Plots with 3 anchors

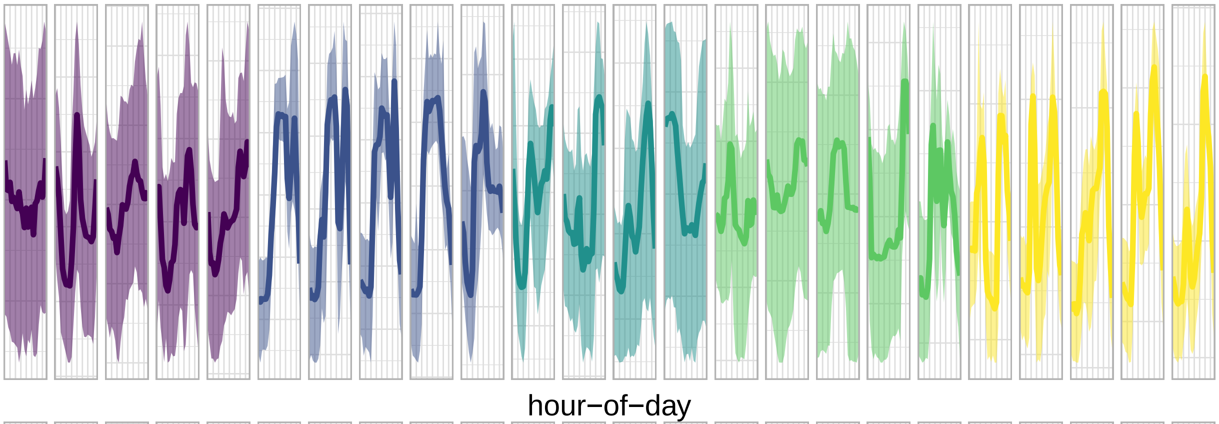
Set number of anchors

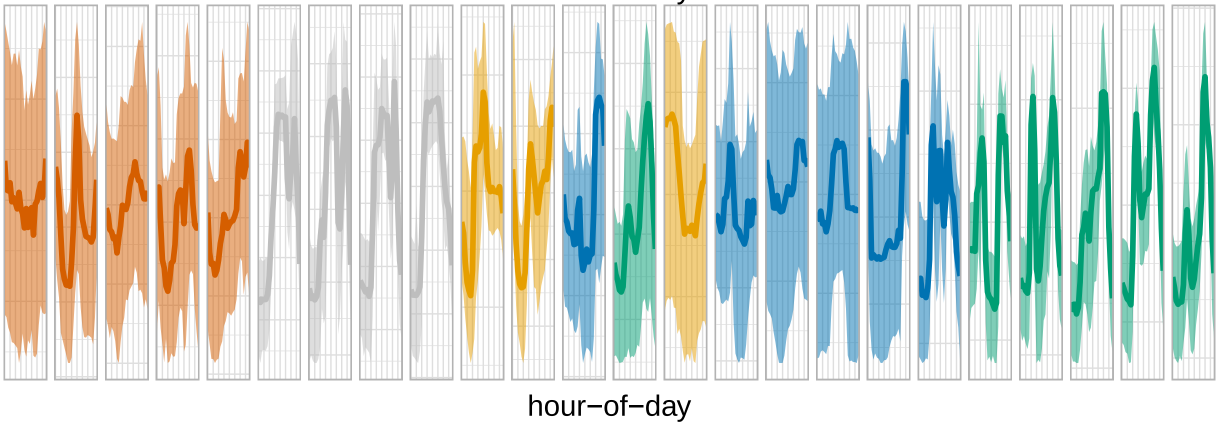
Load the customers you want to cluster

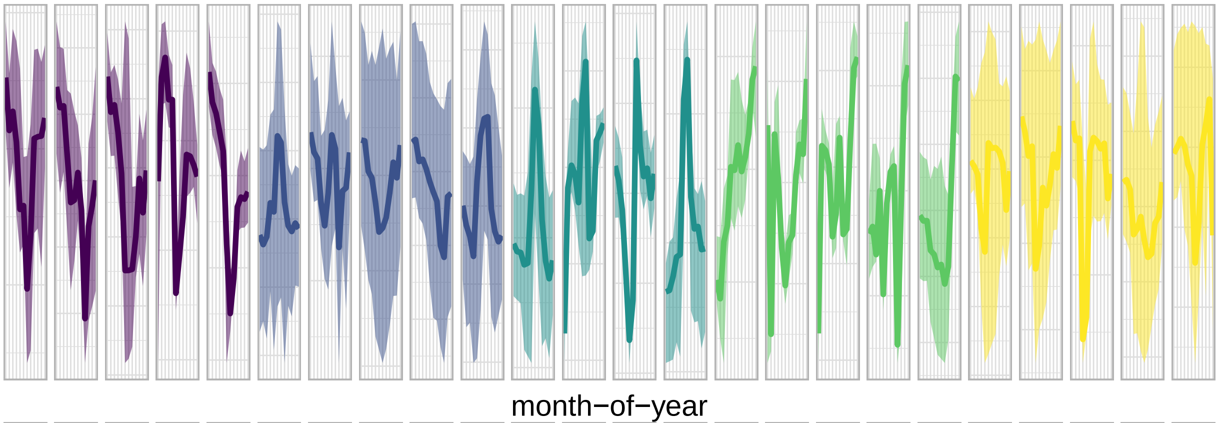
Load raw data for these customers and scale it according to robust method

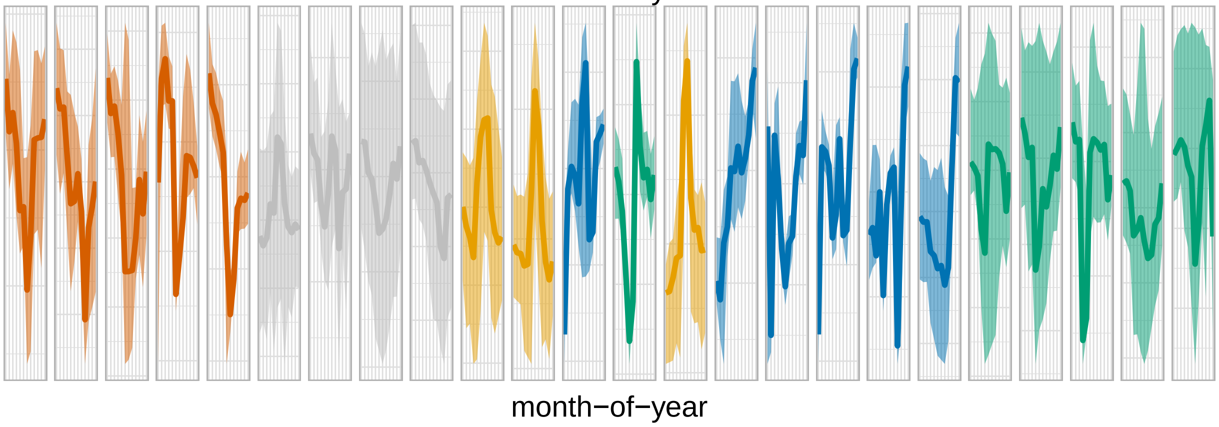
```
## Joining, by = "customer_serial_id"
## Joining, by = "customer_serial_id"
## Joining, by = "customer_serial_id"
```

```
## Joining, by = c("customer_id", "group")
```

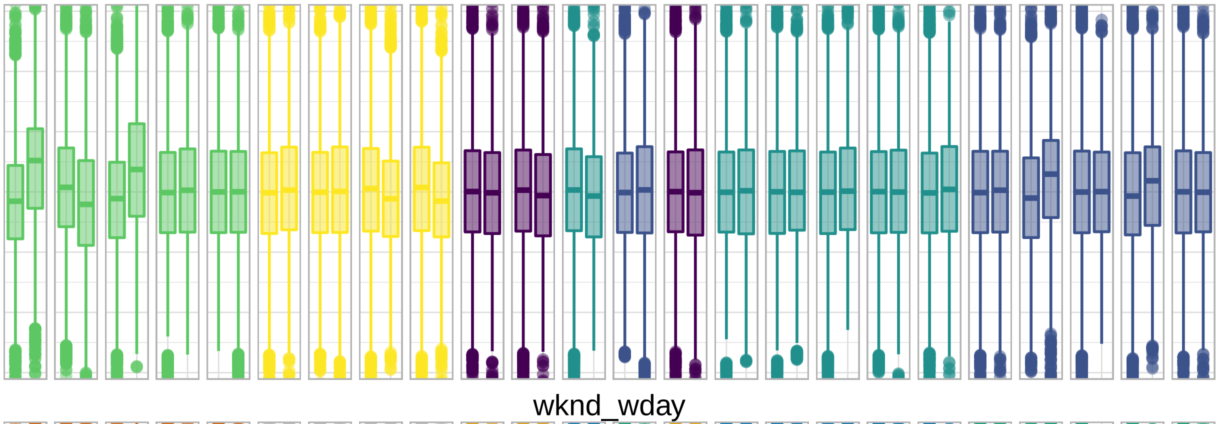


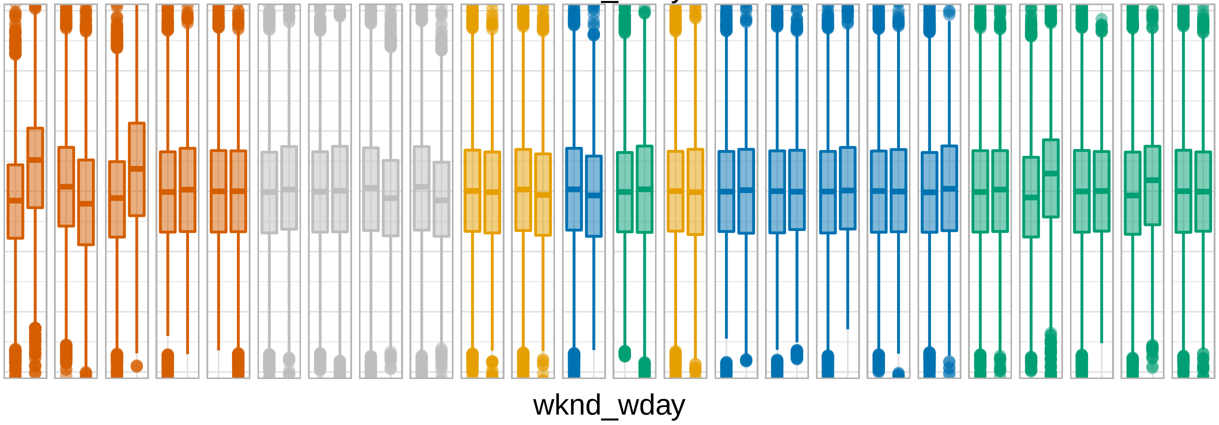






```
## Joining, by = "customer_id"
```



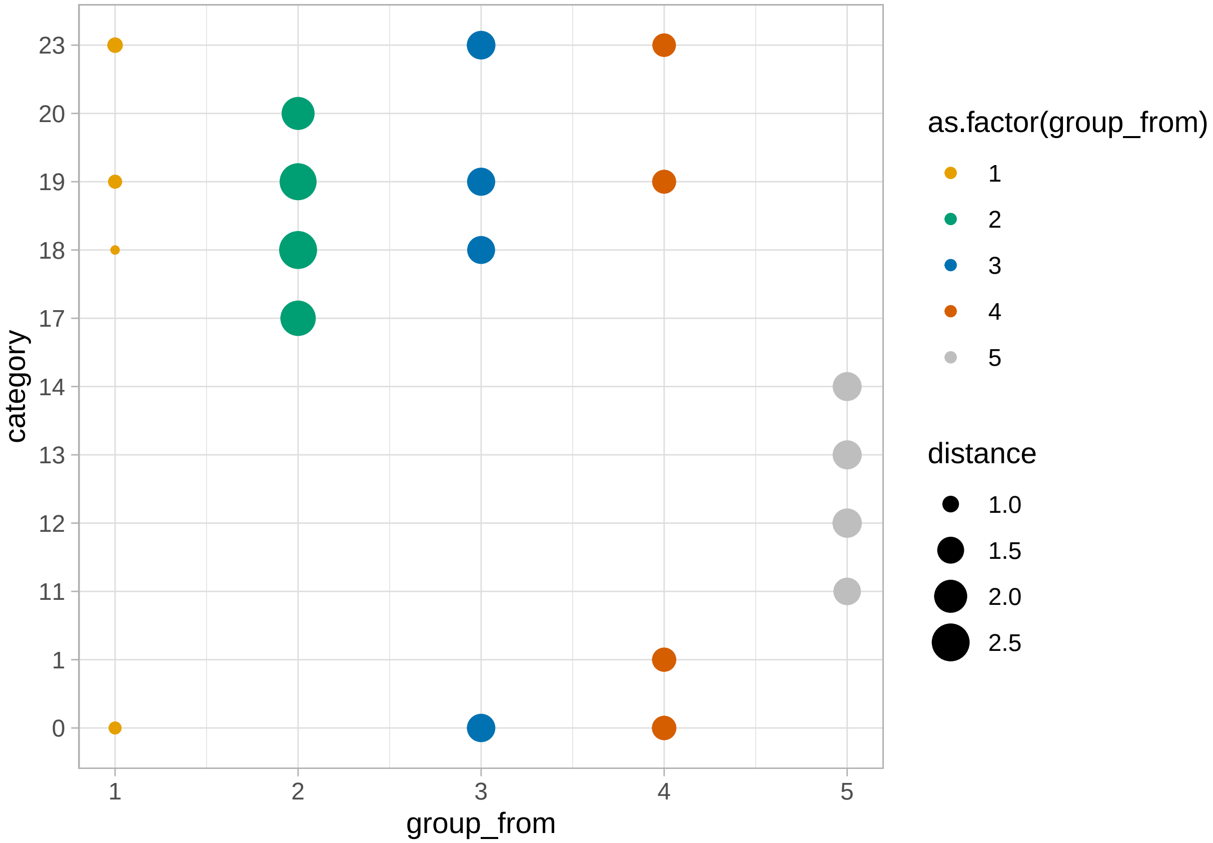


contribution of each levels

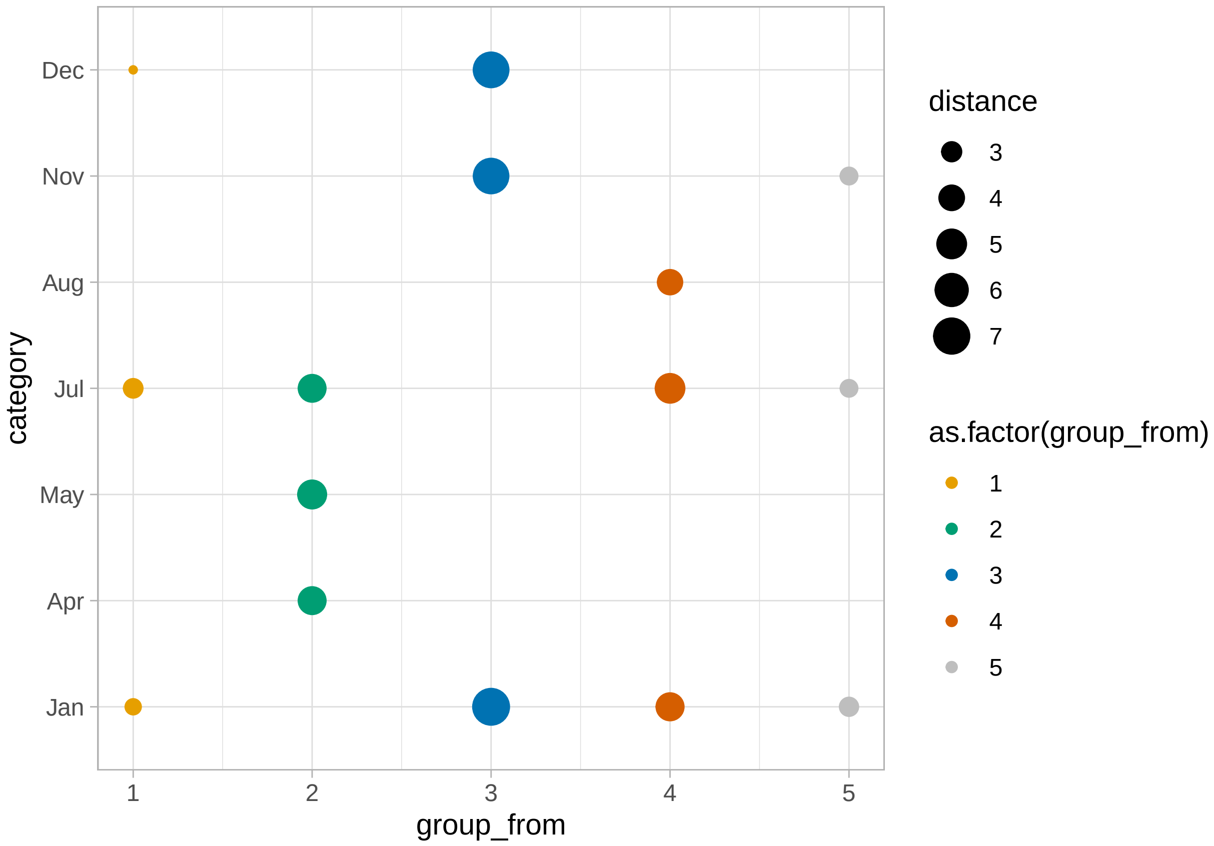
Draw group ones and make interpretation:

```
## # A tibble: 3 x 2
##   gran          d
##   <chr>     <dbl>
## 1 hod      0.0794
## 2 moy      0.132
## 3 wkndwday 0.0226

## `summarise()` has grouped output by 'gran', 'group_item1'. You can override using the `.groups` argu

## # A tibble: 3 x 6
## # Groups:   gran [3]
##   gran       `1`   `2`   `3`   `4`   `5`
##   <chr>    <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 hod       7.80  29.4 14.4   9.54 14.8
## 2 moy      10.4   32.7 29.1  15.9  16.0
## 3 wkndwday  3.10  11.0  6.75  6.12  5.42

## Joining, by = "customer_serial_id"
## Joining, by = "customer_serial_id"
## Joining, by = "customer_serial_id"

## `summarise()` has grouped output by 'gran', 'category', 'group_from'. You can override using the `.g
```

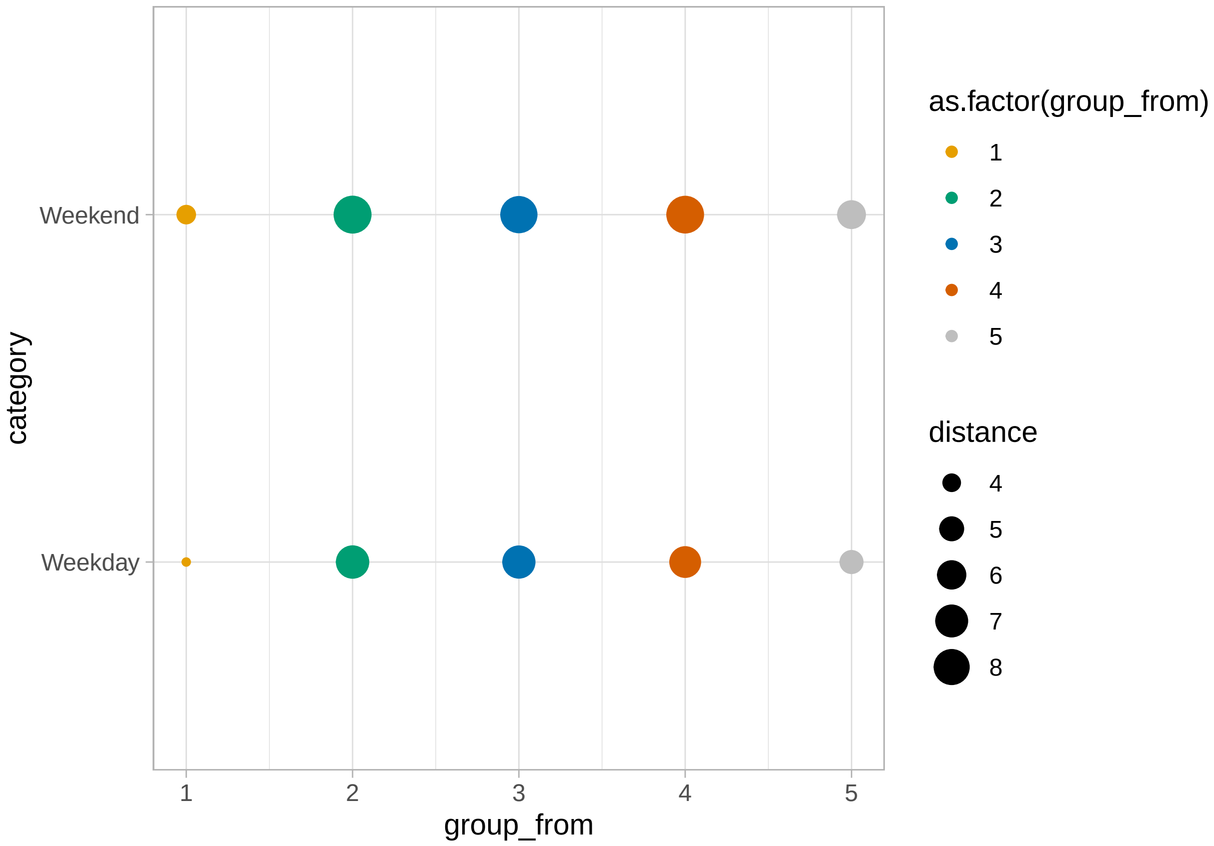
Contribution of individual category for hod



Contribution of individual category for moy



Contribution of individual category for wknwday


Weekend
Weekday
category
group_from
as.factor(group_from)
distance