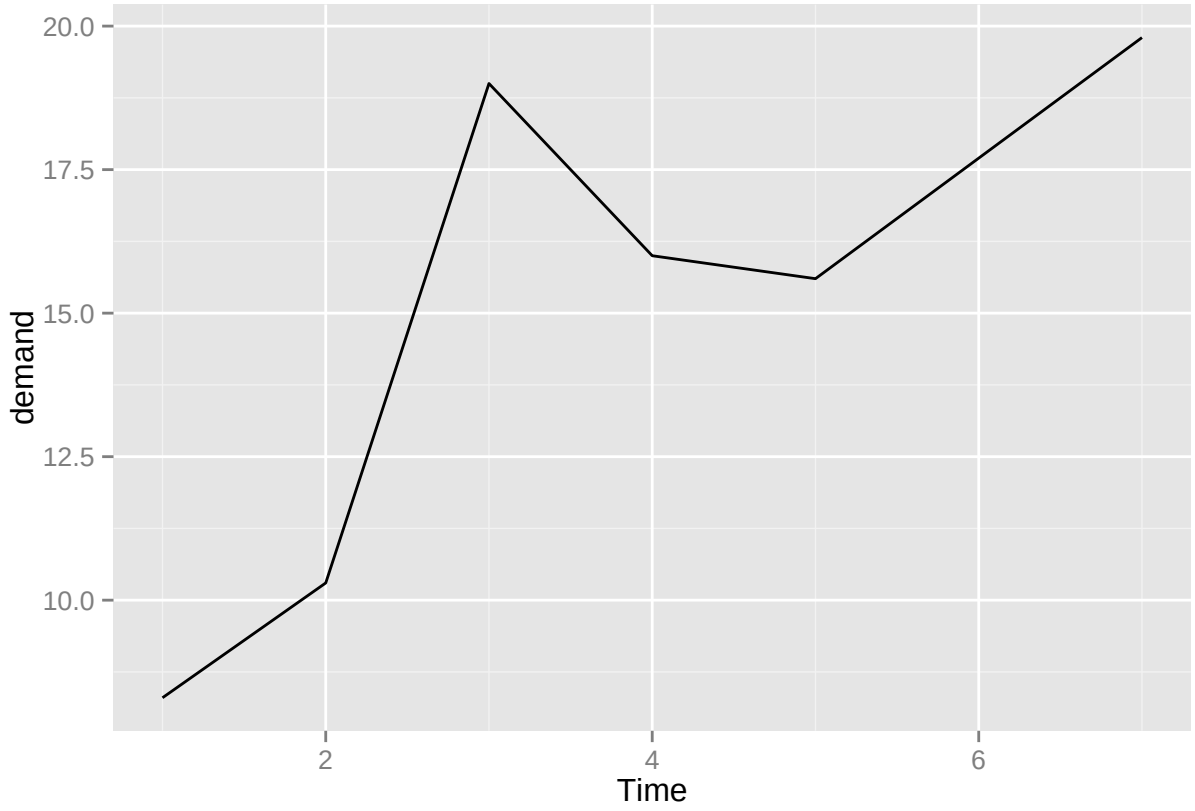
4_Line_Graphs

Gino Tesei

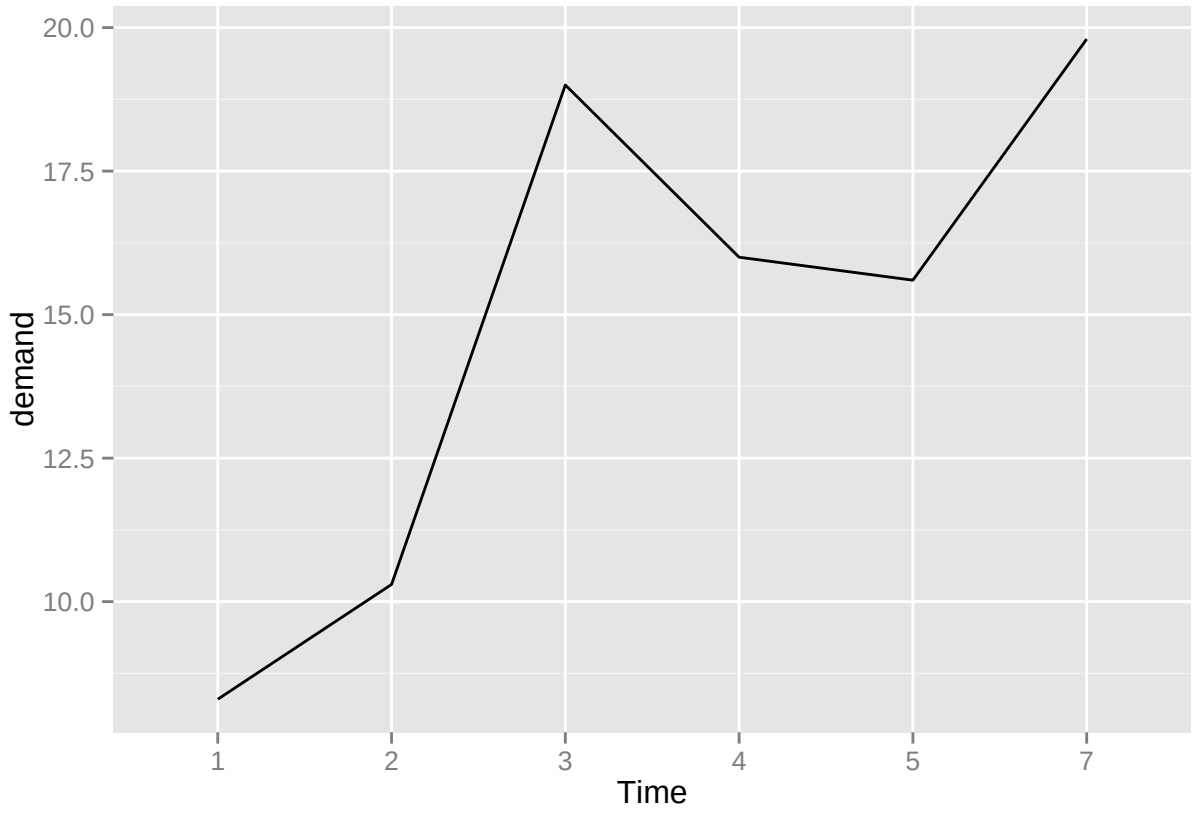
December 12, 2015

1. Making a Basic Line Graph

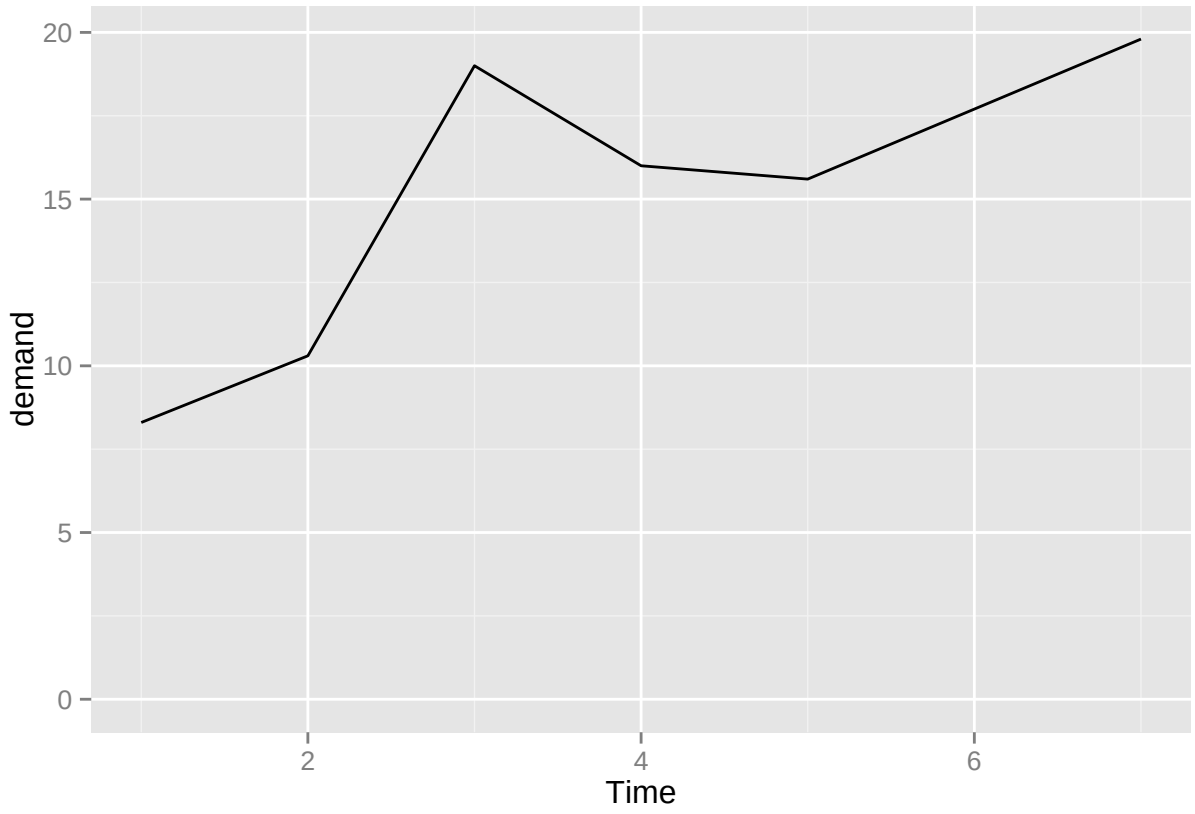
```
library(ggplot2)
library(gcookbook) # For the data set

library(plyr) ##

## numeric
ggplot( BOD, aes( x = Time, y = demand)) +
  geom_line()
```




```
## categorical
BOD1 <- BOD # Make a copy of the data
BOD1$Time <- factor( BOD1$Time)
ggplot( BOD1, aes( x = Time, y = demand, group = 1)) +
  geom_line()
```

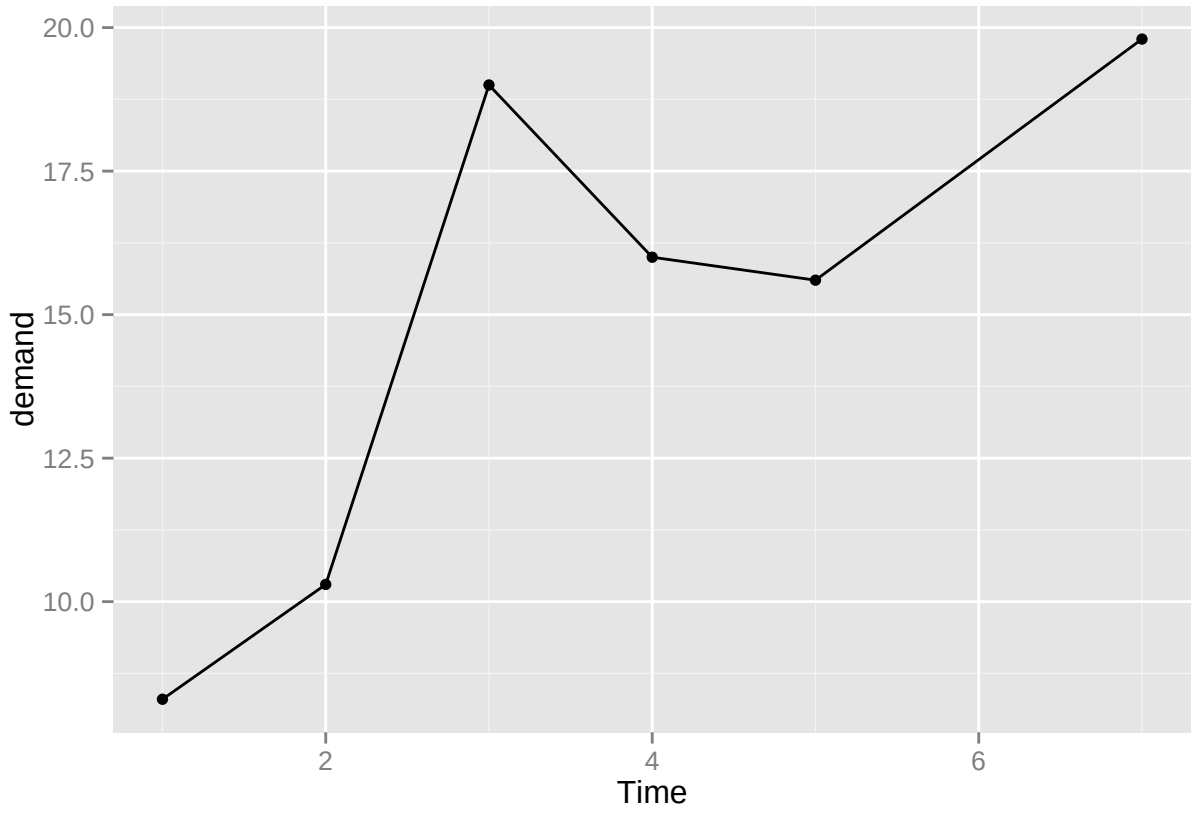



```
# expands ylim
ggplot( BOD, aes( x = Time, y = demand)) +
  geom_line() +
  expand_limits( y = 0)
```

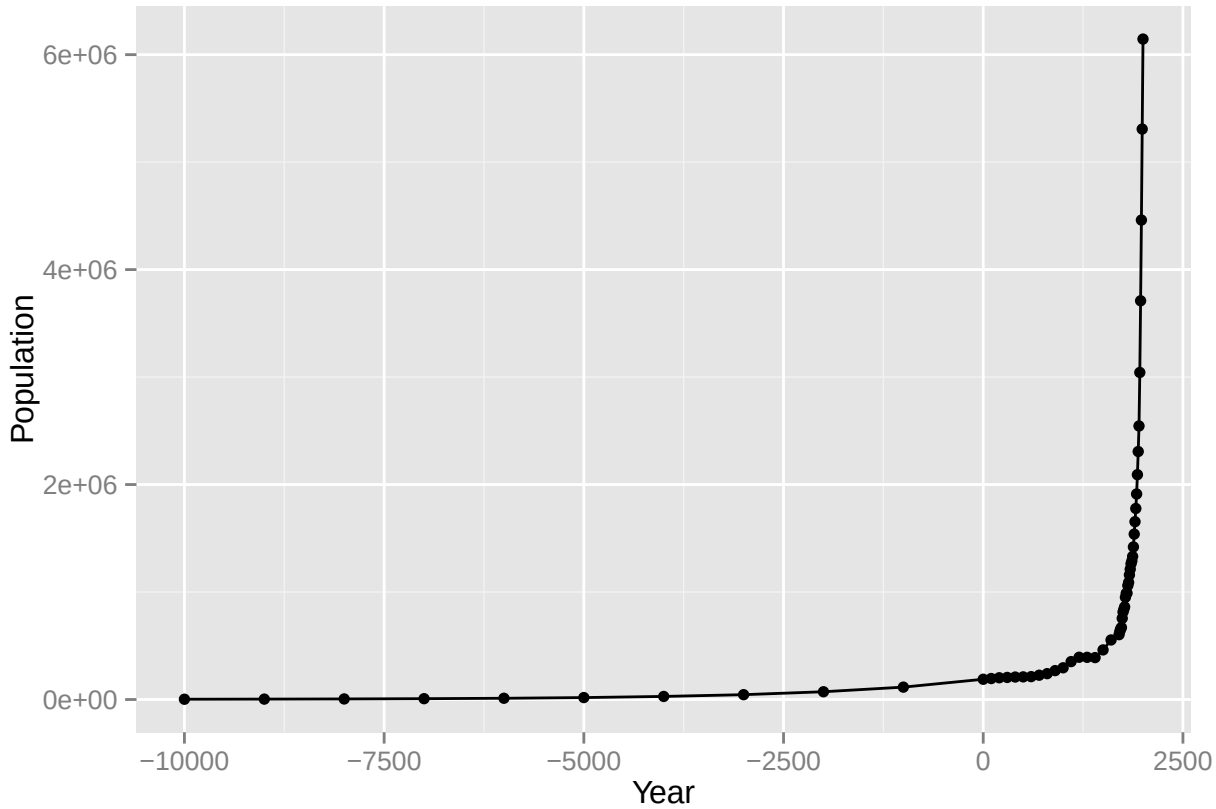


2. Adding Points to a Line Graph

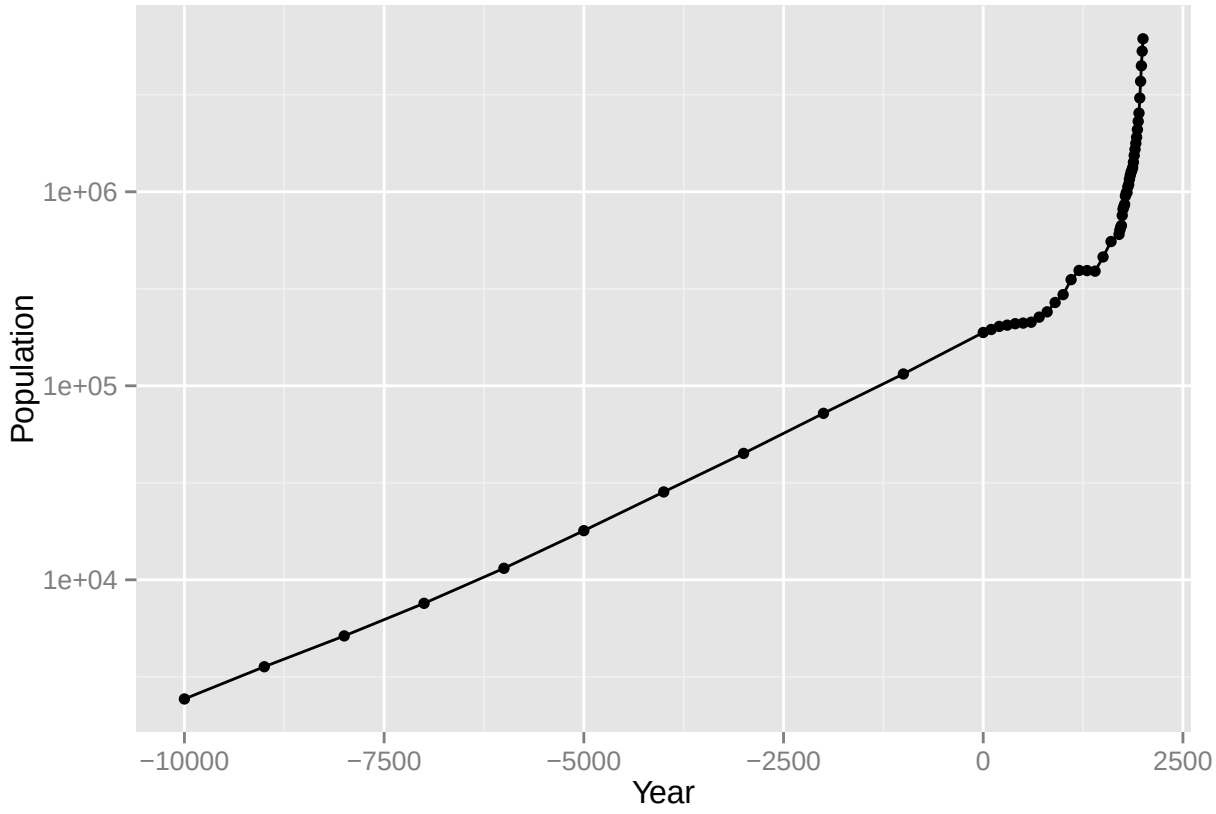
```
ggplot( BOD, aes( x = Time, y = demand)) + geom_line() +
  geom_point()
```




```
ggplot( worldpop, aes( x = Year, y = Population)) + geom_line() +
  geom_point()
```

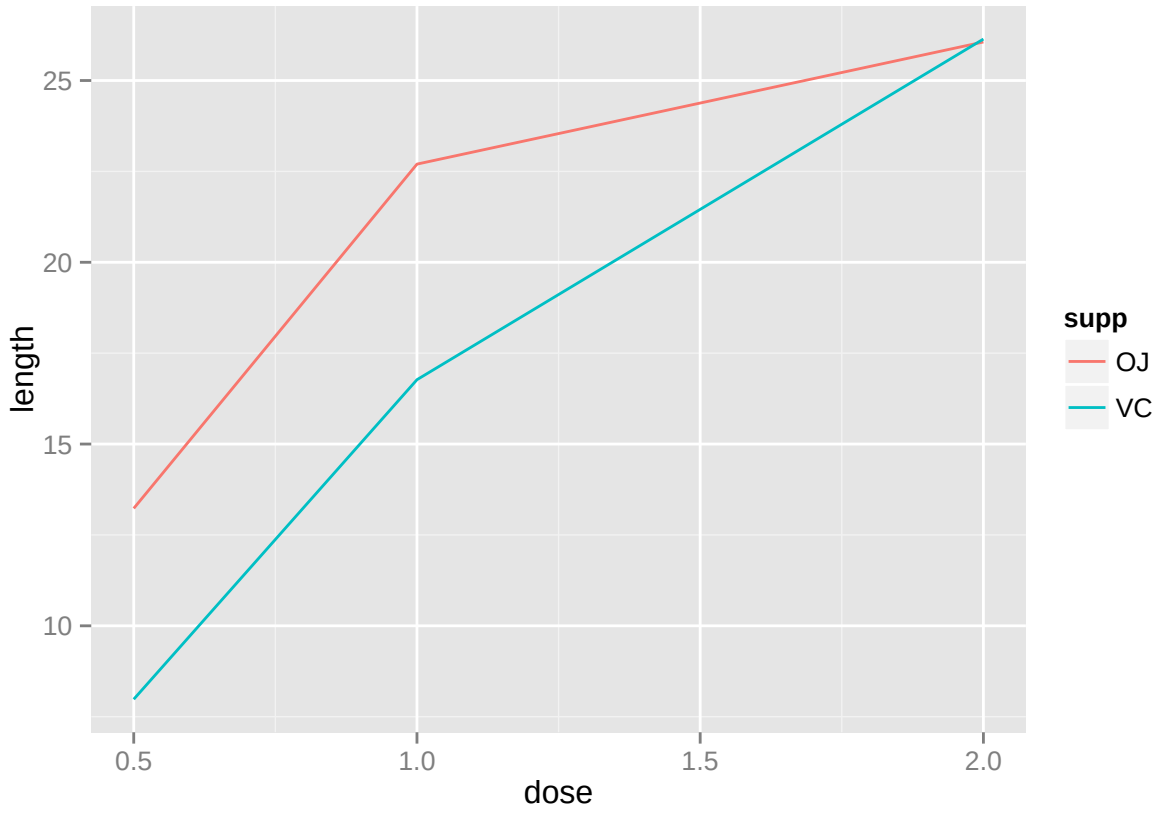


```
# Same with a log y-axis
ggplot( worldpop, aes( x = Year, y = Population)) + geom_line() +
  geom_point() +
  scale_y_log10()
```

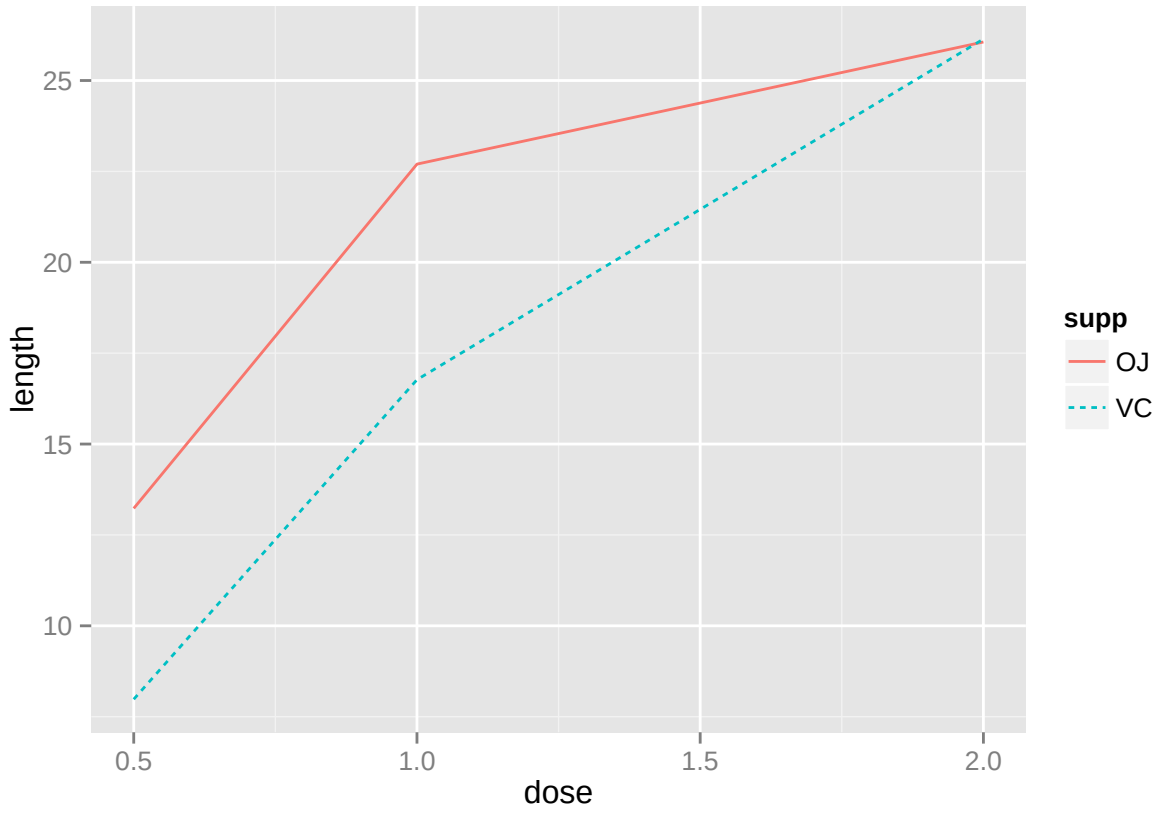


3. Making a Line Graph with Multiple Lines

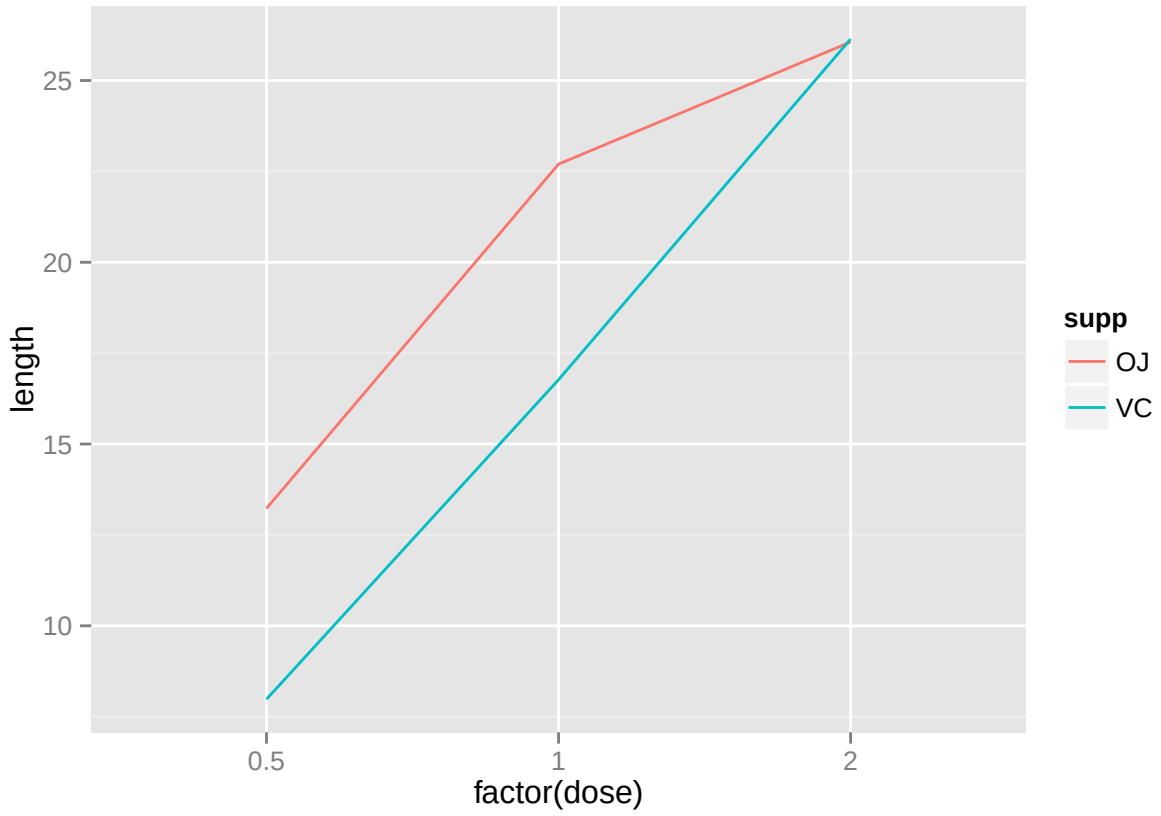
```
# Summarize the ToothGrowth data
tg <- ddply( ToothGrowth, c("supp", "dose"), summarise, length = mean( len))

# Map supp to colour
ggplot( tg, aes( x = dose, y = length, colour = supp)) +
  geom_line()
```



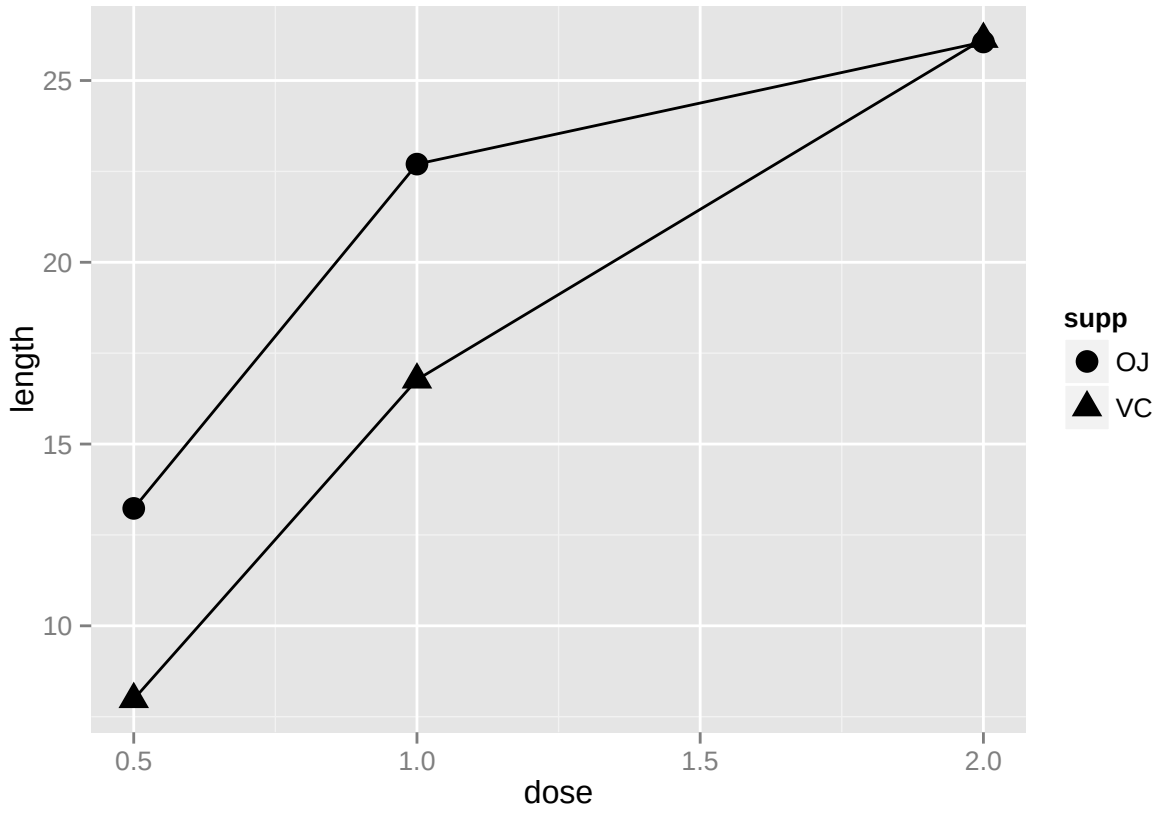

```
# Map supp to linetype
ggplot( tg, aes( x = dose, y = length, linetype = supp , colour = supp)) +
  geom_line()
```



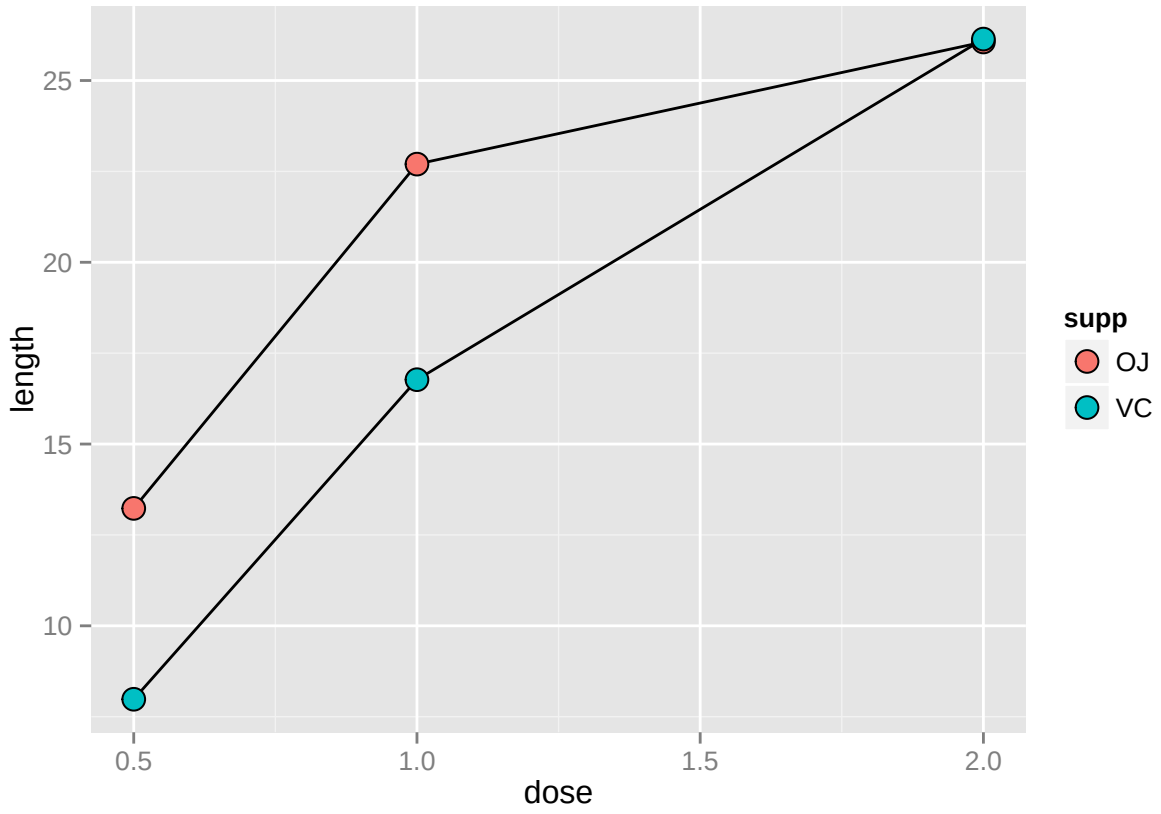

```
## x-axis is conceived of as being categorical,
ggplot( tg, aes( x = factor( dose), y = length, colour = supp, group = supp)) +
  geom_line()
```




```
## shape
ggplot( tg, aes( x = dose, y = length, shape = supp)) +
  geom_line() +
  geom_point( size = 4)
```

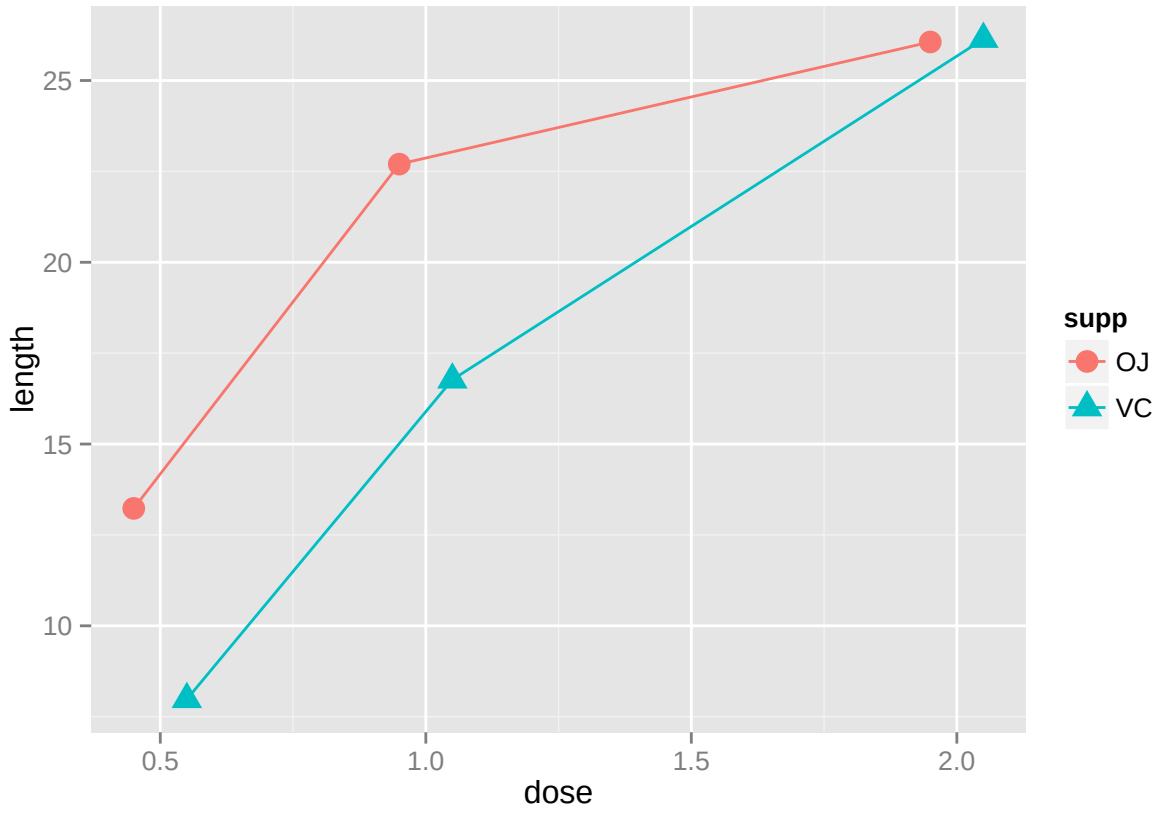



```
# Make the points a little larger
ggplot( tg, aes( x = dose, y = length, fill = supp)) +
  geom_line() +
  geom_point( size = 4, shape = 21)
```



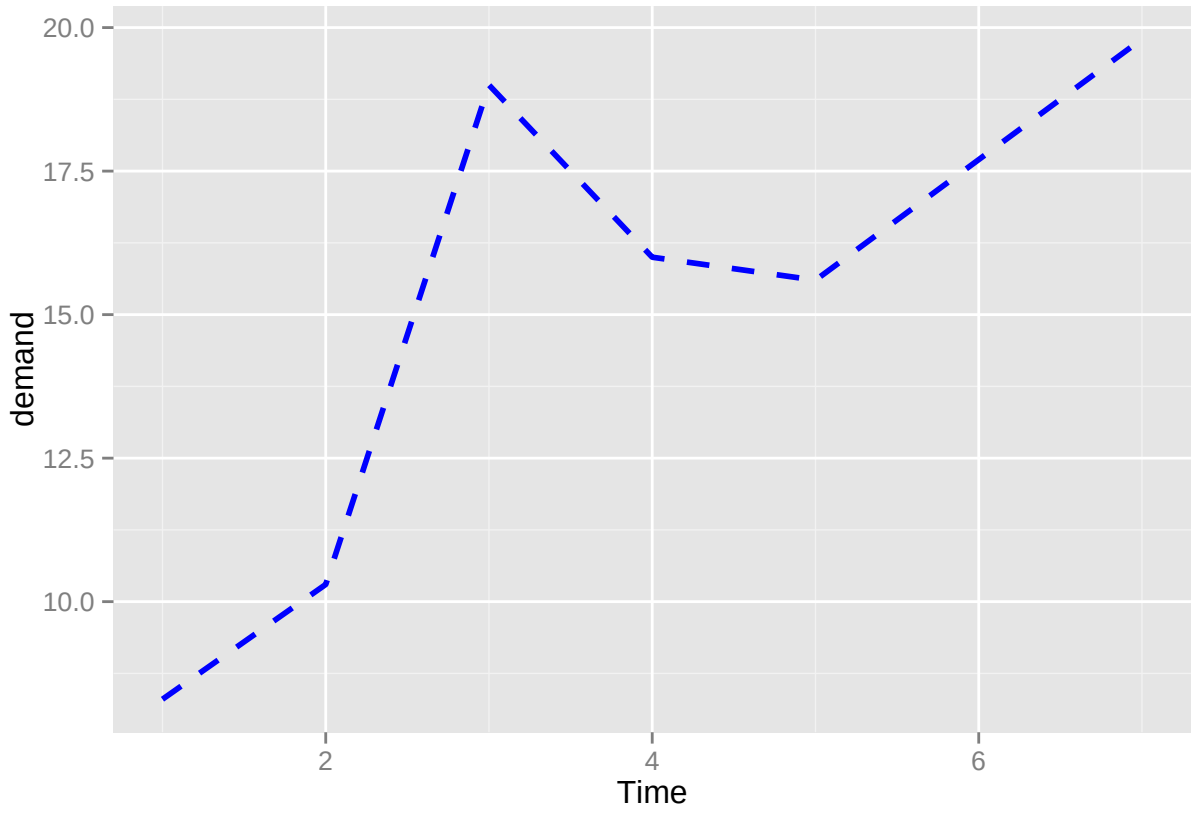

```
## in case points overlap dodge them
ggplot( tg, aes( x = dose, y = length, shape = supp , colour = supp )) + ## linetype = supp)) +
  geom_line( position = position_dodge( 0.2)) + # Dodge lines by 0.2
  geom_point( position = position_dodge( 0.2), size = 4) # Dodge points by 0.2
```

```
## ymax not defined: adjusting position using y instead
## ymax not defined: adjusting position using y instead
```

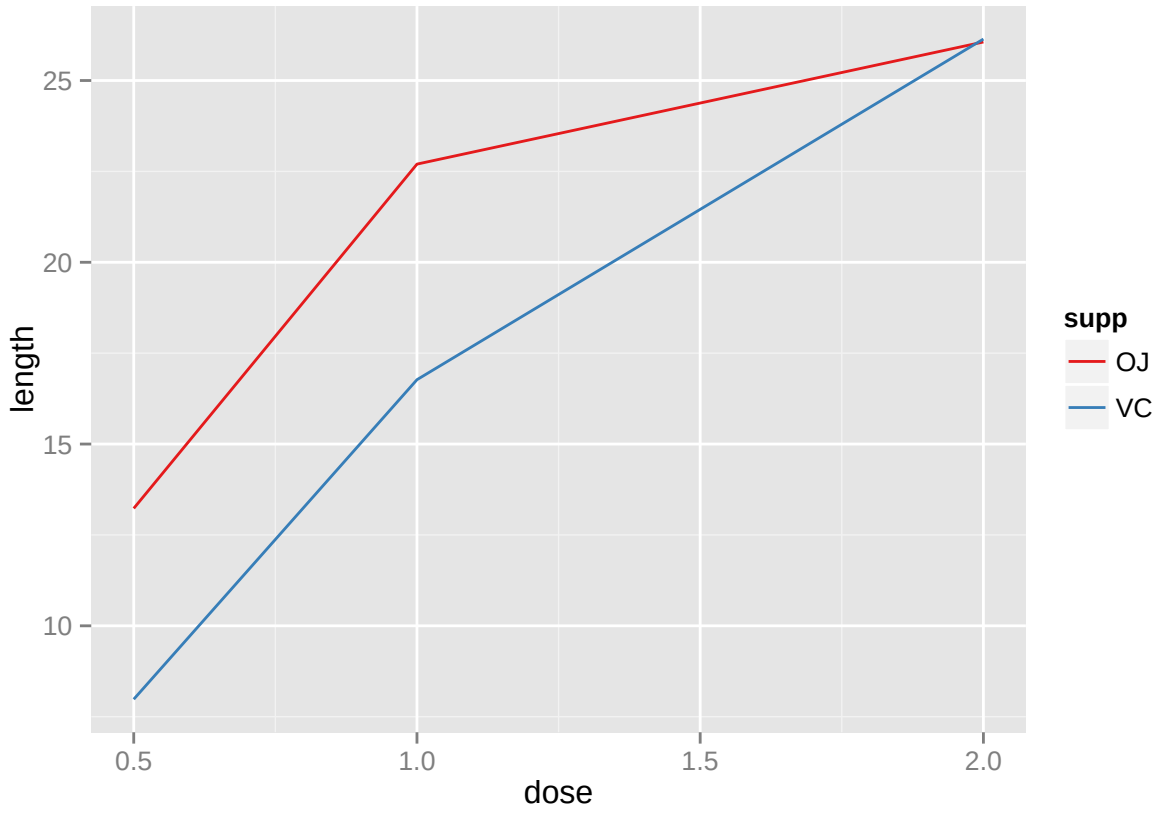


4. Changing the Appearance of Lines

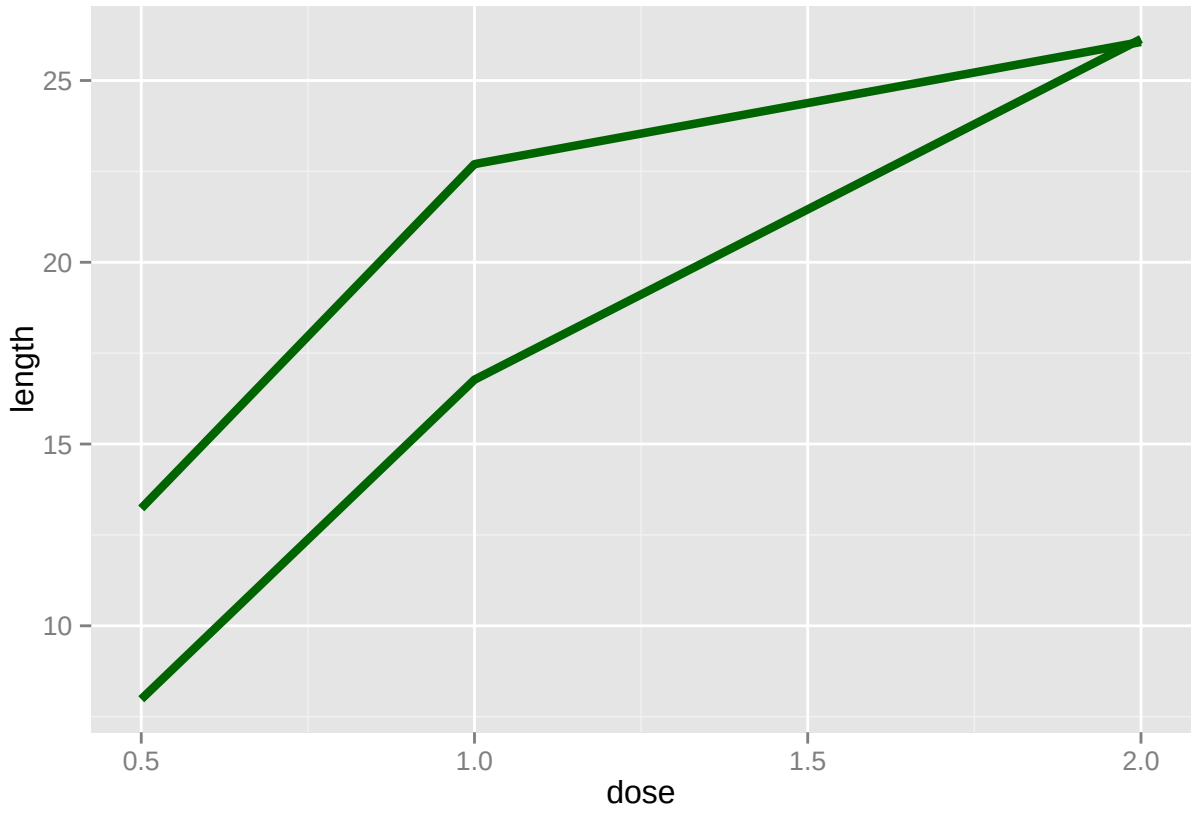
```
ggplot( BOD, aes( x = Time, y = demand)) +
  geom_line( linetype ="dashed", size = 1, colour ="blue")
```



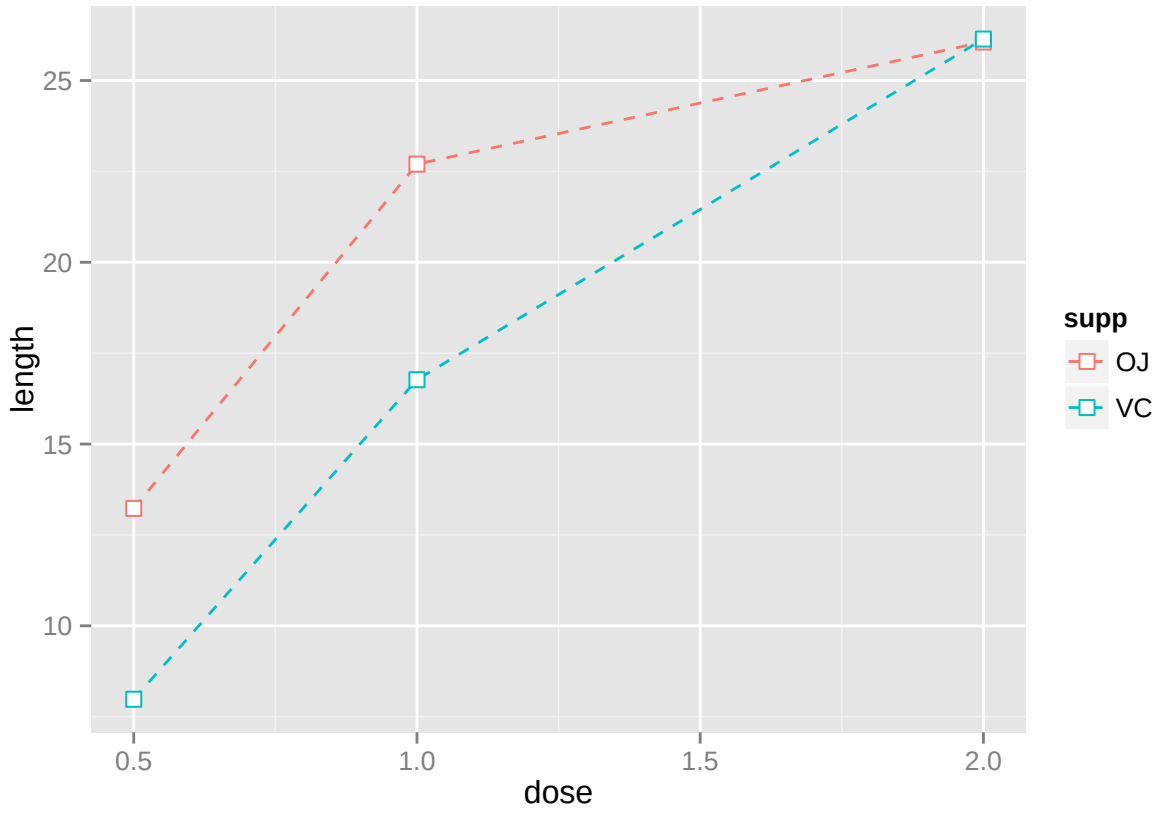

```
# Summarize the ToothGrowth data
tg <- ddply( ToothGrowth, c("supp", "dose"), summarise, length = mean( len))

ggplot( tg, aes( x = dose, y = length, colour = supp)) +
  geom_line() +
  scale_colour_brewer( palette ="Set1")
```




```
# If both lines have the same properties, you need to specify a variable to # use for grouping
ggplot( tg, aes( x = dose, y = length, group = supp)) +
  geom_line( colour ="darkgreen", size = 1.5)
```

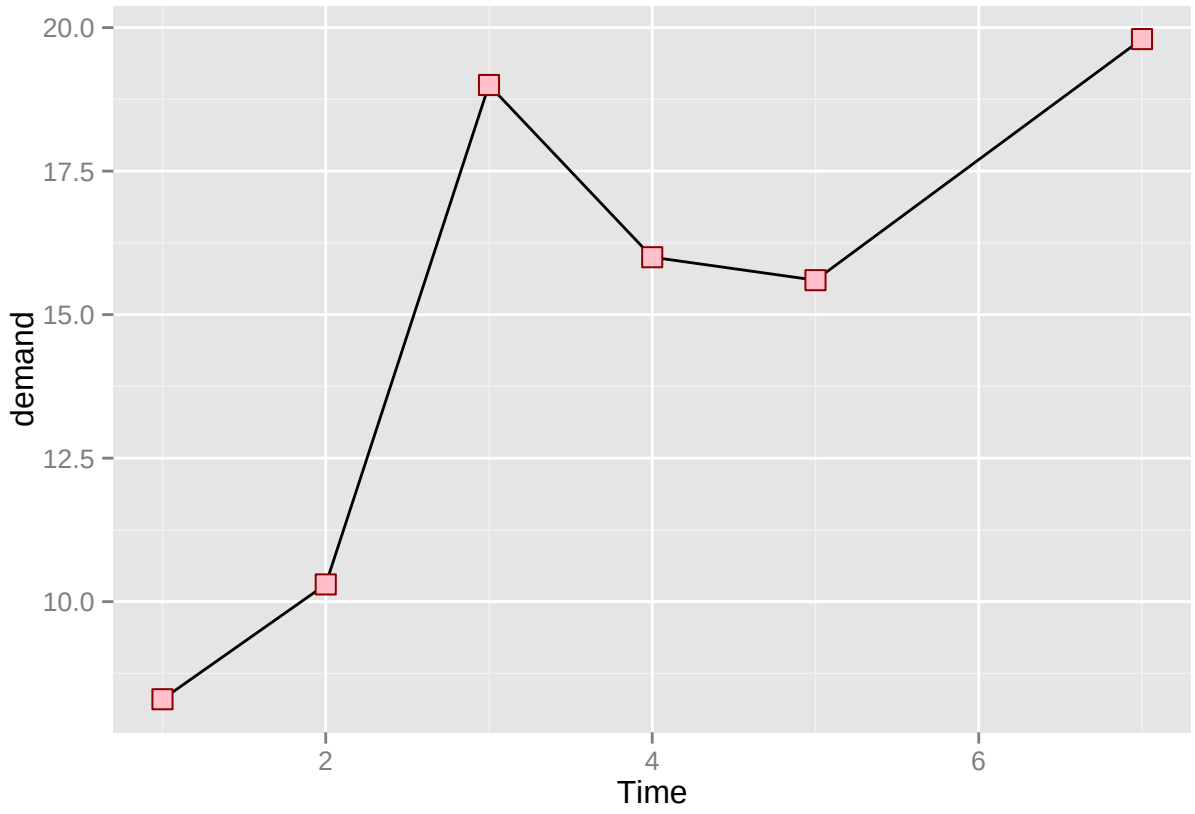



```
# Since supp is mapped to colour, it will automatically be used for grouping
ggplot( tg, aes( x = dose, y = length, colour = supp)) +
  geom_line( linetype ="dashed") +
  geom_point( shape = 22, size = 3, fill ="white")
```

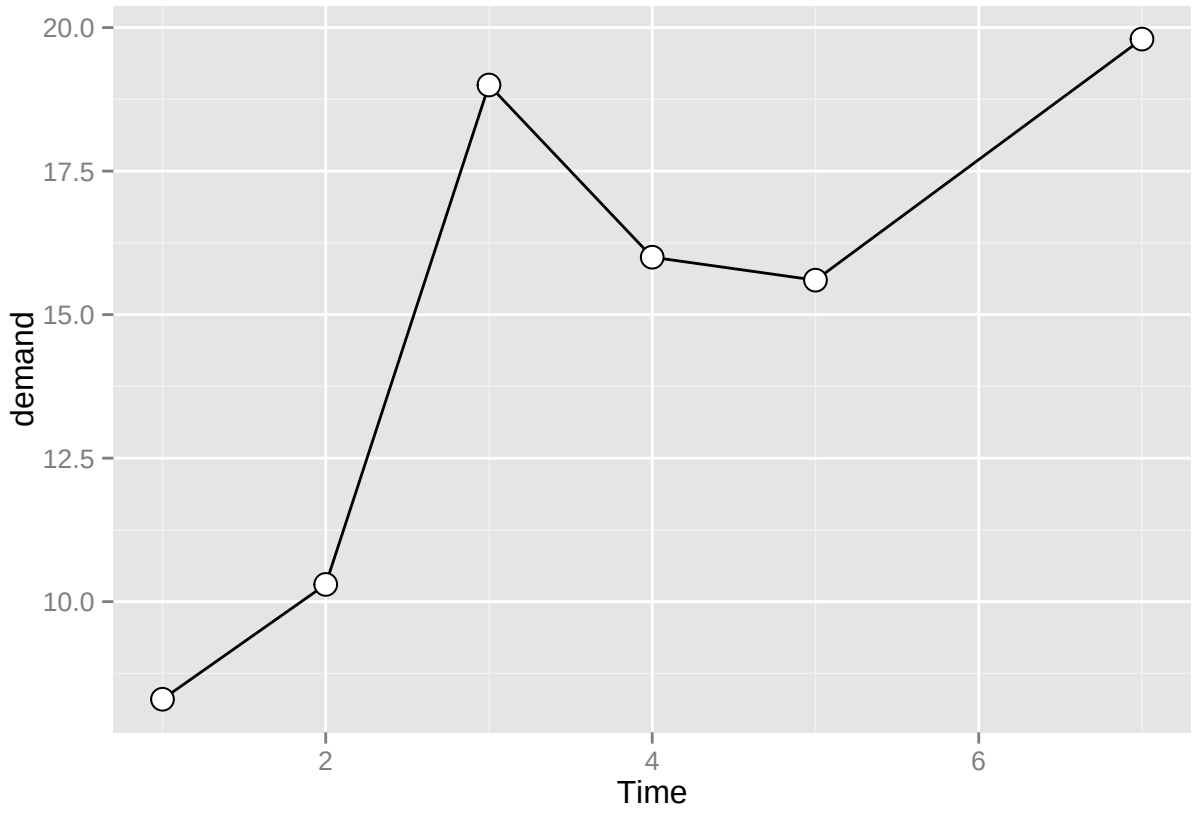


5. Changing the Appearance of Points

```
ggplot( BOD, aes( x = Time, y = demand)) +
  geom_line() +
  geom_point( size = 4, shape = 22, colour ="darkred", fill ="pink")
```

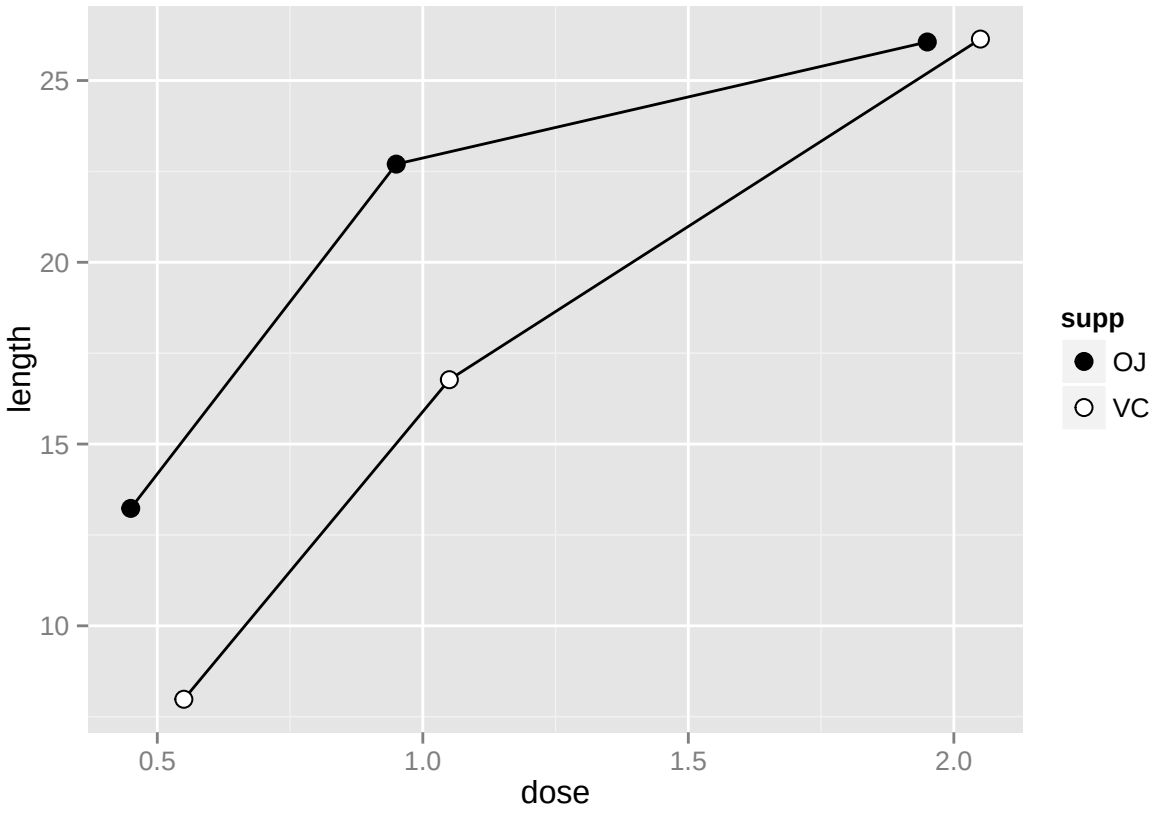



```
## The default shape for points is a solid circle, the default size is 2, and the default colour is "bla
## The fill color is relevant only for some point shapes (numbered 21- 25), which have separate outline
ggplot( BOD, aes( x = Time, y = demand)) +
  geom_line() +
  geom_point( size = 4, shape = 21, fill ="white")
```



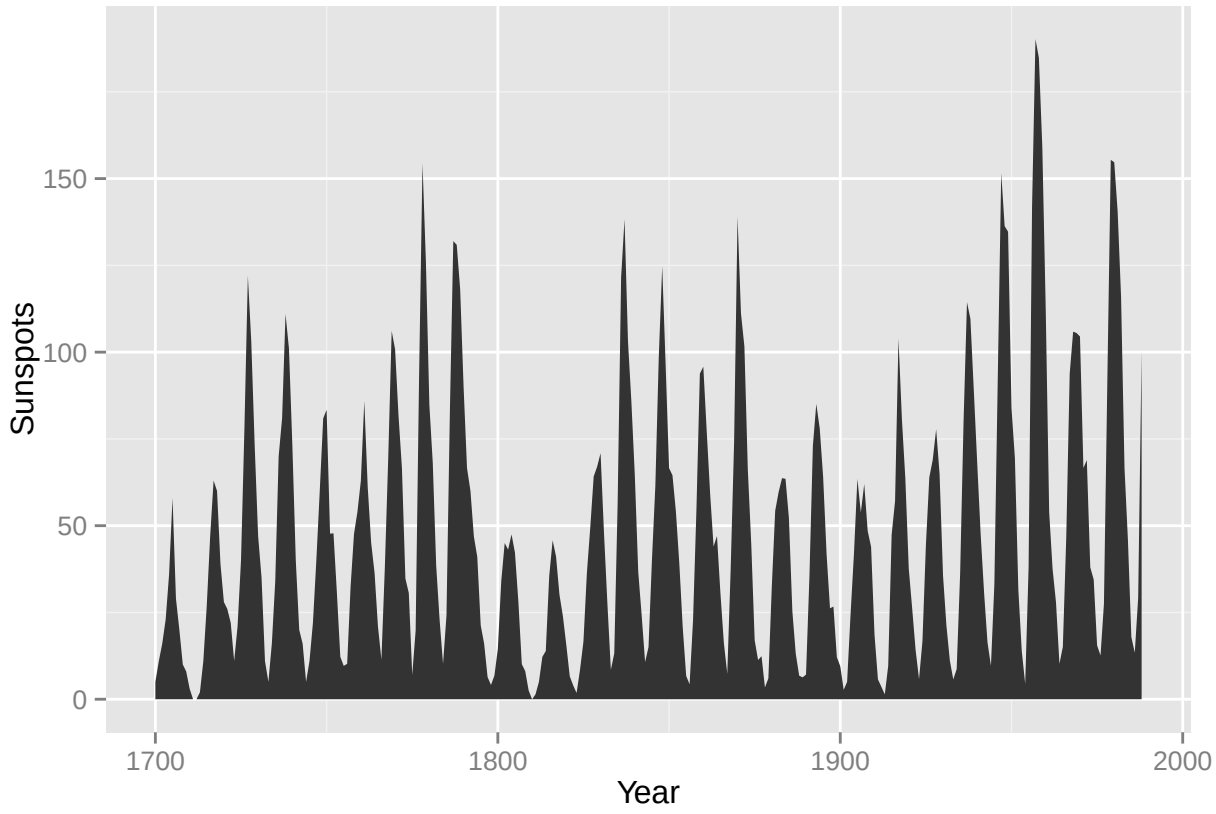

```
## If the points and lines have different colors, you should specify the points after the lines
pd <- position_dodge(0.2)
ggplot( tg, aes( x = dose, y = length, fill = supp)) +
  geom_line( position = pd) +
  geom_point( shape = 21, size = 3, position = pd) +
  scale_fill_manual( values = c("black","white"))
```

```
## ymax not defined: adjusting position using y instead
## ymax not defined: adjusting position using y instead
```

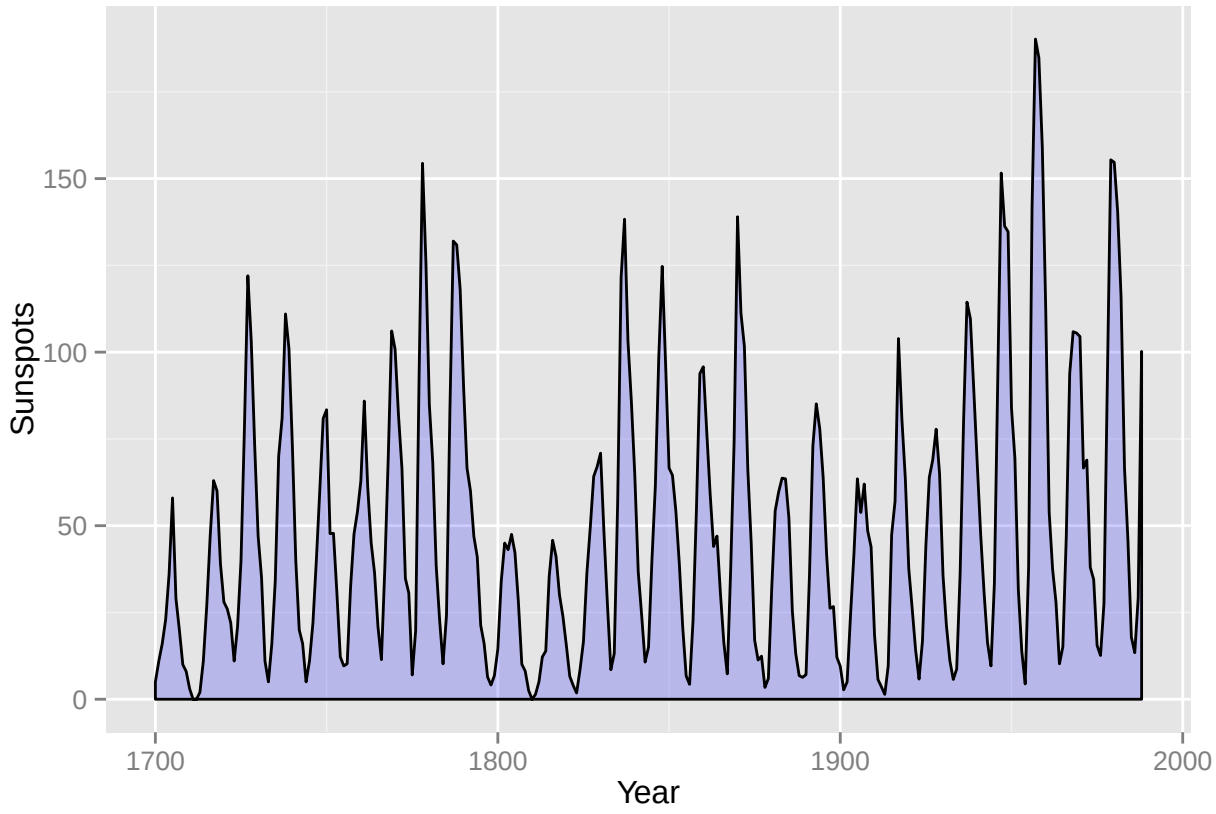


6. Making a Graph with a Shaded Area

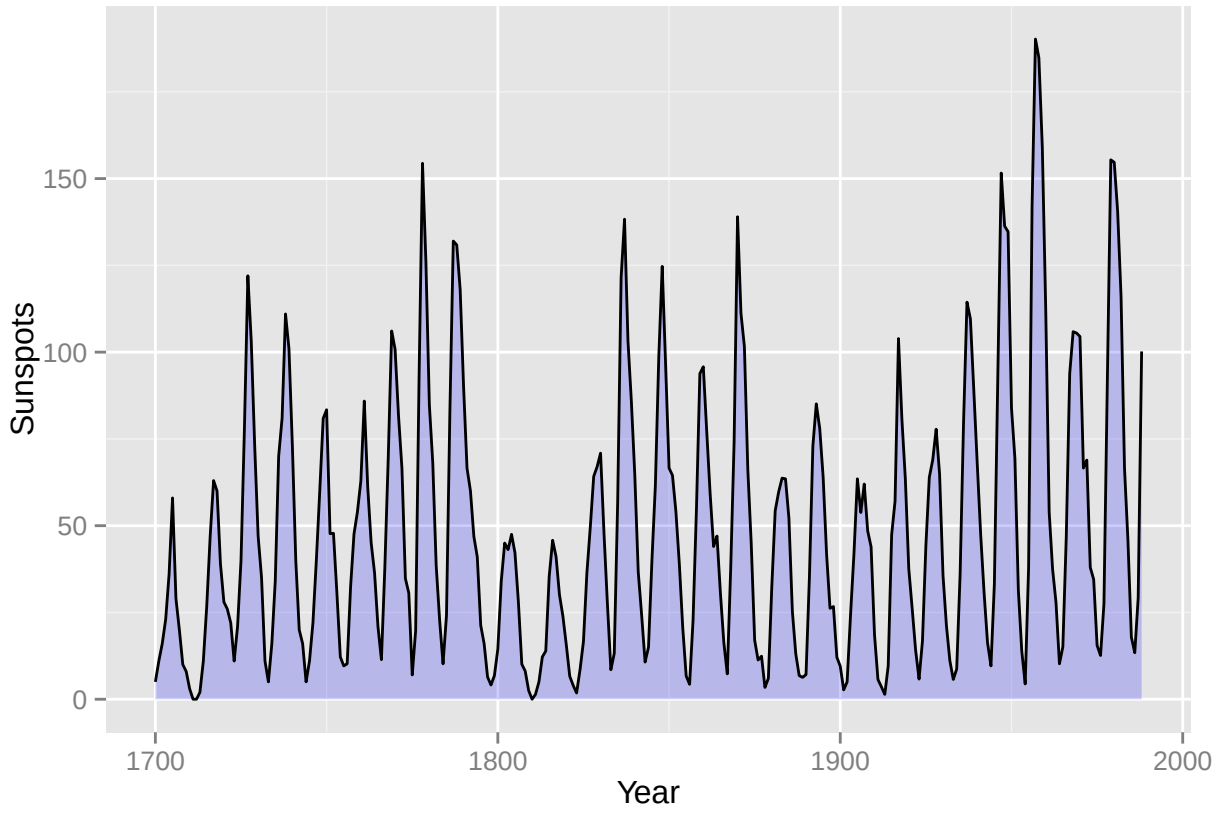
```
# Convert the sunspot.year data set into a data frame for this example
sunspotyear <- data.frame( Year = as.numeric( time( sunspot.year)), Sunspots = as.numeric( sunspot.year

ggplot( sunspotyear, aes( x = Year, y = Sunspots)) +
  geom_area()
```




```
## We'll also add an outline, by setting colour:
ggplot( sunspotyear, aes( x = Year, y = Sunspots)) +
  geom_area( colour ="black", fill ="blue", alpha =.2)
```




```
## Having an outline around the entire area might not be desirable,
ggplot( sunspotyear, aes( x = Year, y = Sunspots)) +
  geom_area( fill ="blue", alpha =.2) +
  geom_line()
```

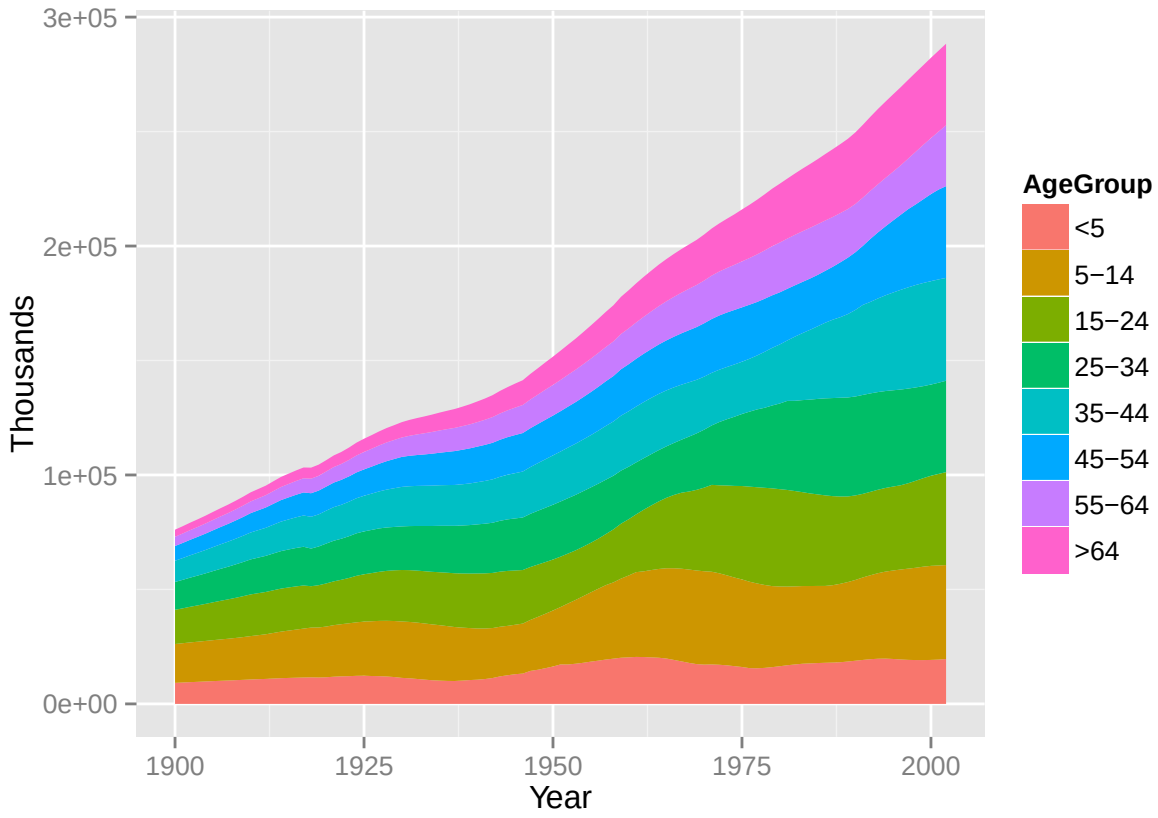


7. Making a Stacked Area Graph

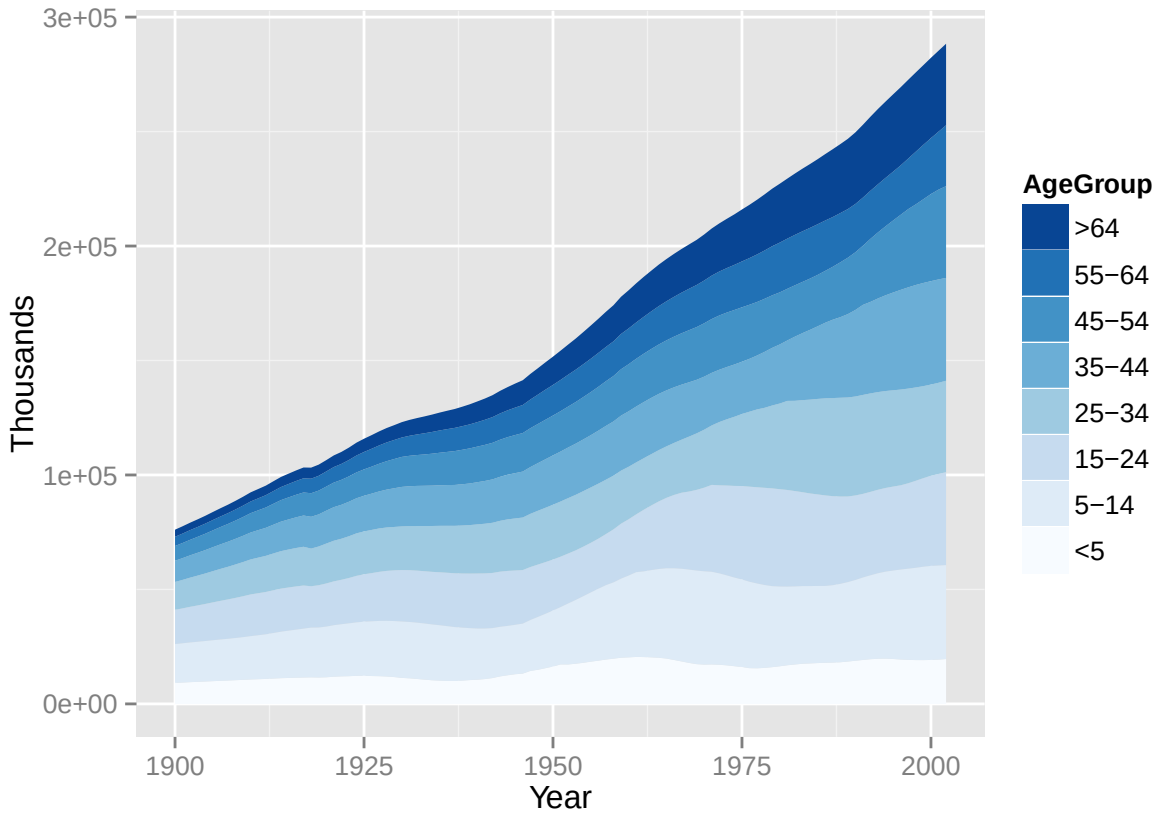
```
str(uspopage) ## AgeGroup is a factor
```

```
## 'data.frame':    824 obs. of  3 variables:
##  $ Year     : int  1900 1900 1900 1900 1900 1900 1900 1900 1901 1901 ...
##  $ AgeGroup : Factor w/ 8 levels "<5","5-14","15-24",..: 1 2 3 4 5 6 7 8 1 2 ...
##  $ Thousands: int  9181 16966 14951 12161 9273 6437 4026 3099 9336 17158 ...
```

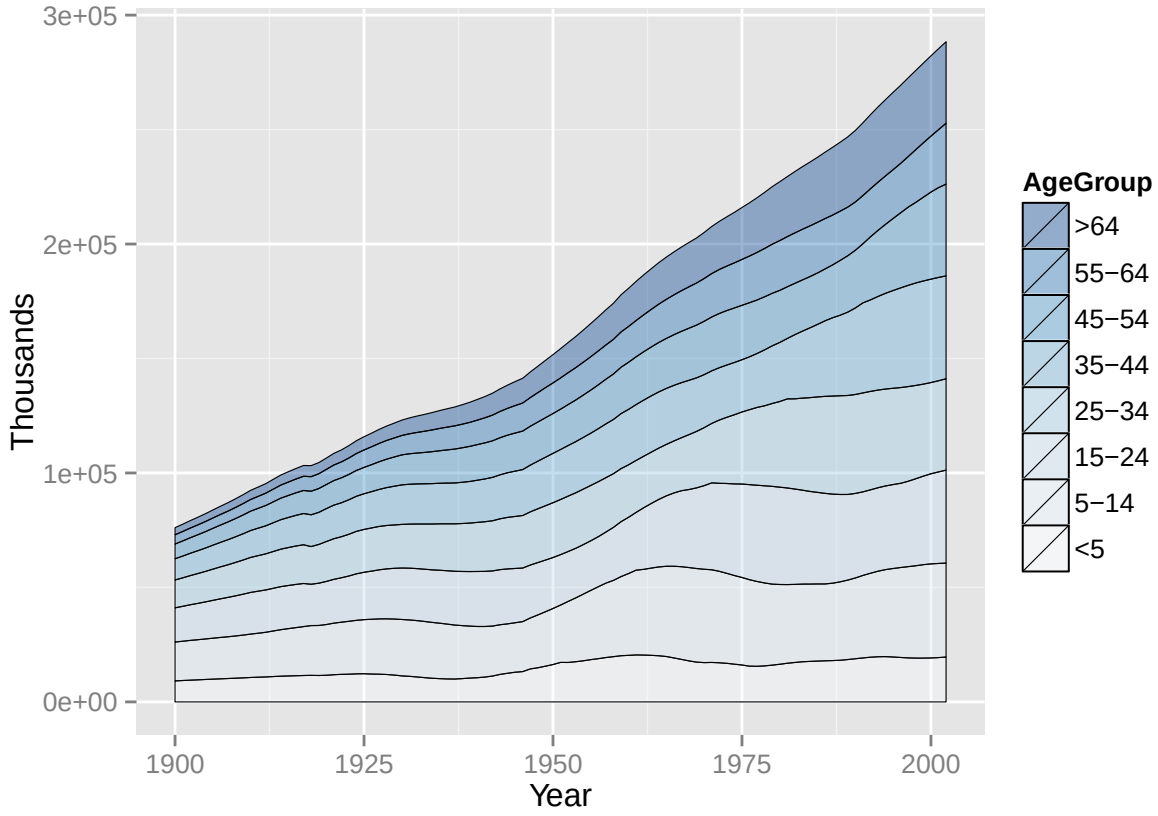
```
ggplot( uspopage, aes( x = Year, y = Thousands, fill = AgeGroup) ) +
  geom_area() ##+ scale_fill_brewer( breaks = rev( levels( uspopage$AgeGroup)))
```



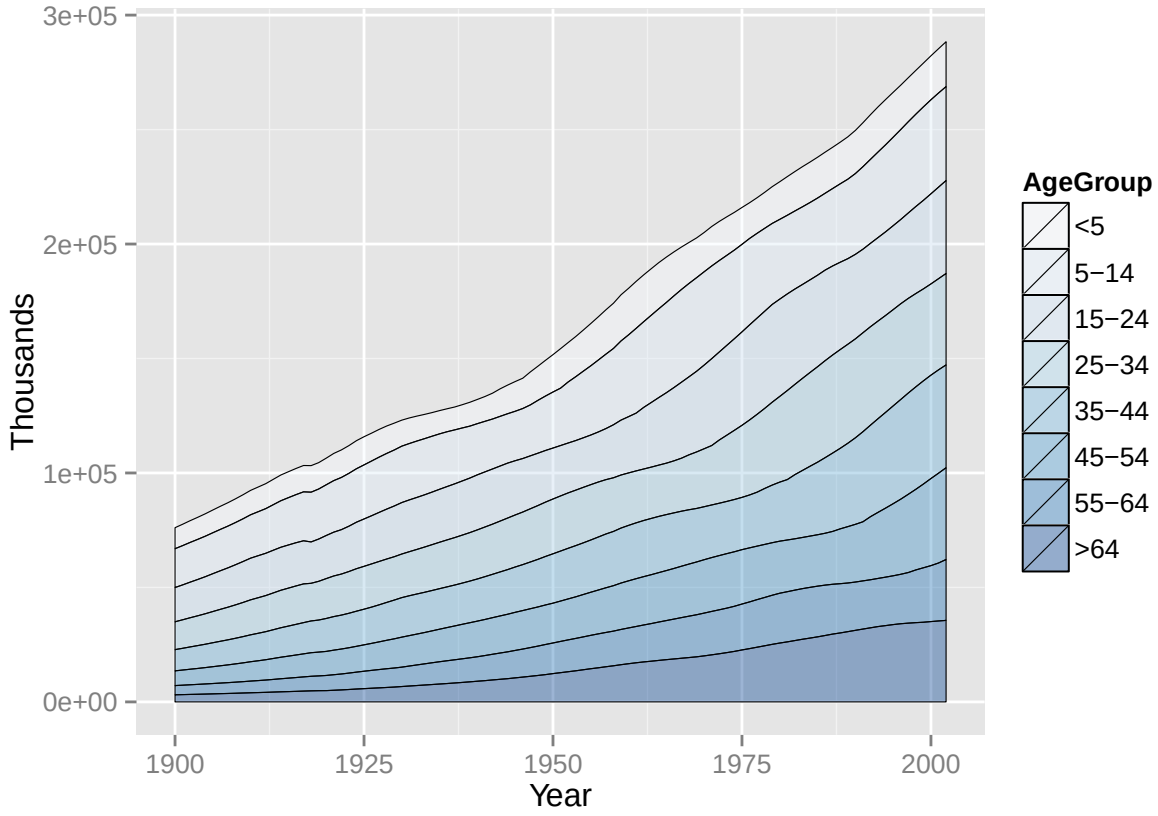

```
ggplot( uspopage, aes( x = Year, y = Thousands, fill = AgeGroup) ) +
  geom_area() +
  scale_fill_brewer( breaks = rev( levels( uspopage$AgeGroup)))
```




```
## The legend can be reversed by setting the breaks in the scale.
ggplot( uspopage, aes( x = Year, y = Thousands, fill = AgeGroup)) +
  geom_area( colour ="black", size =.2, alpha =.4) +
  scale_fill_brewer( palette ="Blues", breaks = rev( levels( uspopage$AgeGroup)))
```

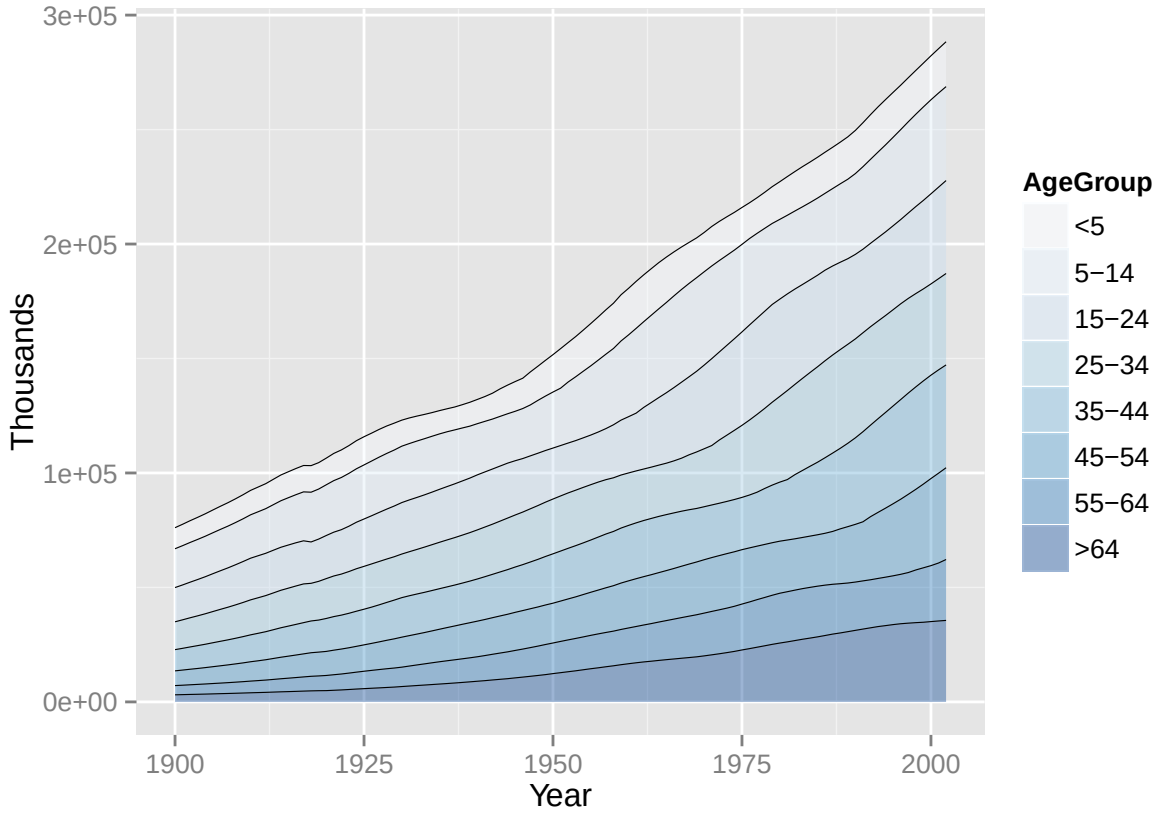



```
## To reverse the stacking order, we'll put order = desc( AgeGroup)
ggplot( uspopage, aes( x = Year, y = Thousands, fill = AgeGroup, order = desc( AgeGroup))) +
  geom_area( colour ="black", size =.2, alpha =.4) +
  scale_fill_brewer( palette ="Blues")
```



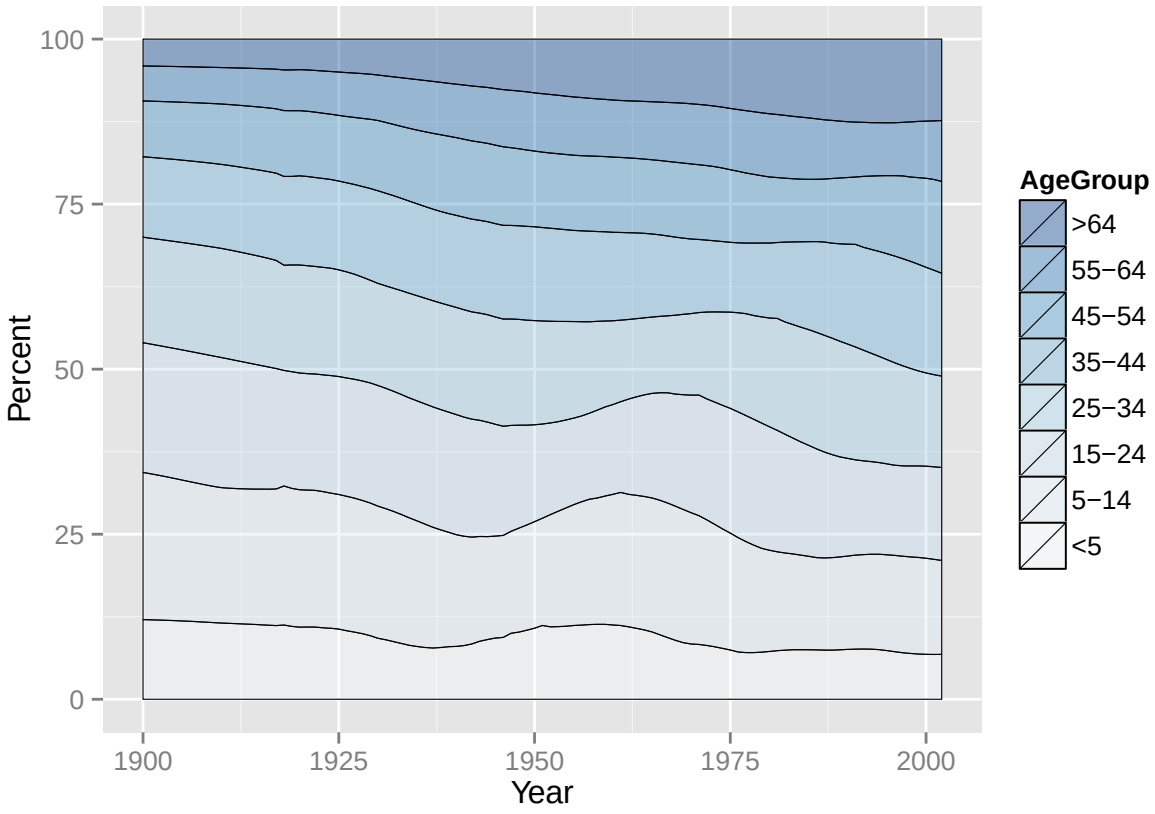

```
## Since each filled area is drawn with a polygon, the outline includes the left and right sides. This r
ggplot( uspopage, aes( x = Year, y = Thousands, fill = AgeGroup, order = desc( AgeGroup))) +
  geom_area( colour = NA, alpha =.4) +
  scale_fill_brewer( palette ="Blues") +
  geom_line( position ="stack", size =.2)
```

```
## ymax not defined: adjusting position using y instead
```



8. Making a Proportional Stacked Area Graph

```
# Convert Thousands to Percent
uspopage_prop <- ddply( uspopage, "Year", transform, Percent = Thousands / sum( Thousands) * 100)

ggplot( uspopage_prop, aes( x = Year, y = Percent, fill = AgeGroup)) +
  geom_area( colour =" black", size =.2, alpha =.4) +
  scale_fill_brewer( palette ="Blues", breaks = rev( levels( uspopage$AgeGroup)))
```

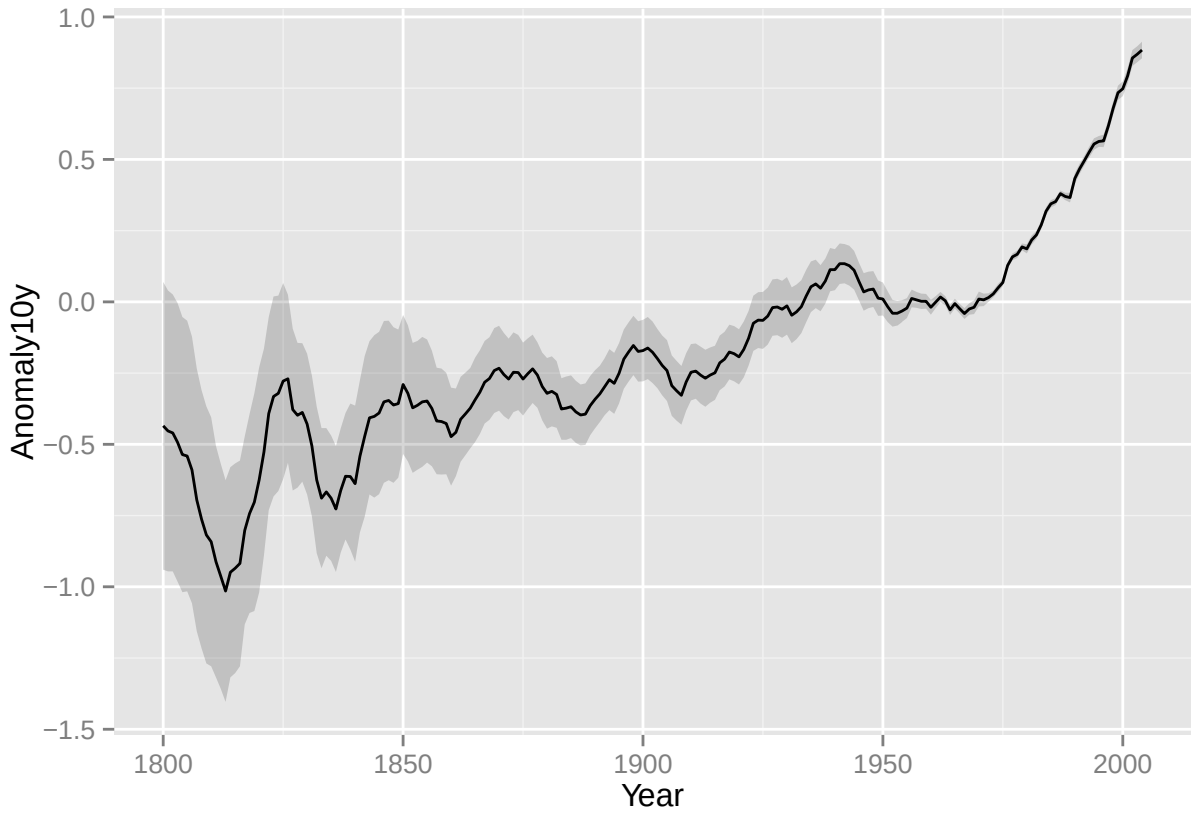


9. Adding a Confidence Region

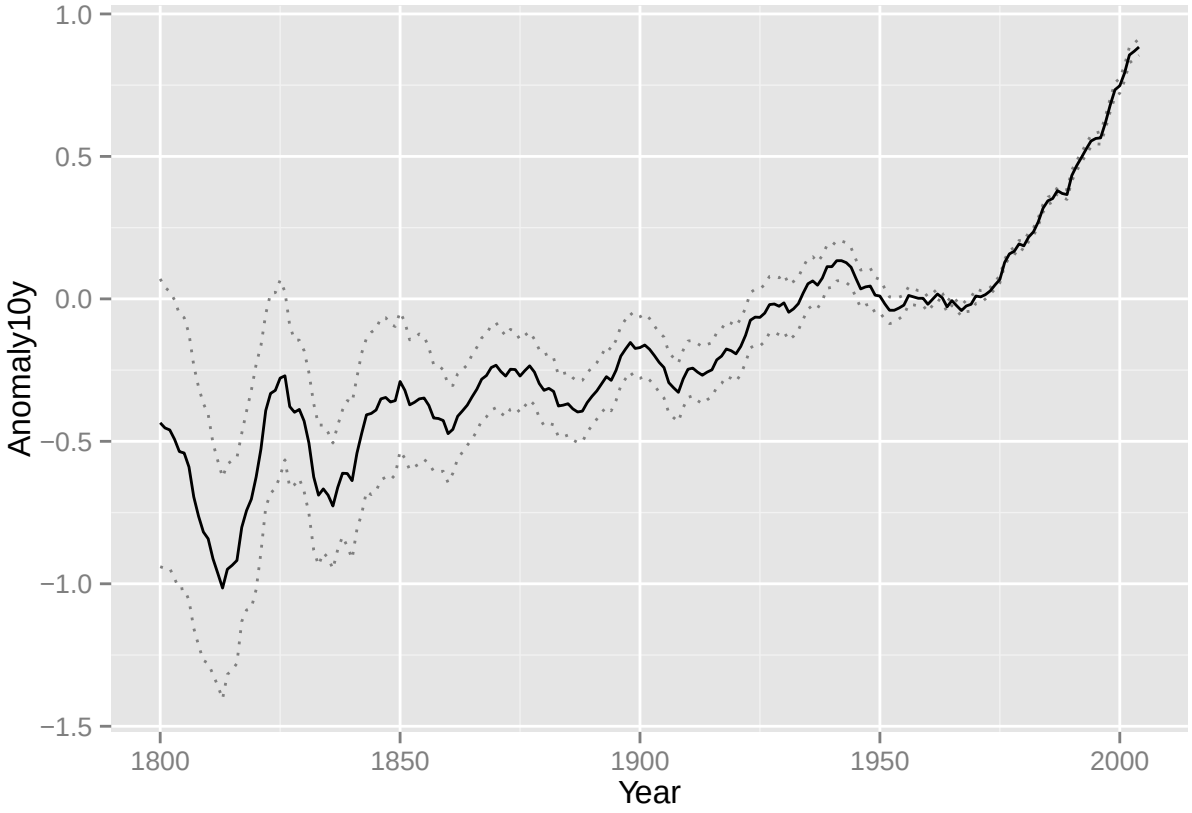
```
# Grab a subset of the climate data
clim <- subset( climate, Source == "Berkeley", select = c("Year", "Anomaly10y", "Unc10y"))
str(clim)
```

```
## 'data.frame':    205 obs. of  3 variables:
##  $ Year      : num  1800 1801 1802 1803 1804 ...
##  $ Anomaly10y: num  -0.435 -0.453 -0.46 -0.493 -0.536 -0.541 -0.59 -0.695 -0.763 -0.818 ...
##  $ Unc10y    : num  0.505 0.493 0.486 0.489 0.483 0.475 0.468 0.461 0.453 0.451 ...
```

```
## Anomaly10y is a 10-year running average of the deviation
## Unc10y is the 95% confidence interval of Anomaly10y

# Shaded region
ggplot( clim, aes( x = Year, y = Anomaly10y)) +
  geom_ribbon( aes( ymin = Anomaly10y-Unc10y, ymax = Anomaly10y + Unc10y), alpha = 0.2) +
  geom_line()
```




```
# With a dotted line for upper and lower bounds
ggplot( clim, aes( x = Year, y = Anomaly10y)) +
  geom_line( aes( y = Anomaly10y-Unc10y), colour ="grey50", linetype ="dotted") +
  geom_line( aes( y = Anomaly10y + Unc10y), colour ="grey50", linetype ="dotted") +
  geom_line()
```


1.0
0.5
0.0
−0.5
−1.0
−1.5
Anomaly10y
1800
1850
1900
1950
2000
Year