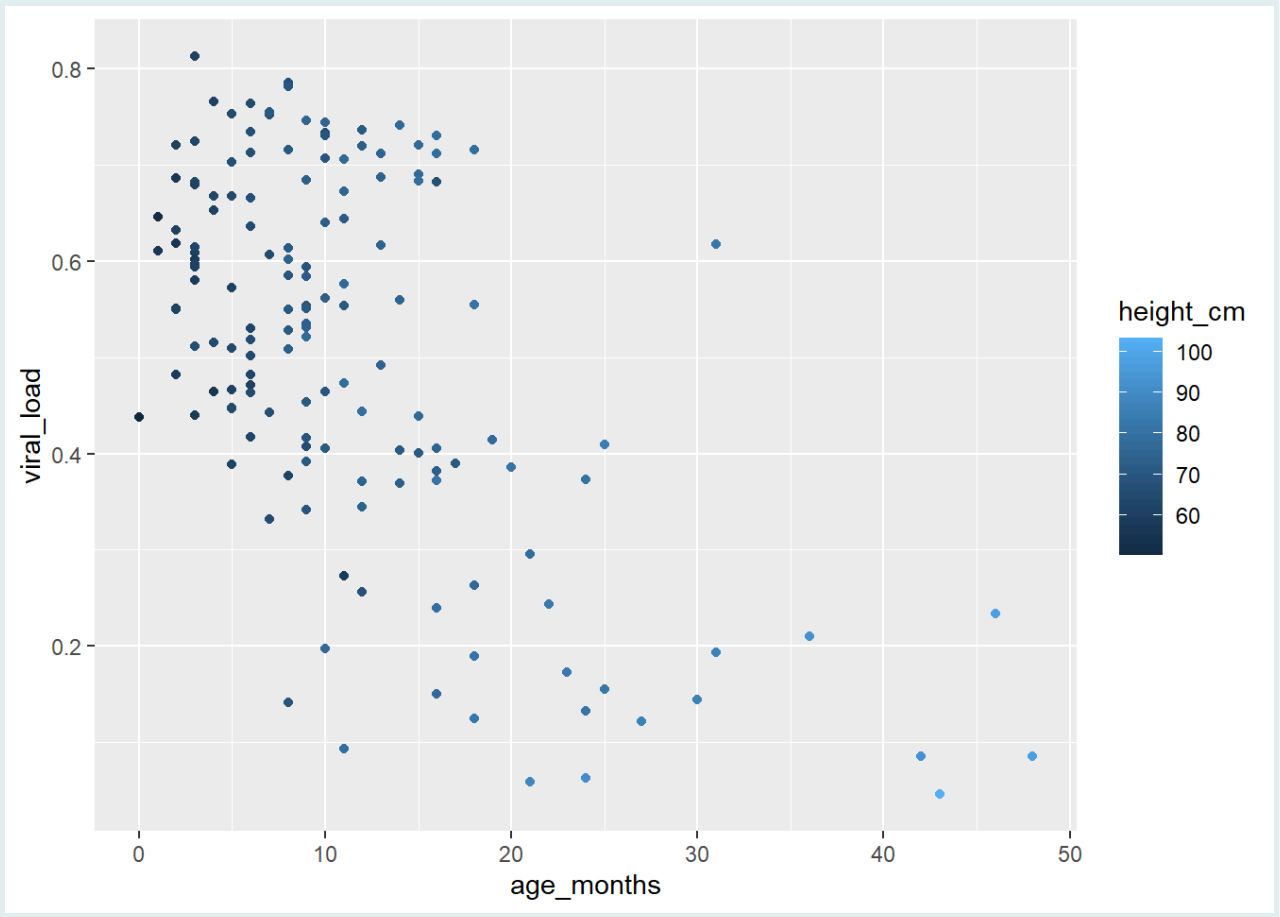
Mapping des données aux esthétiques

En plus de mapper des variables aux axes des **x** et des **y** comme nous l'avons fait précédemment, les variables peuvent être mappées à la couleur, à la forme, à la taille, à l'opacité et à d'autres caractéristiques visuelles du `geom`. Cela permet de superposer plusieurs observations sur un seul graphique.

Pour mapper une variable à une esthétique, il faut associer le nom de l'esthétique au nom de la variable à l'intérieur de `aes()`. De cette façon, vous pouvez visualiser une troisième variable dans un graphique de dispersion à deux dimensions en la mappant à une nouvelle esthétique.

Par exemple, supposons que nous voulons montrer comment la taille (`height_cm`) varie avec l'âge et la charge virale. Nous pouvons mapper la variable `height_cm` à la couleur de nos points de la manière suivante :

```
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(mapping = aes(color = height_cm))
```



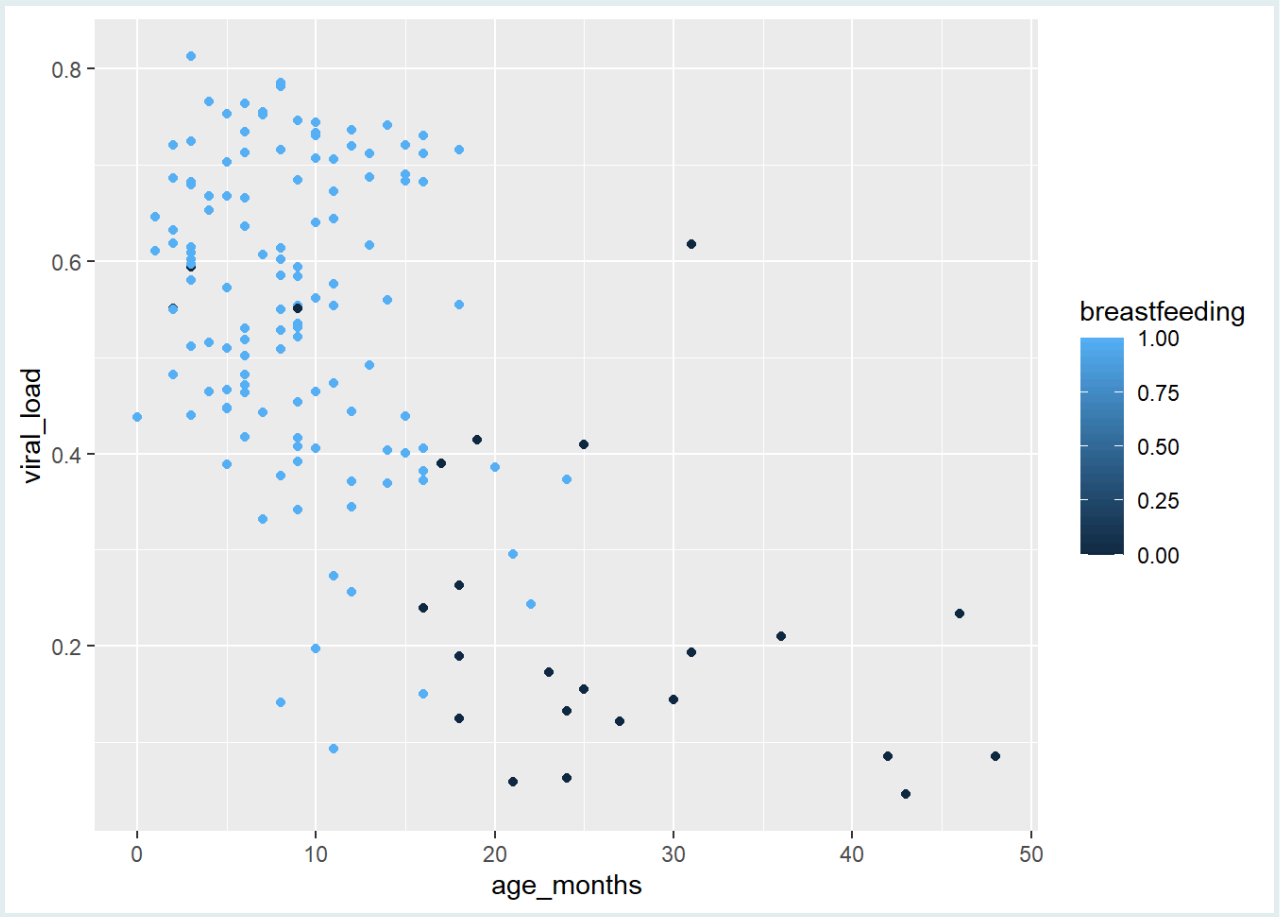
Comme vous pouvez le constater, {ggplot2} a automatiquement assigné les valeurs de notre variable à une esthétique, un processus connu sous le nom de **scaling**. {ggplot2} ajoutera également une légende pour expliquer quelles couleurs correspondent à quelles valeurs.

Dans cet exemple, les points sont colorés par différentes nuances de bleu, avec des couleurs plus foncées représentant des valeurs plus basses.

Cela nous indique que la taille augmente avec l'âge, comme on pouvait s'y attendre.

Au lieu d'une variable continue comme `height_cm`, nous pouvons également mapper une variable binaire comme `breastfeeding`, pour indiquer les enfants qui sont allaités et ceux qui ne le sont pas :

```
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(mapping = aes(color = breastfeeding))
```



Nous obtenons la même échelle de couleurs que celle obtenue avec la variable `height_cm`. Cette échelle de couleurs suggère une variabilité continue, ce qui n'est pas idéal dans notre cas, car notre variable `breastfeeding` ne peut prendre que deux valeurs distinctes : 0 ou 1.

Cela est dû à la classe de la variable `breastfeeding` dans `malidd` :

```
class(malidd$breastfeeding)
```

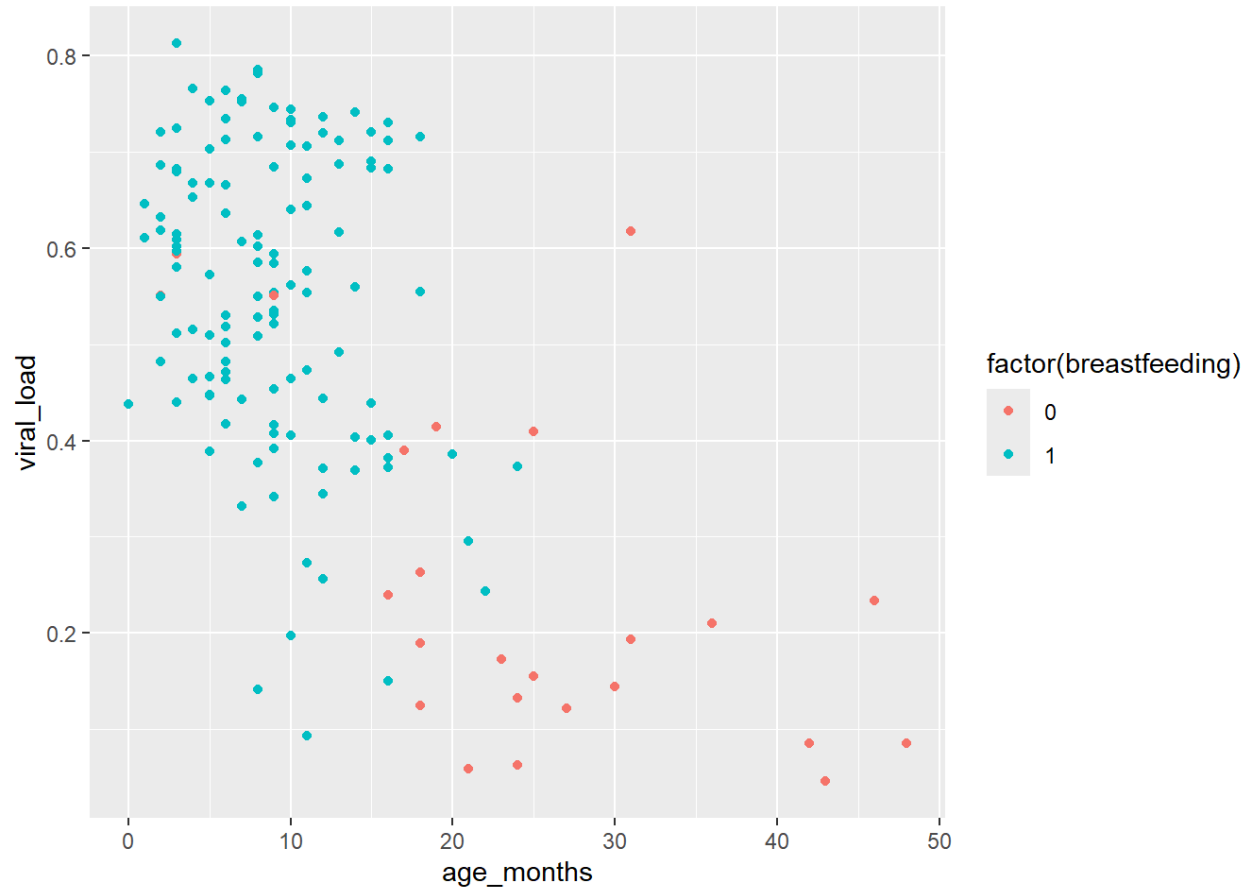
```
## [1] "numeric"
```

Même si cette variable est numérique, elle ne peut prendre que deux valeurs. Par conséquent, une échelle de couleurs continue n'est pas la représentation la plus appropriée pour ce cas.

Dans des cas comme celui-ci, nous ajoutons la fonction `factor()` autour de la variable `breastfeeding` pour dire à `ggplot()` de traiter la variable comme un facteur. Voyons ce qui se passe lorsque nous faisons cela :

```
ggplot(data = malidd,
       mapping = aes(x = age_months,
```

```
                      y = viral_load)) +
  geom_point(mapping = aes(color = factor(breastfeeding)))
```



Lorsque la variable est traitée comme un facteur, les couleurs attribuées sont nettement distinctes. Avec les facteurs, {ggplot2} attribue automatiquement un niveau unique de l'esthétique (dans ce cas, une couleur unique) à chaque valeur de la variable. C'est ce qui s'est produit avec la variable `region` du dataframe `malidd` que nous avons utilisé dans la leçon précédente.

Ce graphique met en évidence une relation claire entre l'âge et l'allaitement maternel, comme on pouvait s'y attendre. Les enfants sont susceptibles d'arrêter l'allaitement vers 20 mois. Dans cette étude, aucun enfant de 25 mois ou plus n'était allaité.

L'ajout de couleurs à notre diagramme de dispersion nous a permis de visualiser une **troisième variable** en plus de la relation entre l'âge et la charge virale. Cette troisième variable peut être soit discrète (comme dans le cas de l'allaitement), soit continue.

- Créez un graphique de dispersion à partir du dataframe `malidd`, montrant la relation entre l'âge et la charge virale, et mappez

une troisième variable `freqrespi` à la couleur.

- Recréez le même graphique de dispersion âge vs. taille, mais cette fois, mappez la variable binaire `fever` à la couleur des points. N'oubliez pas que `fever` doit être traité comme un facteur.

Définir les esthétiques fixes

Les esthétiques fixes sont constantes sur tout le graphique et ne varient pas en fonction des données. Pour ajouter une esthétique fixe, nous l'ajoutons comme argument direct de la fonction `geom_*()` *en dehors* de `mapping = aes()`.

Voici certains des arguments esthétiques que nous pouvons placer directement dans `geom_point()` pour apporter des modifications visuelles aux points de notre graphique de dispersion :

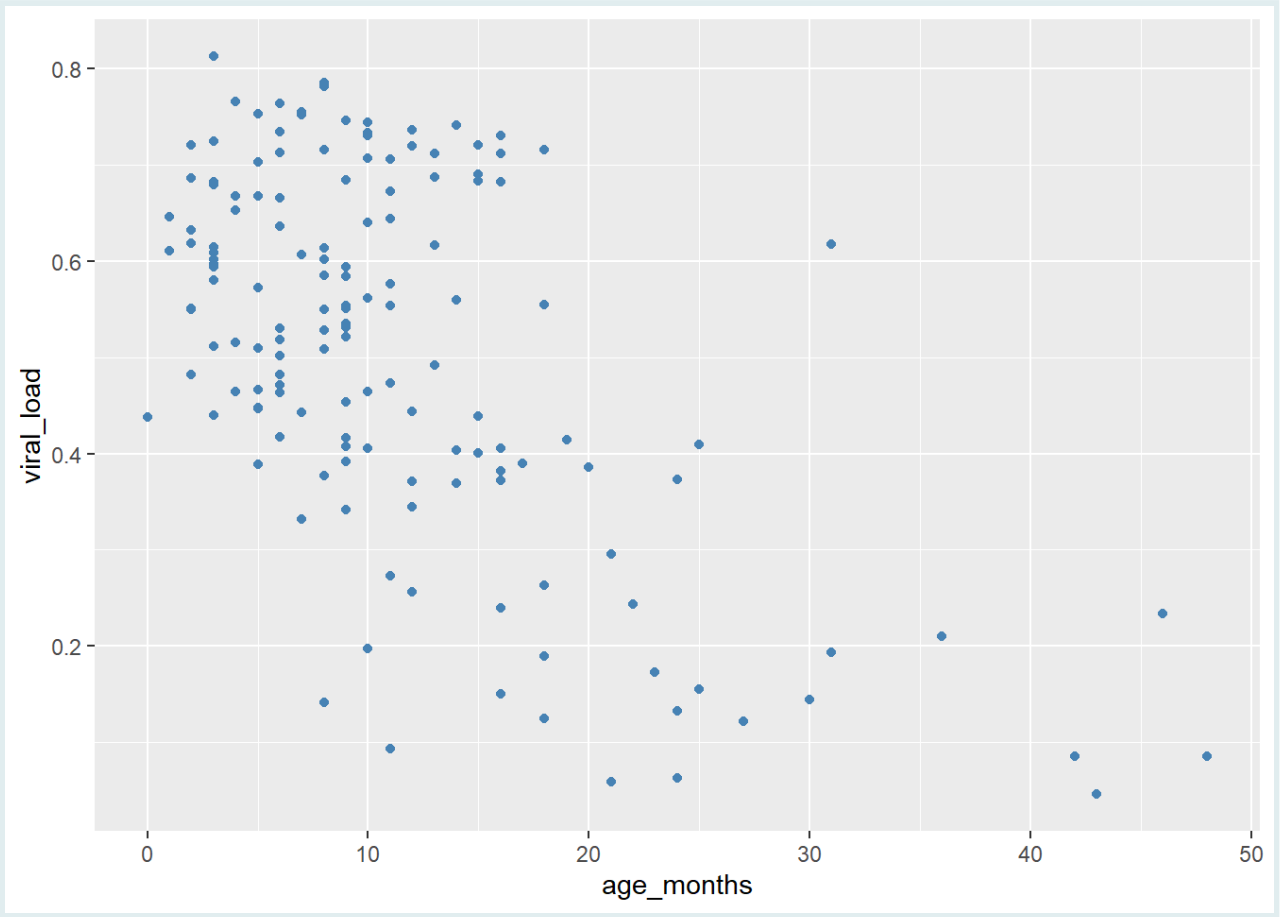
- `color` - couleur du point ou couleur du contour du point
- `size` - taille du point
- `alpha` - opacité du point
- `shape` - forme du point
- `fill` - couleur de remplissage du point (ne s'applique que si le point a un contour)

Pour utiliser ces options afin de créer un graphique de dispersion plus attractif, vous devrez choisir une valeur appropriée pour chaque argument esthétique, comme le montrent les exemples ci-dessous.

Modifier `color`, `size` et `alpha`

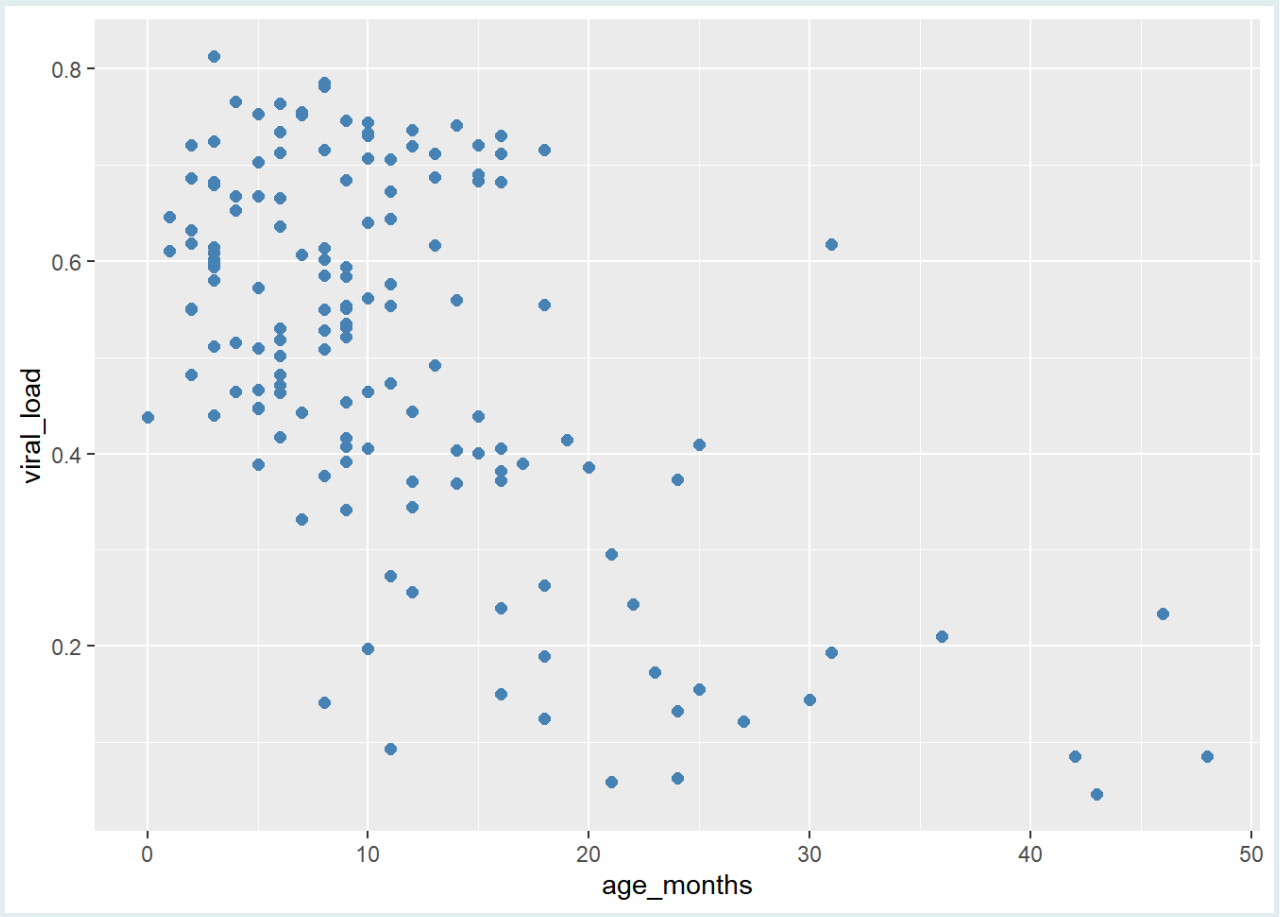
Pour modifier la couleur des points en leur attribuant une valeur fixe, il faut positionner l'argument `color` directement dans `geom_point()`. La couleur que nous choisissons doit être une chaîne de caractères que R reconnaît comme une couleur. Pour définir la couleur des points en bleu acier ("steelblue"), vous pouvez utiliser le code suivant :

```
#  Modifier le graphique de dispersion d'origine en définissant `color =
       "steelblue"`
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(color = "steelblue")        # définir la couleur
```



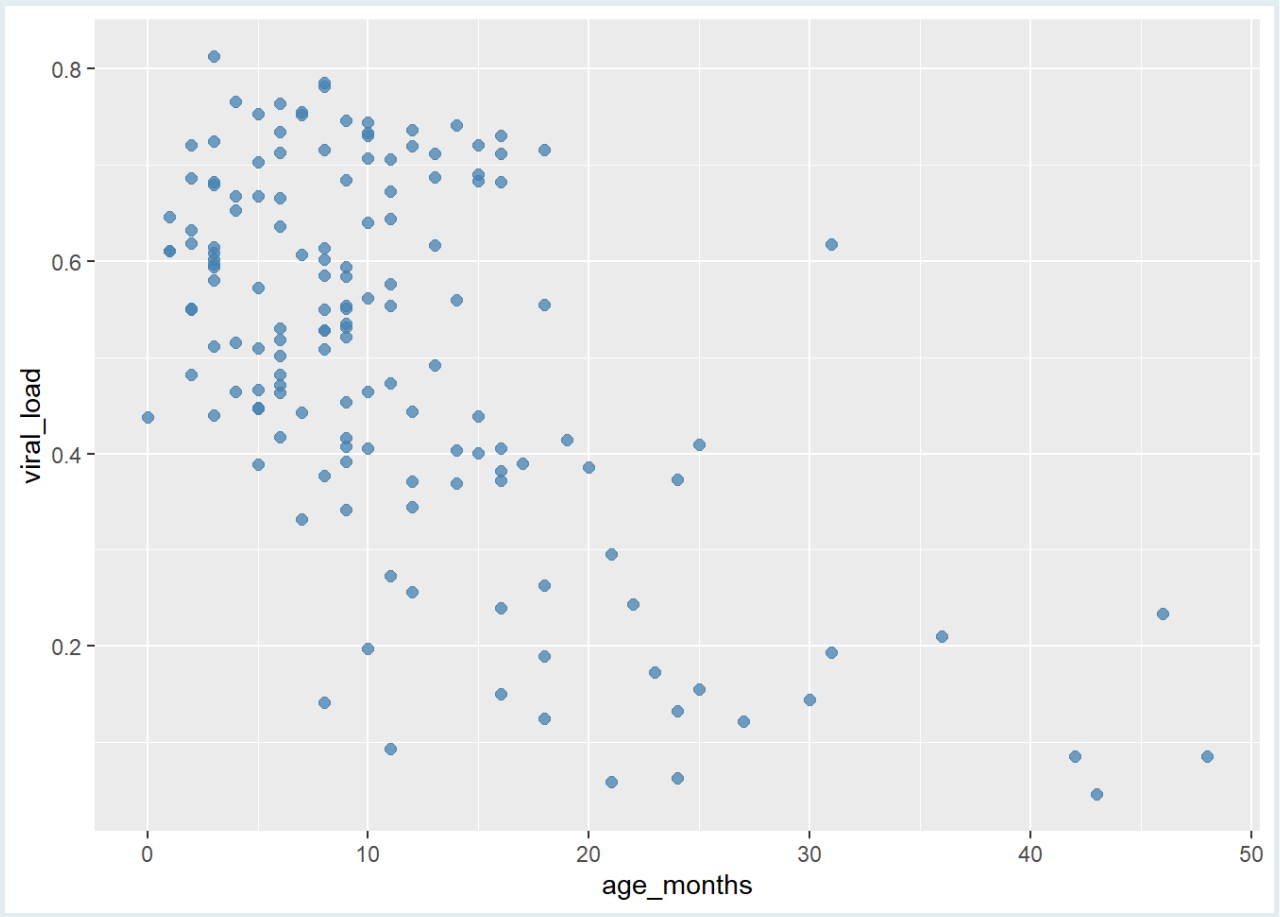
En plus de changer la couleur par défaut, nous allons maintenant modifier l'esthétique `size` des points en lui attribuant un nombre fixe (en millimètres). La taille par défaut est de 1 mm, nous allons donc choisir une valeur plus grande :

```
#  Définir la taille à 2 mm en ajoutant `size = 2`
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(color = "steelblue",        # définir la couleur
             size = 2)                   # définir la taille (mm)
```



L'esthétique `alpha` contrôle le niveau d'opacité des `geom`. `alpha` est une valeur numérique qui varie de 0 (complètement transparent) à la valeur par défaut de 1 (complètement opaque). Nous allons réduire l'opacité des points en diminuant la valeur de `alpha` :

```
#  Définir l'opacité à 75% en ajoutant `alpha = 0.75`
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(color = "steelblue",        # définir la couleur
             size = 2,                   # définir la taille (mm)
             alpha = 0.75)               # définir le niveau d'opacité
```



En rendant les points semi-transparents, nous pouvons voir où les points se chevauchent. C'est particulièrement utile pour les graphiques de dispersion où il y a beaucoup de **superposition** de points.

Souvenez-vous, changer la couleur, la taille, ou l'opacité de nos points ici ne transmet aucune information supplémentaire sur les données - ce sont des choix de conception que nous faisons pour rendre nos graphiques plus esthétiques.

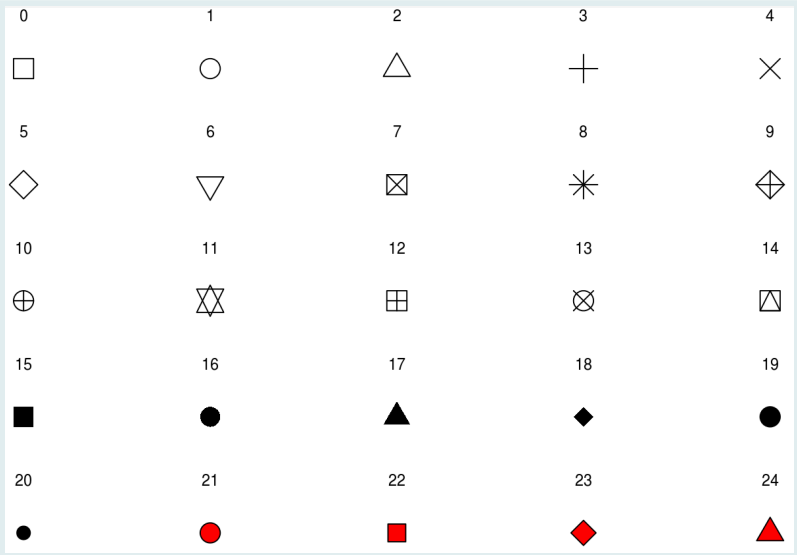
- Créez un graphique de dispersion avec les mêmes variables que dans l'exemple précédent, mais changez la couleur des points à `cornflowerblue`, augmentez la taille des points à 3 mm et réglez l'opacité à 60%.

Modifier `shape` et `fill`

Nous pouvons modifier l'apparence des points d'un graphique de dispersion avec l'esthétique `shape`.

Pour fixer la forme de vos `geom` à une valeur spécifique, vous pouvez définir l'esthétique `shape` au nombre correspondant à la forme souhaitée.

Dans {ggplot2}, les formes des points sont définies par les numéros 0 à 24 :

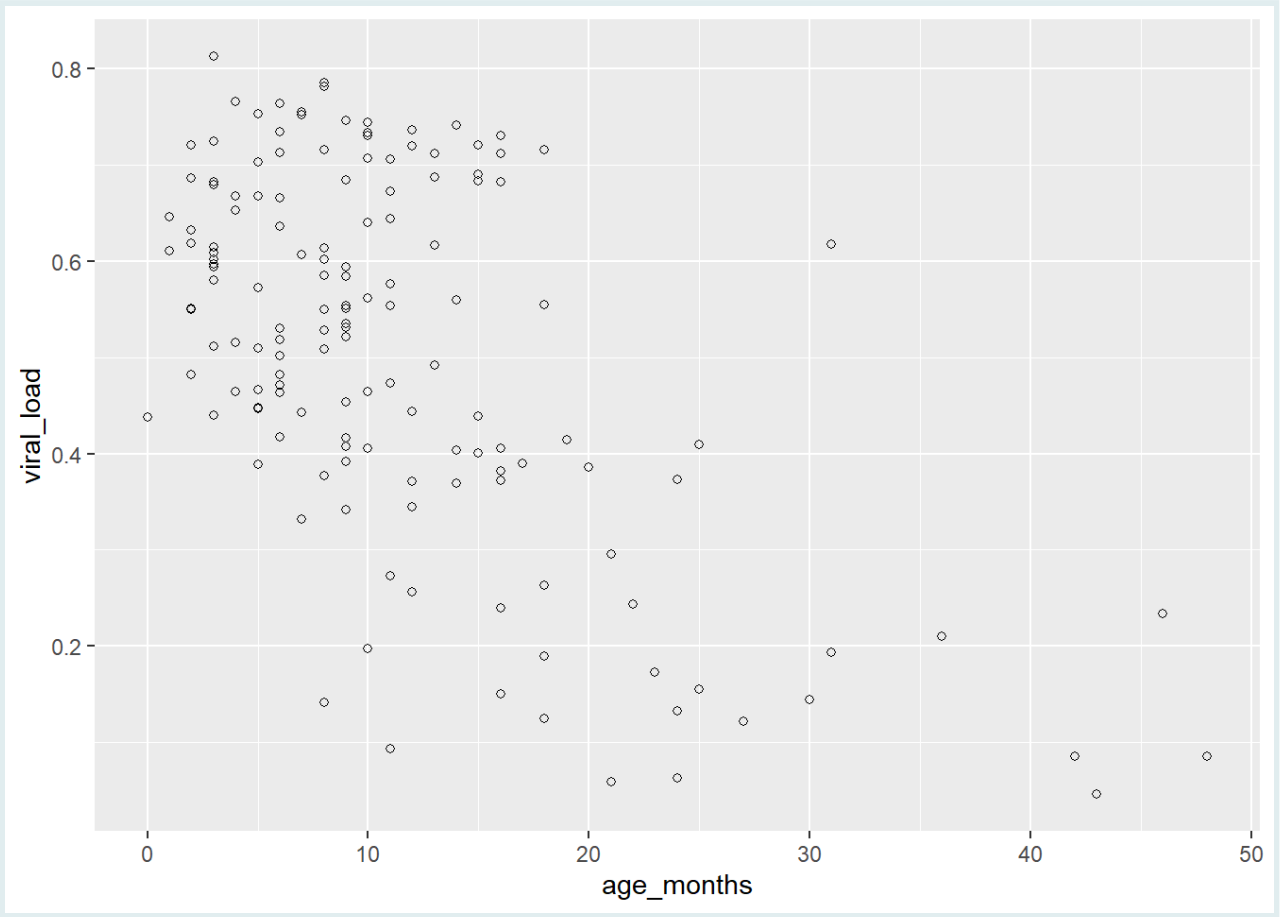


Notez que les formes 21 à 24 peuvent être remplies (`fill`) et avoir une couleur de contour (`color`), tandis que les autres formes ne sont sensibles qu'à la couleur de contour `color`.

Tout d'abord, modifions notre graphique de dispersion d'origine en choisissant une forme de point remplissable :

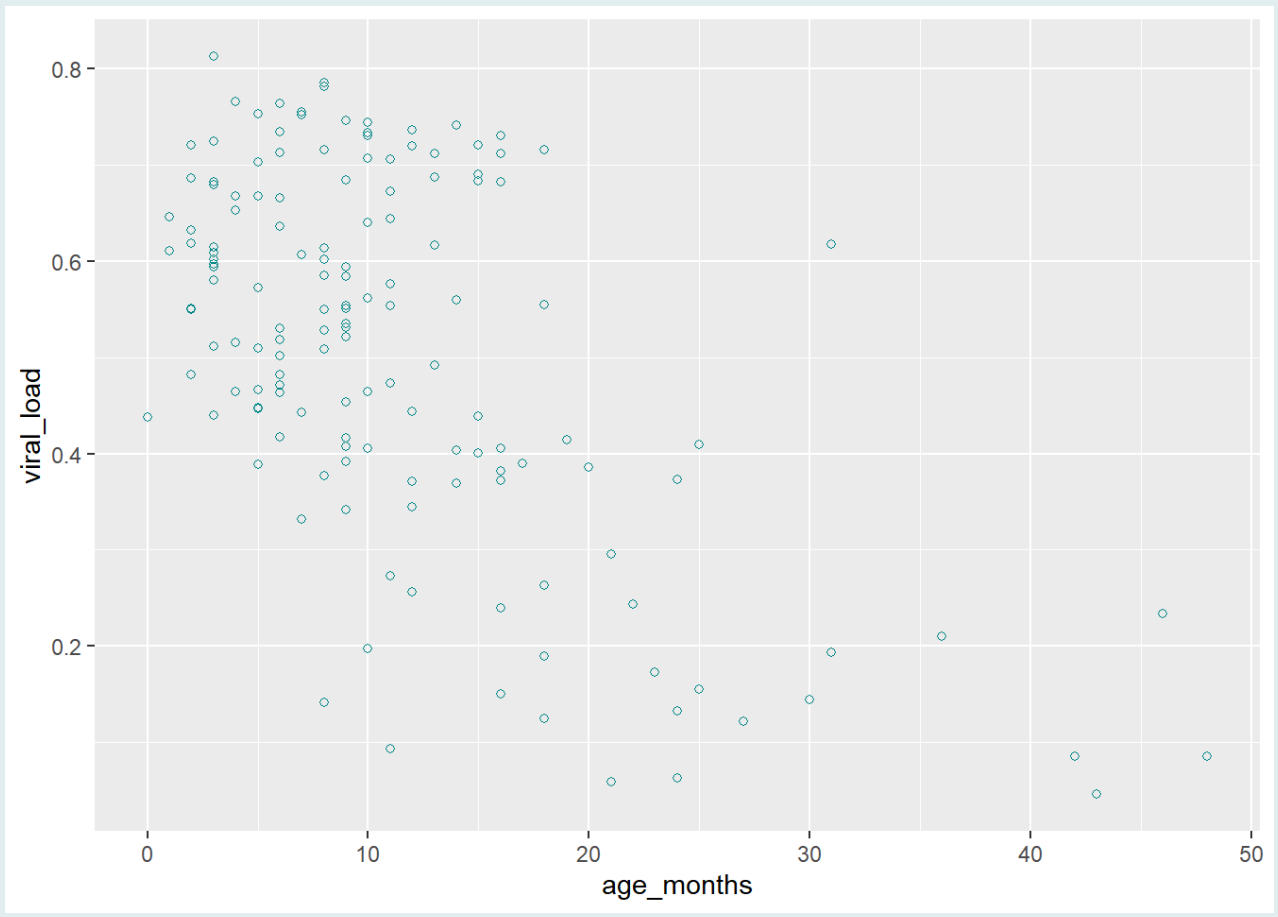
```
# Définir la forme à des cercles remplissables en ajoutant `shape = 21`

ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(shape = 21)                # définir la forme
```



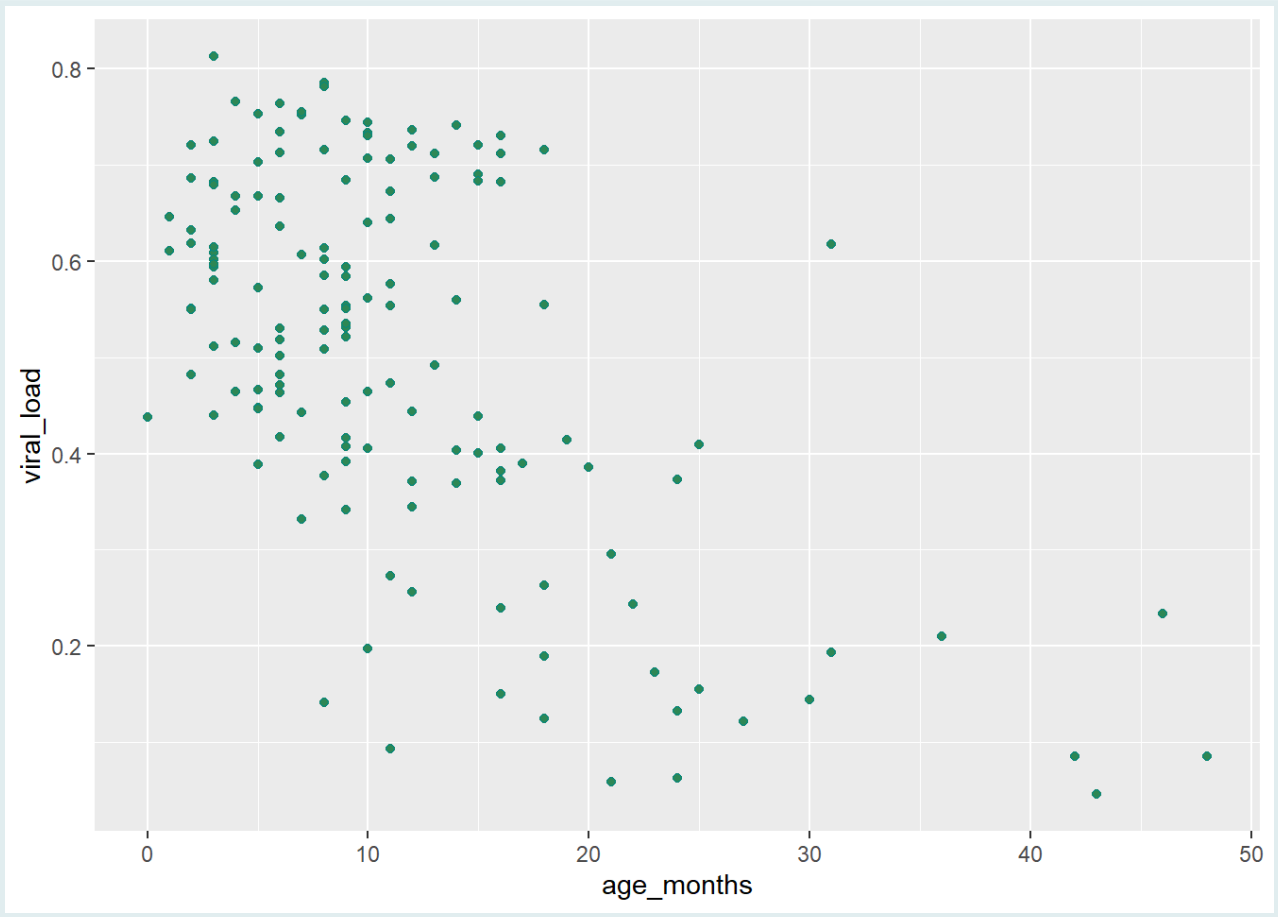
Les formes remplissables peuvent avoir des couleurs différentes pour le contour et l'intérieur. Modifier l'esthétique `color` ne changera que le contour de nos points :

```
# Définir la couleur du contour des formes en ajoutant `color = cyan4`
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(shape = 21,                # définir la forme
             color = "cyan4")           # définir la couleur du contour
```



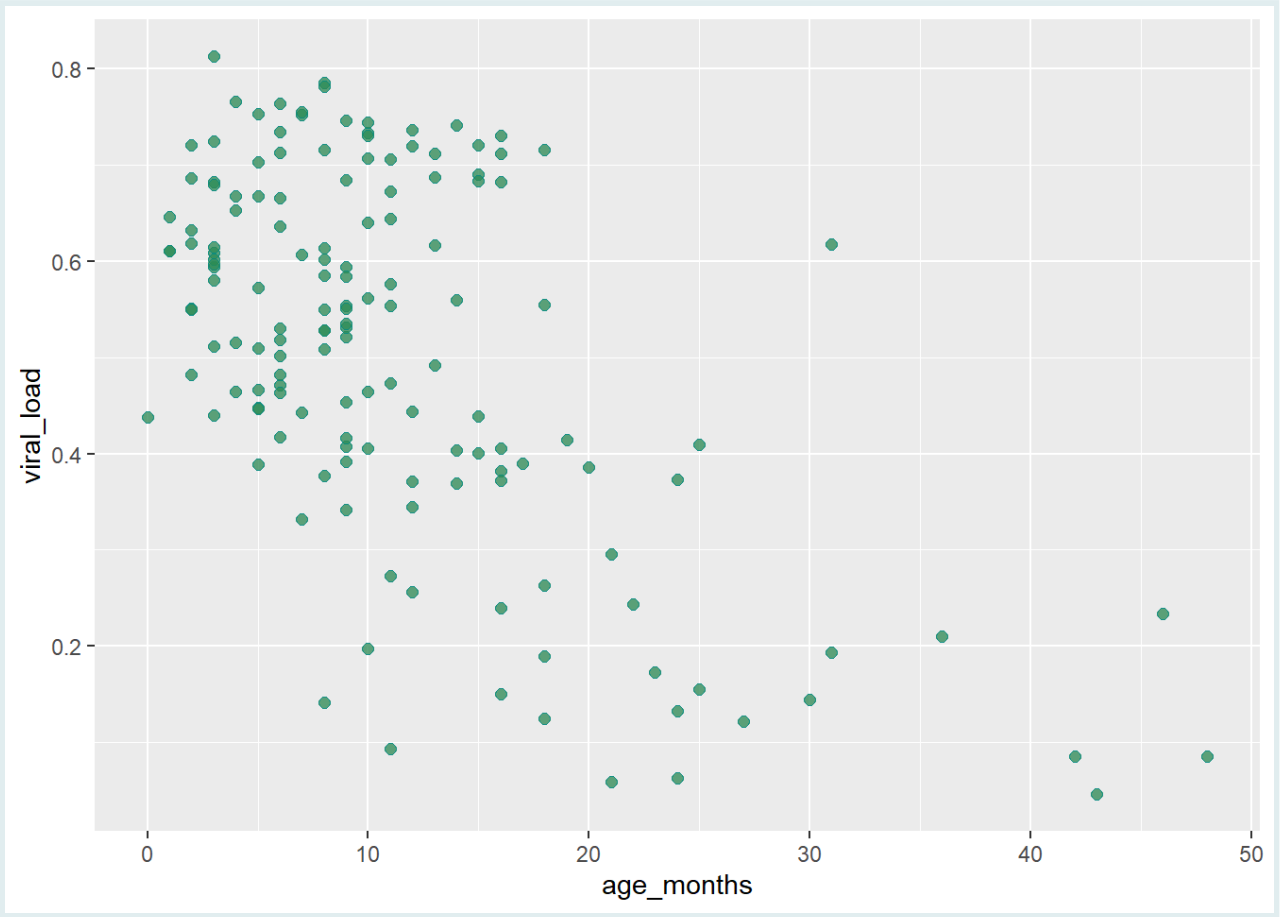
Maintenant, remplissons les points :

```
# Définir la couleur intérieure des formes en ajoutant `fill = "seagreen"`
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(shape = 21,                # définir la forme
             color = "cyan4",           # définir la couleur du contour
             fill = "seagreen")         # définir la couleur de remplissage
```



Nous pouvons améliorer la lisibilité en augmentant la taille et en réduisant l'opacité avec `size` et `alpha`, comme nous l'avons fait précédemment :

```
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(shape = 21,                # définir la forme
             color = "cyan4",           # définir la couleur du contour
             fill = "seagreen",         # définir la couleur de remplissage
             size = 2,                  # définir la taille (mm)
             alpha = 0.75)              # définir le niveau d'opacité
```



Ajouter une ligne de tendance

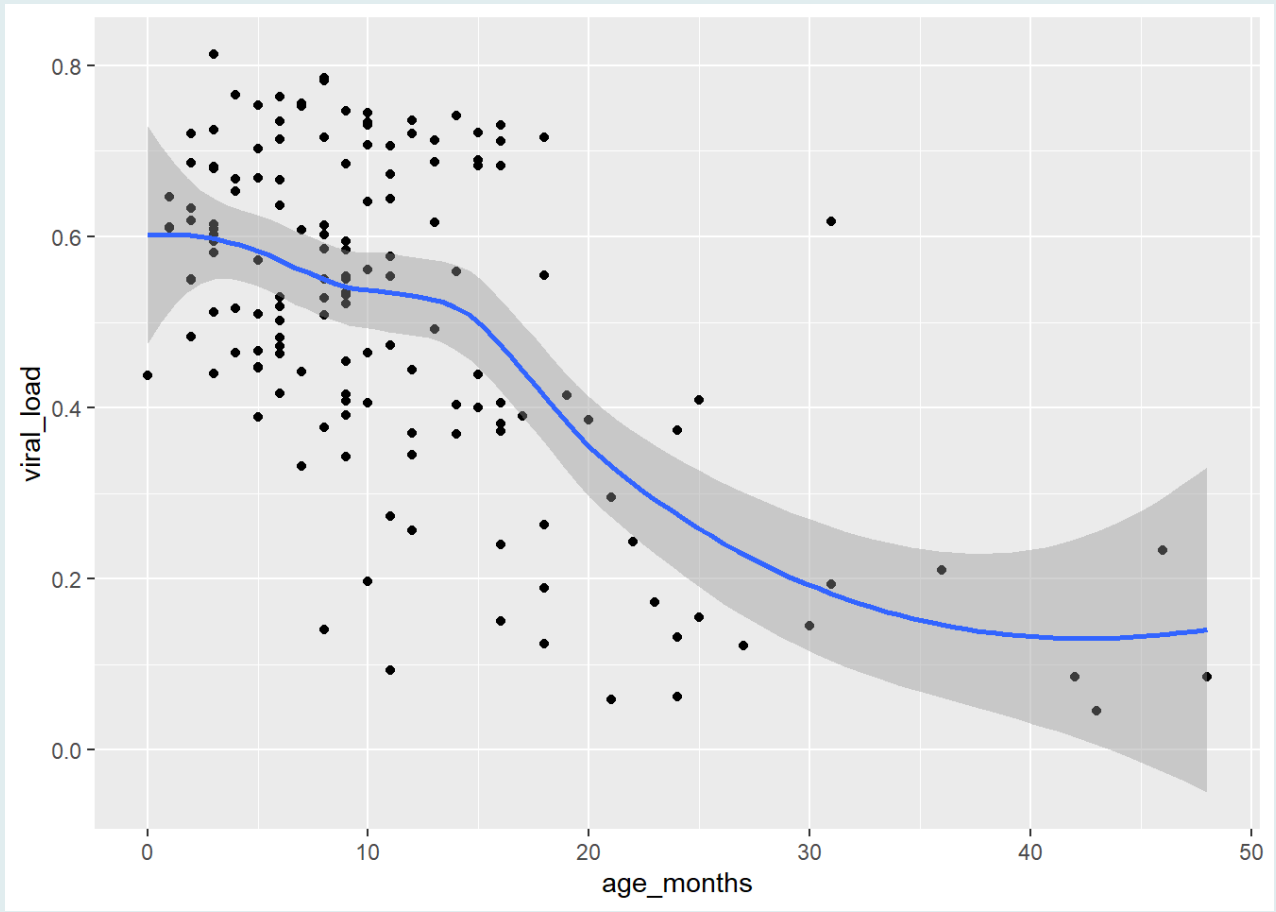
Il peut être difficile de discerner à première vue des relations ou des tendances en se basant uniquement sur les points du graphique. Une solution courante pour améliorer la lisibilité de ces tendances consiste à ajouter une ligne de lissage. Ceci est particulièrement utile lorsque nous cherchons à comprendre les relations de régression entre les variables.

Pour avoir une meilleure idée de la relation entre ces deux variables, nous pouvons ajouter une ligne de tendance (également connue sous le nom de ligne de meilleure adéquation ou ligne de lissage).

Pour ce faire, nous ajoutons la fonction `geom_smooth()` à notre code :

```
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~
## x'
```



La ligne de lissage est ajoutée à notre graphique comme une autre couche géométrique et vient après les points.

La fonction de lissage par défaut utilisée dans ce graphique de dispersion est “loess” qui signifie **l**ocally **w**eighted **scatter plot s**moothing. Le lissage loess est une technique couramment employée par de nombreux logiciels de statistiques. Dans {ggplot2}, l’usage de loess est généralement recommandé lorsque vous avez moins de 1000 points, car son calcul prend beaucoup de temps.

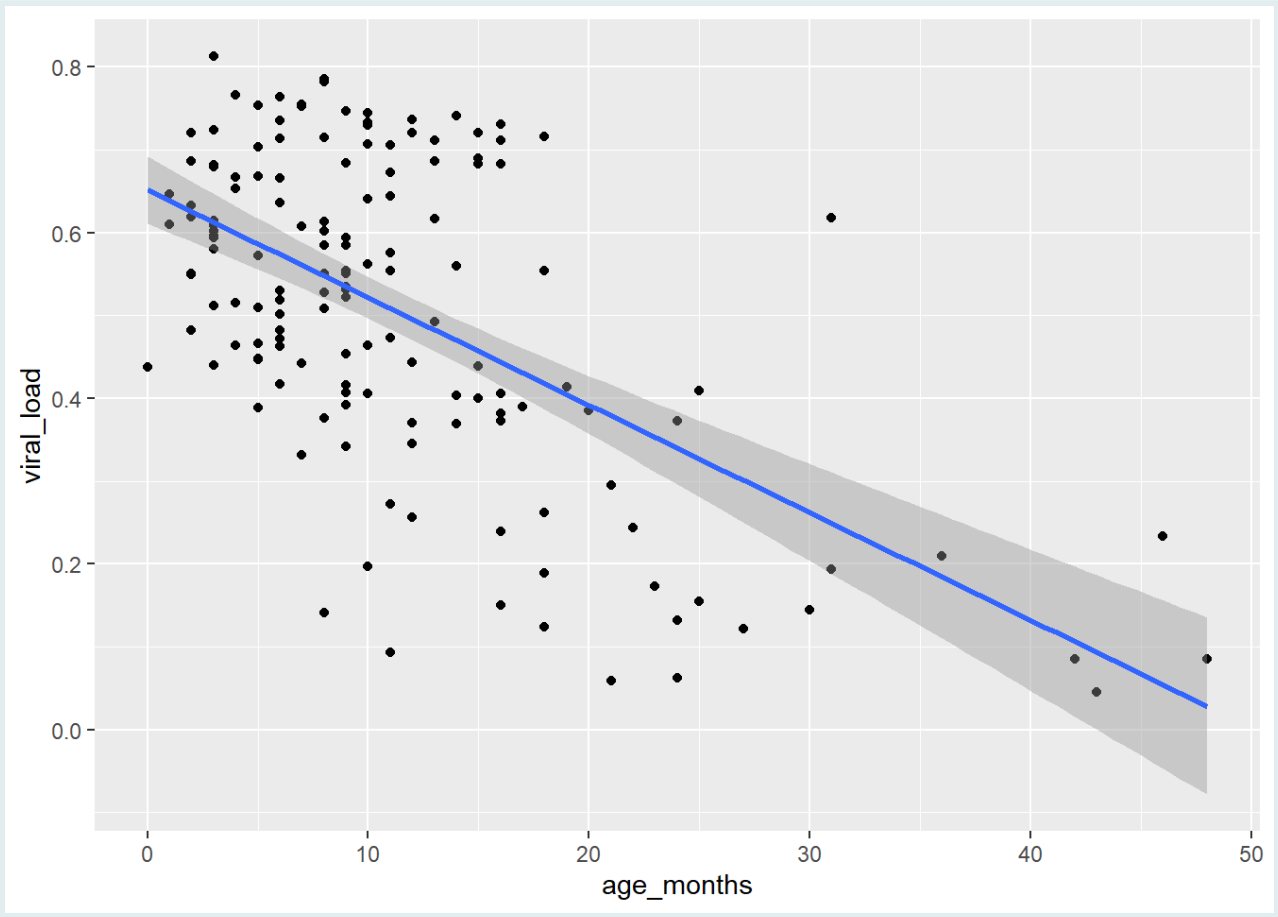
De nombreuses fonctions de lissage peuvent également être utilisées dans `geom_smooth()`.

Cette fois-ci, utilisons plutôt une méthode de régression linéaire. Nous allons opter pour un modèle linéaire généralisé. Pour ce faire, nous définissons l’argument `method` à l’intérieur de `geom_smooth()` :

```
# Changer pour une fonction de lissage par régression linéaire avec `method =
        "glm"`
ggplot(data = malidd,
```

```
        mapping = aes(x = age_months,
                      y = viral_load)) +   geom_point() +
  geom_smooth(method = "glm")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

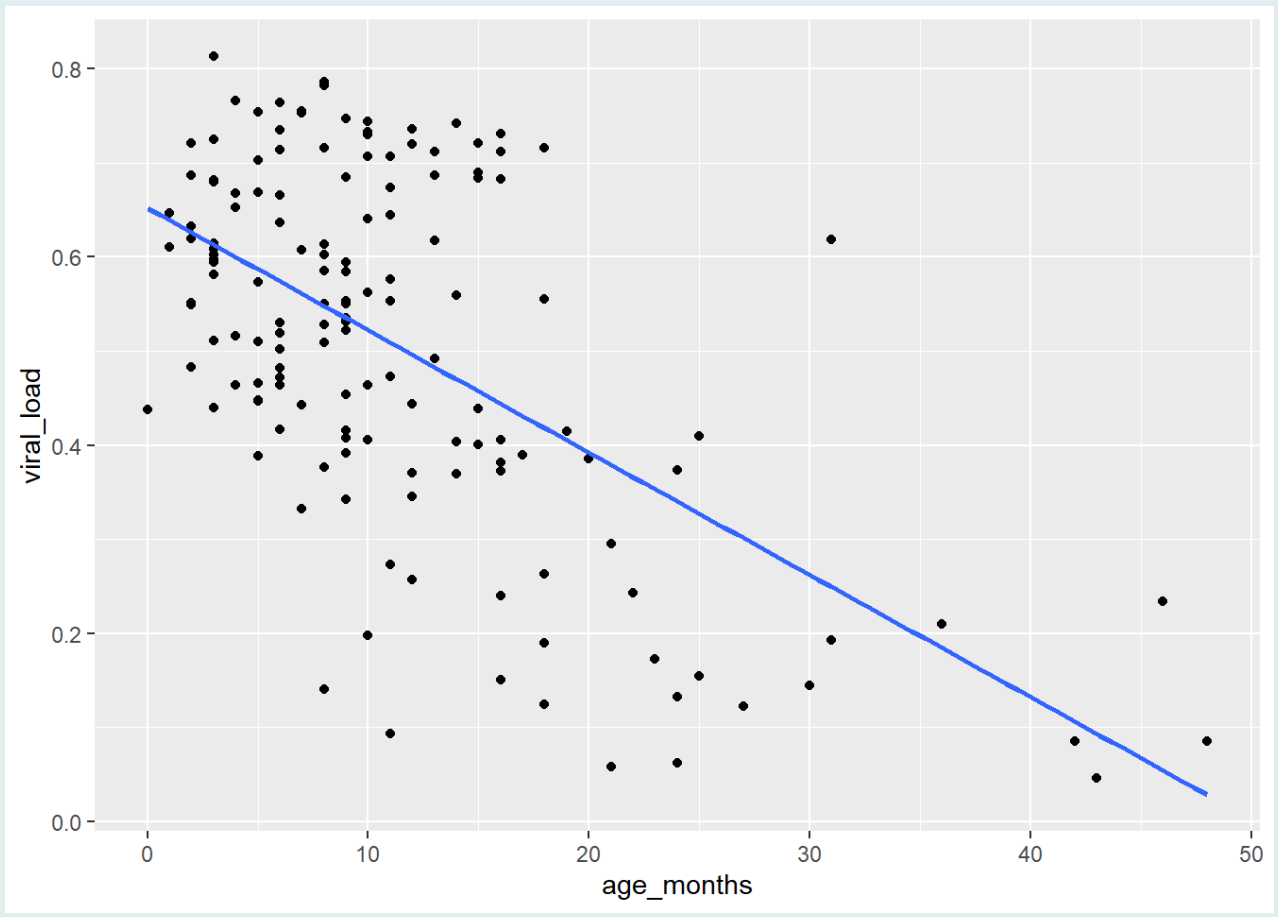


Par défaut, des intervalles de confiance à 95% sont affichés autour de ces lignes de lissage.

Pour supprimer ces intervalles, vous pouvez ajouter l'argument `se = FALSE` à l'intérieur de `geom_smooth()` :

```
# Supprimer les intervalles de confiance en ajoutant `se = FALSE`
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point() +
  geom_smooth(method = "glm",
              se = FALSE)
```

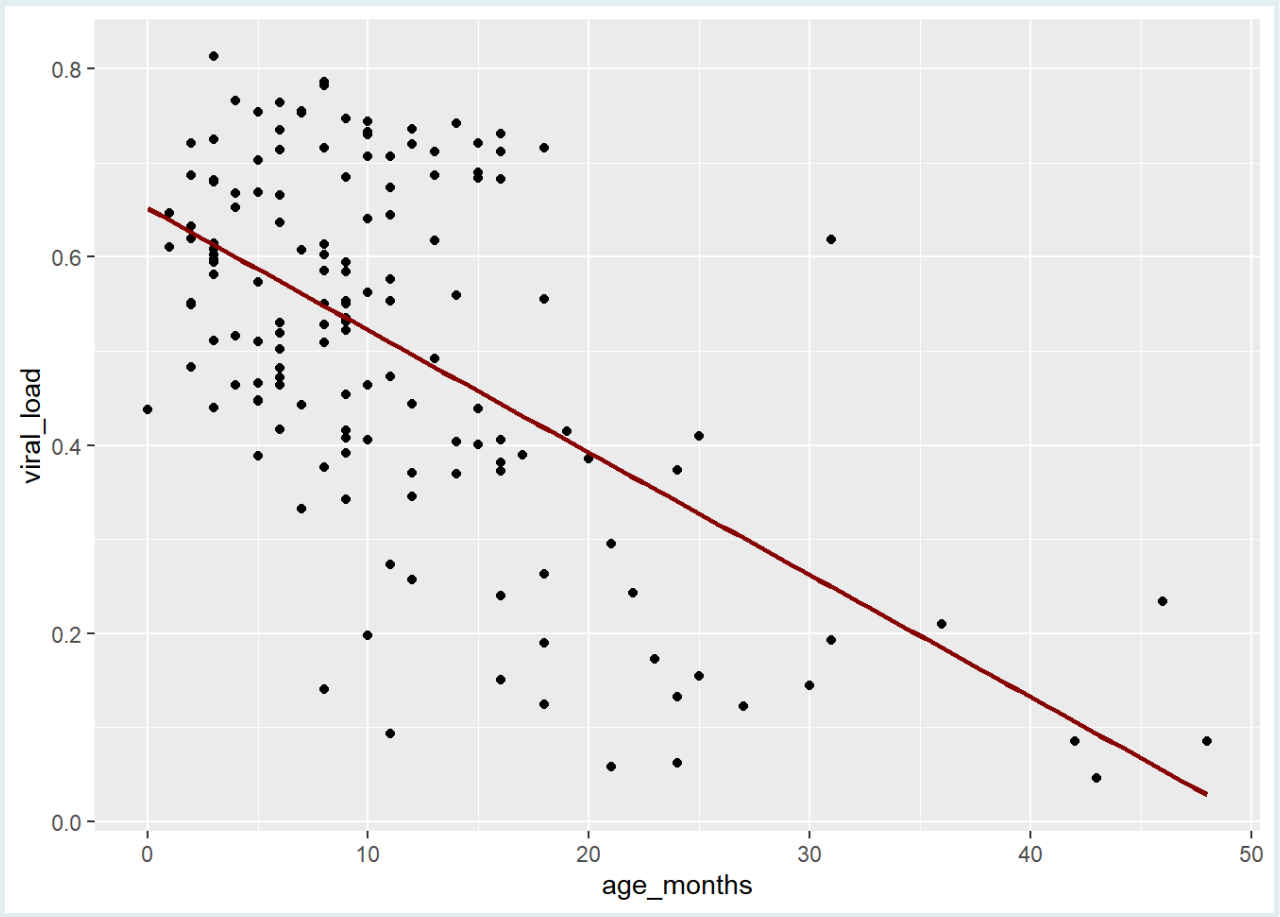
```
## `geom_smooth()` using formula = 'y ~ x'
```



Ajoutons aussi l'argument `color` à l'intérieur de `geom_smooth()` pour changer la couleur de la ligne :

```
# Changer la couleur de la ligne de tendance en ajoutant `color = "darkred"`
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point() +
  geom_smooth(method = "glm",
              se = FALSE,
              color = "darkred")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

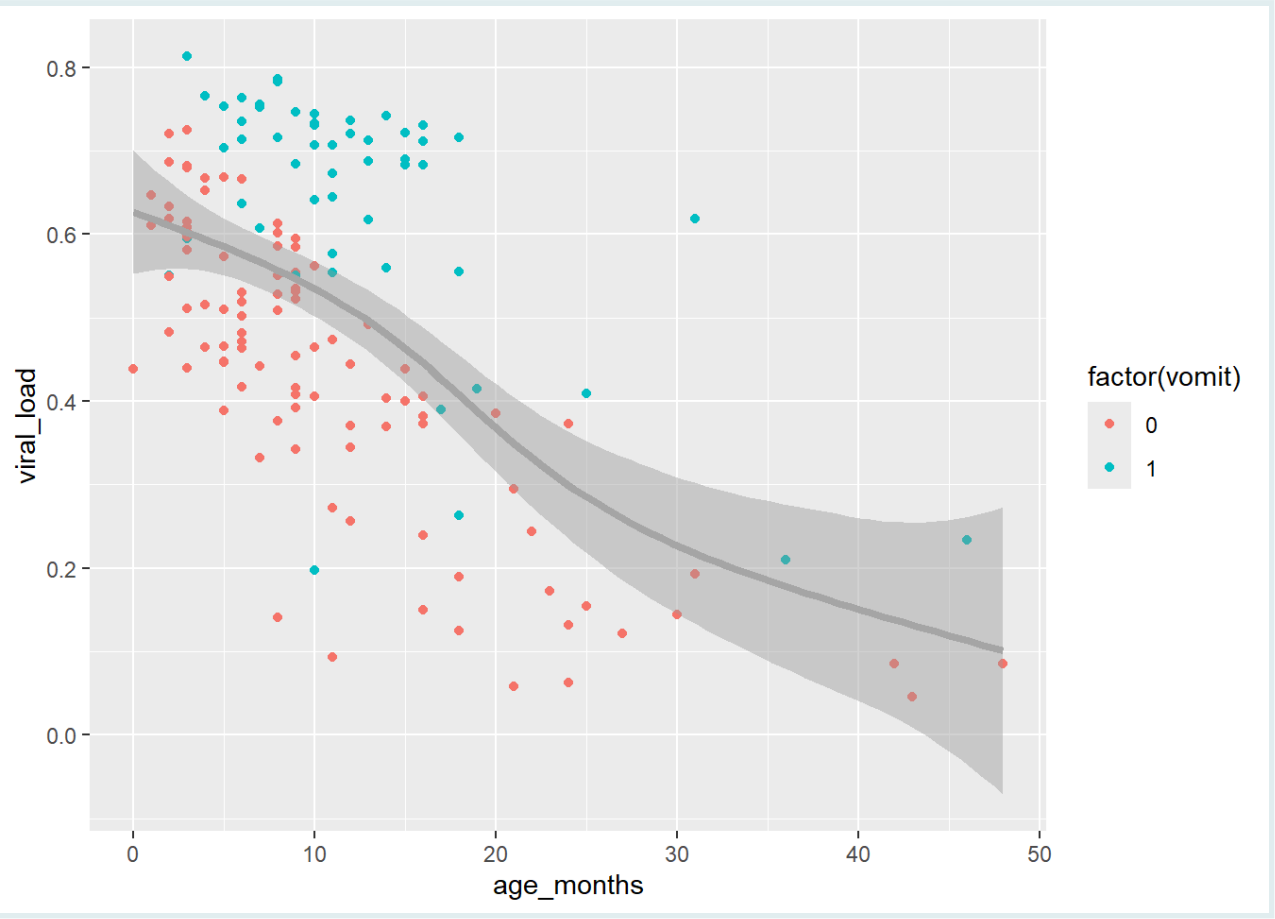


Cette régression linéaire confirme ce que nous avions initialement observé dans le premier nuage de points : Une *relation négative* existe entre `age_months` et `viral_load` : lorsque l'âge augmente, la charge virale tend à diminuer.

Introduisons maintenant une troisième variable du dataset `malidd` appelée `vomit`. Cette variable binaire enregistre si le patient a vomi ou non. Nous allons incorporer la variable `vomit` au graphique en la mappant à l'esthétique de la couleur. Nous allons changer à nouveau la méthode de lissage pour le modèle additif généralisé ("`gam`") et nous apporterons quelques modifications esthétiques à la ligne dans la couche `geom_smooth()`.

```
ggplot(data = malidd,
       mapping = aes(x = age_months,
                     y = viral_load)) +
  geom_point(mapping = aes(color = factor(vomit))) +
  geom_smooth(method = "gam",
              size = 1.5,
              color = "darkgray")
```

```
## `geom_smooth()` using formula = 'y ~ s(x, bs = "cs")'
```



Notez la distribution des points bleus (représentant les enfants qui ont vomi) par rapport aux points rouges (représentant les enfants qui n'ont pas vomi). Les points bleus se situent principalement au-dessus de la ligne de tendance, ce qui indique que les charges virales plus élevées étaient non seulement associées à des enfants plus jeunes, mais aussi à des enfants plus susceptibles de présenter des symptômes de vomissements.

- Créez un graphique de dispersion avec les variables `age_months` et `viral_load`. Définissez la couleur des points à "steelblue", la taille à 2,5 mm et l'opacité à 80%. Puis ajoutez une ligne de tendance avec la méthode de lissage "lm" (modèle linéaire). Pour faire ressortir la ligne de tendance, définissez sa couleur à "indianred3".
- Recréez le graphique de la question précédente, mais cette fois adaptez le code pour changer la forme des points à la forme numéro 23, et ajoutez la variable de température corporelle (`temp`) en **la mappant** à la couleur de remplissage des points.

En résumé

Les graphiques de dispersion permettent de visualiser la relation entre deux variables quantitatives.

Avec un dataset de taille moyenne à grande, vous pouvez expérimenter avec différentes variations des éléments d'un graphique de dispersion que nous avons vus, comme l'ajout de lignes de tendance, la modification de la couleur, de la taille, de la forme, du remplissage ou de l'opacité des points. Cette personnalisation est souvent un aspect ludique de la visualisation de données, car elle vous permet de découvrir différentes relations qui se dévoilent au fur et à mesure que vous ajustez vos graphiques.

Contributeurs

Ont contribué à ce cours :

JOY VAZ

R Developer and Instructor, the GRAPH Network
Loves doing science and teaching science

IMANE BENSOUDA KORACHI

R Developer and Instructor, the GRAPH Network

ADMIN TEAM

GRAPH Courses Administration Team
The GRAPH Courses team is building epidemiological training courses to enhance disease surveillance and data science for public health across the globe

Références

Le contenu de ce cours a été adapté en partie des sources suivantes :

- Ismay, Chester, and Albert Y. Kim. 2022. *A ModernDive into R and the Tidyverse*. https://moderndive.com/.

- Kabacoff, Rob. 2020. *Data Visualization with R*. https://rkabacoff.github.io/datavis/.
- Giroux-Bougard, Xavier, Maxwell Farrell, Amanda Winegardner, Étienne Low-Decarie and Monica Granados. 2020. *Workshop 3: Introduction to Data Visualisation with {ggplot2}*. http://r.qcbs.ca/workshop03/book-en/.