Load the Data, Explore the Data

The data for this example can be read directly from our course webpage. When we first examine the data, we will see that the rows correspond to different types of food/drink and the columns correspond to the 4 countries within the UK. Our first matter of business is transposing this data so that the 4 countries become our observations (i.e. rows).

```
> library(reshape2) #melt data matrix into 3 columns
> library(ggplot2) #heatmap
> food=read.csv("http://birch.iaa.ncsu.edu/~slrace/LinearAlgebra2021/Code/ukfood.csv",
+               header=TRUE,row.names=1)
> head(food)
```

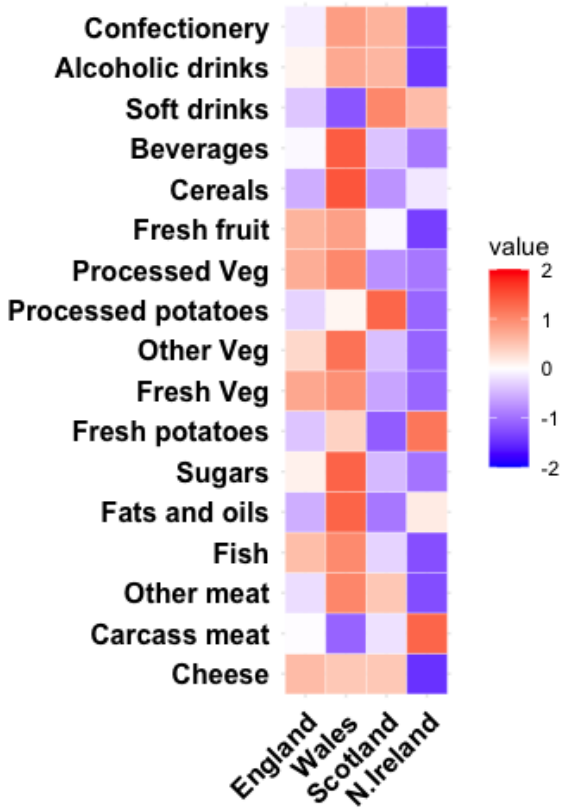
```
             England Wales Scotland N.Ireland
Cheese           105   103      103        66
Carcass meat     245   227      242       267
Other meat       685   803      750       586
Fish             147   160      122        93
Fats and oils    193   235      184       209
Sugars           156   175      147       139
```

```
> food=as.data.frame(t(food))
> head(food)
```

```
          Cheese Carcass meat Other meat Fish Fats and oils Sugars
England      105          245        685  147           193    156
Wales        103          227        803  160           235    175
Scotland     103          242        750  122           184    147
N.Ireland     66          267        586   93           209    139
          Fresh potatoes Fresh Veg Other Veg Processed potatoes Processed Veg
England              720       253       488                198           360
Wales                874       265       570                203           365
Scotland             566       171       418                220           337
N.Ireland           1033       143       355                187           334
          Fresh fruit Cereals Beverages Soft drinks Alcoholic drinks
England          1102    1472        57        1374              375
Wales            1137    1582        73        1256              475
Scotland          957    1462        53        1572              458
N.Ireland         674    1494        47        1506              135
          Confectionery
England              54
Wales                64
Scotland             62
N.Ireland            41
```

Next we will visualize the information in this data using a simple heat map. To do this we will standardize and then melt the data using the reshape2 package, and then use a ggplot() heatmap.

```
> food.std = scale(food, center=T, scale = T)
> food.melt = melt(food.std, id.vars = row.names(food.std), measure.vars = 1:17)
> ggplot(data = food.melt, aes(x=Var1, y=Var2, fill=value)) +
+   geom_tile(color = "white")+
+   scale_fill_gradient2(low = "blue", high = "red", mid = "white",
+                        midpoint = 0, limit = c(-2,2), space = "Lab"
+                        ) +  theme_minimal()+
+   theme(axis.title.x = element_blank(),axis.title.y = element_blank(),
+         axis.text.y = element_text(face = 'bold', size = 12, colour = 'black'),
+         axis.text.x = element_text(angle = 45, vjust = 1, face = 'bold',
+                                    size = 12, colour = 'black', hjust = 1))+coord_fixed()
```



prcomp() function for PCA

The prcomp() function is the one I most often recommend for reasonably sized principal component calculations in R. This function returns a list with class "prcomp" containing the following components (from help prcomp):

1. sdev: the standard deviations of the principal components (i.e., the square roots of the eigenvalues of the covariance/correlation matrix, though the calculation is actually done with the singular values of the data matrix).

2. rotation: the matrix of *variable loadings* (i.e., a matrix whose columns contain the eigenvectors). The function princomp returns this in the element loadings.

3. x: if retx is true *the value of the rotated data (i.e. the scores)* (the centred (and scaled if requested) data multiplied by the rotation matrix) is returned. Hence, cov(x) is the diagonal matrix $diag(sdev^2)$. For the formula method, napredict() is applied to handle the treatment of values omitted by the na.action.

4. center, scale: the centering and scaling used, or FALSE.

The option scale = TRUE inside the prcomp() function instructs the program to use **correlation PCA**. The **default is covariance PCA**.

```
> pca=prcomp(food, scale = T)
```

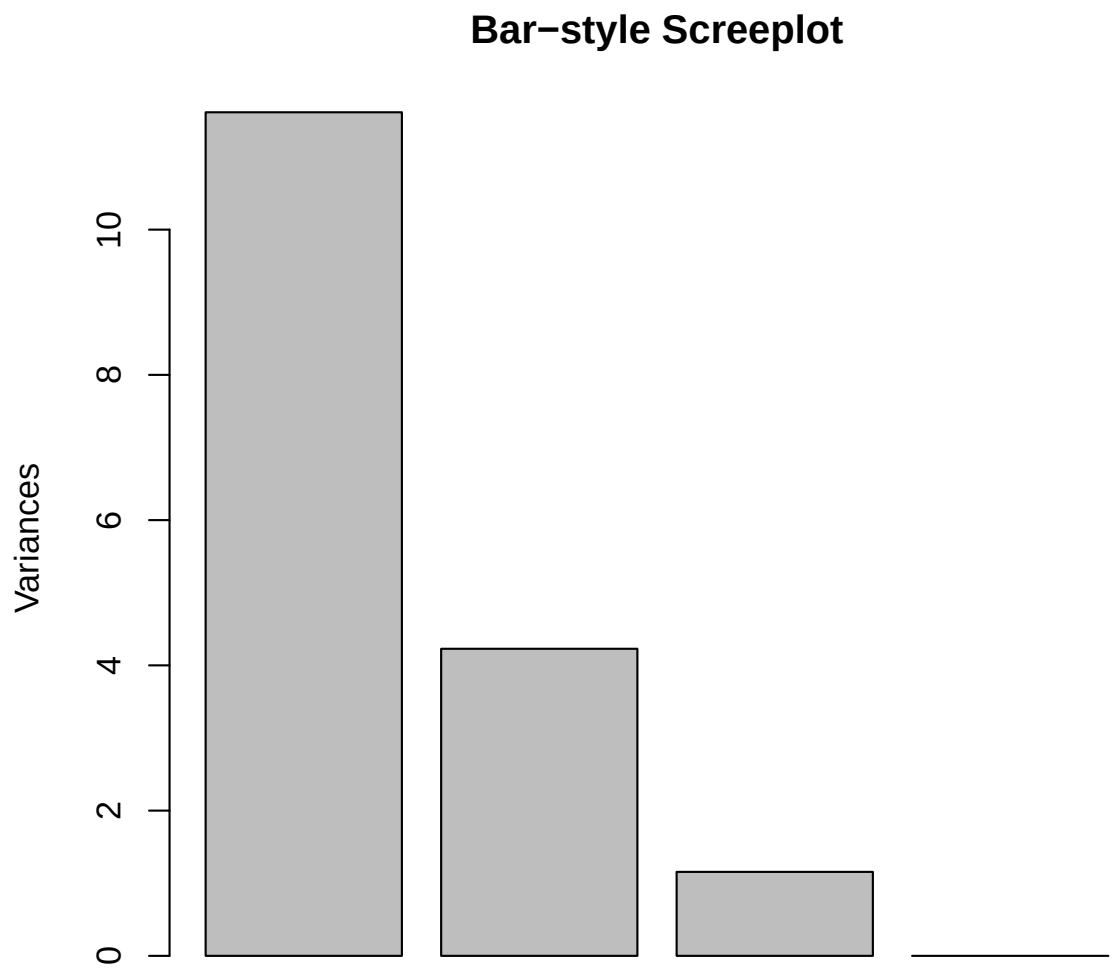
This first plot just looks at magnitudes of eigenvalues - it is essentially the screeplot in barchart form.

The next plot views our four datapoints (locations) projected onto the 2-dimensional subspace (from 17 dimensions) that captures as much information (i.e. variance) as possible.

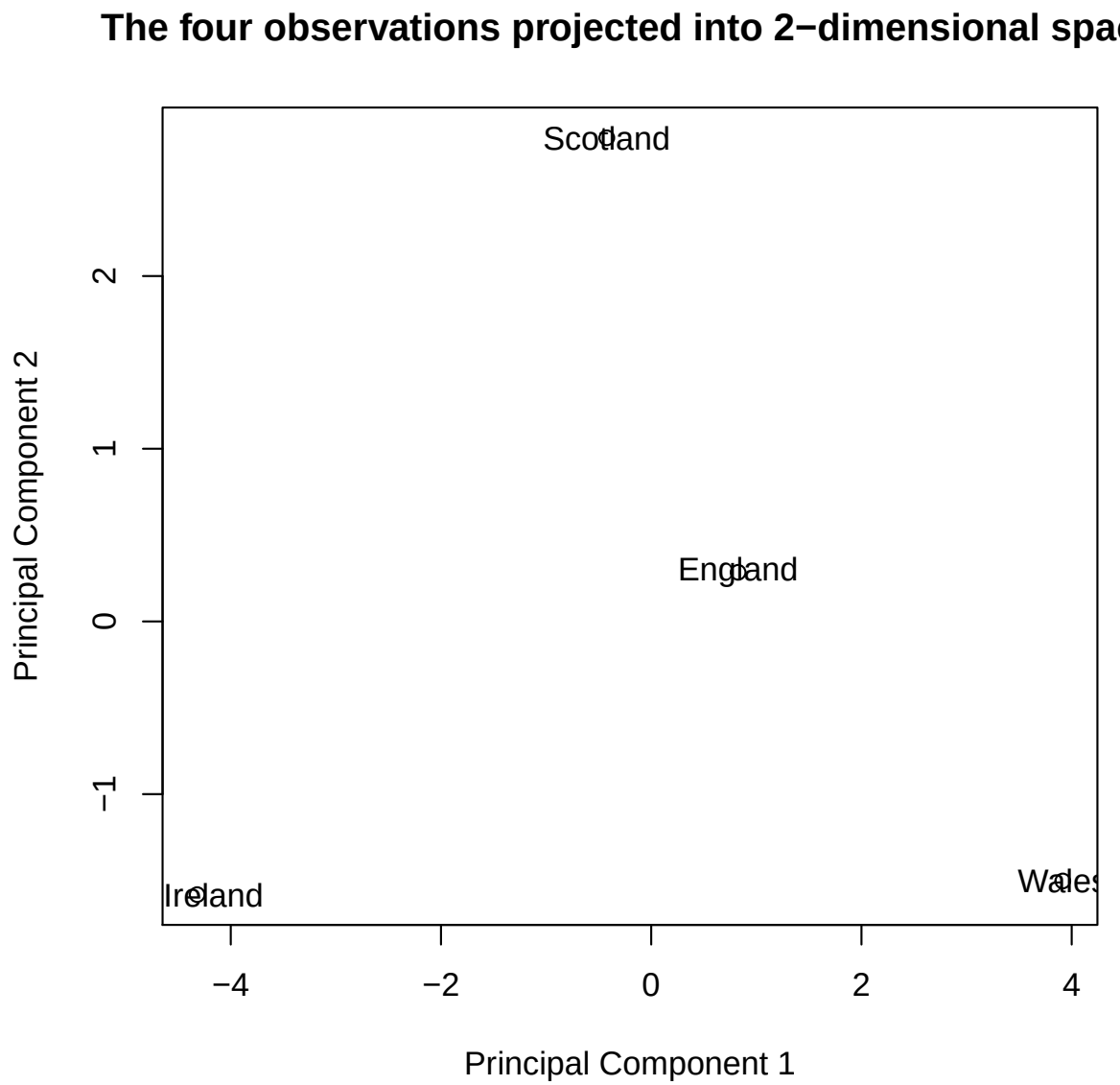
```
> summary(pca)
```

```
Importance of components:
                          PC1    PC2     PC3       PC4
Standard deviation     3.4082 2.0562 1.07524 6.344e-16
Proportion of Variance 0.6833 0.2487 0.06801 0.000e+00
Cumulative Proportion  0.6833 0.9320 1.00000 1.000e+00
```

```
> plot(pca, main = "Bar-style Screeplot")
```



```
> plot(pca$x,
+      xlab = "Principal Component 1",
+      ylab = "Principal Component 2",
+      main = 'The four observations projected into 2-dimensional space')
> text(pca$x[,1], pca$x[,2],row.names(food))
```



The BiPlot

Now we can also view our original variable axes projected down onto that same space!

```
> biplot(pca, cex = c(1.5, 1), col = c('black','red'),
+        xlim = c(-0.8,0.8), ylim = c(-0.6,0.7))
```

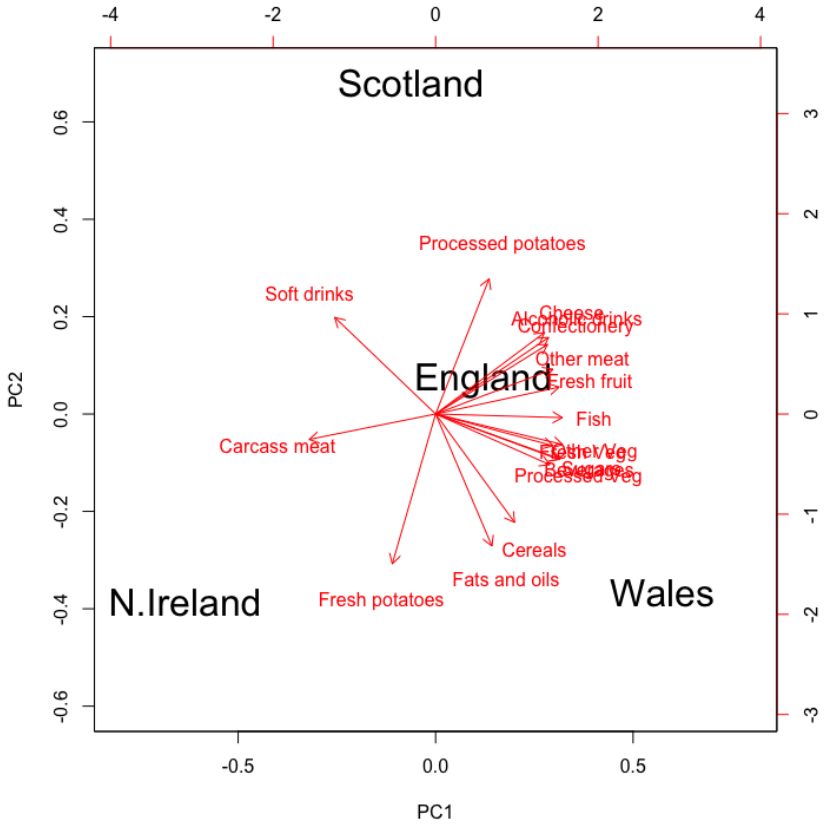
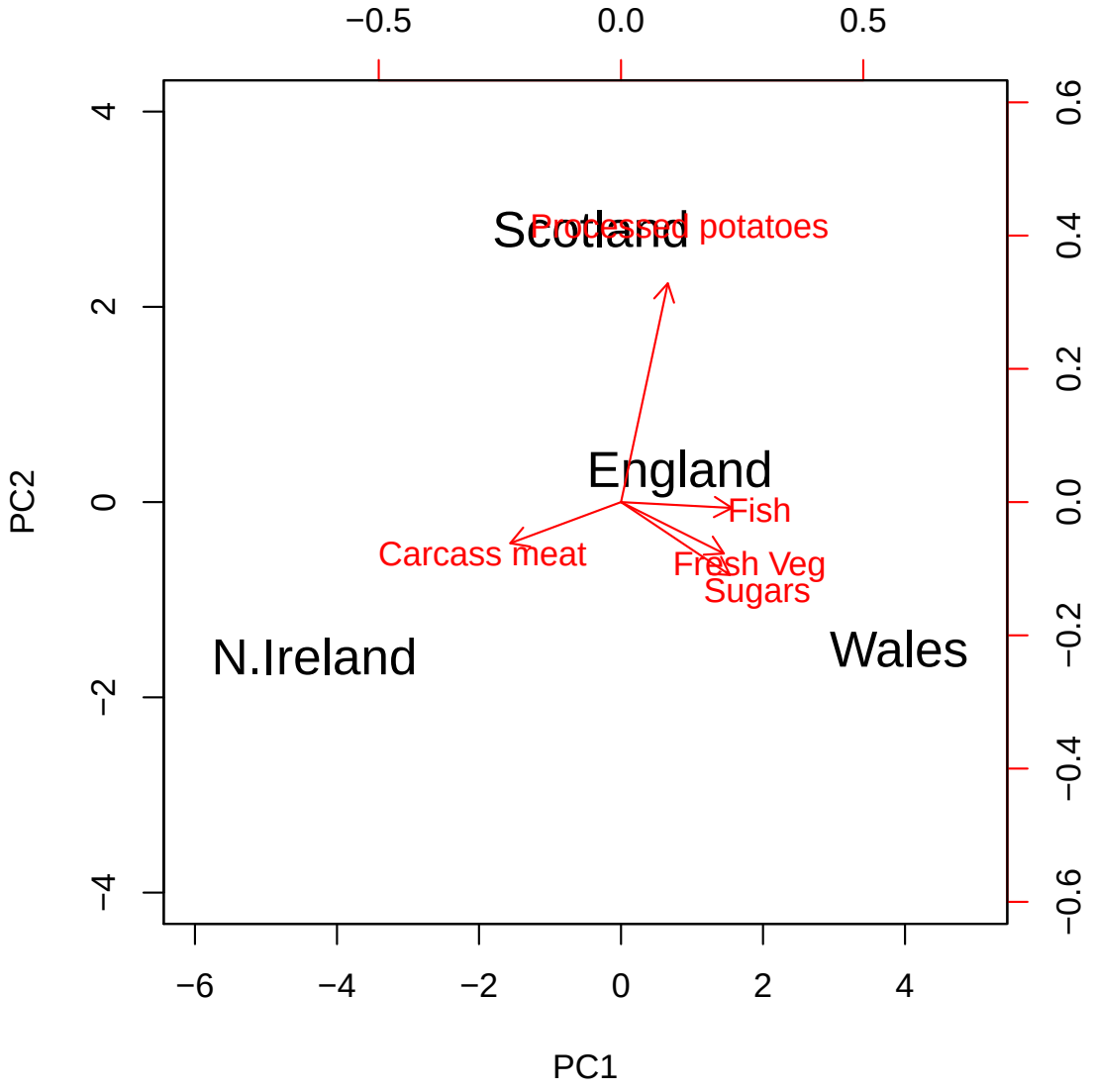


Figure 1: BiPlot: The observations and variables projected onto the same plane.

Formatting the biplot for readability

I will soon introduce the autoplot() function from the ggfortify() package, but for now I just want to show you that you can specify *which* variables (and observations) to include in the biplot by directly specifying the loadings matrix and scores matrix of interest in the biplot function:

```
> desired.variables = c(2,4,6,8,10)
> biplot(pca$x, pca$rotation[desired.variables,1:2], cex = c(1.5, 1),
+        col = c('black','red'), xlim = c(-6,5), ylim = c(-4,4))
```



What are all these numbers on the axes?

Those numbers relate to the scores on PC1 and PC2 (sometimes normalized so that each new variable has variance 1 - and sometimes not) and the loadings on PC1 and PC2 (sometimes normalized so that each variable vector is a unit vector - and sometimes scaled by the eigenvalues or square roots of the eigenvalues in some fashion).

Generally, I've never found it useful to hunt down how each package is rendering the biplot, as they should be providing the same information regardless of the *numbers* on the axes. We don't actually use those numbers to help us draw conclusions. We use the directions of the arrows and the layout of the points in reference to those direction arrows.