ggplot: Lexis surface plots and the effective use of color

Jonas Schöley

September 19, 2017

Contents


```
library(tidyverse)
```

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ----------------------------------------------
```

```
## filter(): dplyr, stats
## lag():    dplyr, stats
```

Lexis Surfaces in GGplot

Lexis surfaces show the value of a third variable on a period-age-grid. If the value of the third variable is given via colour the resulting plot is known as "Heatmap" and used in many disciplines. In ggplot we produce heatmaps using `geom_tile()` or `geom_rect()`. These geometries draw rectangles at specified xy-positions. By default all rectangles are equal in size. They can be coloured according to some variable in the data.

`geom_rect()` is faster and produces smaller pdf's, while `geom_tile()` allows to specify the dimensions of the rectangles making it useful for data that does not come in single year period and age intervals.

For now we will ignore the colouring aspect and just look at how ggplot draws rectangles.

1x1 year data

We work with data from the Human Mortality Database – Swedish period mortality rates by sex.

```
swe <- read_csv("mortality_surface_sweden.csv")
```

```
## Parsed with column specification:
## cols(
##   Country = col_character(),
##   Timeframe = col_character(),
##   Sex = col_character(),
##   Year = col_integer(),
##   Age = col_integer(),
```

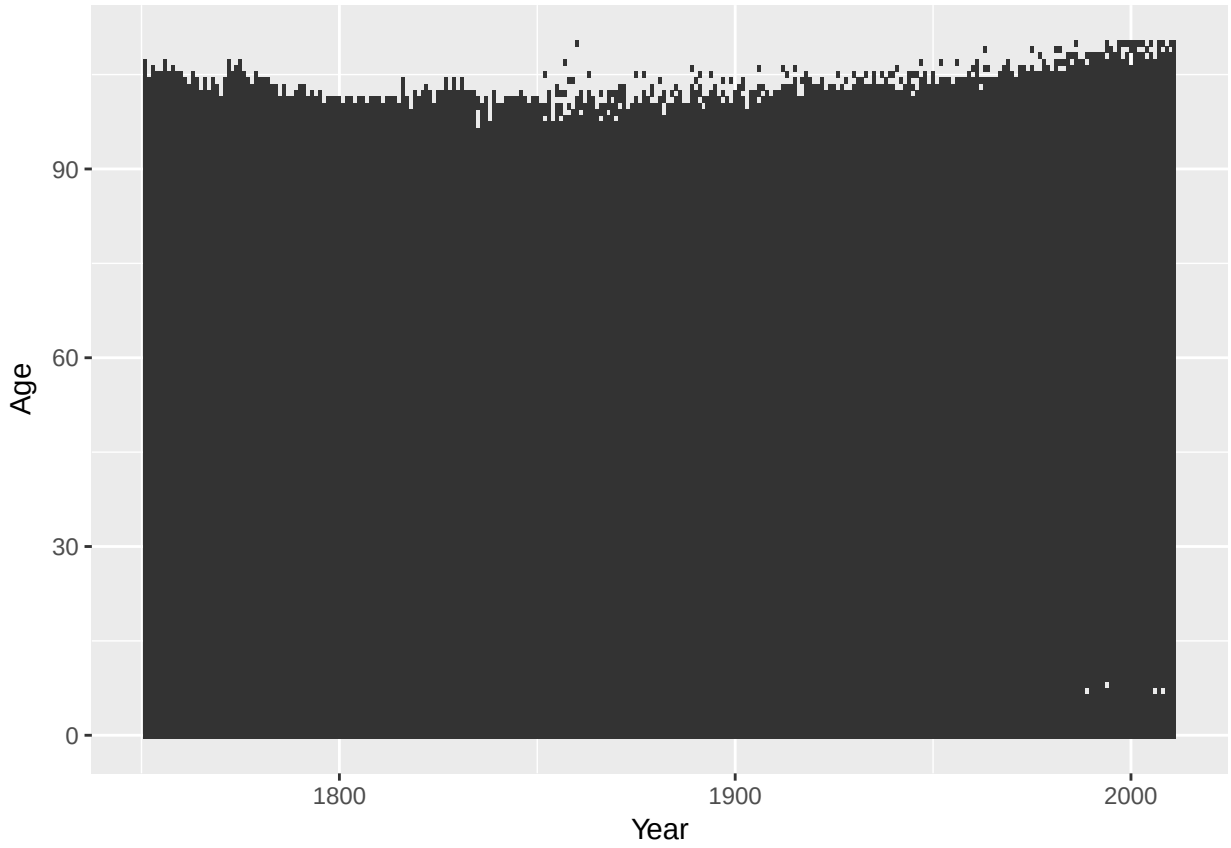
```
##   Dx = col_double(),
##   Nx = col_double(),
##   mx = col_double()
## )
```

```
head(swe)
```

```
## # A tibble: 6 x 8
##   Country Timeframe    Sex  Year   Age      Dx       Nx      mx
##     <chr>     <chr>  <chr> <int> <int>   <dbl>    <dbl>   <dbl>
## 1     SWE    Period Female  1751     0 5988.00 28214.05 0.21223
## 2     SWE    Period   Male  1751     0 6902.00 28626.58 0.24110
## 3     SWE    Period Female  1751     1 1286.45 26035.03 0.04941
## 4     SWE    Period   Male  1751     1 1359.88 25682.81 0.05295
## 5     SWE    Period Female  1751     2  834.56 25880.11 0.03225
## 6     SWE    Period   Male  1751     2  882.13 25504.40 0.03459
```

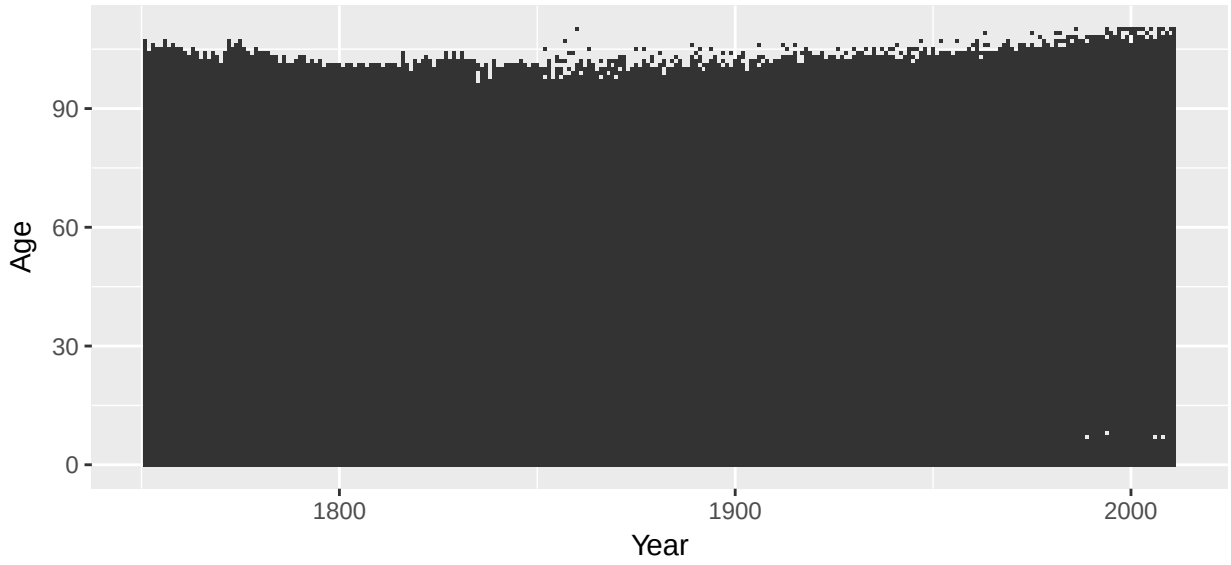
Only specifying x and y position and omitting colour puts a grey rectangle at every xy position that appears in the data. The resulting plot gives us information about the period-ages where we have mortality data on Swedish females.

```
swe %>% filter(Sex == "Female") %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age))
```



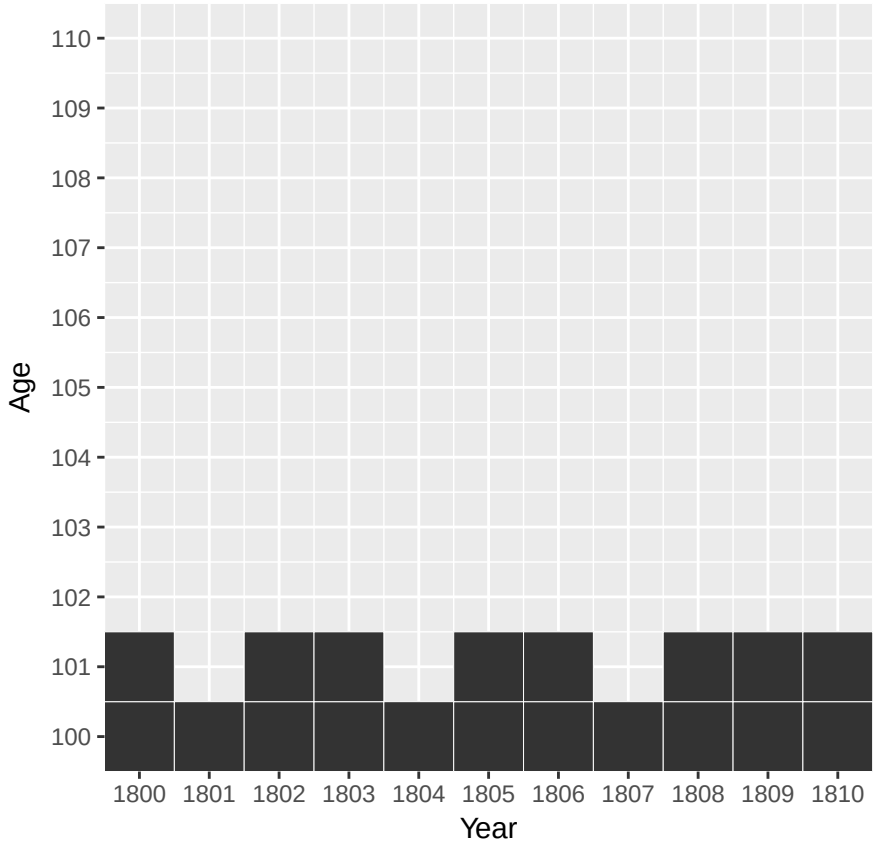
When constructing Lexis surfaces it is a good idea to use isometric scales. The distance corresponding to a single year should be the same on the x and the y scales (a 1x1 rectangle should actually be a square). We can force such an equality by adding a suitable coordinate layer.

```
swe %>% filter(Sex == "Female") %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age)) +
  coord_equal()
```



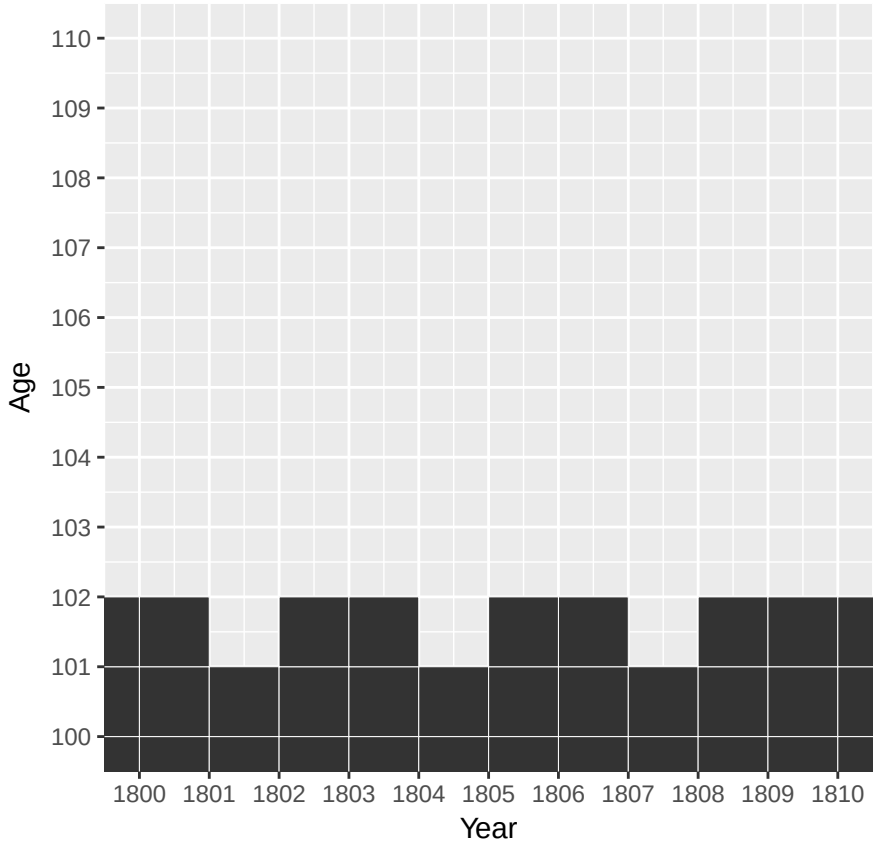
By default the small rectangles have a width and height of 1 scale unit and are drawn over the mid-points of the corresponding x and y values.

```
swe %>% filter(Sex == "Female") %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age), colour = "white") +
  scale_x_continuous(breaks = 1800:1810) +
  scale_y_continuous(breaks = 100:110) +
  coord_equal(xlim = c(1800, 1810), ylim = c(100, 110))
```



Shifting the data by 0.5 in x and y aligns things neatly.

```
swe %>% filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age), colour = "white") +
  scale_x_continuous(breaks = 1800:1810) +
  scale_y_continuous(breaks = 100:110) +
  coord_equal(xlim = c(1800, 1810), ylim = c(100, 110))
```



nxm year data

If our data does not come in single year and age groups we have to adjust the `width` and/or `height` of the rectangles. `width` and `height` are regular aesthetics and can be mapped to variables in the data.

```
cod <- read_csv("cod.csv")
```

```
## Parsed with column specification:
## cols(
##   Year = col_integer(),
##   Age = col_integer(),
##   AgeGr = col_character(),
##   w = col_integer(),
##   Sex = col_character(),
##   COD = col_character()
## )
```

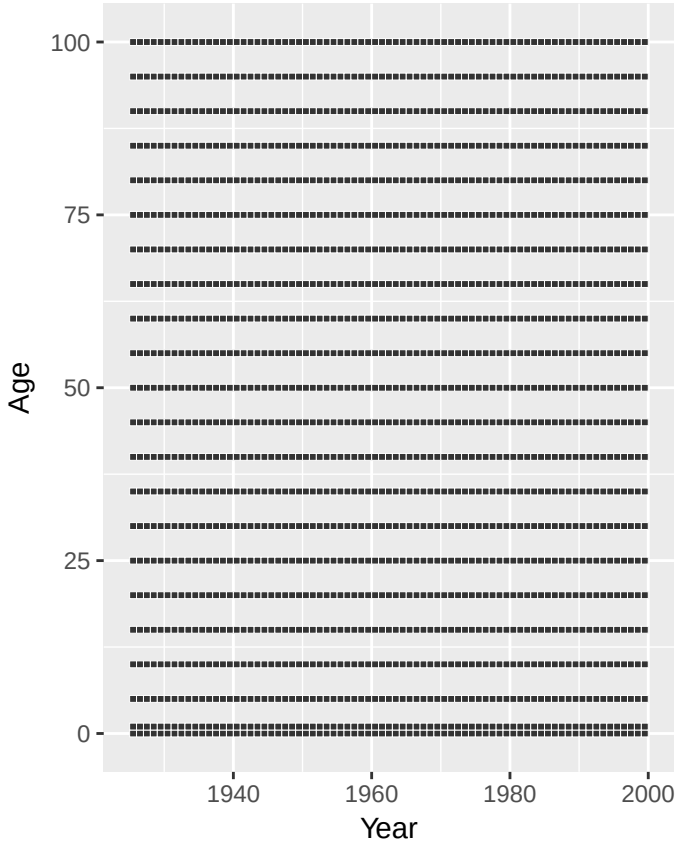
```
head(cod)
```

```
## # A tibble: 6 x 6
##    Year   Age AgeGr     w    Sex   COD
##   <int> <int> <chr> <int>  <chr> <chr>
## 1  1925     0    <1     1 Female Other
## 2  1925     0    <1     1   Male Other
## 3  1925     1   1-4     4 Female Other
## 4  1925     1   1-4     4   Male Other
## 5  1925     5   5-9     5 Female Other
```

```
## 6  1925     5   5-9     5   Male Other
```

The Cause of Death data features age groups of different sizes (1, 4, or 5 years). This is how it looks like if we plot it without any regard to the size of the age groups.

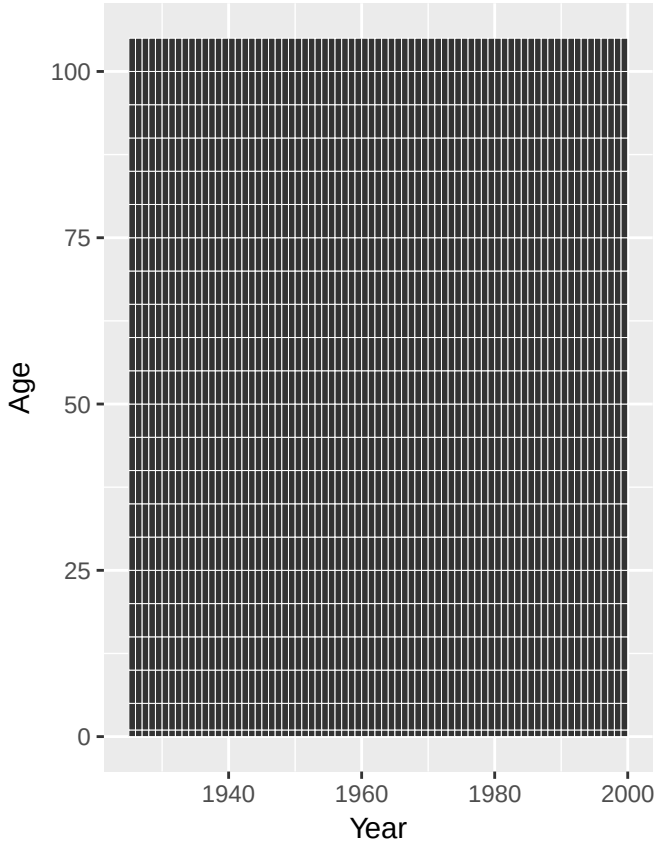
```
cod %>% filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age),
            colour = "white") +
  coord_equal()
```



Now we shift the rectangles away from the age midpoint and scale them in height according to the width of the age group.

```
cod %>% filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5, Age = Age + w/2) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, height = w),
            colour = "white") +
  coord_equal()
```

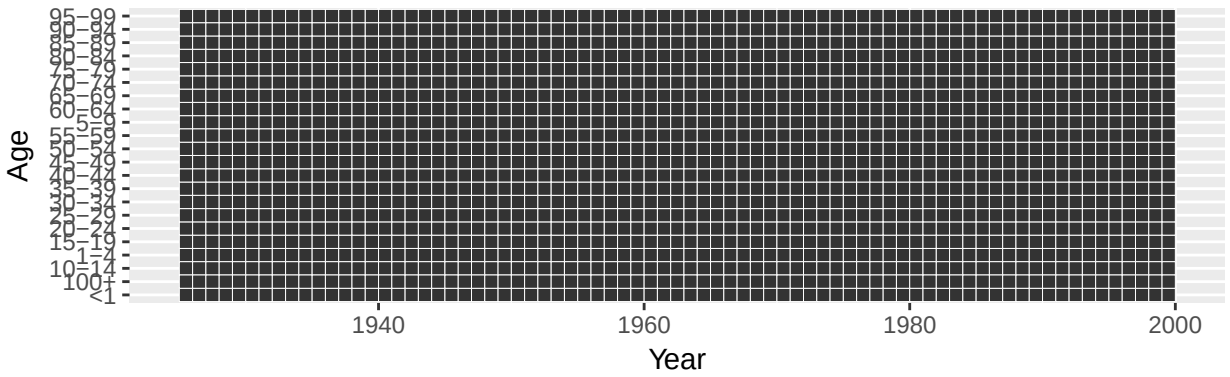
```
## Warning: Ignoring unknown aesthetics: height
```



Discrete Period and Age Scales

If we use discrete axis (happens automatically if we supply a non-numeric variable to the x or y aesthetic) we loose any control over the placement of the age or period groups. They will be equally spaced along the axis.

```
cod %>% filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5, Age = AgeGr) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age), colour = "white") +
  coord_equal()
```

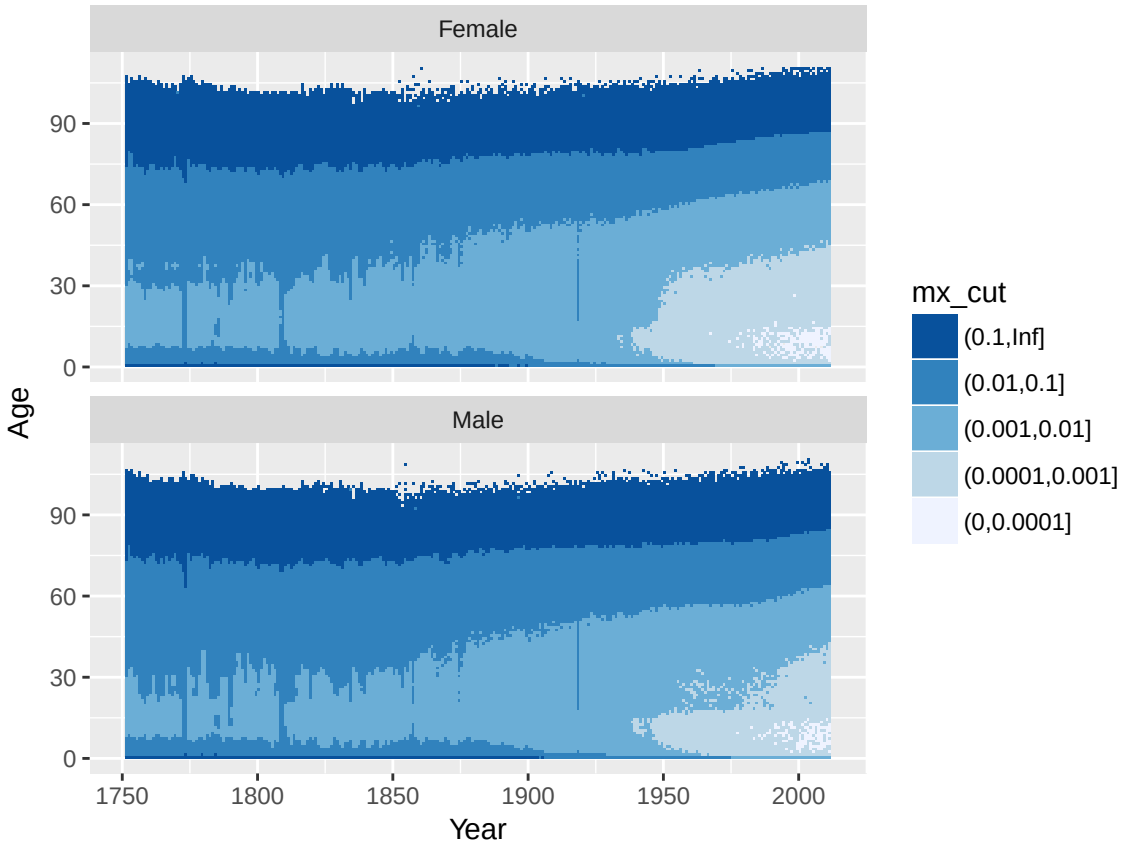


Avoid character or factor variables as your period or age groups. Whenever possible go with numeric "Start of Interval" and "Interval Width" variables.

Sequential Colour Scales: Plotting Magnitudes

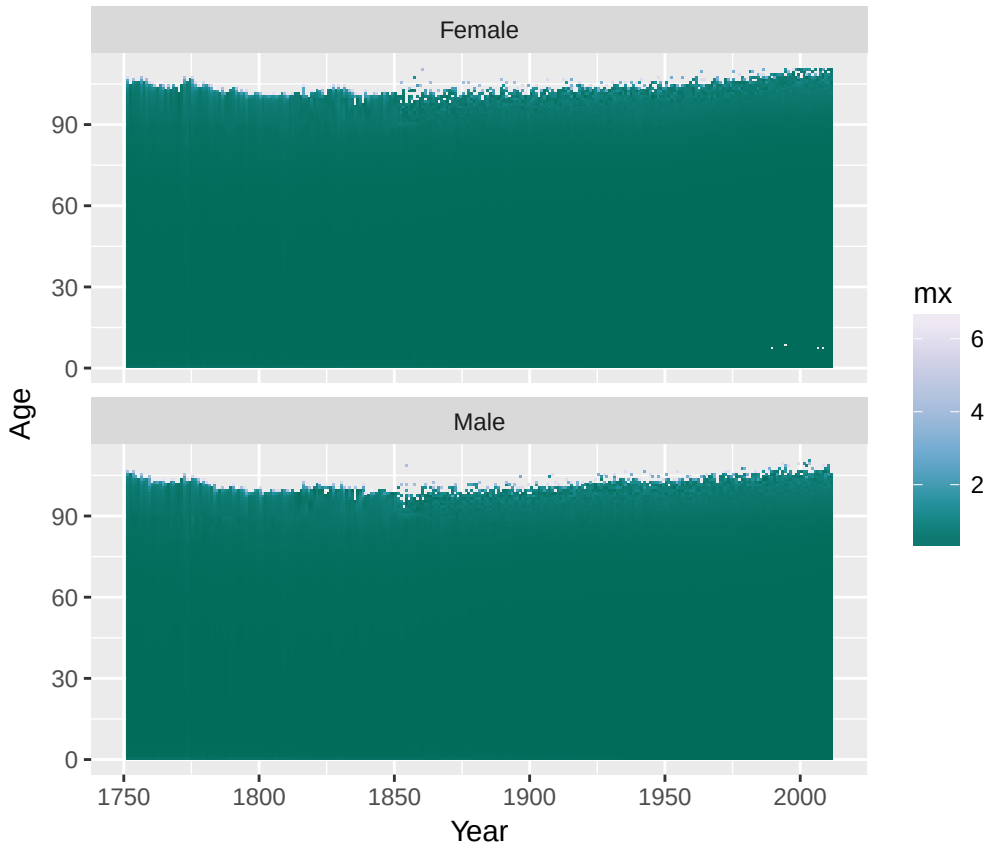
If we plot magnitudes we would like to use a colour scale which has an intrinsic ordering to it. Scales that vary from dark to light are suitable and we call them "sequential". `scale_fill_brewer(type = "seq")` provides you with such a scale.

```
breaks_mx <- c(0, 0.0001, 0.001, 0.01, 0.1, Inf)
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5,
         mx_cut = cut(mx, breaks = breaks_mx)) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = mx_cut)) +
  scale_fill_brewer(type = "seq") +
  facet_wrap(~Sex, ncol = 1) +
  guides(fill = guide_legend(reverse = TRUE)) +
  coord_equal()
```



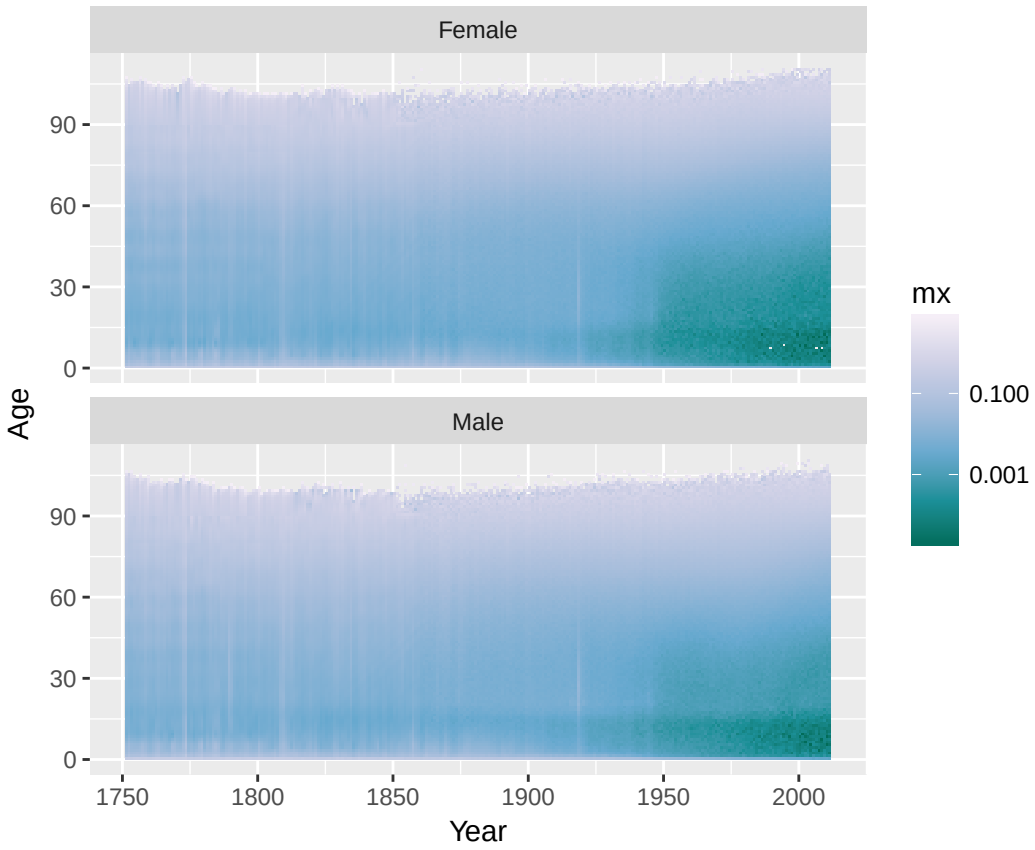
Continuous colour scales take the form of a smooth colour gradient. Getting the gradient to look like you want can be tricky.

```
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = mx)) +
  scale_fill_distiller(type = "seq", palette = "PuBuGn") +
  facet_wrap(~Sex, ncol = 1) +
  coord_equal()
```



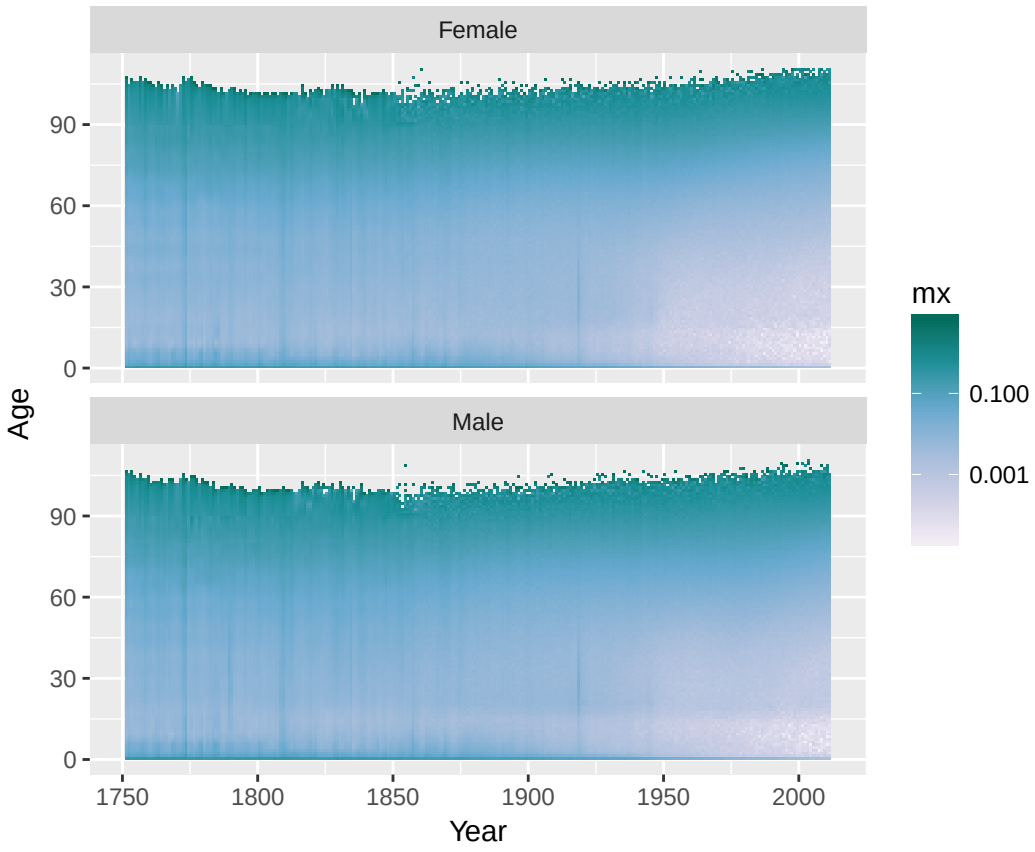
Log transform the colour scale.

```
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = mx)) +
  scale_fill_distiller(type = "seq", palette = "PuBuGn", trans = "log10") +
  facet_wrap(~Sex, ncol = 1) +
  coord_equal()
```



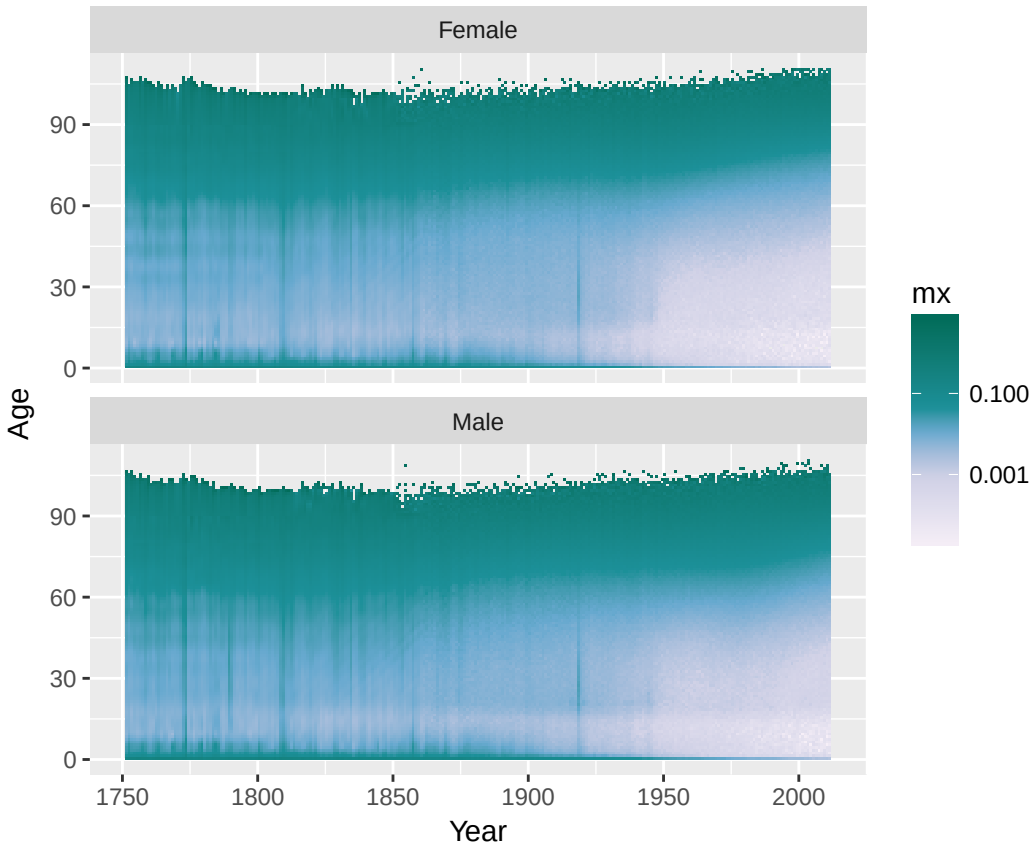
Make high values dark and low values light.

```
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = mx)) +
  scale_fill_distiller(type = "seq", palette = "PuBuGn", trans = "log10",
                       direction = 1) +
  facet_wrap(~Sex, ncol = 1) +
  coord_equal()
```



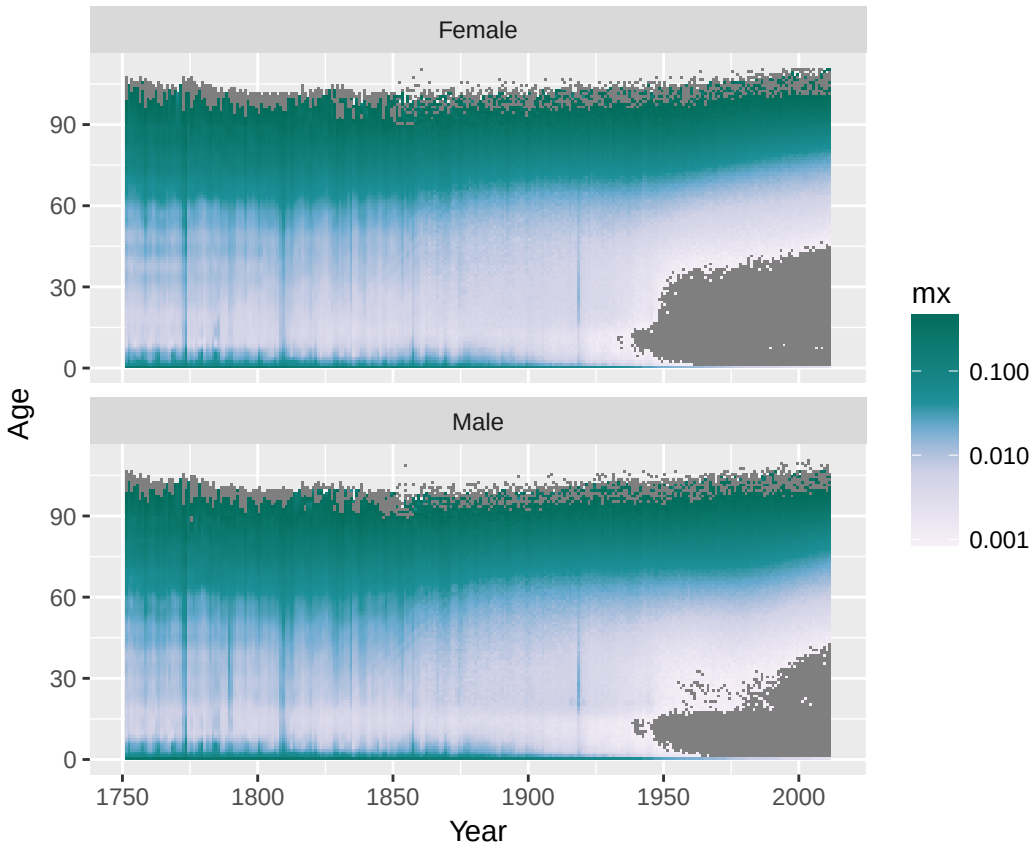
Rescale the colour gradient to increase contrast.

```
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = mx)) +
  scale_fill_distiller(type = "seq", palette = "PuBuGn", trans = "log10",
                       direction = 1,
                       values = c(0, 0.3, 0.4, 0.5, 0.6, 1)) +
  facet_wrap(~Sex, ncol = 1) +
  coord_equal()
```



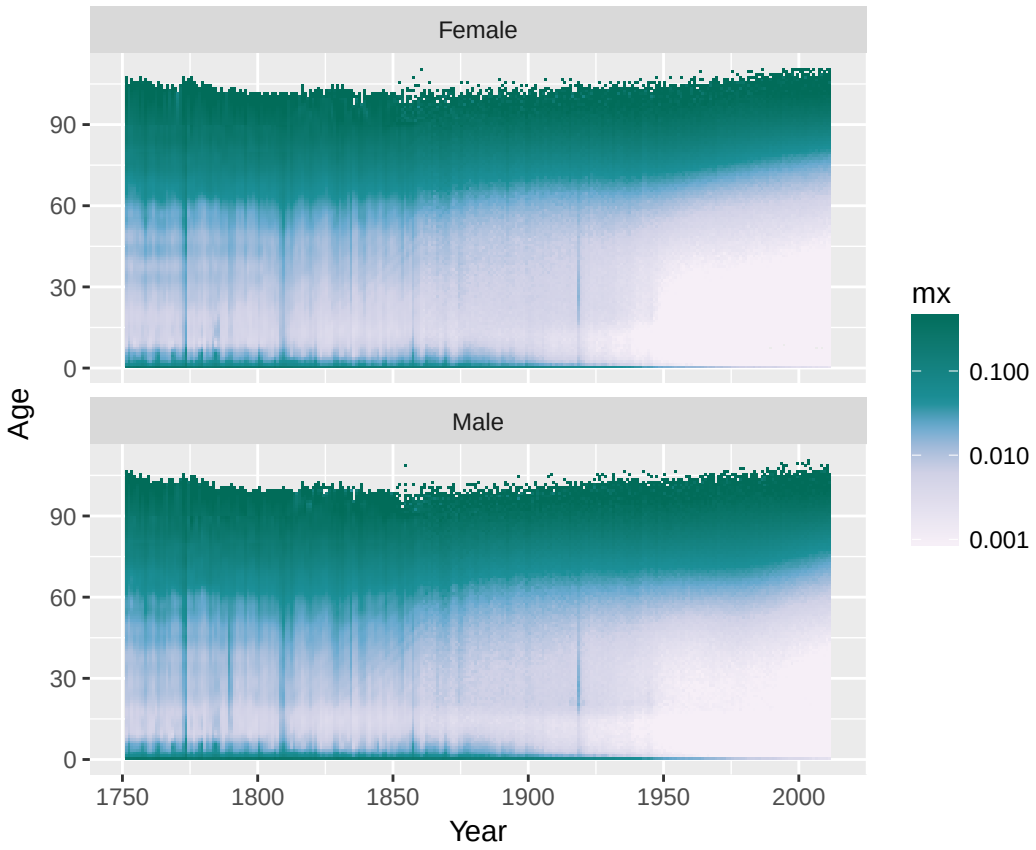
Throw away data outside of the limits.

```
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = mx)) +
  scale_fill_distiller(type = "seq", palette = "PuBuGn", trans = "log10",
                       direction = 1,
                       values = c(0, 0.3, 0.4, 0.5, 0.6, 1),
                       limits = c(0.001, 0.5)) +
  facet_wrap(~Sex, ncol = 1) +
  coord_equal()
```



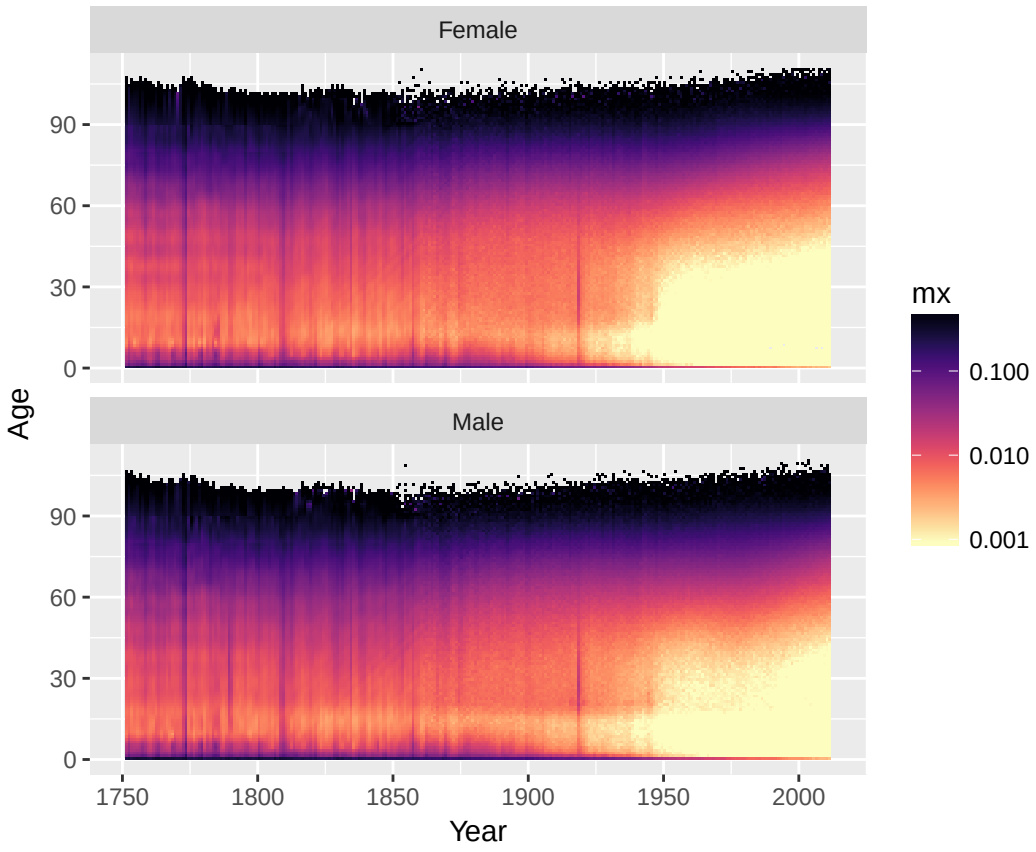
Or instead *squish* the out-of-bounds data into the limits.

```
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = mx)) +
  scale_fill_distiller(type = "seq", palette = "PuBuGn", trans = "log10",
                       direction = 1,
                       values = c(0, 0.3, 0.4, 0.5, 0.6, 1),
                       limits = c(0.001, 0.5),
                       oob = scales::squish) +
  facet_wrap(~Sex, ncol = 1) +
  coord_equal()
```



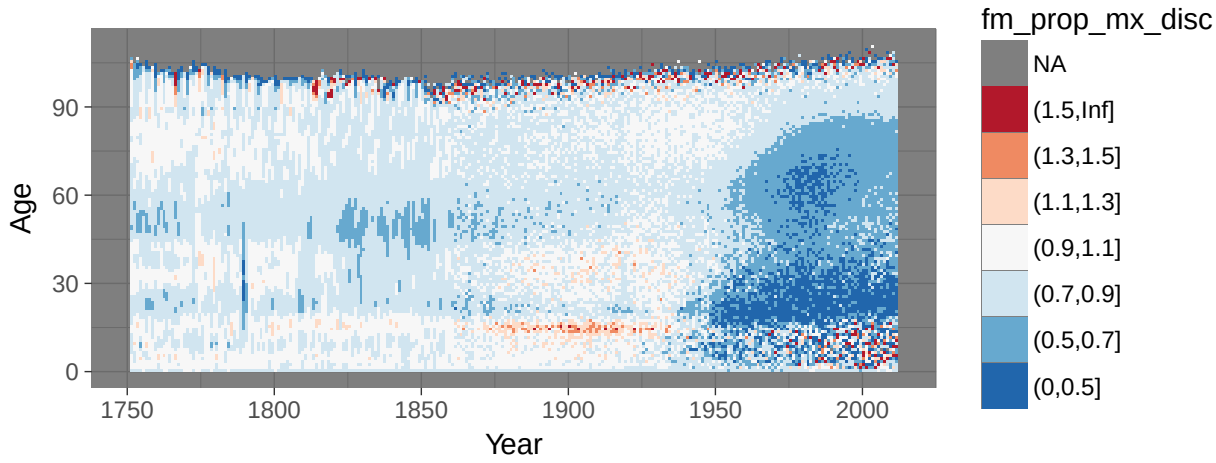
The viridis scales pair perceptual uniformity with strong shifts in hue, thereby increasing discriminability between different magnitude regions.

```
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = mx)) +
  viridis::scale_fill_viridis(direction = -1,
                              trans = "log10",
                              option = "magma",
                              limits = c(0.001, 0.5),
                              oob = scales::squish) +
  facet_wrap(~Sex, ncol = 1) +
  coord_equal()
```



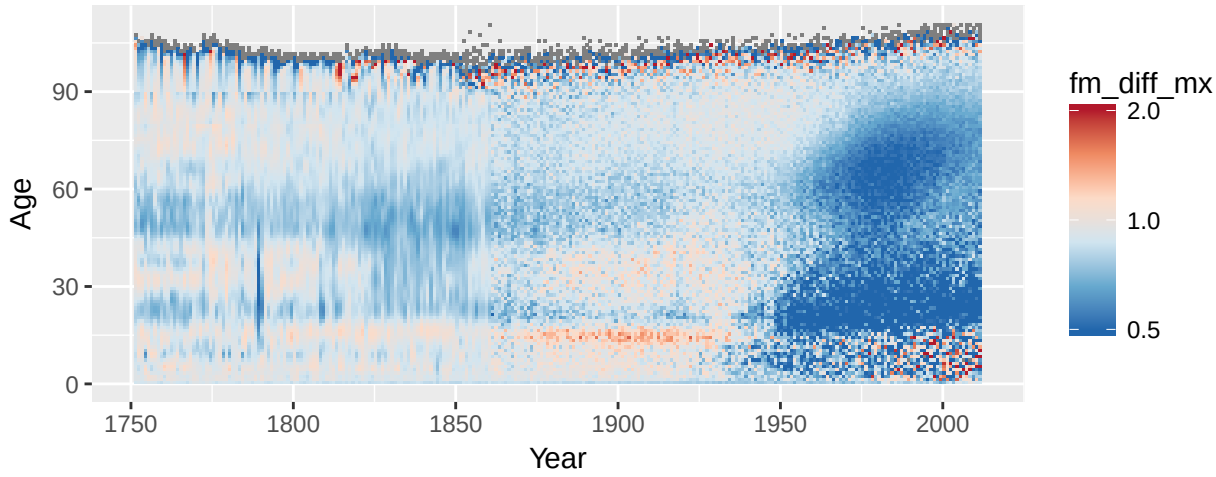
Divergent Colour Scales: Plotting Differences & Proportions

```
breaks_prop_mx <- c(0, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, Inf)
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  select(-Dx, -Nx) %>%
  tidyr::spread(key = Sex, value = mx) %>%
  mutate(fm_prop_mx = Female / Male,
         fm_prop_mx_disc = cut(fm_prop_mx, breaks_prop_mx)) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = fm_prop_mx_disc)) +
  scale_fill_brewer(type = "div", palette = 5, direction = -1) +
  guides(fill = guide_legend(reverse = TRUE)) +
  coord_equal() +
  theme_dark()
```



Continuous variant.

```
swe %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  select(-Dx, -Nx) %>%
  tidyr::spread(key = Sex, value = mx) %>%
  mutate(fm_diff_mx = Female / Male) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, fill = fm_diff_mx)) +
  # takes 6 colours from a brewer palette and interpolates
  scale_fill_distiller(type = "div",
                       palette = "RdBu",
                       trans = "log2",
                       limits = c(0.5, 2),
                       oob = scales::squish) +
  coord_equal()
```

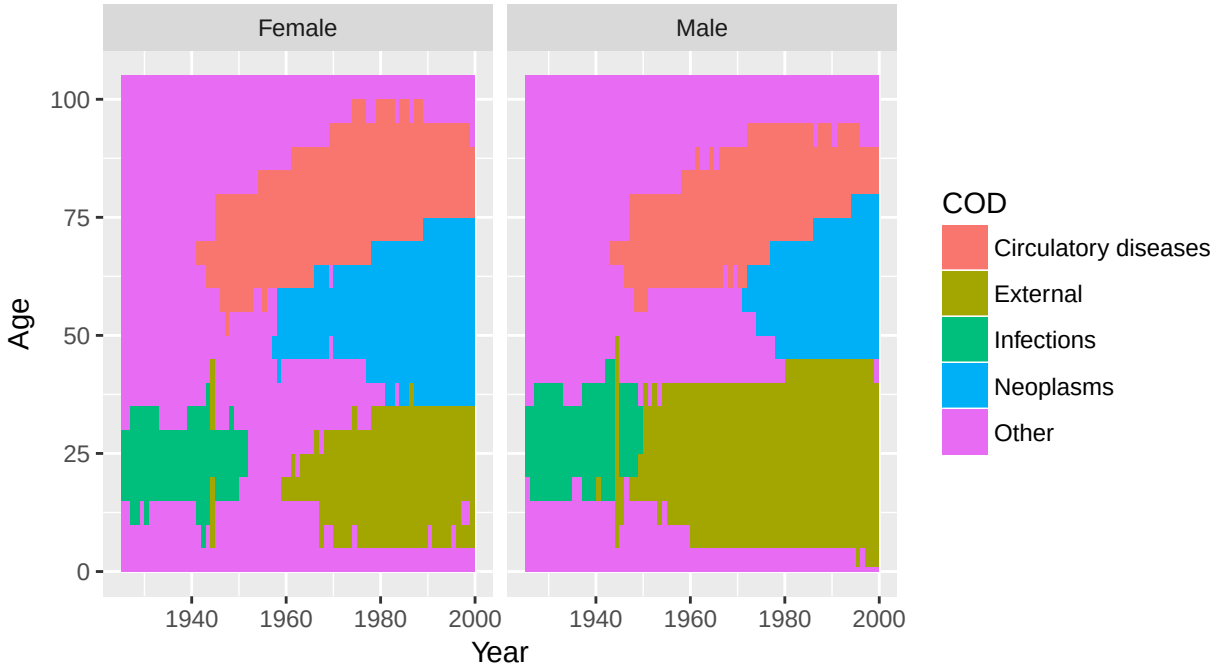


Qualitative Colour Scales: Plotting Group Membership

```
cod %>%
  mutate(Year = Year + 0.5, Age = Age + w/2) %>%
  ggplot() +
  geom_tile(aes(x = Year, y = Age, height = w, fill = COD)) +
```

```
  coord_equal() +
  facet_wrap(~Sex, ncol = 2)
```

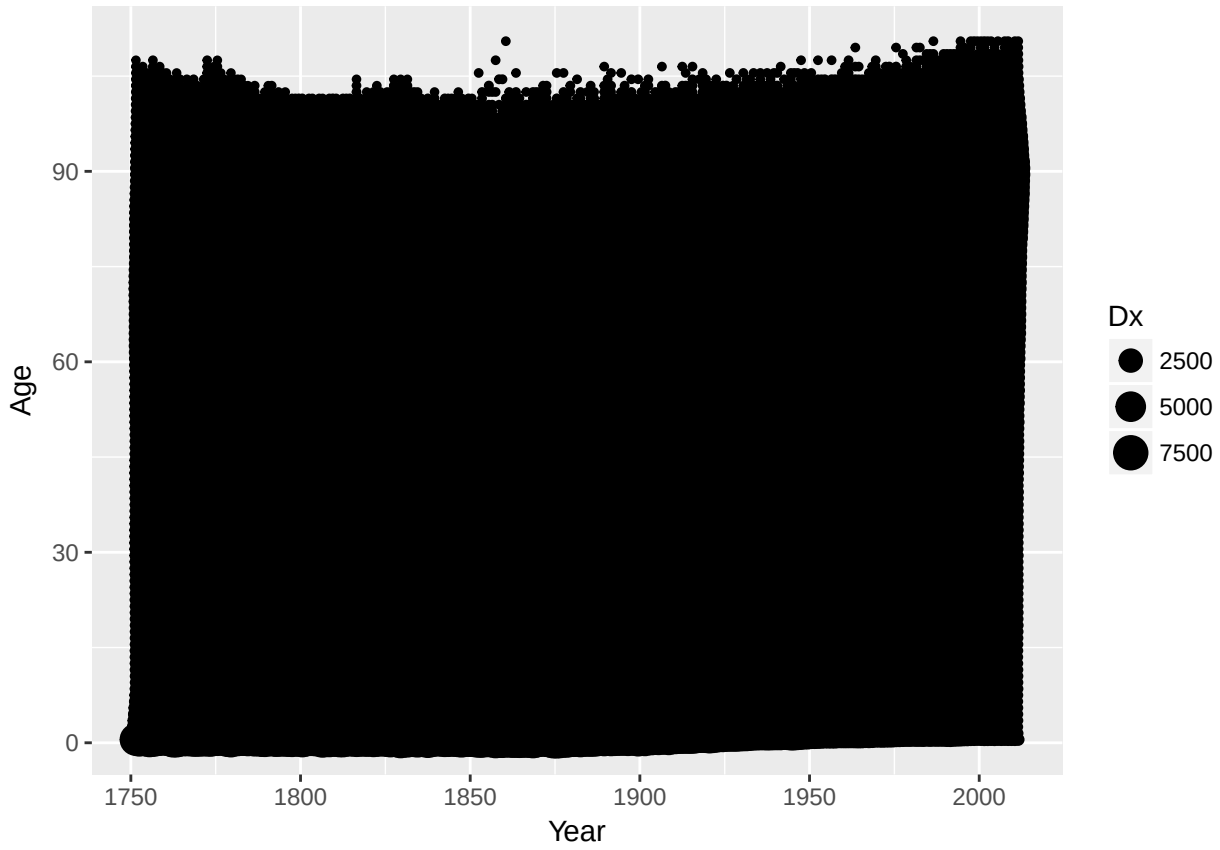
```
## Warning: Ignoring unknown aesthetics: height
```



Experiment in perception: using area instead of colour

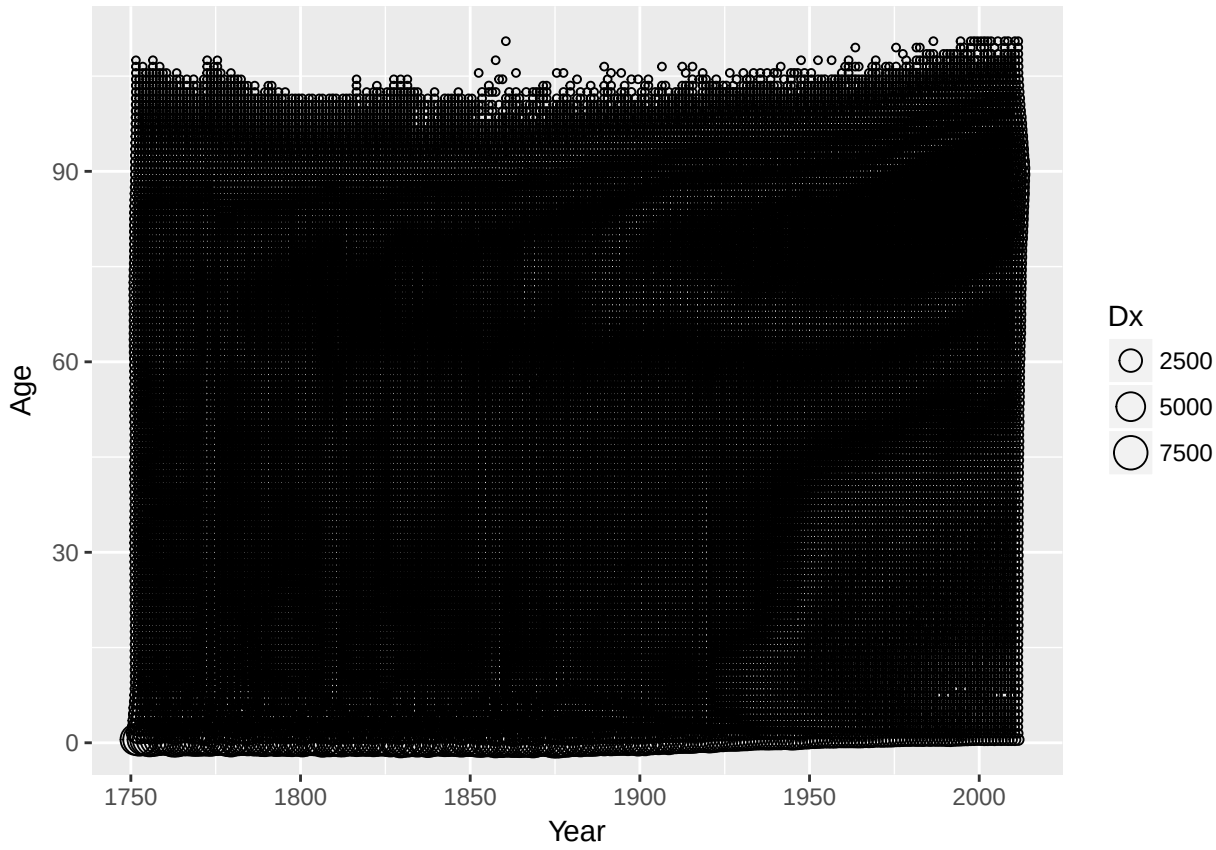
Let's plot a surface of the number of deaths by age. Instead of color as primary visual encoding we draw points of varying size.

```
swe %>%
  filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot(aes(x = Year, y = Age)) +
  geom_point(aes(size = Dx))
```



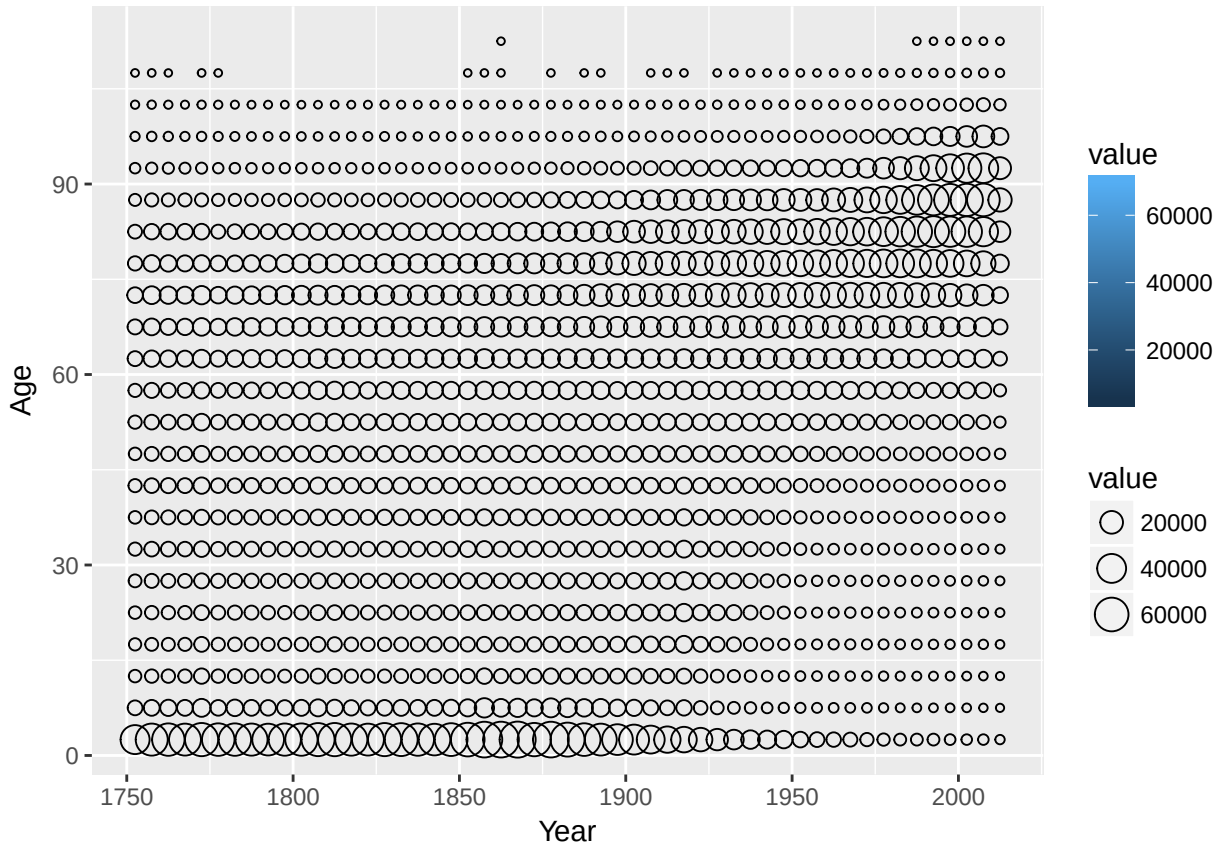
Not helpful. We can set the shape of the points to an open circle to clean up a bit.

```
swe %>%
  filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot(aes(x = Year, y = Age)) +
  geom_point(aes(size = Dx), shape = 1)
```



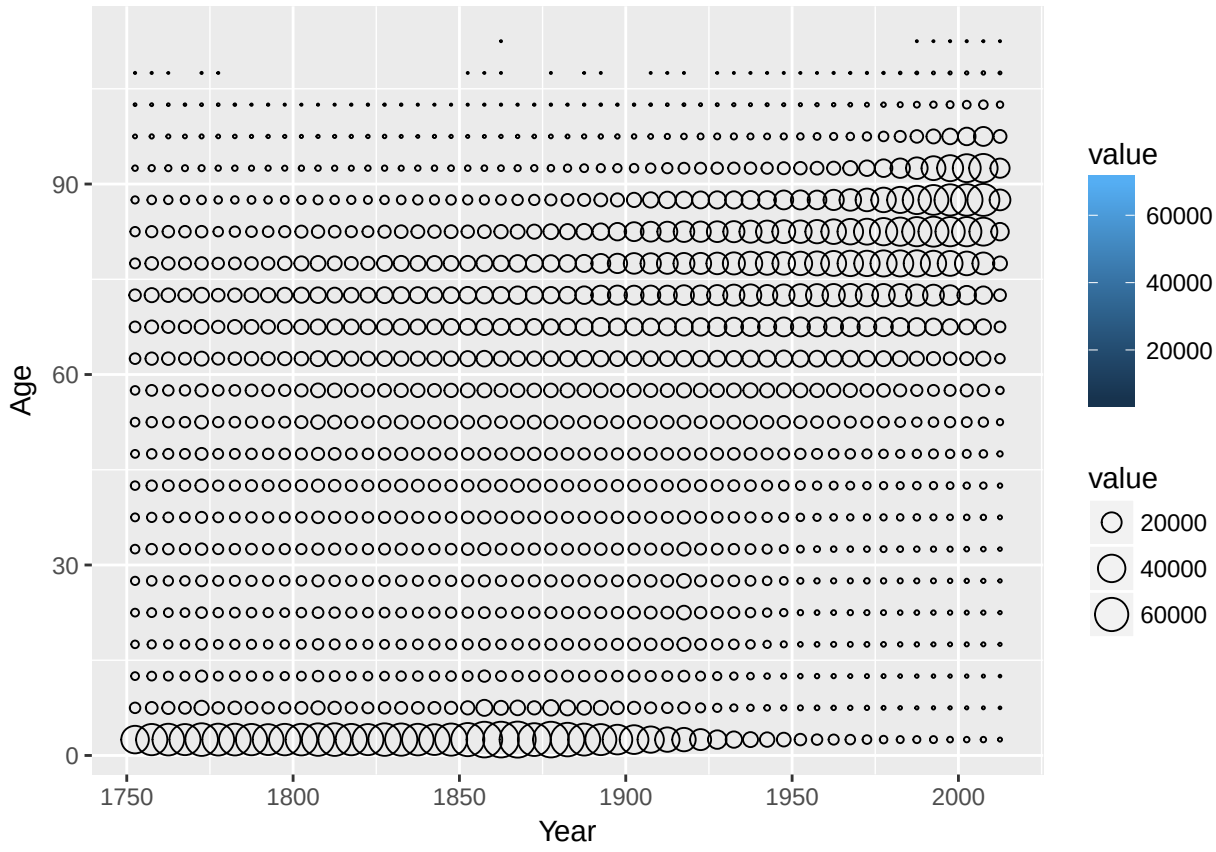
Still not good. Maybe reducing the number of circles would work. In order to do so we need to aggregate the data into wider period and age groups. For simple aggregations we can use the *summary2d* feature in ggplot. It cuts up the area of the plot into larger bins and performs a summary operation on the data in each bin. We will sum up the death counts in each 5x5 year square.

```
swe %>%
  filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot(aes(x = Year, y = Age)) +
  geom_point(aes(z = Dx, size = ..value..), shape = 1,
             stat = "summary2d",
             binwidth = c(5, 5),
             fun = sum)
```



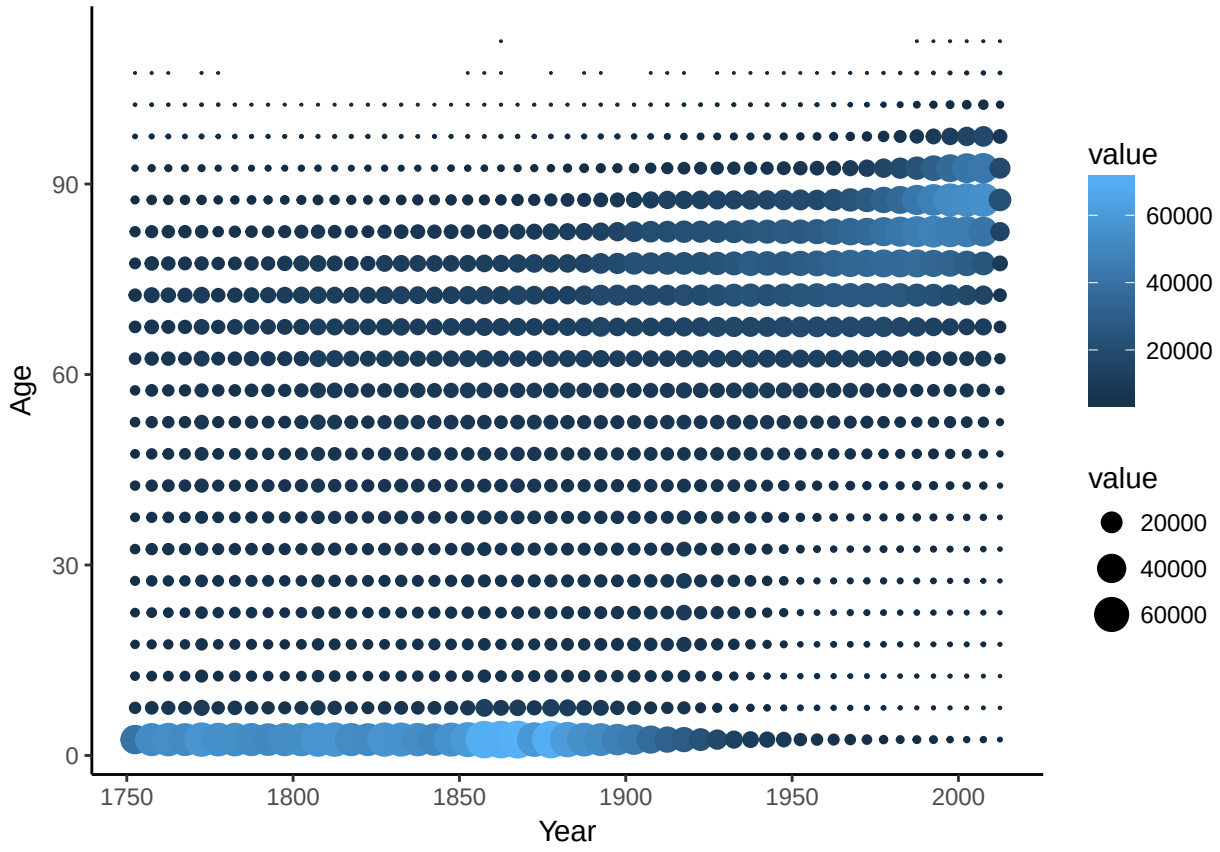
Change the size scale to area.

```
swe %>%
  filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot(aes(x = Year, y = Age)) +
  geom_point(aes(z = Dx, size = ..value..), shape = 1,
             stat = "summary2d",
             binwidth = c(5, 5),
             fun = sum) +
  scale_size_area()
```



Colour the points.

```
swe %>%
  filter(Sex == "Female") %>%
  mutate(Year = Year + 0.5, Age = Age + 0.5) %>%
  ggplot(aes(x = Year, y = Age)) +
  geom_point(aes(z = Dx, size = ..value.., colour = ..value..),
             stat = "summary2d",
             binwidth = c(5, 5),
             fun = sum) +
  scale_size_area() +
  theme_classic()
```



Further Reading

- Brilliant color advice from NASA
- Generator for categorical color scales
- A perceptually uniform continuous color scale
- Color scales for data-viz
- Brewer, Cynthia A. 1994. "Guidelines for Use of the Perceptual Dimensions of Color for Mapping and Visualization." In SPIE, edited by Jan Bares, 2171:54–63. doi:10.1117/12.175328.