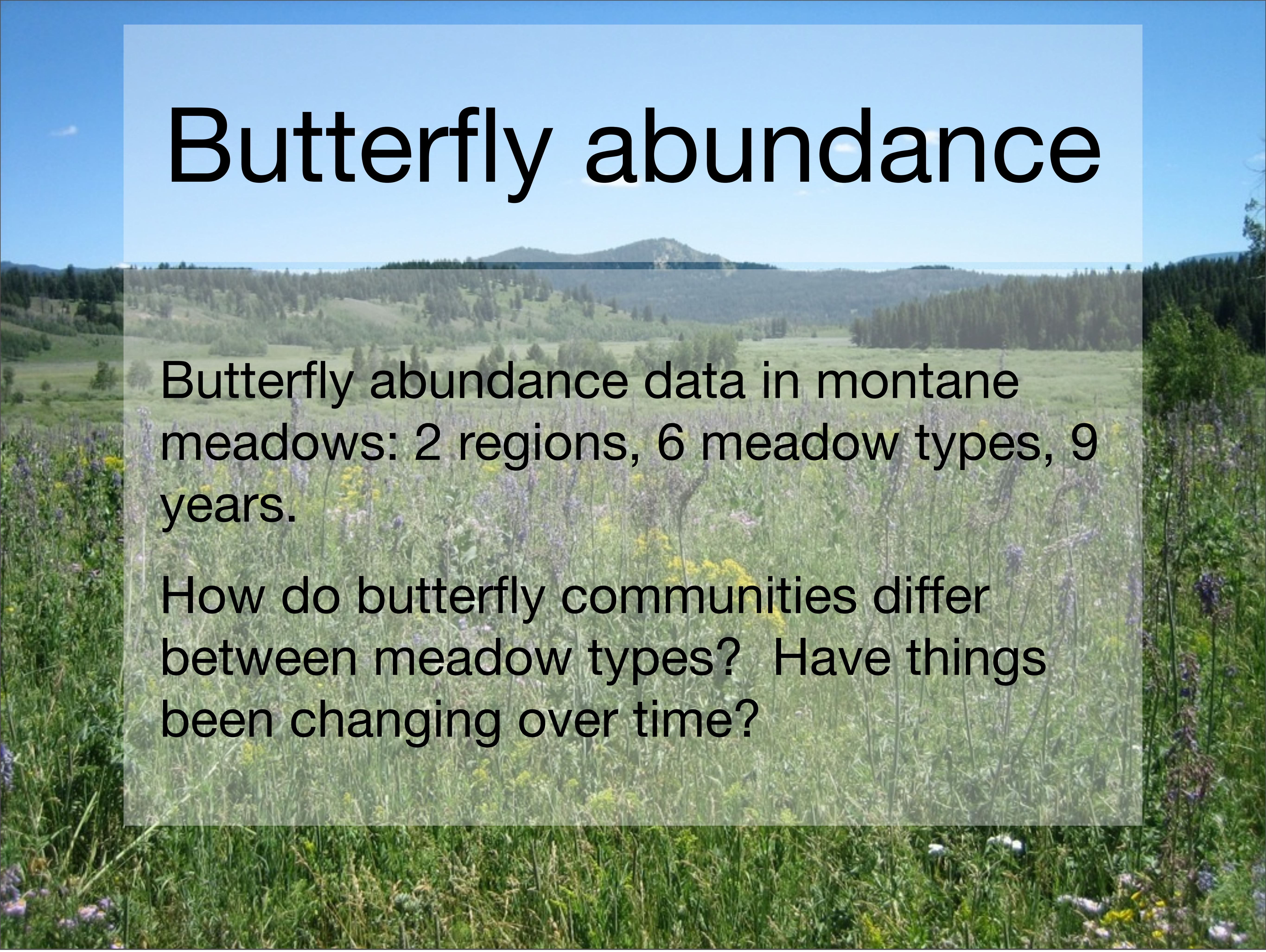
Model visualisation

Hadley Wickham

October 2009

1. Model visualisation
2. Extracting data from a single model: butterfly abundance
3. Fitting and visualising multiple models: Texas housing data



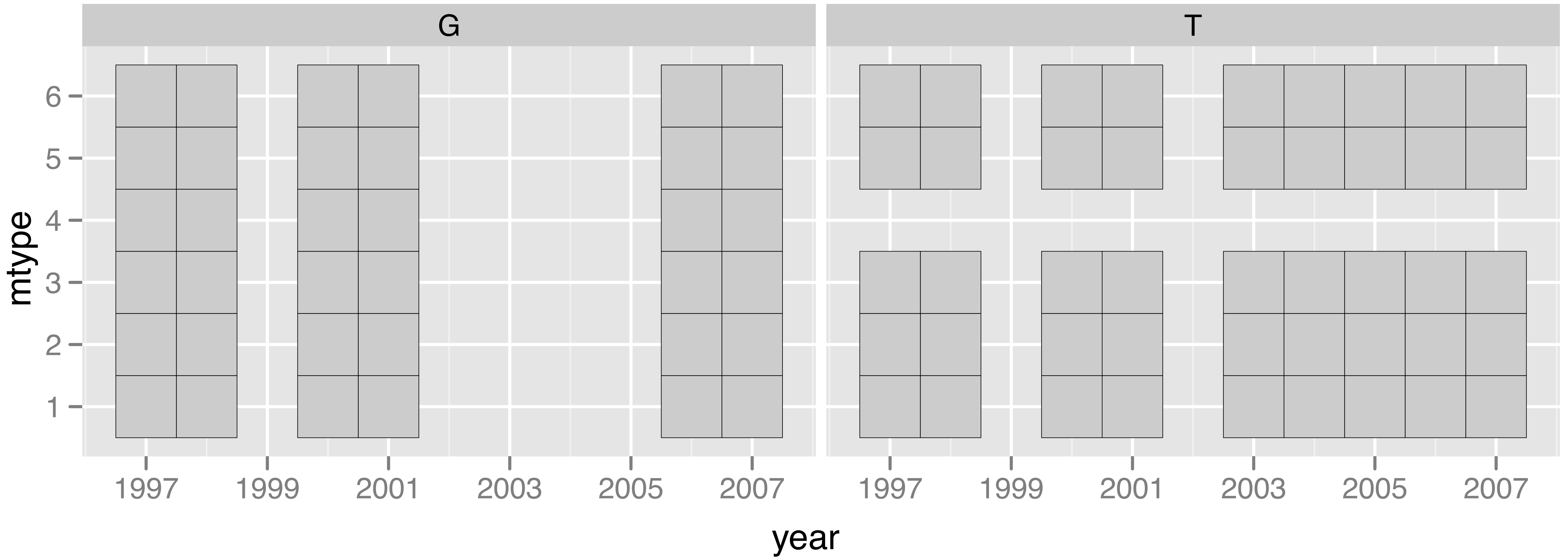
Butterfly abundance

Butterfly abundance data in montane meadows: 2 regions, 6 meadow types, 9 years.

How do butterfly communities differ between meadow types? Have things been changing over time?

Getting started

```
library(ggplot2)
library(vegan)

b <- read.csv(”butterflies.csv”)
b$mtype <- factor(b$mtype)
```



```
ggplot(b, aes(year, mtype)) +
  geom_tile(fill = "grey80", colour = "black") +
  facet_wrap(~ region) +
  scale_x_continuous(breaks = seq(1997, 2007, by = 2))
```

```
# Perform 2-d ordination with vegan package
mds <- metaMDS(b[, -(1:3)], k = 2,
  distance = 'bray', autotransform = F, expand = F)

plot(mds) # Ugh!
?plot.metaMDS

# How can we create a similar plot in ggplot2?
# ggplot2 works exclusively with data frames
# so we need to figure out how to extract the data
# of interest.  As usual, str is our friend.

str(mds)

sites <- as.data.frame(mds$points)
ord <- cbind(b, sites)
names(ord)

# This is a very simple case, but it illustrates
# what you might have to do.  Joining back to the
# original data usually most useful for plotting.
```

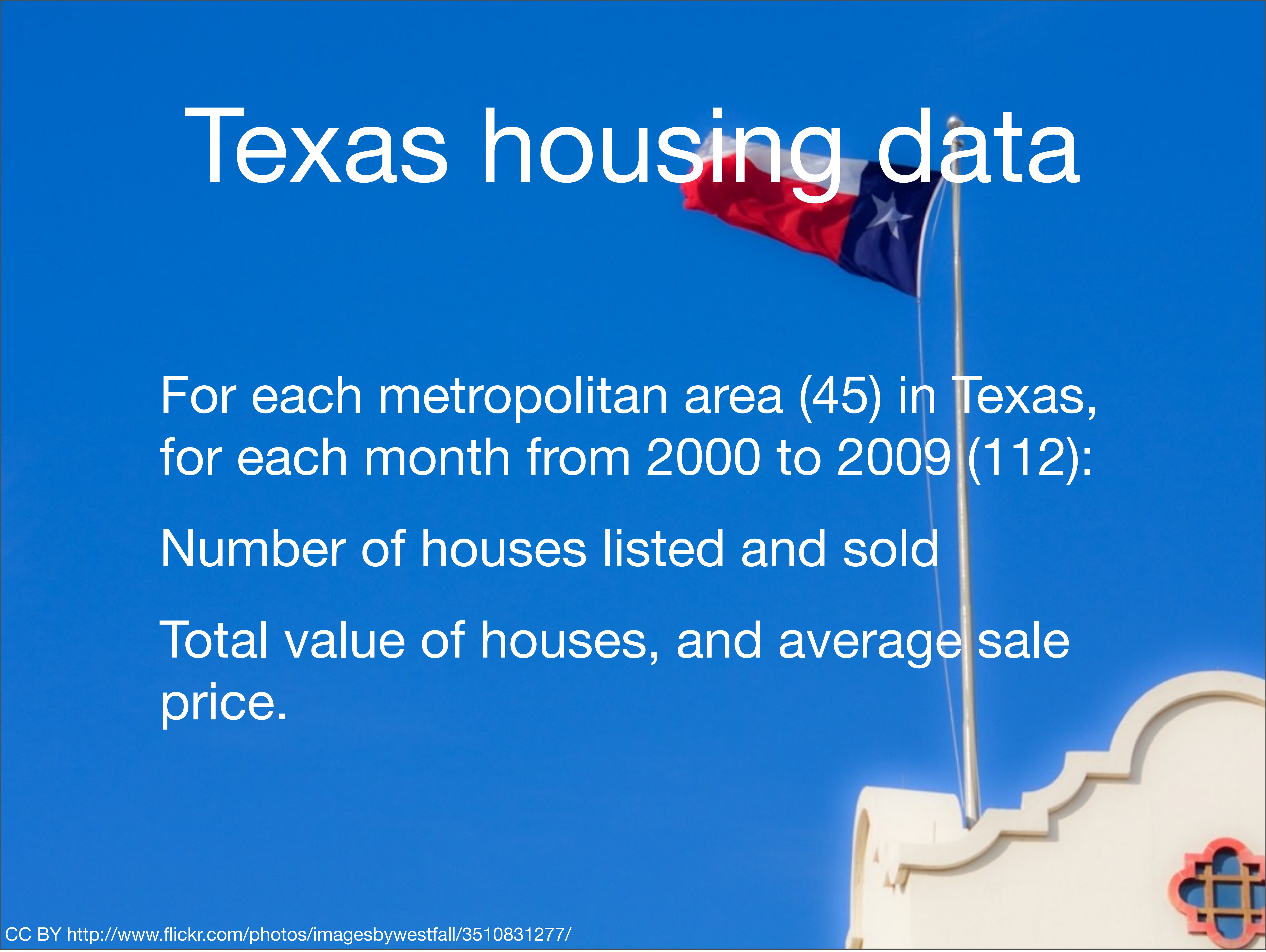
Your turn

What can you say about the differences between meadow types? Is there a difference between regions?

Is the pattern changing over time?

```
qplot(V1, V2, data = ord)
qplot(V1, V2, data = ord, colour = mtype)
qplot(V1, V2, data = ord, colour = mtype) +
  facet_wrap(~ region)

qplot(V1, V2, data = ord, colour = year)
qplot(V1, V2, data = ord, colour = region)
qplot(V1, V2, data = ord, colour = region) +
  facet_wrap(~ mtype)

qplot(V1, V2, data = ord, size = Plebejus.saepiolus) +
  scale_area()
qplot(V1, V2, data = ord, size = Speyeria.mormonia) +
  scale_area()
```

```
teutons <- subset(ord, region == "T")

qplot(V1, V2, data = teutons, colour = mtype, geom = "path",
  arrow = arrow(length = unit(0.05, "npc")))

# Compute velocities / deltas
delta <- ddply(teutons, "mtype", summarise,
  V1_v = diff(V1) / diff(year),
  V2_v = diff(V2) / diff(year))

qplot(V1_v, V2_v, data = delta, colour = mtype,
  geom = "path")
```



Texas housing data

For each metropolitan area (45) in Texas, for each month from 2000 to 2009 (112):

Number of houses listed and sold

Total value of houses, and average sale price.

Getting started

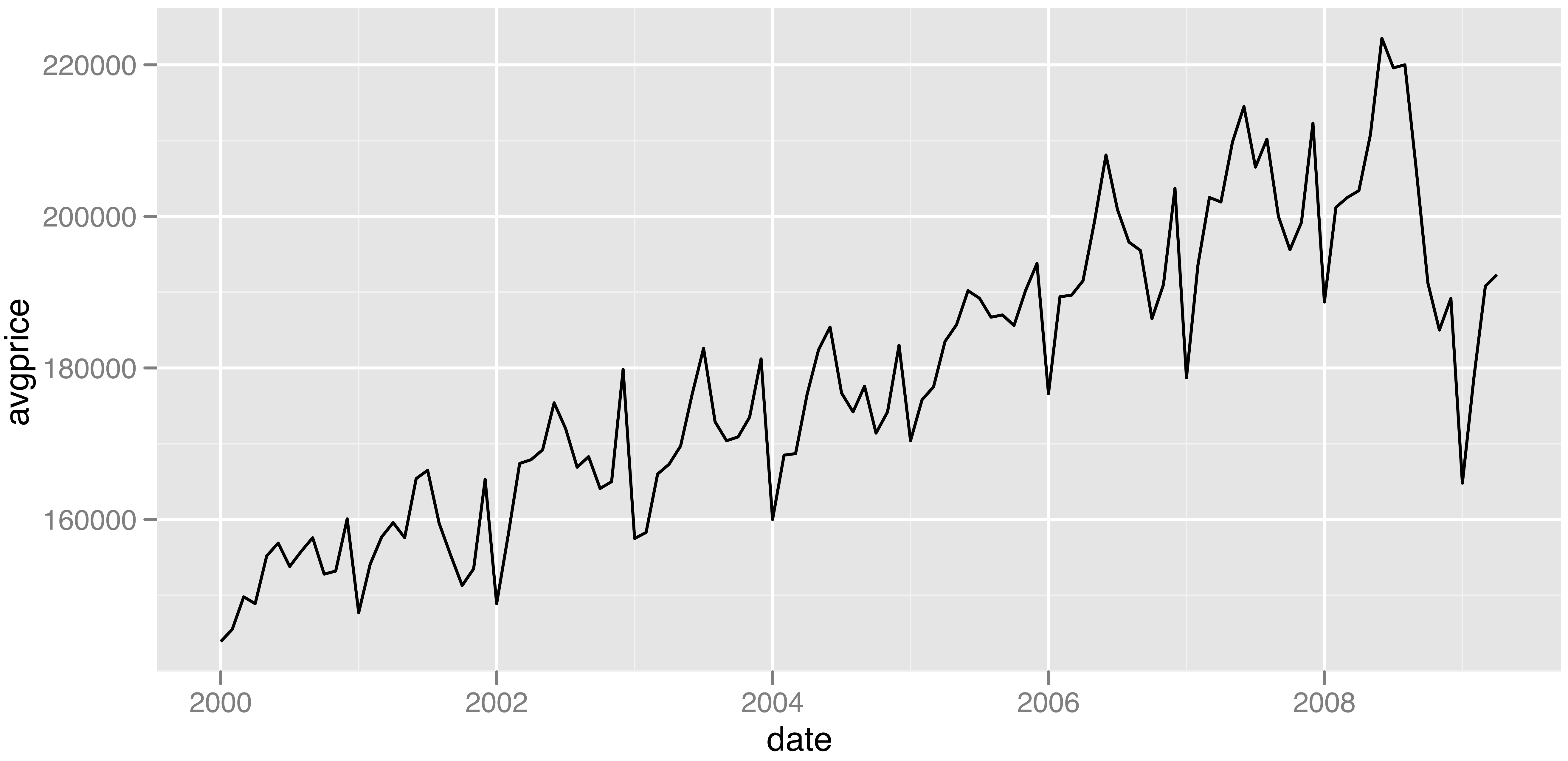
```
library(ggplot2)
tx <- read.csv("tx-house-sales.csv")
houston <- subset(tx, city == "Houston")
```

Strategy

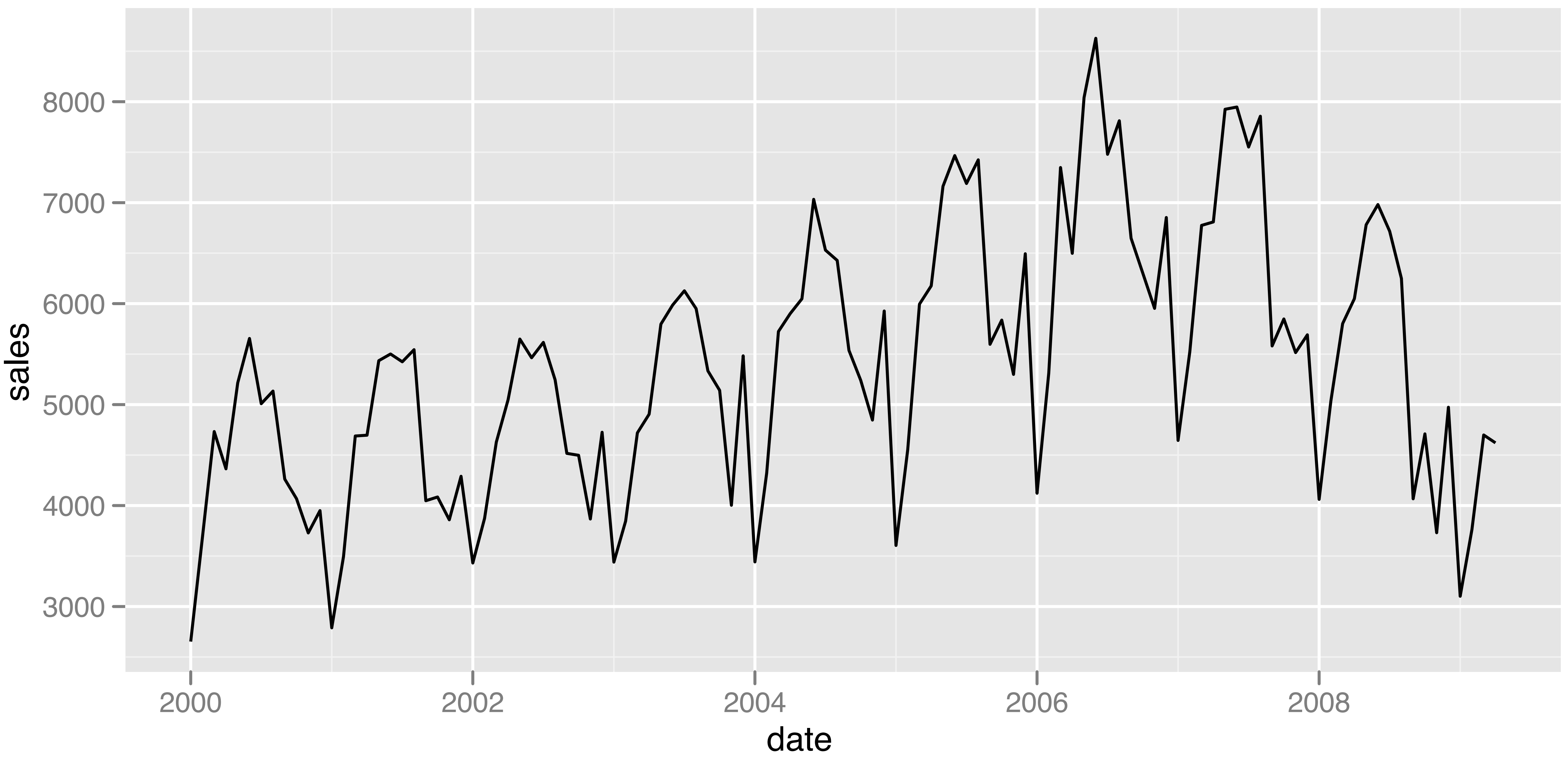
Start with a single city (Houston).

Explore patterns & fit models.

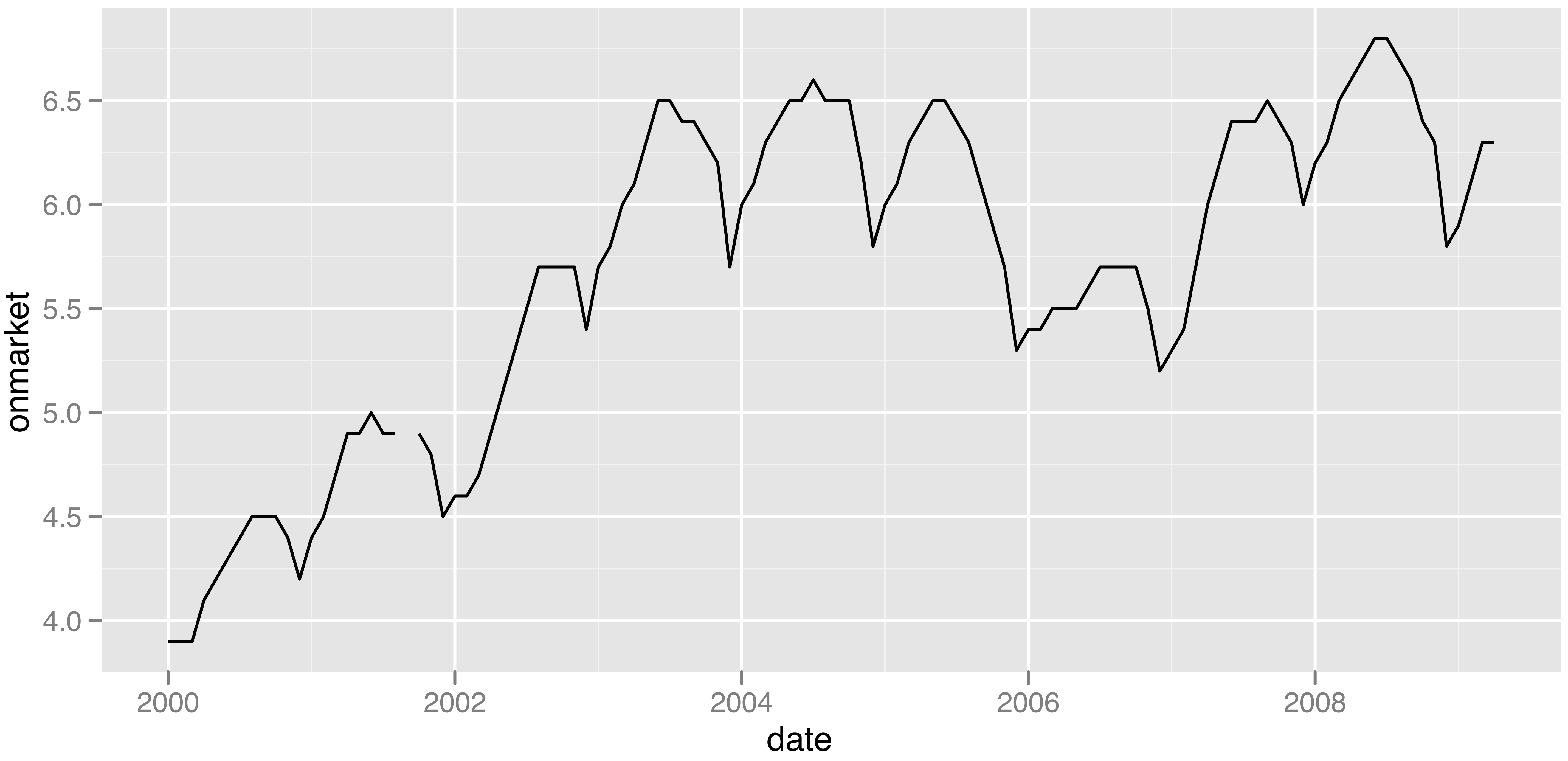
Apply model to all cities.



```
qplot(date, avgprice, data = houston, geom = "line")
```



```
qplot(date, sales, data = houston, geom = "line")
```

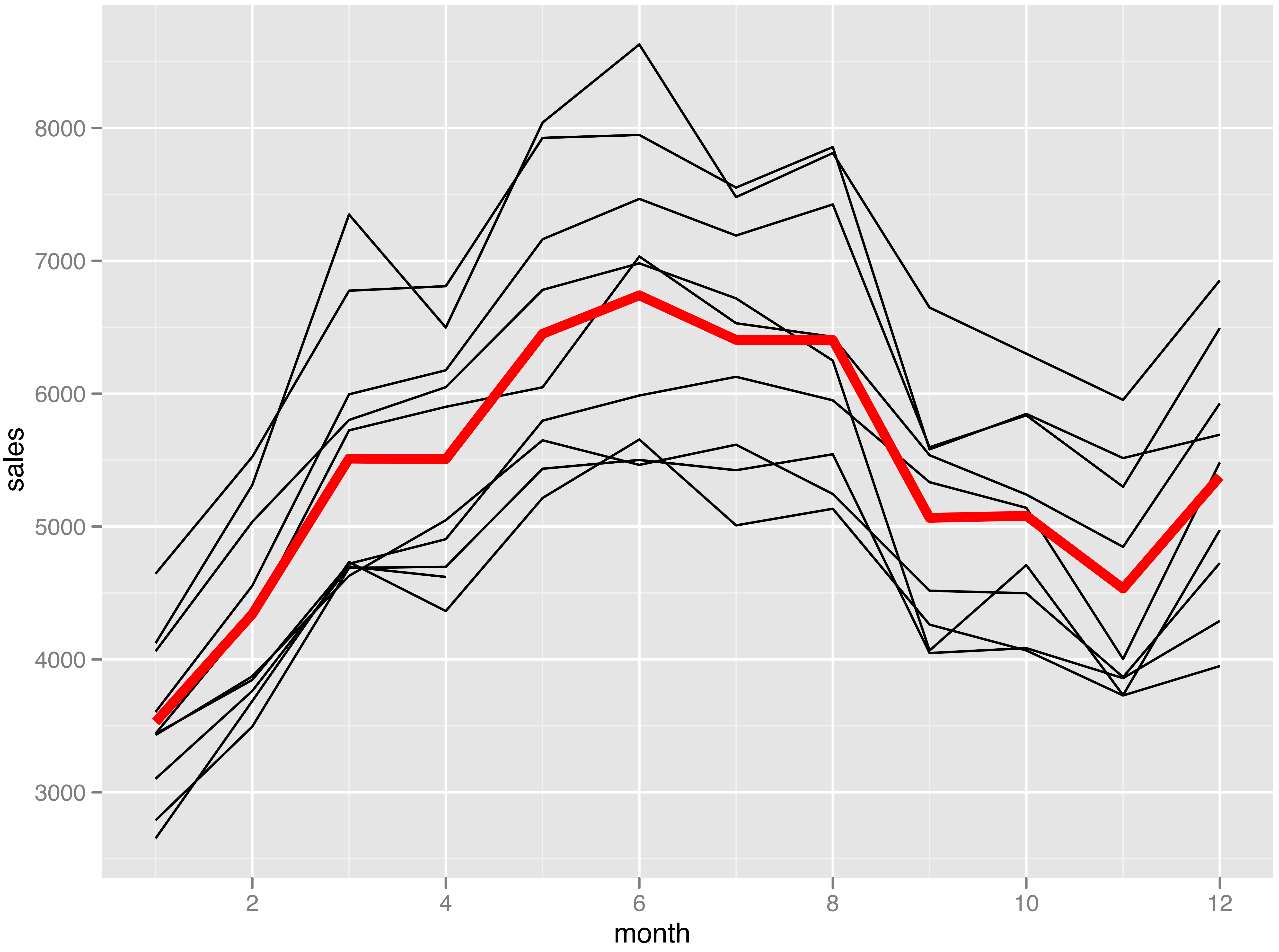


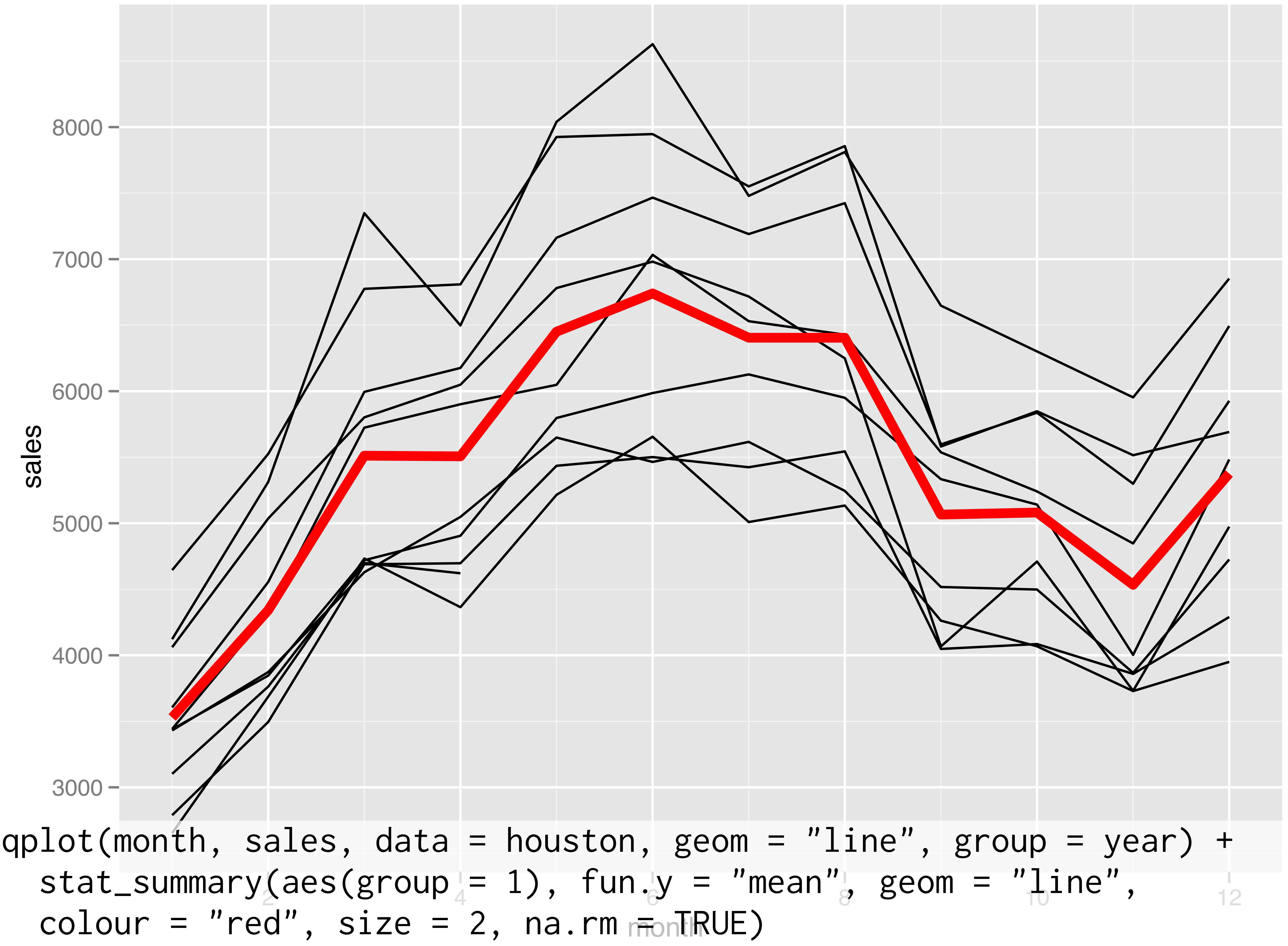
```
qplot(date, onmarket, data = houston, geom = "line")
```

Your turn

One problem with all these plots makes it hard to focus on the long-term trend.

What is it? And how can we get rid of it?


8000
7000
6000
5000
4000
3000
sales
2
4
6
8
10
12
month



```
qplot(month, sales, data = houston, geom = "line", group = year) +
  stat_summary(aes(group = 1), fun.y = "mean", geom = "line",
  colour = "red", size = 2, na.rm = TRUE)
```

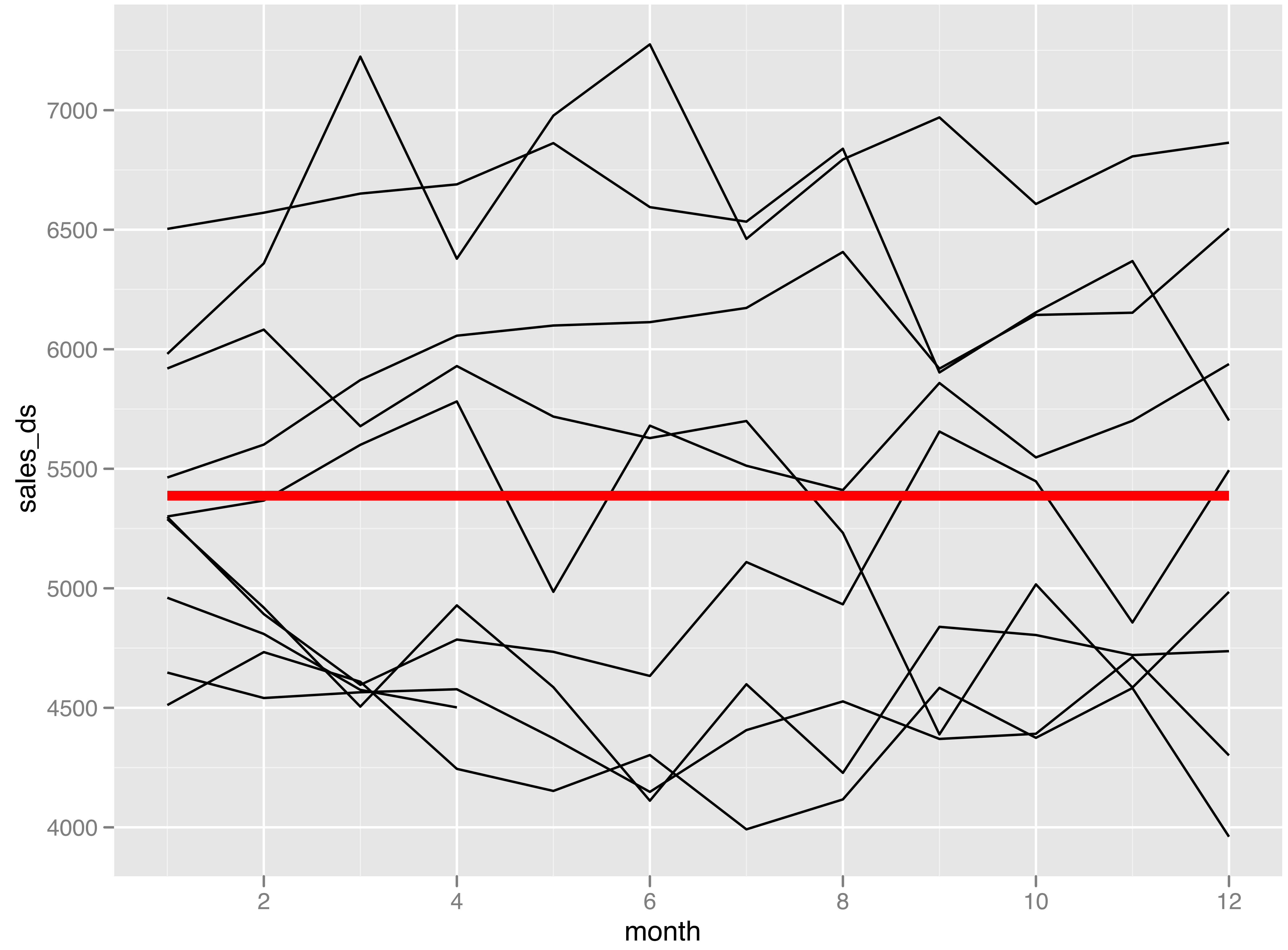
Challenge

What does the following function do?

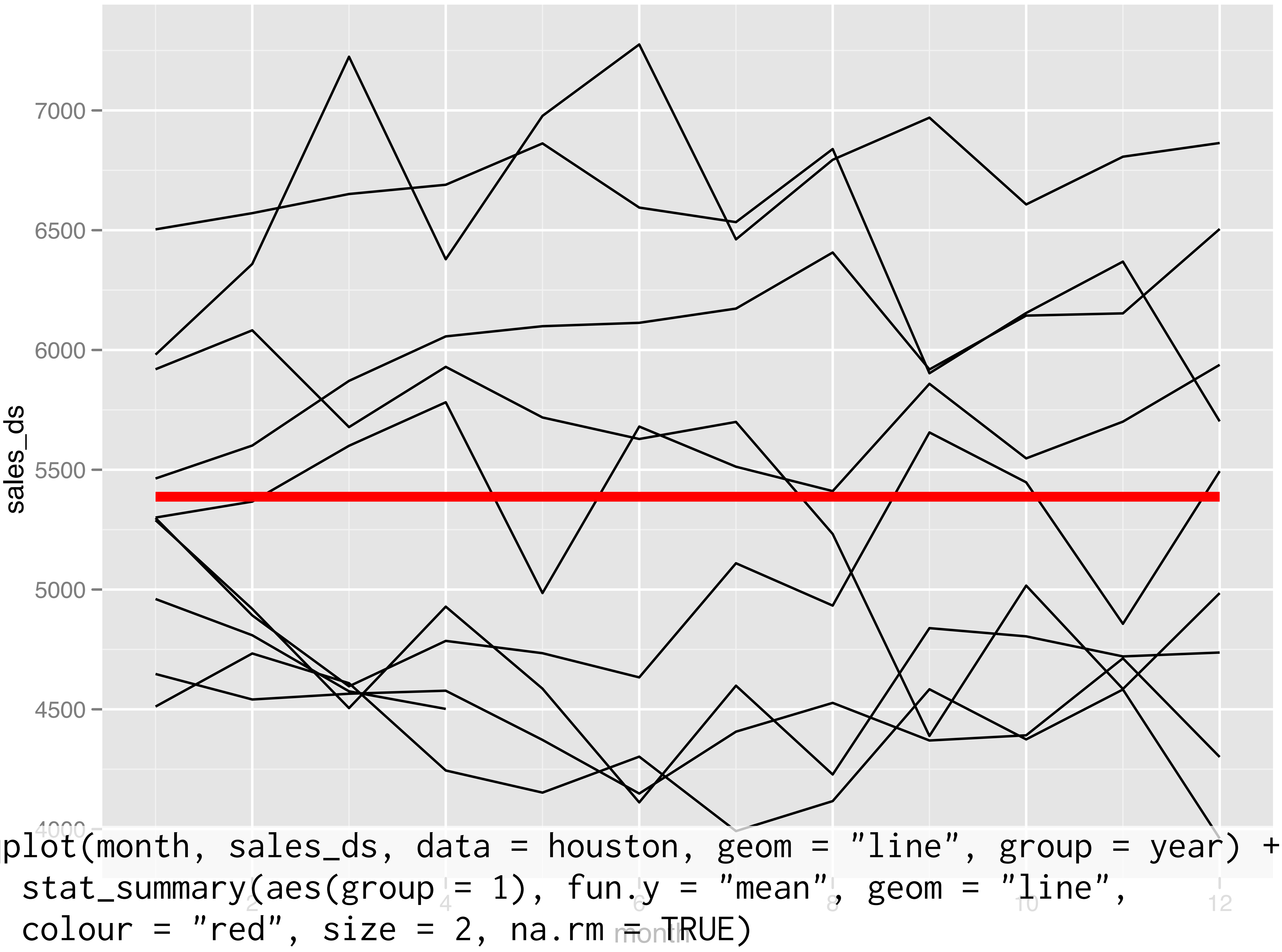
```
deseas <- function(var, month) {
  resid(lm(var ~ factor(month)) +
    mean(var, na.rm = TRUE)
}
```

How could you use it in conjunction with transform to deasonalise the data? What if you wanted to deasonalise every city?

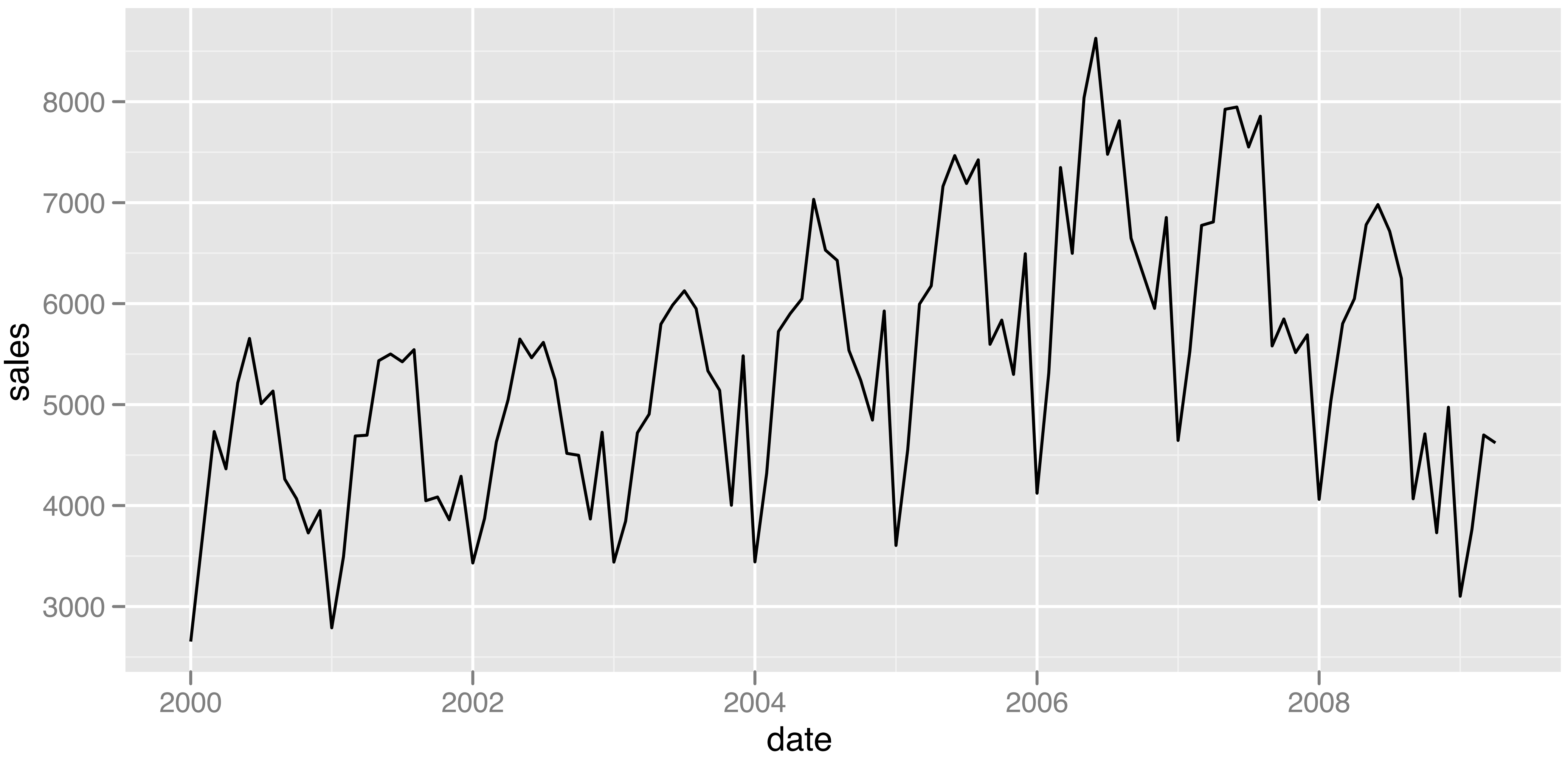
```
deseas <- function(var, month) {
  resid(lm(var ~ factor(month), na = "na.exclude")) +
    mean(var, na.rm = TRUE)
}

houston <- transform(houston,
  sales_ds = deseas(sales, month),
  avgprice_ds = deseas(avgprice, month),
  listings_ds = deseas(listings, month),
  onmarket_ds = deseas(onmarket, month)
)

qplot(month, sales_ds, data = houston, geom = "line",
  group = year) + stat_summary(aes(group = 1),
  fun.y = "mean", geom = "line", colour = "red", size = 2)
```



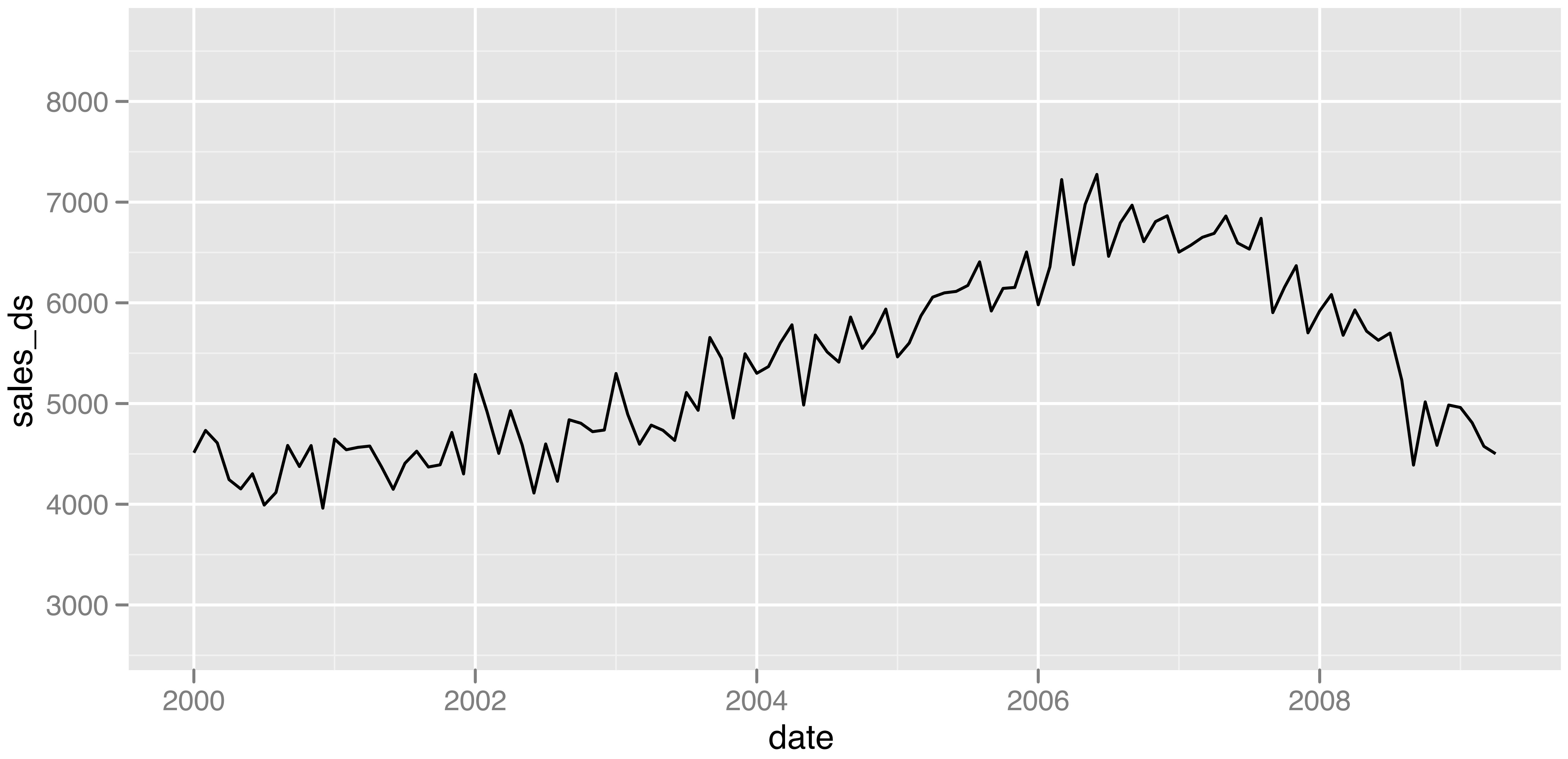
7000
6500
6000
5500
5000
4500
4000
sales_ds
2
4
6
8
10
12
month



```
qplot(month, sales_ds, data = houston, geom = "line", group = year) +
  stat_summary(aes(group = 1), fun.y = "mean", geom = "line",
  colour = "red", size = 2, na.rm = TRUE)
```



```
qplot(date, sales, data = houston, geom = "line")
```

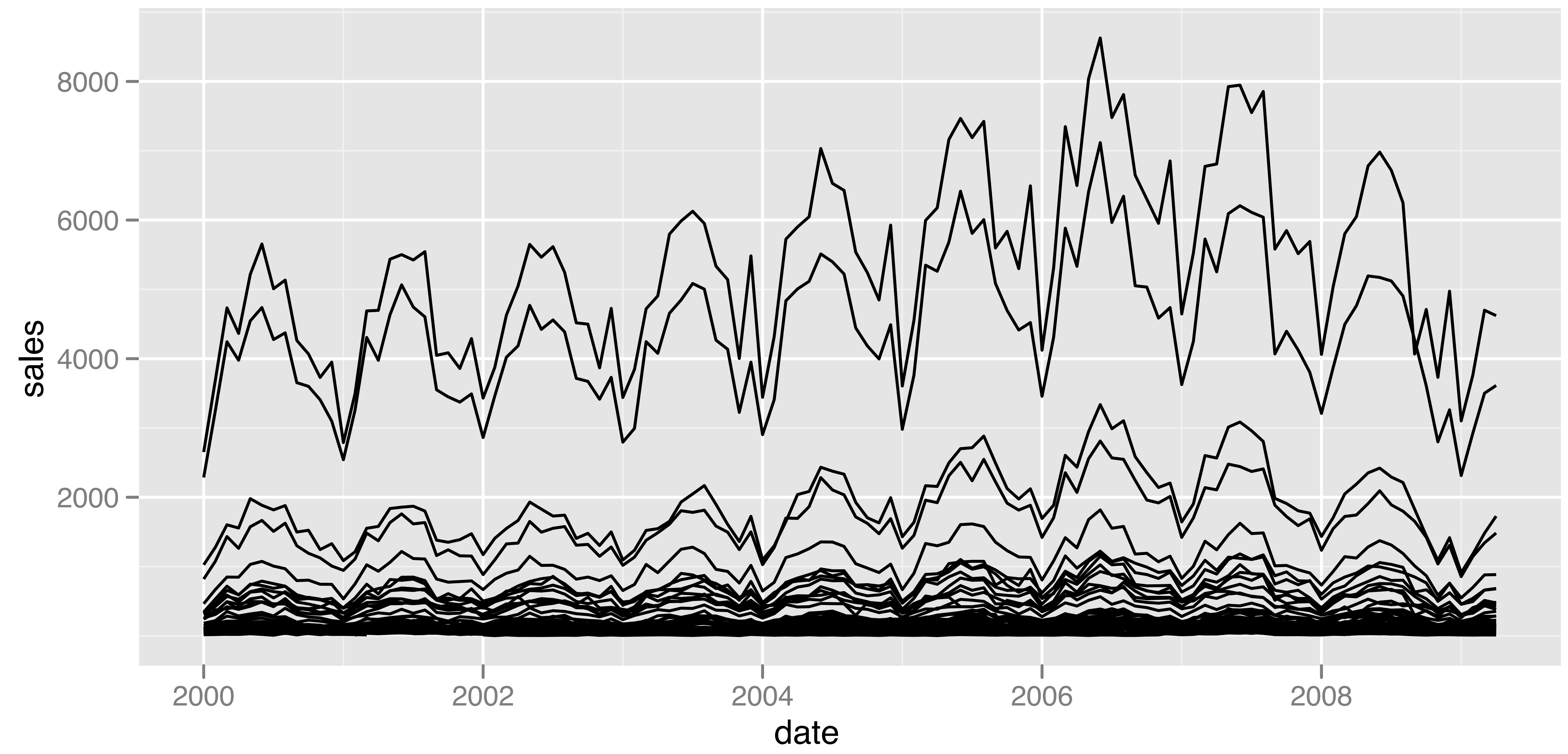


```
qplot(date, sales_ds, data = houston, geom = "line") +
  ylim(range(houston$sales))
```

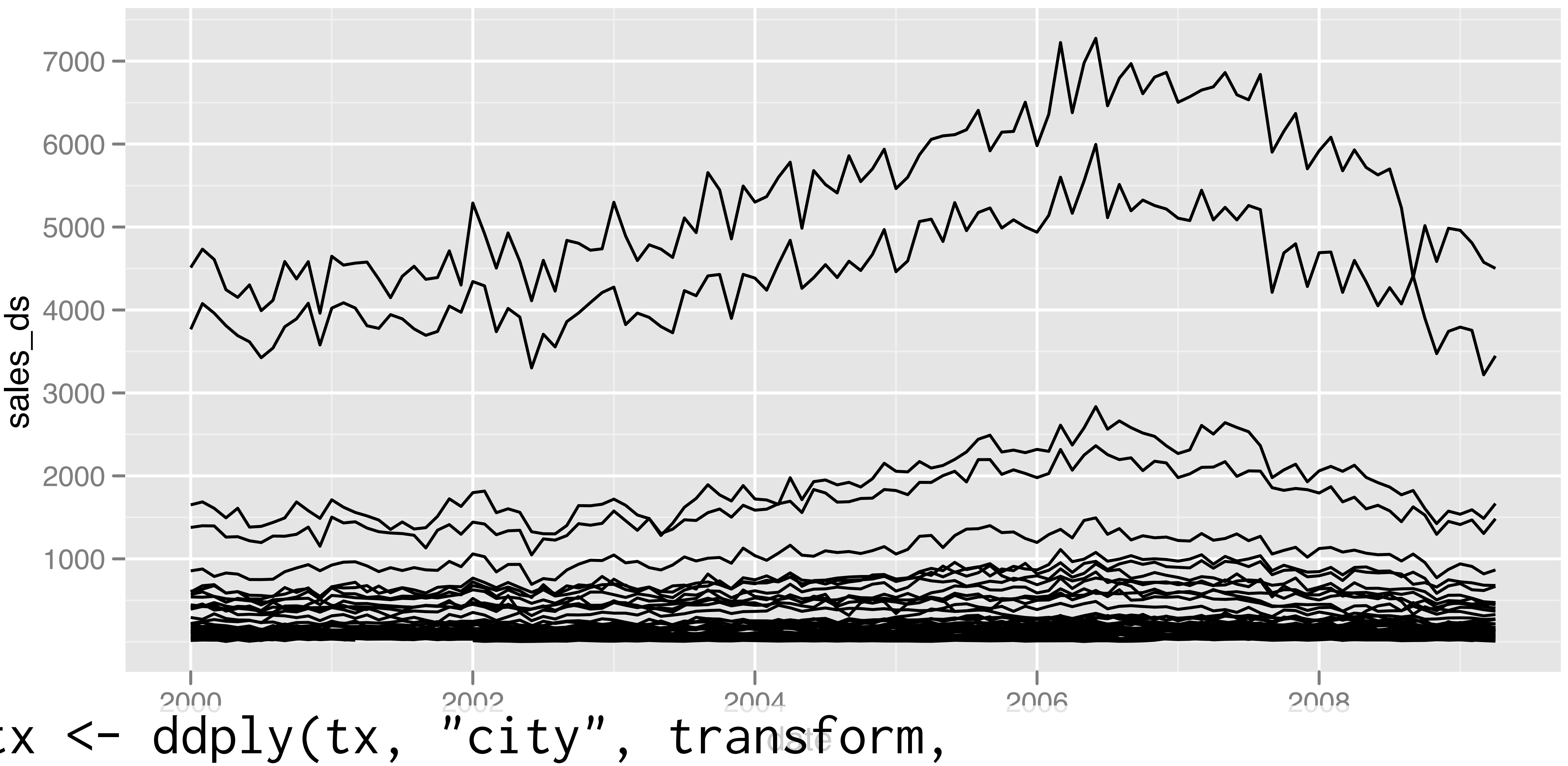
Model as tools

Here we're using the linear model as a tool - we don't care about the coefficients or the standard errors, just using it to get rid of a striking pattern.

Is it still appropriate to do this if we want to look at all cities in Texas? Why/Why not?

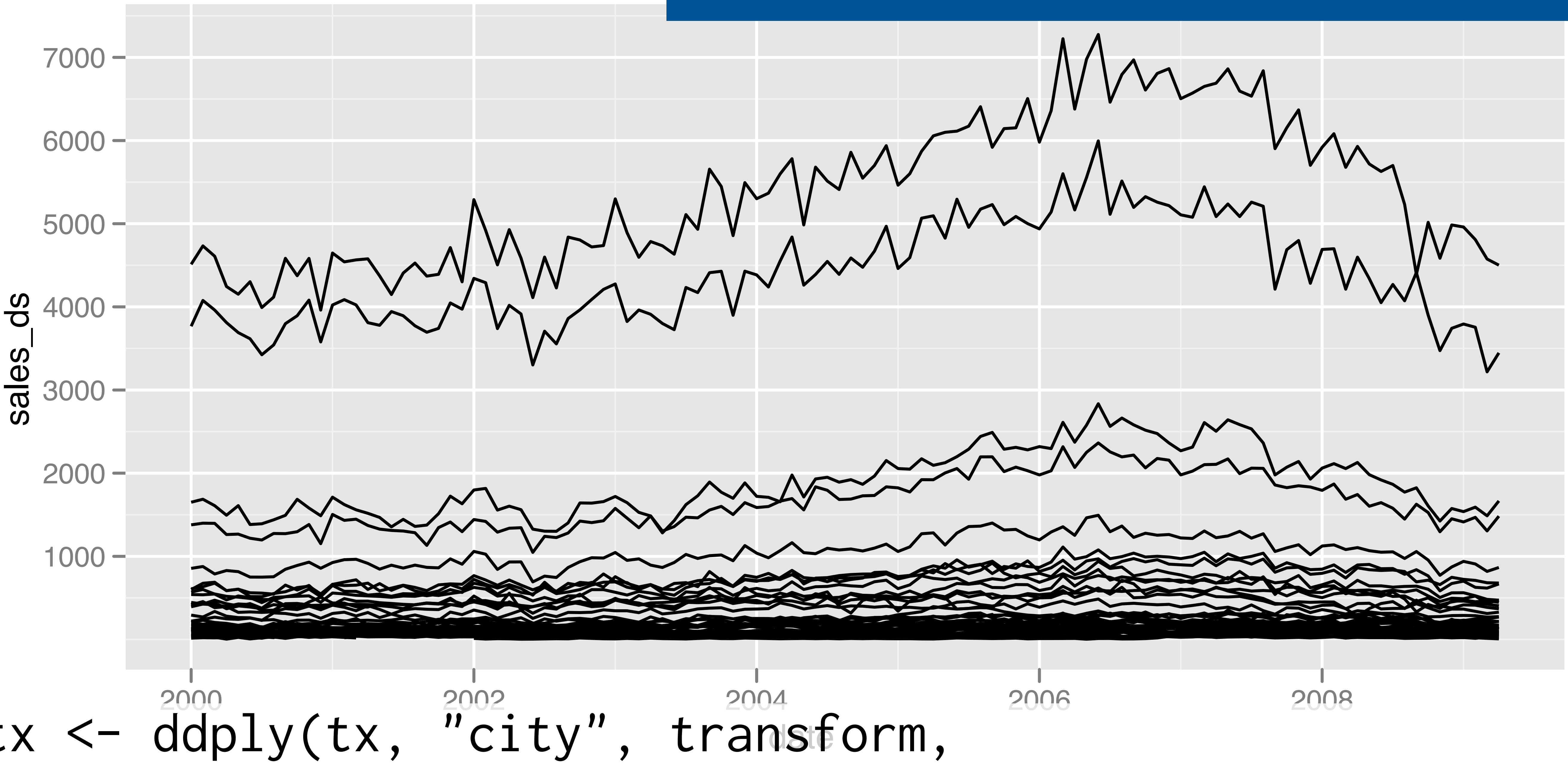


```
qplot(date, sales, data = tx, geom = "line", group = city)
```



```
tx <- ddply(tx, "city", transform,
  sales_ds = deseas(sales, month))
qplot(date, sales_ds, data = tx, geom = "line",
  group = city)
```

Is this still such a good idea? What do we lose? Is there anything else we could do to improve the plot?

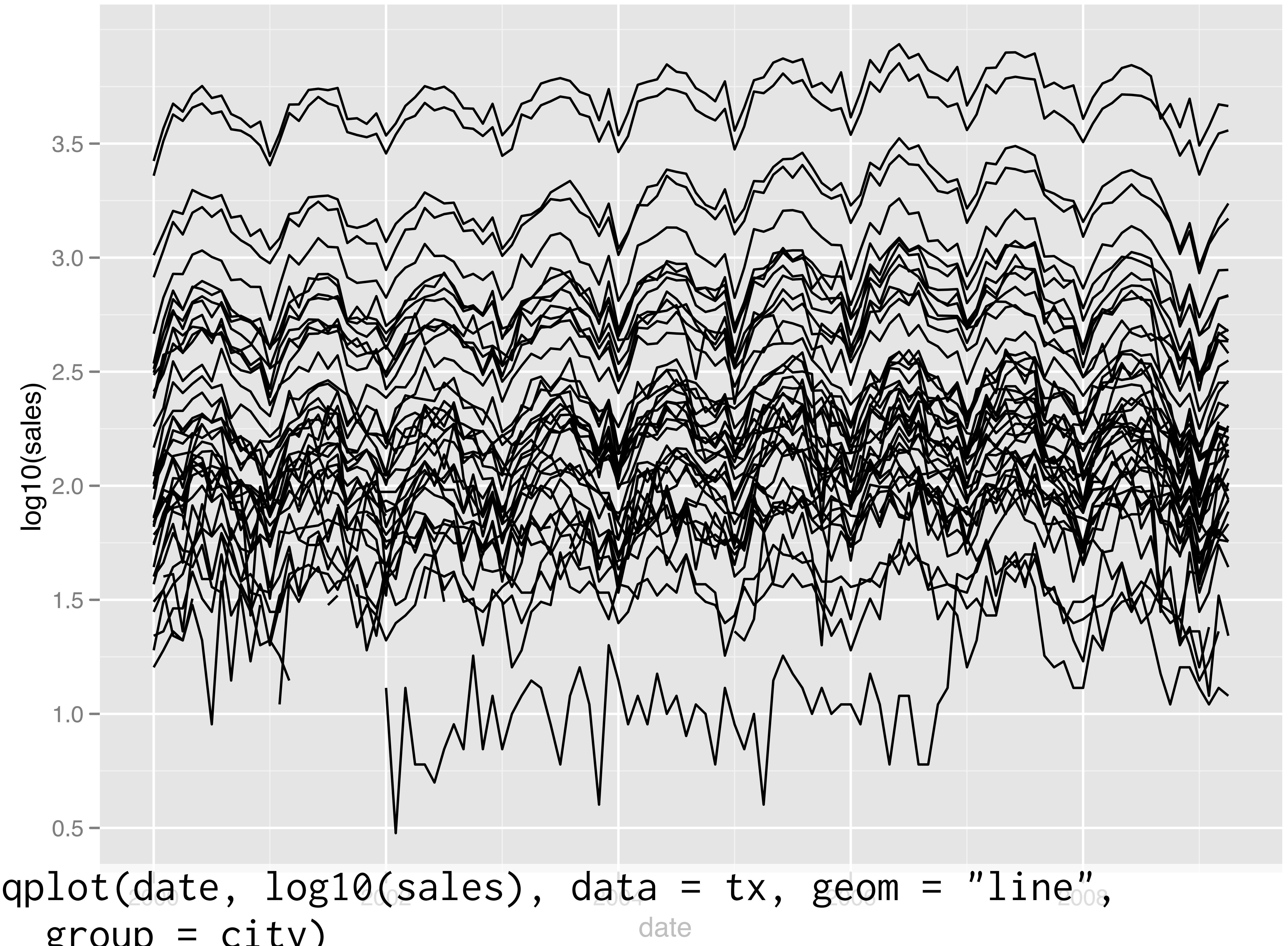


```
tx <- ddply(tx, "city", transform,
  sales_ds = deseas(sales, month))
qplot(date, sales_ds, data = tx, geom = "line",
  group = city)
```

It works, but...

Instead of throwing the models away and just using the residuals, let’s keep the models and explore them in more depth.

And let’s log transform to put all of the cities on a similar scale.



```
qplot(date, log10(sales), data = tx, geom = "line",
  group = city)
```

Two new tools

`dlply`: takes a data frame, **splits** up in the same way as `ddply`, **applies** function to each piece and **combines** the results into a list

`ldply`: takes a list, **splits** up into elements, **applies** function to each piece and then **combines** the results into a data frame

`dlply + ldply = ddply`

```
models <- dlply(tx, "city", function(df)
  lm(log10(sales) ~ factor(month), data = df))


models[[1]]
coef(models[[1]])

ldply(models, coef)
```

Labelling

Notice we didn't have to do anything to have the coefficients labelled correctly.

Behind the scenes plyr records the labels used for the split step, and ensures they are preserved across multiple plyr calls.

Back to the model

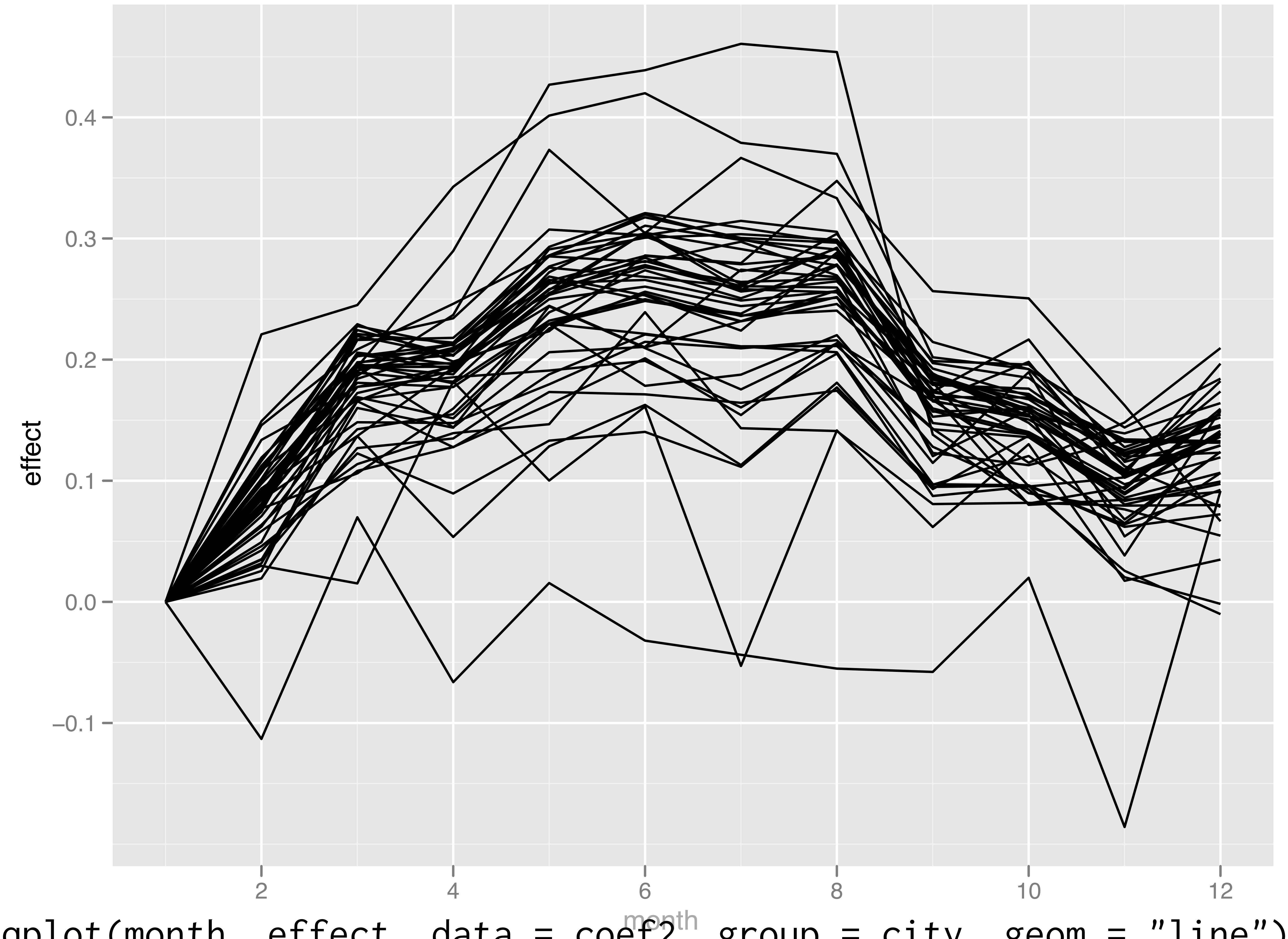
What would be a natural way of visualising these coefficients? How would we need to arrange the coefficients?

Turn to the person next to you and discuss for 2 minutes.

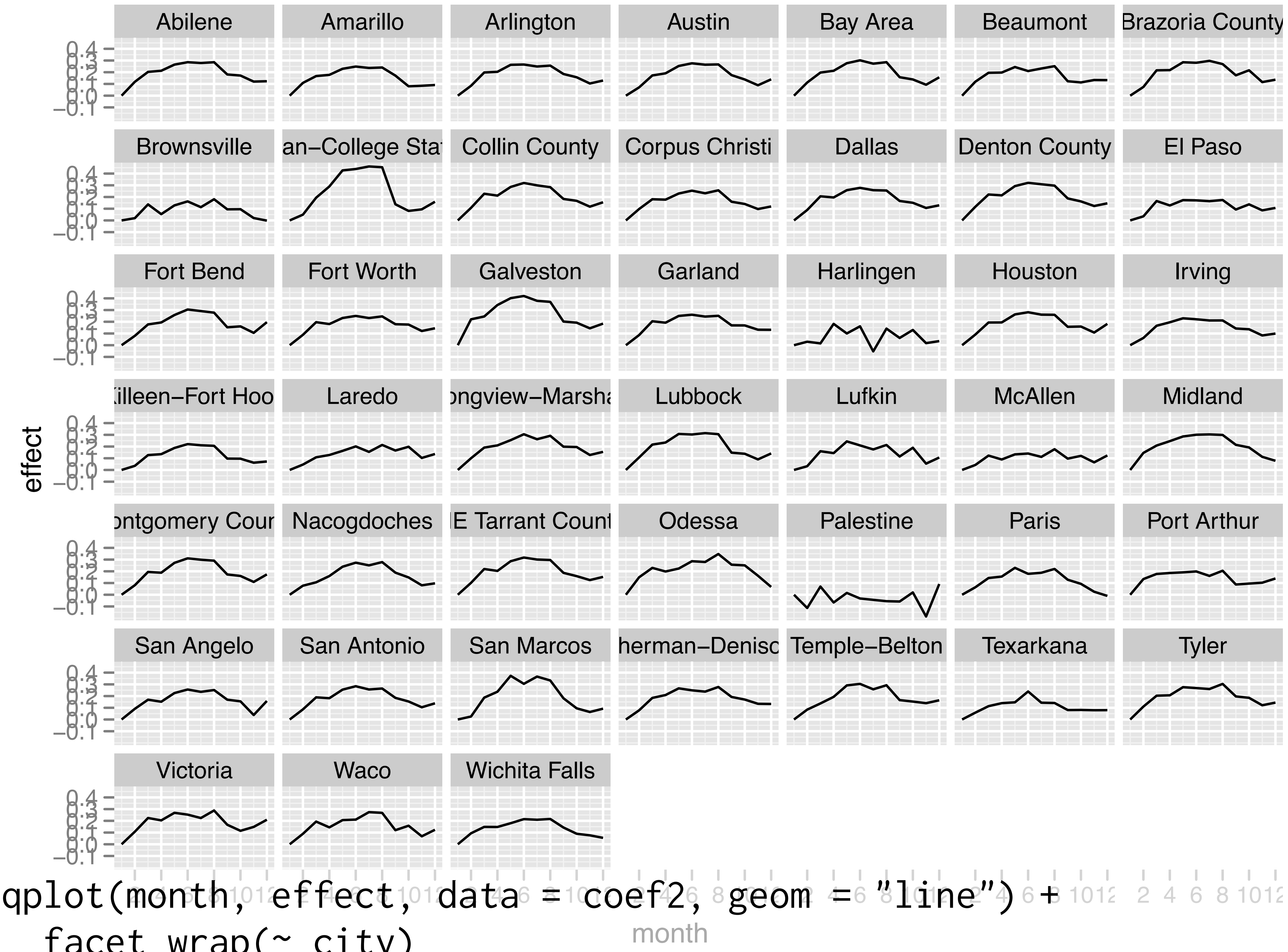
```
coef2 <- ldply(models, function(mod) {
  data.frame(
    month = 1:12,
    effect = c(0, coef(mod)[-1]),
    intercept = coef(mod)[1])
})
```

```
coef2 <- ldply(models, function(mod) {
  data.frame(
    month = 1:12,
    effect = c(0, coef(mod)[-1]),
    intercept = coef(mod)[1])
})
```

Puts coefficients in rows, so they can be plotted more easily



```
qplot(month, effect, data = coef2, group = city, geom = "line")
```



```
qplot(month, effect, data = coef2, geom = "line") +
  facet_wrap(~ city)
```