Red del Sabor

Haydeé Peruyero

2022-10-14

Datos de la recetas

Basado en: https://github.com/lingcheng99/Flavor-Network Reference: Flavor network and the principles of food pairing. Y.-Y. Ahn, S. Ahnert, J. P. Bagrow, and A.-L. Barabási . Scientific Reports 1, 196 (2011).

Cargamos librerías.

```
library(readr)
library(tidyverse)
library(plyr)
library(dplyr)
library(ggplot2)
```

Cargamos la base de datos para explorarla.

```
library(readr)
ingredients <- read_csv("data/srep00196-s3.csv")
```

```
## Rows: 56498 Columns: 24
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (24): Region, Ing1, Ing2, Ing3, Ing4, Ing5, Ing6, Ing7, Ing8, Ing9, Ing1...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(ingredients)
```

```
## # A tibble: 6 x 24
##   Region Ing1  Ing2  Ing3  Ing4  Ing5  Ing6  Ing7  Ing8  Ing9  Ing10 Ing11 Ing12
##   <chr>  <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
## 1 Afric~ chic~ cinn~ soy_~ onion ging~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 2 Afric~ cane~ ging~ cumin garl~ tama~ bread cori~ vine~ onion beef  caye~ pars~
## 3 Afric~ butt~ pepp~ onion card~ caye~ ging~ cott~ garl~ bras~ <NA>  <NA>  <NA>
## 4 Afric~ oliv~ pepp~ wheat beef  onion card~ cumin garl~ rice  leek  <NA>  <NA>
## 5 Afric~ honey wheat yeast <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## 6 Afric~ toma~ cila~ lemo~ onion caye~ scal~ <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
## # ... with 11 more variables: Ing13 <chr>, Ing14 <chr>, Ing15 <chr>,
## #   Ing16 <chr>, Ing17 <chr>, Ing18 <chr>, Ing19 <chr>, Ing20 <chr>,
## #   Ing21 <chr>, Ing22 <chr>, Ing23 <chr>
```

```
ingredients <- as.data.frame(ingredients)
```

Guardamos las recetas por regiones para uso posterior.

```
df_regions <- ingredients %>%
  nest(data = !Region)
```

Obtenemos las regiones.

```
regions <- ingredients %>%
  distinct(Region)

regions
```

```
##              Region
## 1           African
## 2         EastAsian
## 3   EasternEuropean
## 4     LatinAmerican
## 5     MiddleEastern
## 6     NorthAmerican
## 7  NorthernEuropean
## 8        SouthAsian
## 9    SoutheastAsian
## 10 SouthernEuropean
## 11  WesternEuropean
```

Contamos cuantos elementos faltantes hay por cada receta y agregamos una columna a nuestra base de datos con la cantidad de ingredientes por receta.

```
missing_all <- apply(X=is.na(ingredients) , MARGIN = 1, FUN = sum)

ingredients <- ingredients %>%
  mutate(Num_of_Ing = 23 - missing_all)

head(ingredients)
```

```
##    Region          Ing1     Ing2        Ing3     Ing4     Ing5     Ing6
## 1 African       chicken cinnamon   soy_sauce    onion   ginger     <NA>
## 2 African cane_molasses   ginger       cumin   garlic tamarind    bread
## 3 African        butter   pepper       onion cardamom  cayenne   ginger
## 4 African     olive_oil   pepper       wheat     beef    onion cardamom
## 5 African         honey    wheat       yeast     <NA>     <NA>     <NA>
## 6 African        tomato cilantro lemon_juice    onion  cayenne scallion
##             Ing7    Ing8     Ing9 Ing10   Ing11   Ing12       Ing13  Ing14
## 1           <NA>    <NA>     <NA>  <NA>    <NA>    <NA>        <NA>   <NA>
## 2      coriander vinegar    onion  beef cayenne parsley wheat_bread yogurt
## 3 cottage_cheese  garlic brassica  <NA>    <NA>    <NA>        <NA>   <NA>
## 4          cumin  garlic     rice  leek    <NA>    <NA>        <NA>   <NA>
## 5           <NA>    <NA>     <NA>  <NA>    <NA>    <NA>        <NA>   <NA>
## 6           <NA>    <NA>     <NA>  <NA>    <NA>    <NA>        <NA>   <NA>
```

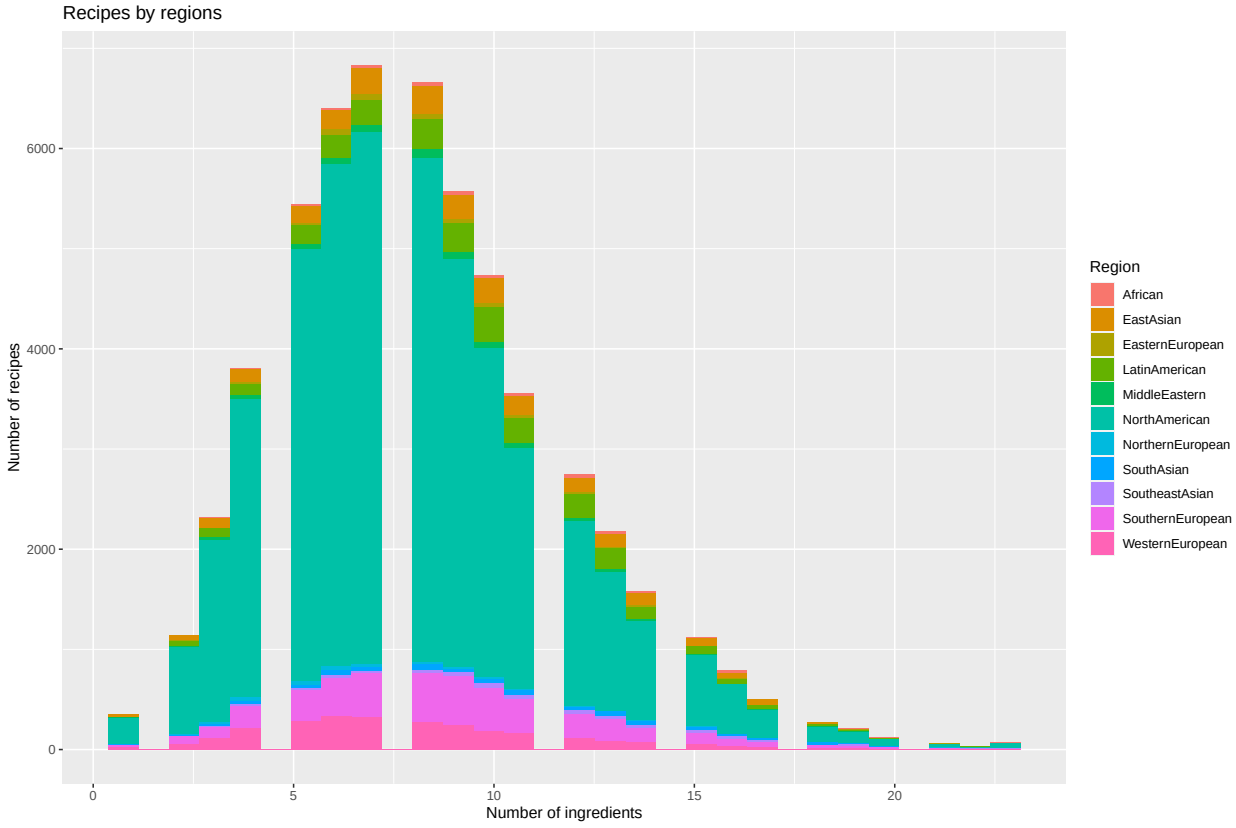
```
##           Ing15 Ing16 Ing17 Ing18 Ing19 Ing20 Ing21 Ing22 Ing23 Num_of_Ing
## 1          <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>          5
## 2 vegetable_oil   egg  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>         16
## 3          <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>          9
## 4          <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>         10
## 5          <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>          3
## 6          <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>          6
```

Gráficos por número de ingredientes por regiones

Seleccionamos solo las columnas de región y número de ingredientes para comparar la cantidad de ingredientes usados por receta y regiones.

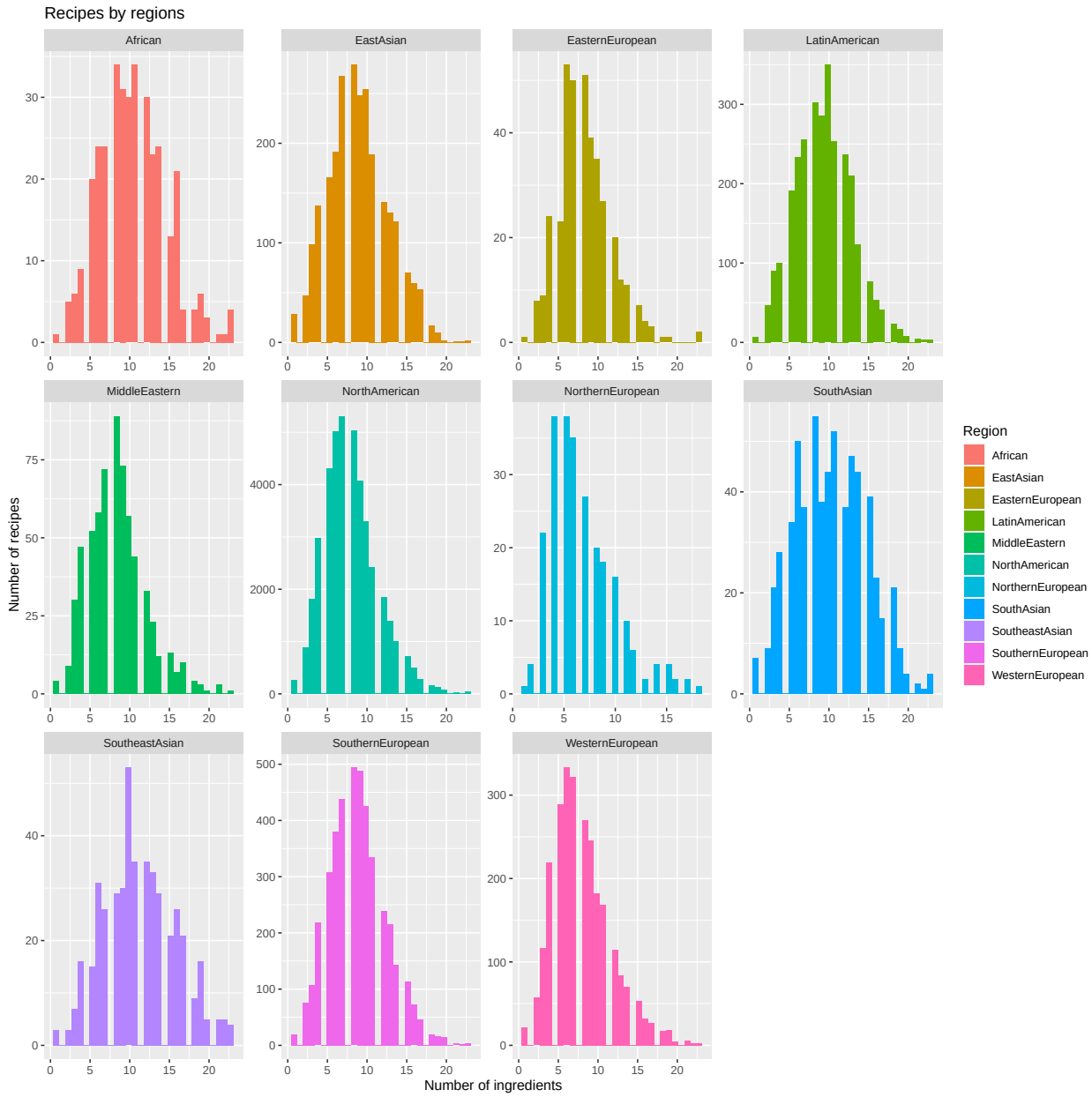
```
df <- ingredients %>%
  select(Region, Num_of_Ing)
```

```
p1 <- ggplot(df, aes(x=Num_of_Ing, fill= Region)) +
  geom_histogram() +
  xlab("Number of ingredients") +
  ylab("Number of recipes") +
  ggtitle("Recipes by regions")

p1
```



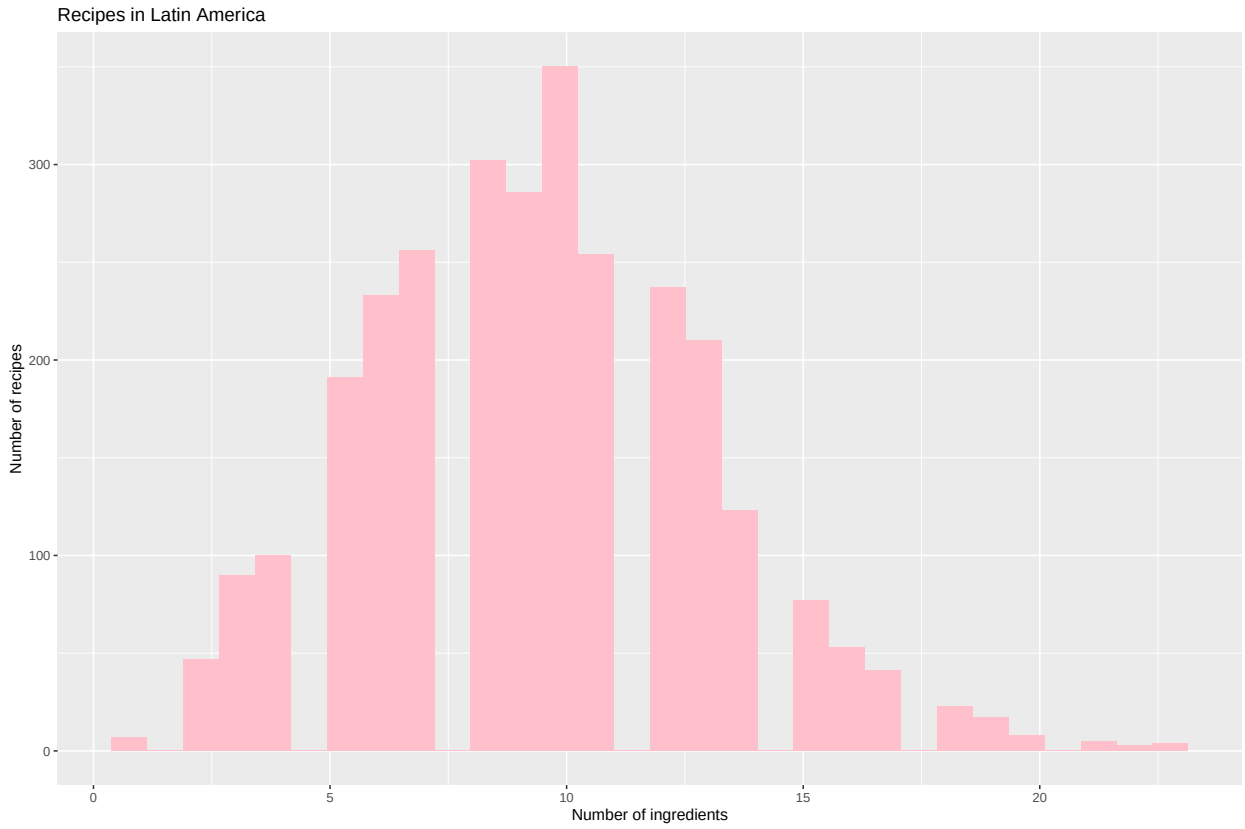
Agregamos la escala libre para cada región.

```
p2 <- ggplot(df, aes(x=Num_of_Ing, fill = Region)) +
  geom_histogram() +
  facet_wrap(~Region, scales = "free") +
  xlab("Number of ingredients") +
  ylab("Number of recipes") +
  ggtitle("Recipes by regions")

p2
```



Filtramos por solo una región.

```
LatinA <- ingredients %>%
  filter(Region == "LatinAmerican")


p3 <- ggplot(LatinA, aes(x=Num_of_Ing)) +
  geom_histogram(fill="pink") +
  xlab("Number of ingredients") +
  ylab("Number of recipes") +
  ggtitle("Recipes in Latin America")

p3
```



Ingredientes únicos

Obtenemos una lista con los ingredientes únicos en toda la base de datos y eliminamos el NA.

```
unique_ing <- as.character(unique(unlist(ingredients[,2:24])))
is.na(unique_ing)
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [193] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [205] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [217] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [229] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [241] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [253] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [265] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [277] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [289] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
## [301] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [313] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [325] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [337] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [349] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [361] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [373] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
unique_ing <- unique_ing[-295]
is.na(unique_ing)
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [193] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [205] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [217] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [229] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [241] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [253] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
## [265] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [277] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [289] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [301] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [313] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [325] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [337] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [349] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [361] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [373] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

Usamos la siguiente función para crear un dataframe por región con la cantidad de veces que se usa un ingrediente en todas las recetas.

```
ing_by_reg <- function(x){
  name <- df_regions$Region[x]
  i=1
  counts <- list()
  for (j in unique_ing) {
    counts[i] <- sum(str_count(df_regions$data[x], j))
    i = i + 1
  }

  count_ing <- unlist(counts)
  df_x <- data.frame(rep(name, length(unique_ing)), unique_ing, count_ing)
  df_p <- df_x %>%
  return(df_x)
}
```

Aplicamos la función a las regiones.

```
ing_African <- ing_by_reg(1)
ing_EastAsian <- ing_by_reg(2)
ing_EasternEuropean <- ing_by_reg(3)
ing_LatinAmerican <- ing_by_reg(4)
ing_MiddleEastern <- ing_by_reg(5)
ing_NorthernEuropean <- ing_by_reg(7)
ing_SouthAsian <- ing_by_reg(8)
ing_SoutheastAsian <- ing_by_reg(9)
ing_westernEuropean <- ing_by_reg(11)
ing_SouthernEuropean <- ing_by_reg(10)
# Como NorthAmerican tiene más de 4000 recetas esto consume mucho tiempo.
#ing_NorthAmerican <- ing_by_reg(6)
```

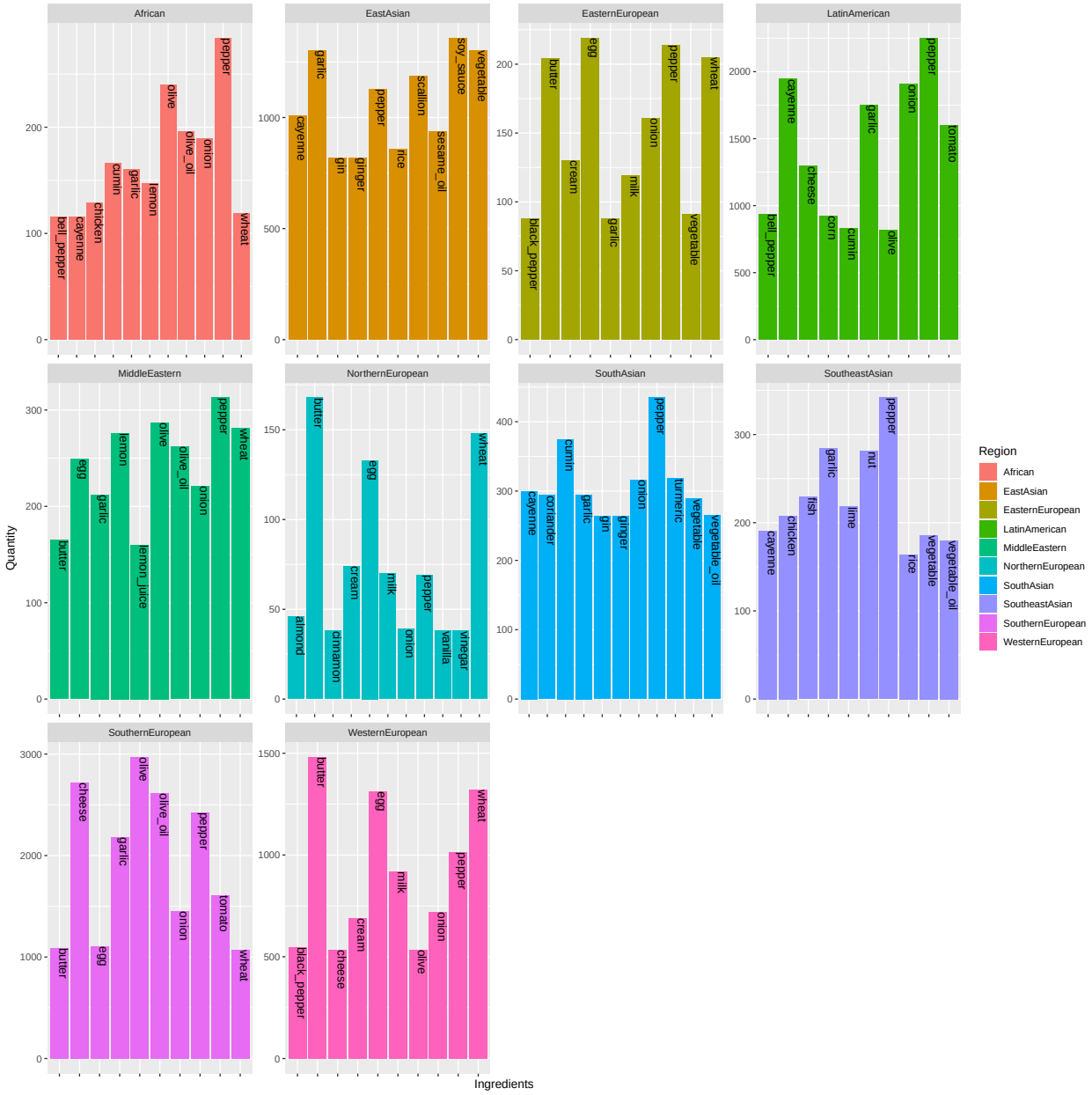
Unimos estos dataframe en uno solo.

```
counts_ingd <- rbind.data.frame(ing_African, ing_EastAsian, ing_EasternEuropean,
                                ing_LatinAmerican, ing_MiddleEastern, ing_NorthernEuropean,
                                ing_SouthAsian, ing_SoutheastAsian, ing_westernEuropean,
                                ing_SouthernEuropean)
names(counts_ingd) <- c("Region", "Ingredients", "Quantity")

head(counts_ingd)
```

```
##    Region   Ingredients Quantity
## 1 African       chicken      129
## 2 African cane_molasses       11
## 3 African        butter       81
## 4 African     olive_oil      196
## 5 African         honey       42
## 6 African        tomato      100
```

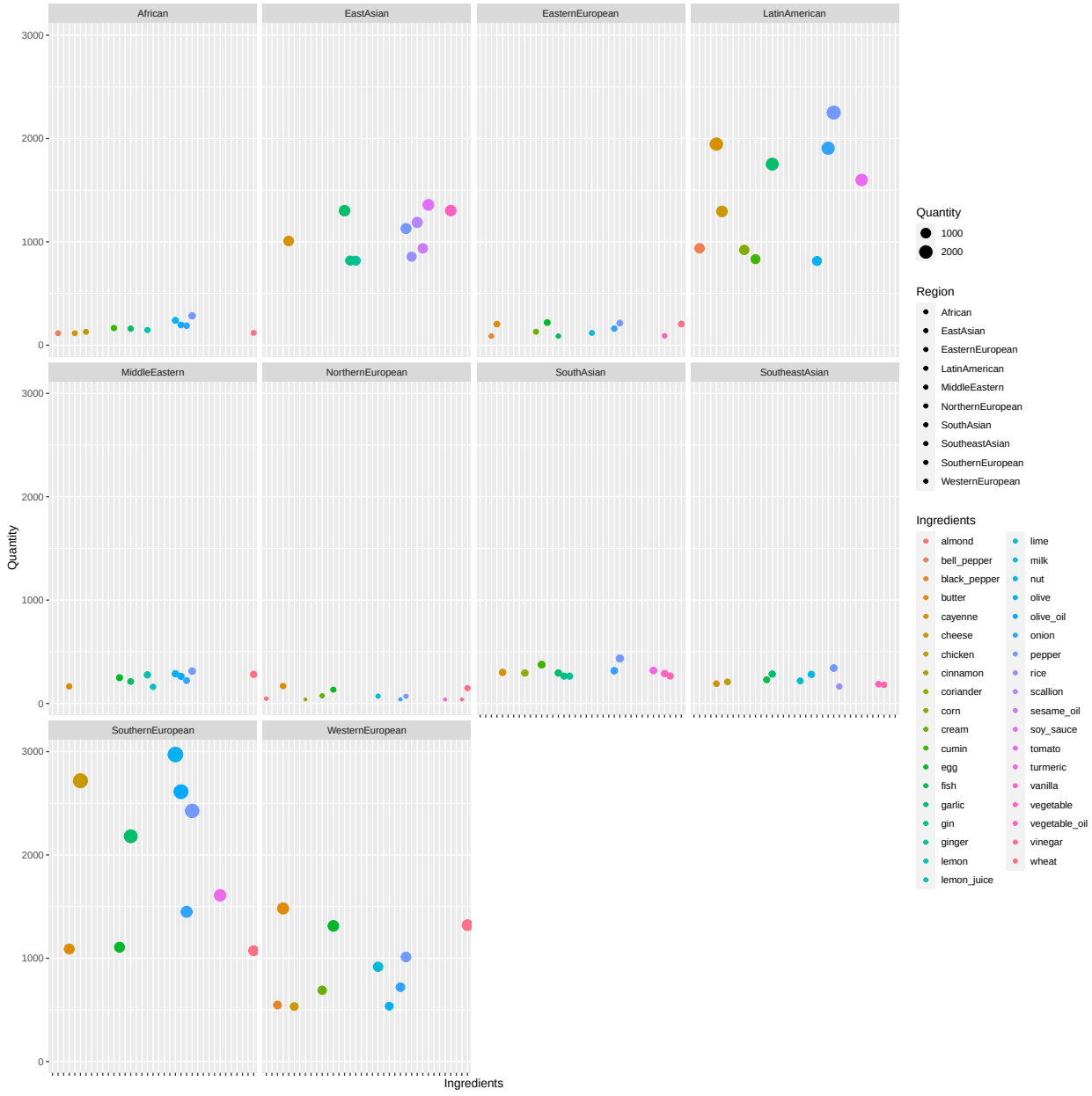
Graficamos los ingredientes más usados por regiones.

```
p4 <- counts_ingd %>%
  group_by(Region) %>%
  top_n(n = 10, Quantity) %>%
  ggplot(., aes(x=Ingredients, y= Quantity, fill= Region)) +
    geom_histogram(stat="identity") +
    theme(axis.text.x = element_blank())+
    facet_wrap(~Region, scales="free")+
    geom_text(aes(label=Ingredients, angle=-90, hjust=0, vjust=0))

p4
```



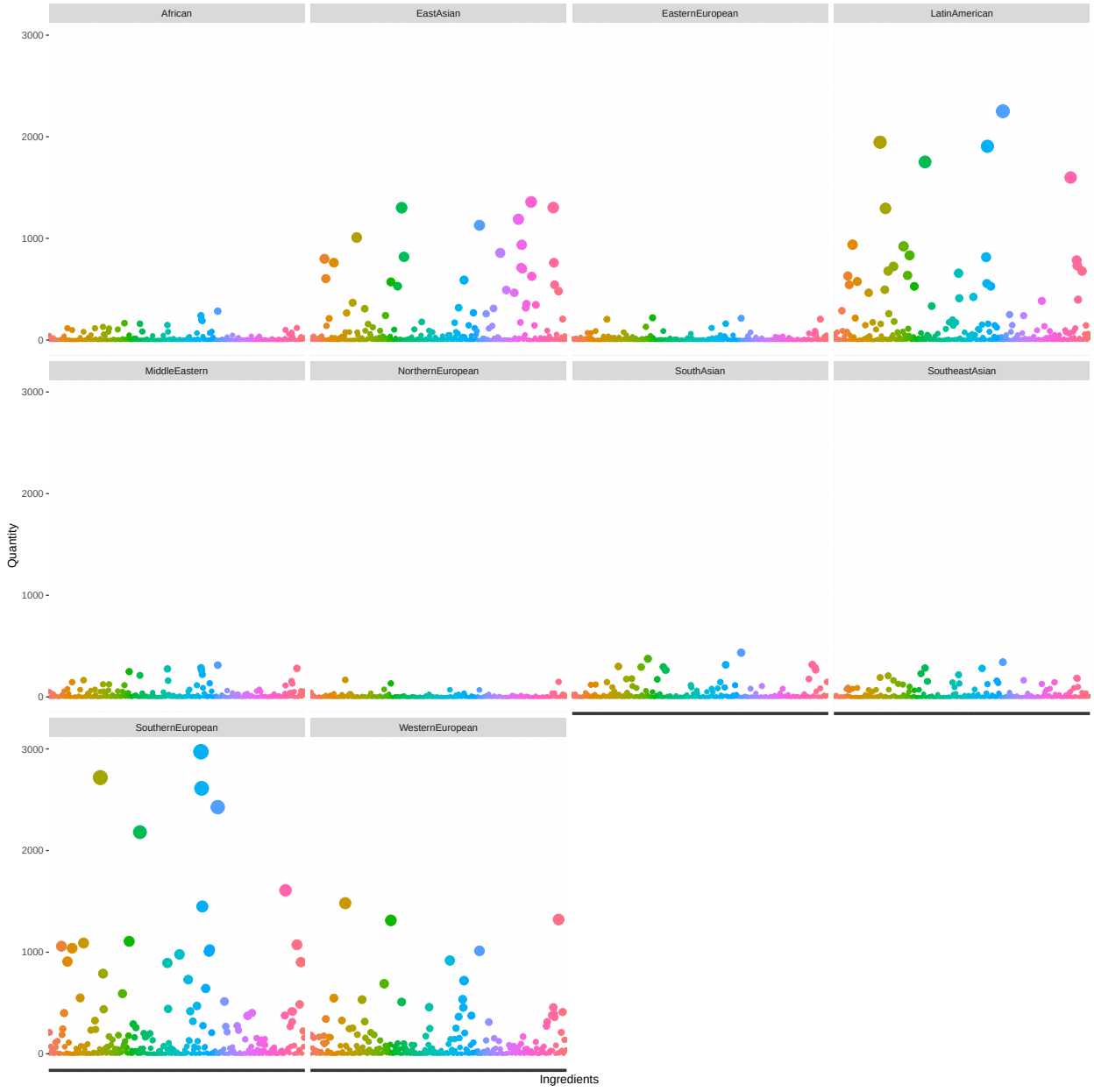
Otro estilo de gráfico.

```
p5 <- counts_ingd %>%
  group_by(Region) %>%
  top_n(n = 10, Quantity) %>%
  ggplot(., aes(x= Ingredients, y=Quantity, fill=Region))+
    geom_point(aes(color=Ingredients, size=Quantity)) +
    theme(axis.text.x = element_blank())+
    facet_wrap(~Region)


p5
```



Graficamos todos los ingredientes.

```
p6 <- ggplot(counts_ingd, aes(x= Ingredients, y=Quantity, fill=Region))+
  geom_point(aes(color=Ingredients, size=Quantity)) +
  theme(axis.text.x = element_blank(), legend.position = "none") +
  facet_wrap(~Region)


p6
```



Porcentajes

Agregamos encabezados a nuestros dataframe por regiones.

```
names(ing_African) <- c("Region", "Ingredient", "Quantity")
names(ing_EastAsian) <- c("Region", "Ingredient", "Quantity")
names(ing_EasternEuropean) <- c("Region", "Ingredient", "Quantity")
names(ing_LatinAmerican) <- c("Region", "Ingredient", "Quantity")
names(ing_MiddleEastern) <- c("Region", "Ingredient", "Quantity")
names(ing_NorthernEuropean) <- c("Region", "Ingredient", "Quantity")
names(ing_SouthAsian) <- c("Region", "Ingredient", "Quantity")
names(ing_SoutheastAsian) <- c("Region", "Ingredient", "Quantity")
```

```
names(ing_westernEuropean) <- c("Region", "Ingredient", "Quantity")
names(ing_SouthernEuropean) <- c("Region", "Ingredient", "Quantity")
```

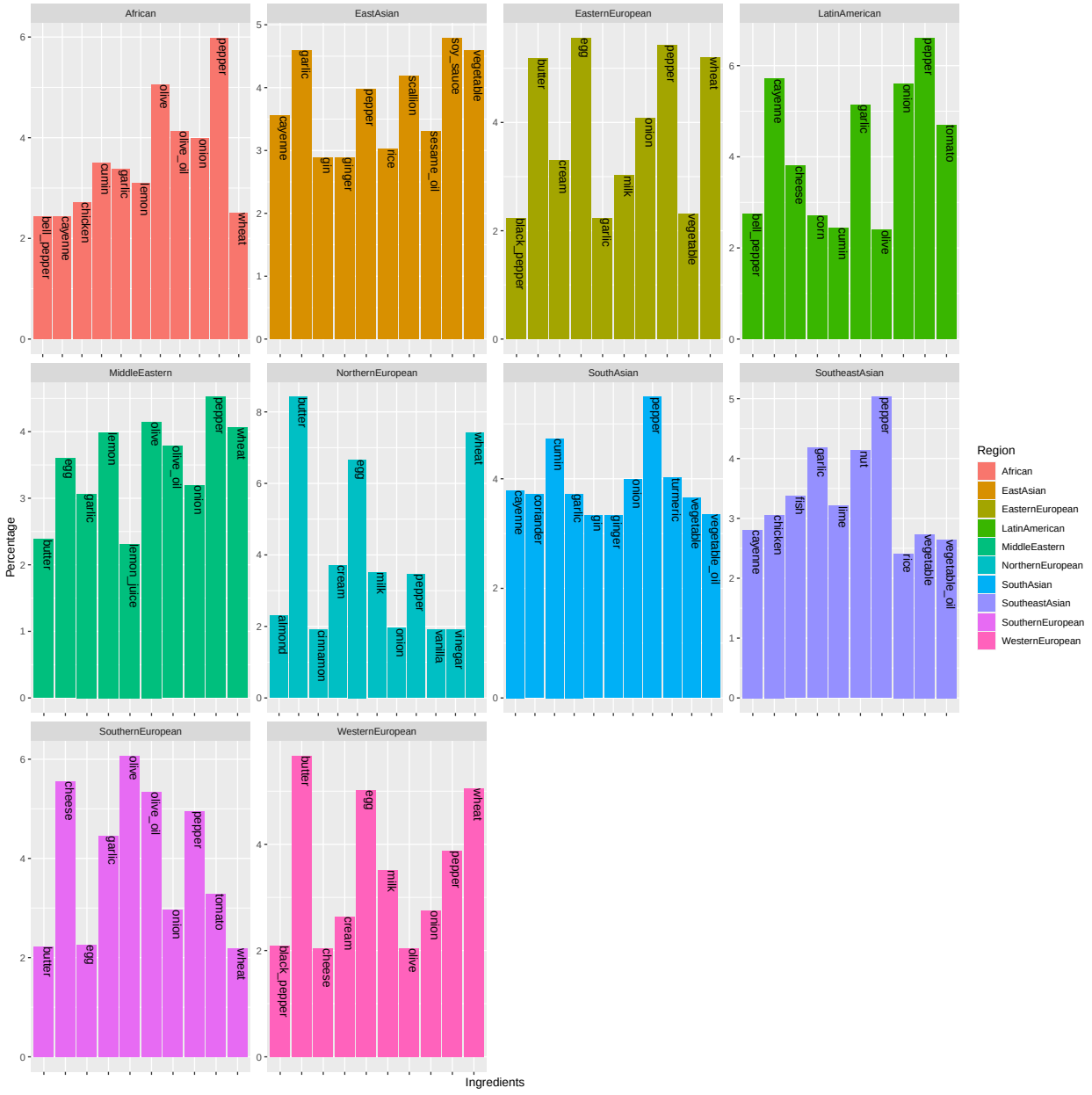
Calculamos los porcentajes de cada ingrediente por región.

```
counts_ingd_rel <- counts_ingd %>%
  group_by(Region) %>%
  dplyr::mutate(Total = sum(Quantity), Percentage = (Quantity/Total)*100 )

head(counts_ingd_rel)
```

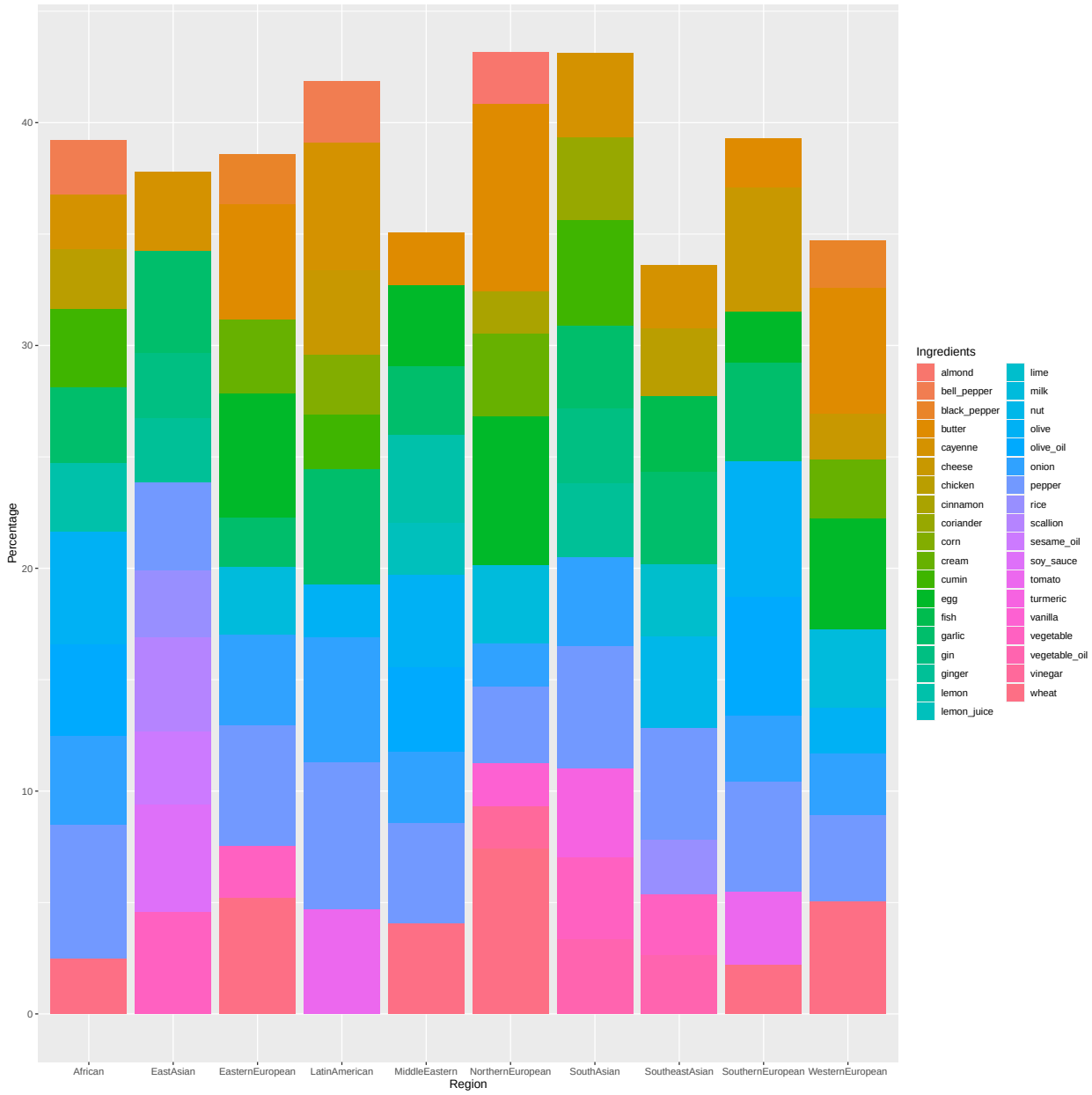
```
## # A tibble: 6 x 5
## # Groups:   Region [1]
##   Region  Ingredients   Quantity Total Percentage
##   <chr>   <chr>            <int> <int>      <dbl>
## 1 African chicken            129  4748      2.72
## 2 African cane_molasses       11  4748      0.232
## 3 African butter              81  4748      1.71
## 4 African olive_oil          196  4748      4.13
## 5 African honey               42  4748      0.885
## 6 African tomato             100  4748      2.11
```

```
top_N <- counts_ingd_rel %>%
  group_by(Region) %>%
  top_n(n = 10, Percentage)
```

```
p7 <- counts_ingd_rel %>%
  group_by(Region) %>%
  top_n(n = 10, Percentage) %>%
  ggplot(., aes(x=Ingredients, y= Percentage, fill= Region)) +
  geom_histogram(stat="identity") +
  theme(axis.text.x = element_blank())+
  facet_wrap(~Region, scales="free")+
  geom_text(aes(label=Ingredients, angle=-90, hjust=0, vjust=0))

p7
```




```
p8 <- counts_ingd_rel %>%
  group_by(Region) %>%
  top_n(n = 10, Percentage) %>%
  ggplot(., aes(x=Region, y= Percentage, fill= Ingredients)) +
  geom_col()

p8
```



A futuro: Matriz con combinaciones de ingredientes

Ordenamos alfabéticamente nuestros ingredientes y creamos una matriz de combinaciones de ingredientes para encontrar cuales combinaciones son más usadas por regiones.

```
ing_alph <- sort(unique_ing)
```