persistenceSanityChecks

Dan Olner

6 December 2017

This is the full model with the weights matrix, a measure of 'other migrant groups apart from mine' and a vector of other variables/controls:

$$x_{ijt} = \gamma_0 + \gamma_1 x_{ijt-k} + \gamma_2 W x_{ijt-k} + \gamma_3 \sum_{r \neq i}^{I} x_{rjt-k} + \gamma_4 \mathbf{Z}_{jt} + \epsilon_{ijt} \quad (1)$$

For working through it here though, we only need look at:

$$x_{ijt} = \gamma_0 + \gamma_1 x_{ijt-k} + \gamma_2 employment + \epsilon_{ijt} \quad (2)$$

I've dropped the weights and all other control variables - this just sticks with the basics:

- Dependent is **share of country of birth in each zone at time t**
- γ_1 is the previous time-step's share of the same.
- Then just one example extra variable, **employment**.

Some other points:

- the 'share' is a proportion **across all geographical zones**: it sums to 100 for all zones. This is done, AFAIK, so that share is comparable across time periods, i.e. they're not interested in changes in raw numbers of people - *only whether concentration has changed spatially.*

Sooo. I've got a few problems with this. The first is simple:

What does the 'spatial persistence' coefficient actually mean?

An example from the actual data. This is the share of Indian-born people in Glasgow:

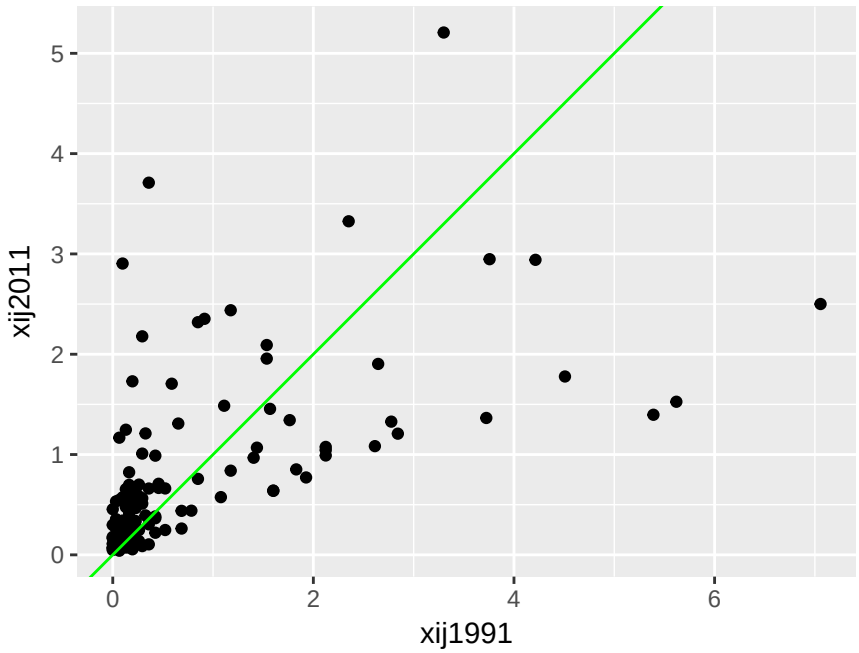
```
## [1] 100
```

```
## Parsed with column specification:
## cols(
##   .default = col_integer(),
##   label = col_character(),
##   fnl_rsL = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
## Parsed with column specification:
## cols(
##   .default = col_integer(),
##   label = col_character(),
##   fnl_rsL = col_character()
## )
```

```
## See spec(...) for full column specifications.
```



- Shares across zones are not independent: if raw numbers drop in one zone, it pushes up the share value in all other zones (and vice versa).
- What this means: if e.g. the coefficient > 1, this could be because higher-value zones **gained numbers** or lower-valued zones **lost numbers**.
- I am unclear what this can tell us about spatial persistence.
- The actual share values are **also** dependent on the **number of zones**. E.g. if people are evenly spread across the area, and you split that into 10 or 20 zones, the shares will be either 10% or 5% per ob. Which may be fine when comparing like with like, but it means any other coefficient values **will be different for differing zone counts** despite the underlying data being identical. (Different from MAUP, that's about different ways to aggregate points, but related.)
- Larger zones have a lot of influence over the result. Per zone percent would deal with this, it won't change (ceteris paribus) if zone size changes. But then that's no good for the time comparison.

To illustrate, let's just take the 1991 raw numbers for Indian-born people in Glasgow. Using the top twenty zones by count to make more clear.

```
#Between 49 and 216 people per zone, twenty zones.
summary(glasgowExample)
```

```
##     label           indianCount1991
##  Length:20          Min.   : 49.0
##  Class :character   1st Qu.: 65.0
##  Mode  :character   Median : 83.0
##                     Mean   : 98.4
##                     3rd Qu.:118.5
##                     Max.   :216.0
```

So, random scenarios:

1. The numbers stay exactly the same between 91 and 2011 - coefficient will be 1. Continued commentary in comments!

```r
#Pretend 2011 numbers exactly the same
glasgowExample$indianCount2011 <- glasgowExample$indianCount1991

#Find across-zone shares for both, summing to 100% across all zones (so down columns)
#https://stackoverflow.com/questions/45947787/create-new-variables-with-mutate-at-while-keeping-the-ori
glasgowExample <- glasgowExample %>%
  mutate_at(vars(indianCount1991:indianCount2011), funs(share = . / sum(.) * 100))
  #mutate(indianShares1991 = indianCount1991/sum(indianCount1991) )

#Sum to 100% across zones? Tick.
apply(glasgowExample[,c(4:5)],2,sum)
```

```
## indianCount1991_share indianCount2011_share
##                   100                   100
```

```r
#So stupid regression, gonna be 1
summary(lm(glasgowExample, formula = indianCount2011_share ~ indianCount1991_share))
```

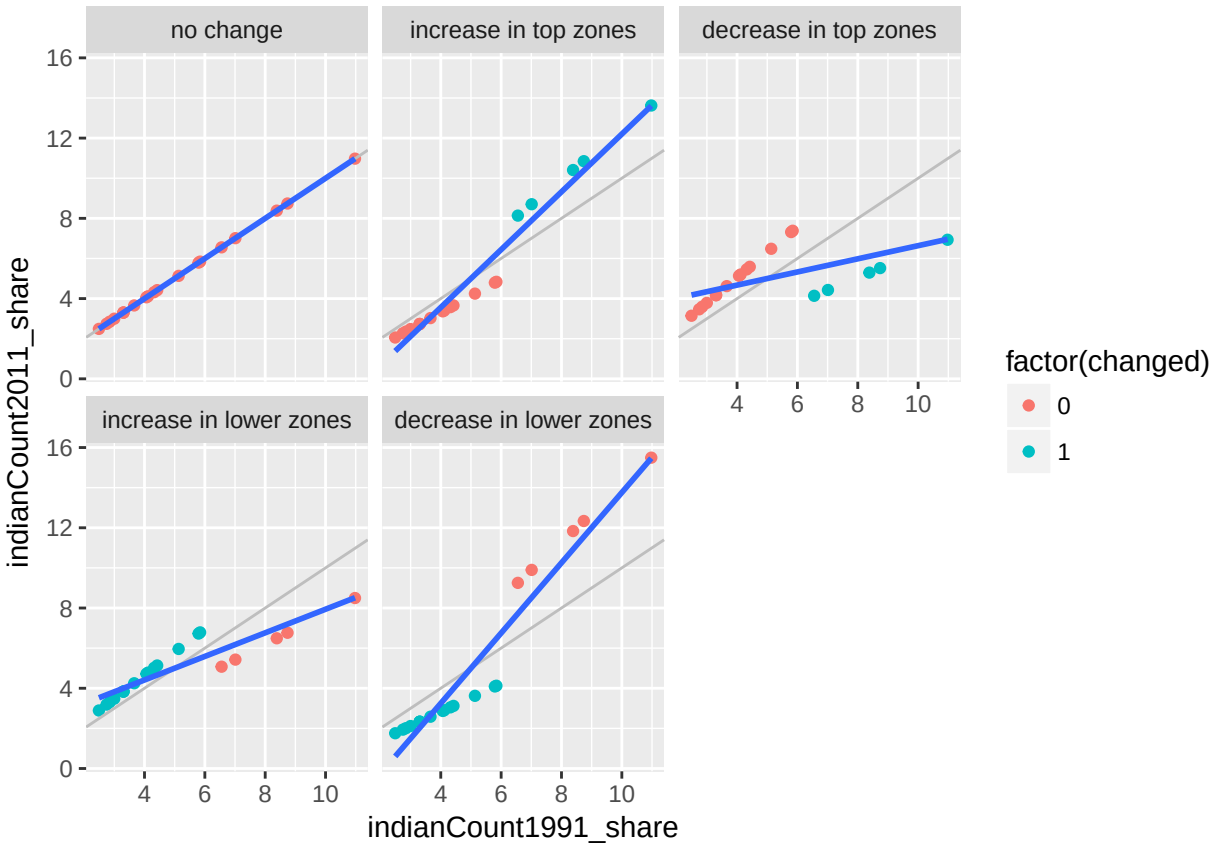
```
## Warning in summary.lm(lm(glasgowExample, formula = indianCount2011_share
## ~ : essentially perfect fit: summary may be unreliable
```

```
##
## Call:
## lm(formula = indianCount2011_share ~ indianCount1991_share, data = glasgowExample)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -6.976e-16 -1.344e-16 -8.897e-17 -4.890e-18  1.629e-15
##
## Coefficients:
##                          Estimate Std. Error    t value Pr(>|t|)
## (Intercept)           -7.944e-16  2.406e-16 -3.301e+00  0.00397 **
## indianCount1991_share  1.000e+00  4.386e-17  2.280e+16  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.428e-16 on 18 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 5.198e+32 on 1 and 18 DF,  p-value: < 2.2e-16
```

```r
#Now let's make a couple of other scenarios. I'll combine them to look at together

#Scenario 2: INCREASE counts in 2011 in top 5 zones and recalculate shares
scenario2 <- glasgowExample
scenario2$indianCount2011[1:5] <- scenario2$indianCount2011[1:5] * 1.5
#Mark the changed points
scenario2$changed <- 0
scenario2$changed[1:5] <- 1

scenario2 <- scenario2 %>%
  mutate_at(vars(indianCount1991:indianCount2011), funs(share = . / sum(.) * 100))

#Scenario 3: DECREASE counts in 2011 in top 5 zones and recalculate shares
scenario3 <- glasgowExample
scenario3$indianCount2011[1:5] <- scenario3$indianCount2011[1:5] * 0.5
```

```r
scenario3$changed <- 0
scenario3$changed[1:5] <- 1

scenario3 <- scenario3 %>%
  mutate_at(vars(indianCount1991:indianCount2011), funs(share = . / sum(.) * 100))


#Then two more, with the numbers changed AT THE LOWER END OF VALUES
#Increase
scenario4 <- glasgowExample
scenario4$indianCount2011[6:20] <- scenario4$indianCount2011[6:20] * 1.5
scenario4$changed <- 0
scenario4$changed[6:20] <- 1

scenario4 <- scenario4 %>%
  mutate_at(vars(indianCount1991:indianCount2011), funs(share = . / sum(.) * 100))

#Decrease
scenario5 <- glasgowExample
scenario5$indianCount2011[6:20] <- scenario5$indianCount2011[6:20] * 0.5
scenario5$changed <- 0
scenario5$changed[6:20] <- 1

scenario5 <- scenario5 %>%
  mutate_at(vars(indianCount1991:indianCount2011), funs(share = . / sum(.) * 100))


#Stick those together to examine at the same time
scenario1 <- glasgowExample
scenario1$changed <- 0

scenario1$scenario <- "no change"
scenario2$scenario <- "increase in top zones"
scenario3$scenario <- "decrease in top zones"
scenario4$scenario <- "increase in lower zones"
scenario5$scenario <- "decrease in lower zones"

alltogether <- do.call(rbind,list(scenario1,scenario2,scenario3,scenario4,scenario5))

alltogether$scenario <- factor(alltogether$scenario, levels = c("no change","increase in top zones","de
```

```r
ggplot(alltogether,aes(y = indianCount2011_share, x = indianCount1991_share)) +
  geom_point(aes(colour = factor(changed))) +
  geom_abline(slope = 1,intercept = 0,colour="grey") +
  geom_smooth(method = "lm", formula = y~x, se = FALSE) +
  #coord_fixed(ratio = 1) +
  facet_wrap(~scenario)
```


no change
increase in top zones
decrease in top zones
increase in lower zones
decrease in lower zones
indianCount2011_share
indianCount1991_share
0
4
8
12
16
4
6
8
10
factor(changed)
0
1