Factor Analysis and Independent Component Analysis

Sam Bowyer

2023-02-03

Task 1

For the first task we will use the dataset `mtcars` which contains 11 values for each of 32 different types of car.

```
data(mtcars)
head(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

We can first identify a suitable choice for the number of components to use using the `factanal` function, which performs maximum-likelihood factor analysis on our dataset, From this we can obtain the p-value of a test on the hypothesis that k factors are suitable for representing the data with the factor analysis model.

```
factanal(mtcars, factors = 1)$PVAL
```

```
##   objective
## 1.49622e-17
```

```
factanal(mtcars, factors = 2)$PVAL
```

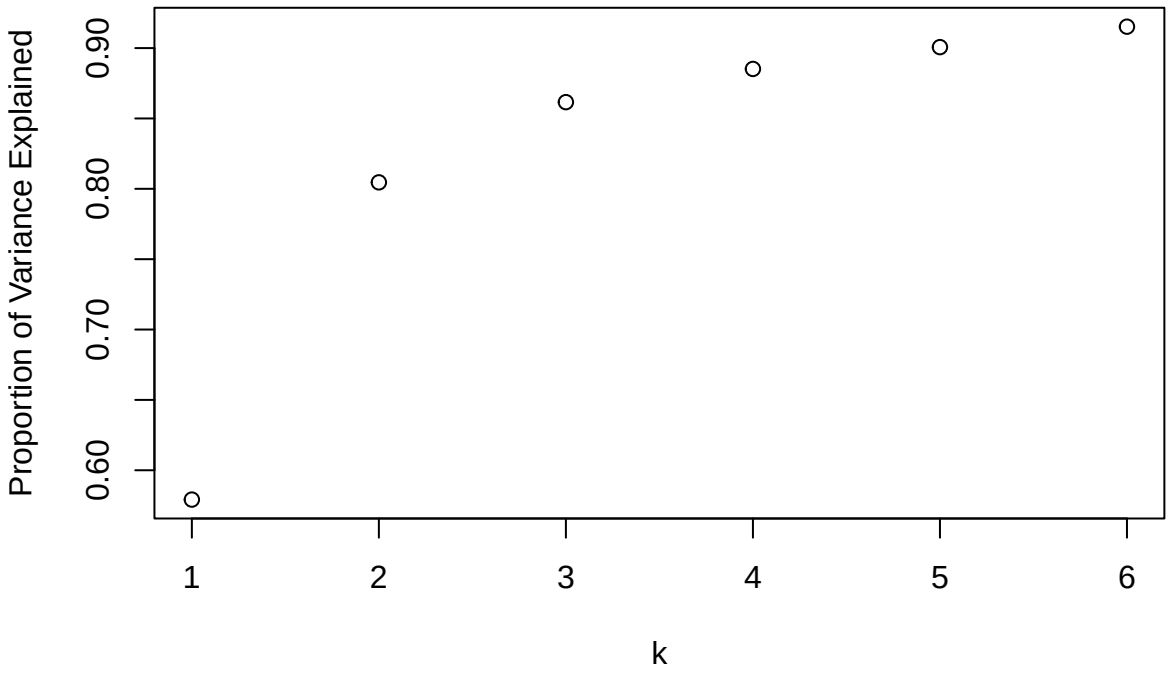
```
##   objective
## 0.000404751
```

```
factanal(mtcars, factors = 3)$PVAL
```

```
## objective
## 0.2051923
```

We see that $k = 3$ is the lowest k with a p-value above 5%, suggesting it would be a suitable choice for the number of factors, but we can further justify this by considering the total variance explained by k factors as k ranges from 1 to 6 (`factanal` breaks down for $k > 6$ on `mtcars` due to insufficient number of free parameters: if $k = 7$ then $\Delta_{k,p} = \frac{(p-k)^2}{2} - \frac{(p+k)}{2} = \frac{(11-7)^2}{2} - \frac{(11+7)}{2} = 8 - 9 = -1$).

```
vars = rep(0,6)
for (i in 1:6){
  loadings = factanal(mtcars, factors=i)$loadings
  varsPerVariable = colSums(loadings^2)/nrow(loadings)
  vars[i] = sum(varsPerVariable)
}
plot(1:6, vars, xlab="k", ylab="Proportion of Variance Explained")
```



Whilst we do see higher amounts of variance explained for $k \geq 4$ than $k = 3$, the significant increase for $k = 3$ compared to $k = 1$ and $k = 2$ reinforces the idea that it would be suitable to use three components in our model in order to reduce the dimensionality of our data. This leads to $\Delta_{k,p} = \frac{(p-k)^2}{2} - \frac{(p+k)}{2} = \frac{(11-3)^2}{2} - \frac{(11+3)}{2} = 32 - 7 = 25$. As this value is positive we must then look for an approximate solution, since there are more constraints than free parameters.

We will be using the iterated principal factor analysis algorithm via the `fapa` command, which requires the correlation matrix of our data as input along with k (`numFactors`) and the method for calculating the initial communality values—we choose the default option of `SMC` where (as explained in the function's documentation) "Initial communalities are estimated by taking the squared multiple correlations of each indicator after regressing the indicator on the remaining variables. The following equation is employed to find the squared multiple correlation: $1 - 1/\text{diag}(R^{-1})$" [1]. Since the communalities are given by the diagonals of $\Lambda\Lambda^T = R - \Psi$, this then tells us that the initial specific variances are given by $\hat{\Psi}^{(0)} = 1/\text{diag}(R^{-1})$.

```
library(fungible)
R = cor(mtcars)
model = fapa(R, numFactors=3, communality="SMC")
```

This then gives us the following estimates for the loading matrix $\hat{\Lambda}$ and specific variances $\{\hat{\psi}_{jj}\}_{j=1}^{p}$.

```
loadings = model$loadings
rownames(loadings) = colnames(mtcars)
loadings
```

```
##            [,1]        [,2]        [,3]
## mpg   0.9241493 -0.03253603 -0.15703674
## cyl  -0.9648148 -0.06577705 -0.15017332
## disp -0.9406706  0.08619078 -0.04948988
## hp   -0.8411009 -0.38962339  0.09310713
## drat  0.7293312 -0.41443372  0.08847318
## wt   -0.8855965  0.23870931  0.25816072
## qsec  0.5183040  0.75730051  0.32427523
## vs    0.7696451  0.34975717  0.24320756
## am    0.5911699 -0.67240793 -0.11308796
```

```
## gear  0.5243656 -0.73759195  0.21704014
## carb -0.5463130 -0.65201435  0.37728569
```

```
specificVars = diag(cov(mtcars)) - model$h2
specificVars
```

```
##           mpg           cyl          disp            hp          drat
##  3.544433e+01  2.231770e+00  1.535991e+04  4.699999e+03 -4.256255e-01
##            wt          qsec            vs            am          gear
##  4.946875e-02  2.245869e+00 -5.198013e-01 -5.654112e-01 -3.217527e-01
##          carb
##  1.742946e+00
```

To aid interpretability, we will consider the varimax-rotated loadings.

```
loadings = varimax(loadings)$loadings
loadings
```

```
##
## Loadings:
##      [,1]   [,2]   [,3]
## mpg   0.488 -0.665  0.447
## cyl  -0.282  0.627 -0.696
## disp -0.301  0.718 -0.537
## hp   -0.557  0.305 -0.682
## drat        -0.807  0.238
## wt   -0.481  0.784 -0.247
## qsec  0.213  0.177  0.933
## vs    0.231 -0.289  0.798
## am          -0.892 -0.106
## gear -0.245 -0.898
## carb -0.775        -0.507
##
##                  [,1]  [,2]  [,3]
## SS loadings    1.720 4.436 3.332
## Proportion Var 0.156 0.403 0.303
## Cumulative Var 0.156 0.560 0.862
```

Recalling that this model leads to the approximation of data points $x_i \approx \hat{\lambda}_{(1)}\hat{f}_{i1} + \hat{\lambda}_{(2)}\hat{f}_{i2} + \hat{\lambda}_{(3)}\hat{f}_{i3}$ we might interpret the first factor as relating to the general size/strength of the car's engine since all of the negative loadings in the first column occur for variables where (roughly speaking) a larger value might suggest a larger engine/car (number of cylinders `cyl`, displacement `disp`, gross horsepower `hp`, weight `wt` and number of gears/carburetors `gear`/`carb`) whereas we see positive values for miles per gallon `mpg` and quarter-mile time `qsec` where a smaller value might suggest a larger engine/car. This is further supported by the fact that the loadings in the first column for the rear axle ratio `drat` and automatic/manual transmission type `am` have are very small as they aren't strong indicators of a car's size/strength. (The blank values indicate that the values are between -0.1 and 0.1.)

```
# The loadings shown as blank in the previous output block have absolute values < 0.1
c(loadings["drat",1], loadings["am",1], loadings["carb",2], loadings["gear",3])
```

```
##         drat           am         carb         gear
##  0.063242506  0.082515883 -0.092791773 -0.005730166
```

Interpreting the second and third factors is slightly trickier. The second has loadings of opposite sign compared to the first factor with the exception of the rear axle ratio `drat`, quarter-mile time `qsec`, transmission type `am` and number of carburetors `carb` (although this last loading is very small. These differences could be seen as refining the distinctions made between 'larger/stronger' cars in the first factor based on the previously

unimportant variables like `drat` and `am`. The third factor has somewhat similar pattern of loading to the first factor and so represents a similar set of characteristics with some further refinement.

We can calculate our approximations to these factors using the linear regression model suggested in the notes, giving the following (with a generalised least squares estimate of the regression coefficients):

```
psi = diag(specificVars)
A = solve(t(loadings) %*% solve(psi) %*% loadings) %*% t(loadings) %*% solve(psi)
A
```

```
##              [,1]        [,2]          [,3]          [,4]       [,5]
## [1,]  0.005292457 -0.18352660 -2.133512e-05 -4.761624e-05 -0.6364783
## [2,] -0.002033426  0.04214365  2.953467e-06  4.369373e-05 -0.1998924
## [3,] -0.011389266  0.44616202  4.348695e-05  1.877718e-04  0.5342328
##             [,6]       [,7]       [,8]       [,9]      [,10]      [,11]
## [1,] -1.98625132  0.1854169 -0.8137240 -0.1426108 -0.8577110 0.02666919
## [2,]  0.07776878 -0.1263091  0.3269106 -0.3081274 -0.6761670 0.16282109
## [3,]  0.15743667 -0.7381727  2.4840768 -0.7880174 -0.2154674 0.29783089
```

```
f = A %*% t(mtcars)
rownames(f) = c("f1","f2", "f3")
t(f)
```

```
##                           f1        f2         f3
## Mazda RX4           -9.099728 -4.800946  -7.647483
## Mazda RX4 Wag       -9.502389 -4.851848  -8.020713
## Datsun 710          -8.588616 -5.336243  -8.636191
## Hornet 4 Drive      -9.100477 -4.144945  -7.597859
## Hornet Sportabout   -9.586867 -3.906942  -6.985191
## Valiant             -9.255333 -4.154053  -8.270805
## Duster 360         -10.075342 -3.422133  -5.402708
## Merc 240D           -9.778690 -4.945716  -8.547271
## Merc 230            -9.317834 -5.356421 -10.547233
## Merc 280           -11.081437 -4.334294  -5.570571
## Merc 280C          -10.977597 -4.407232  -5.997530
## Merc 450SE         -10.700773 -3.722486  -6.887947
## Merc 450SL          -9.983601 -3.776020  -7.099360
## Merc 450SLC        -10.019861 -3.818385  -7.362840
## Cadillac Fleetwood -12.858363 -3.499300  -6.825711
## Lincoln Continental -13.278411 -3.479150  -6.641458
## Chrysler Imperial  -13.319584 -3.488892  -6.282780
## Fiat 128            -8.284478 -5.520964  -9.282719
## Honda Civic         -7.822859 -5.450106  -7.901608
## Toyota Corolla      -7.560726 -5.634763  -9.600407
## Toyota Corona       -7.528574 -4.484667  -8.707371
## Dodge Challenger    -9.540202 -3.798526  -7.040296
## AMC Javelin         -9.541157 -3.936838  -7.159934
## Camaro Z28         -11.027407 -3.448763  -4.754031
## Pontiac Firebird   -10.339390 -3.866141  -6.984926
## Fiat X1-9           -7.890807 -5.459205  -8.845583
## Porsche 914-2       -8.954943 -6.071743  -9.382764
## Lotus Europa        -8.043352 -5.694987  -7.544706
## Ford Pantera L     -12.022950 -5.148586  -5.169816
## Ferrari Dino       -10.211480 -4.957140  -6.658254
## Maserati Bora      -12.265974 -4.338317  -4.332345
## Volvo 142E          -9.651409 -5.184773  -8.100142
```

2D Comparison to PCA

Now we will do the same factor analysis on the `mtcars` dataset but with $k = 2$ and compare the (estimated) factors against the two-dimensional PCA reduction of the dataset.

```
# FA
model = fapa(R, numFactors=2, communality="SMC")

loadings = model$loadings
rownames(loadings) = colnames(mtcars)

specificVars = diag(cov(mtcars)) - model$h2
psi = diag(specificVars)

A = solve(t(loadings) %*% solve(psi) %*% loadings) %*% t(loadings) %*% solve(psi)
f = A %*% t(mtcars)
rownames(f) = c("f1","f2")
f = t(f)
f
```

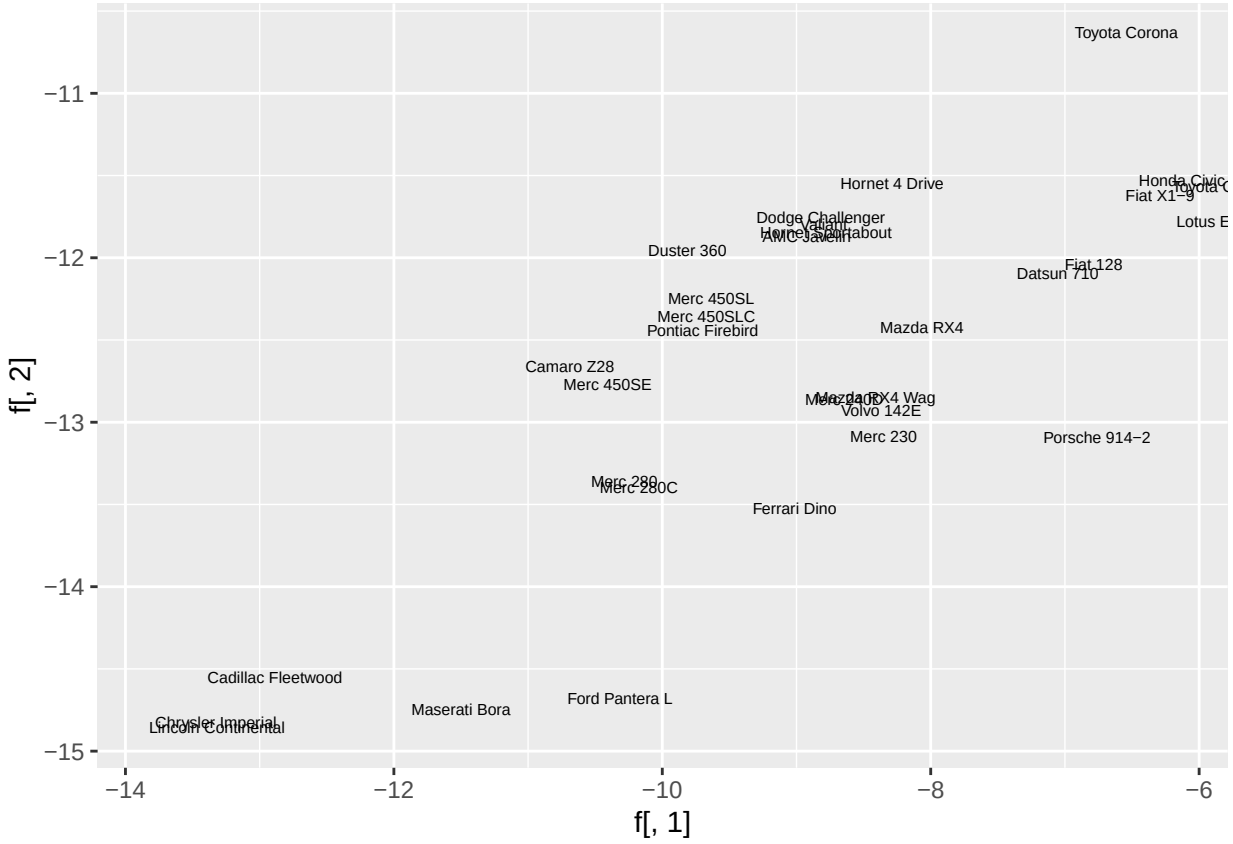
```
##                           f1        f2
## Mazda RX4           -8.375812 -12.45744
## Mazda RX4 Wag       -8.858872 -12.88496
## Datsun 710          -7.358374 -12.12992
## Hornet 4 Drive      -8.672619 -11.57957
## Hornet Sportabout   -9.270919 -11.88194
## Valiant             -8.976369 -11.83329
## Duster 360         -10.104149 -11.99056
## Merc 240D           -8.935054 -12.89574
## Merc 230            -8.599639 -13.12165
## Merc 280           -10.531030 -13.39286
## Merc 280C          -10.465925 -13.43005
## Merc 450SE         -10.735050 -12.80560
## Merc 450SL          -9.957564 -12.27992
## Merc 450SLC        -10.034837 -12.38830
## Cadillac Fleetwood -13.388220 -14.58568
## Lincoln Continental -13.824696 -14.88984
## Chrysler Imperial  -13.778152 -14.85822
## Fiat 128            -7.000553 -12.07395
## Honda Civic         -6.449146 -11.56403
## Toyota Corolla      -6.202031 -11.59870
## Toyota Corona       -6.925136 -10.66479
## Dodge Challenger    -9.298918 -11.78897
## AMC Javelin         -9.254356 -11.90379
## Camaro Z28         -11.019728 -12.69467
## Pontiac Firebird   -10.112165 -12.47479
## Fiat X1-9           -6.547319 -11.65379
## Porsche 914-2       -7.158894 -13.12551
## Lotus Europa        -6.169530 -11.81553
## Ford Pantera L     -10.706873 -14.71553
## Ferrari Dino        -9.326712 -13.55964
## Maserati Bora      -11.869689 -14.77936
## Volvo 142E          -8.666841 -12.96051
```

```
library(ggfortify)
```

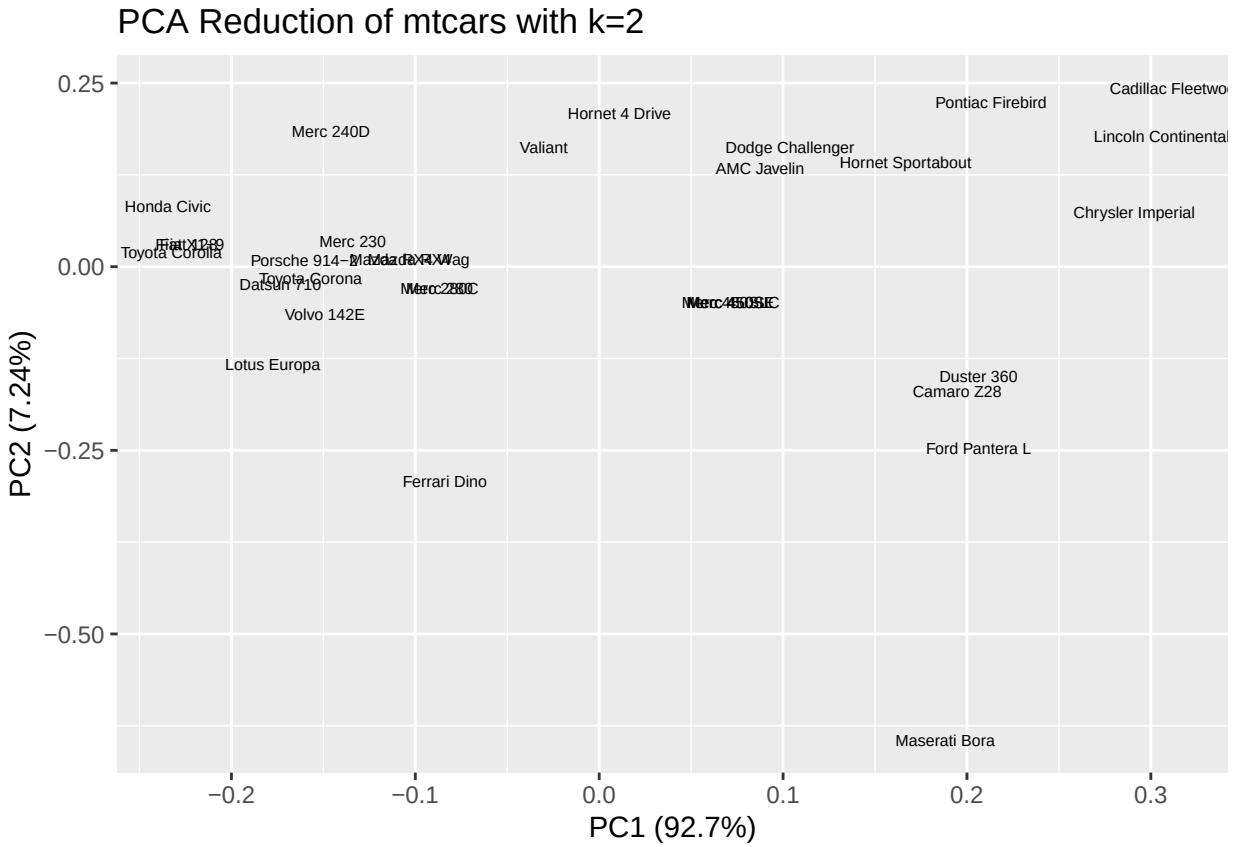
```
## Loading required package: ggplot2
```

```
ggplot(f,aes(x = f[,1], y = f[,2], label=rownames(f))) +
    geom_point(shape=NA) + geom_text(hjust=0, vjust=0, size=2)
```

```
## Warning: Removed 32 rows containing missing values (`geom_point()`).
```




```
# PCA
pca = prcomp(mtcars, center=TRUE)
autoplot(pca, data=mtcars, label=T, label.size=2, shape=F) +
    ggtitle("PCA Reduction of mtcars with k=2")
```



We do see some similarities between the two representations, for example notice how in both plots we have `Cadillac Fleetwood`, `Lincoln Continental` and `Chrysler Imperial` close together (bottom left in the FA plot and top right for PCA). Similarly, in the top right of the FA plot and top left for PCA we find a close group of `Honda Civic`, `Toyota Corolla` and `Fiat X1-9` (with `Fiat 128` nearby). However, the two reductions seem fairly comparable in quality overall, with the slight exception that the FA model has slightly fewer extreme overlaps of data reductions. It would probably be about equally as good to use either technique for this dataset, however, the lack of easily explainable (and distinct) factors in the factor model suggest that PCA might be the safer option.

Task 2

For this independent component analysis we will work with the `icamusical`[1] dataset and attempt to recover the 3 original audio files which have been combined to produce the $p = 3$ available audio files.

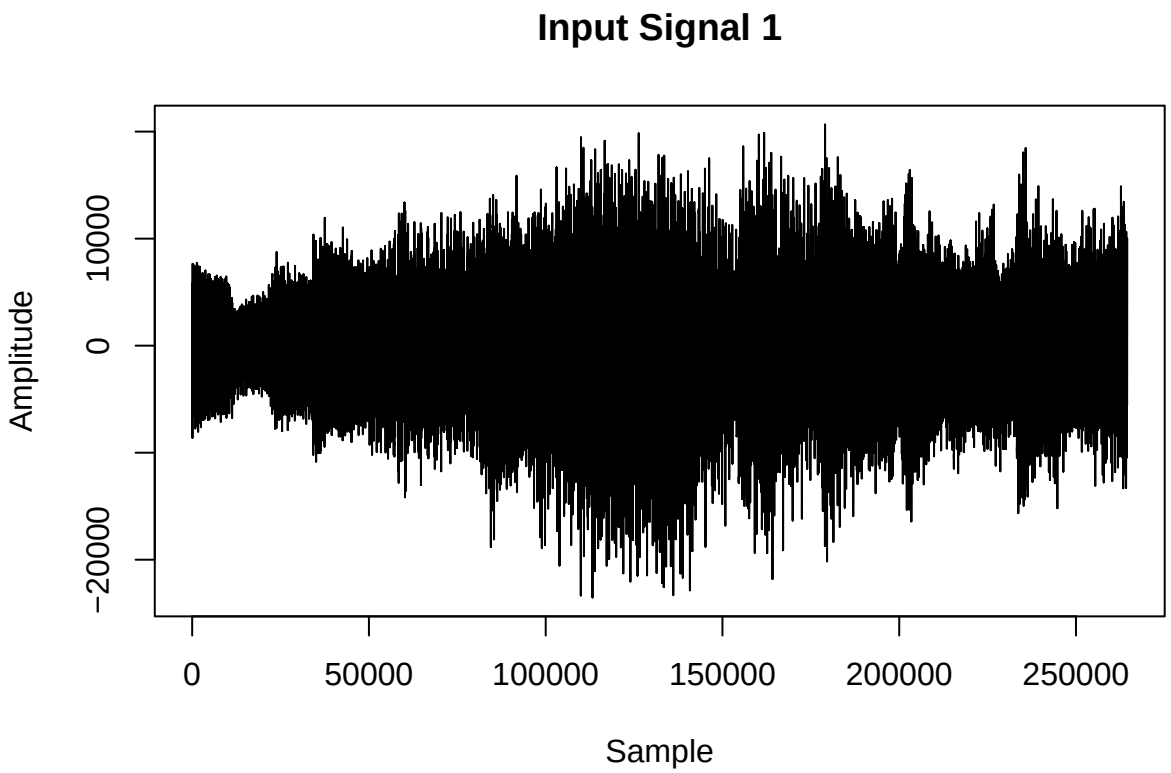
First we'll read the audio files into `R`.

```
library(tuneR)
f1 = readWave("data/ICA mix 1.wav")
X1 = f1@left

f2 = readWave("data/ICA mix 2.wav")
X2 = f2@left

f3 = readWave("data/ICA mix 3.wav")
X3 = f3@left

X0 = cbind(X1, X2, X3)
```

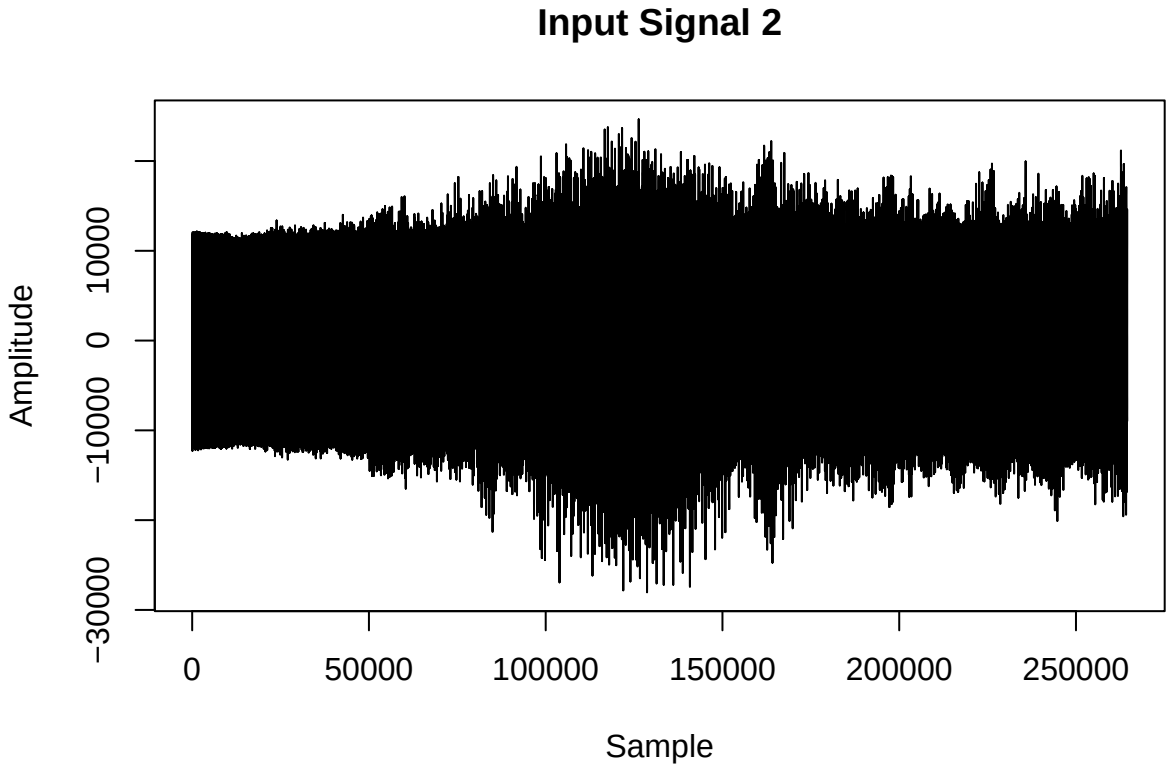
[1] https://www.kaggle.com/chittalpatel/icamusical/activity

We can visualise these input signals as seen below.

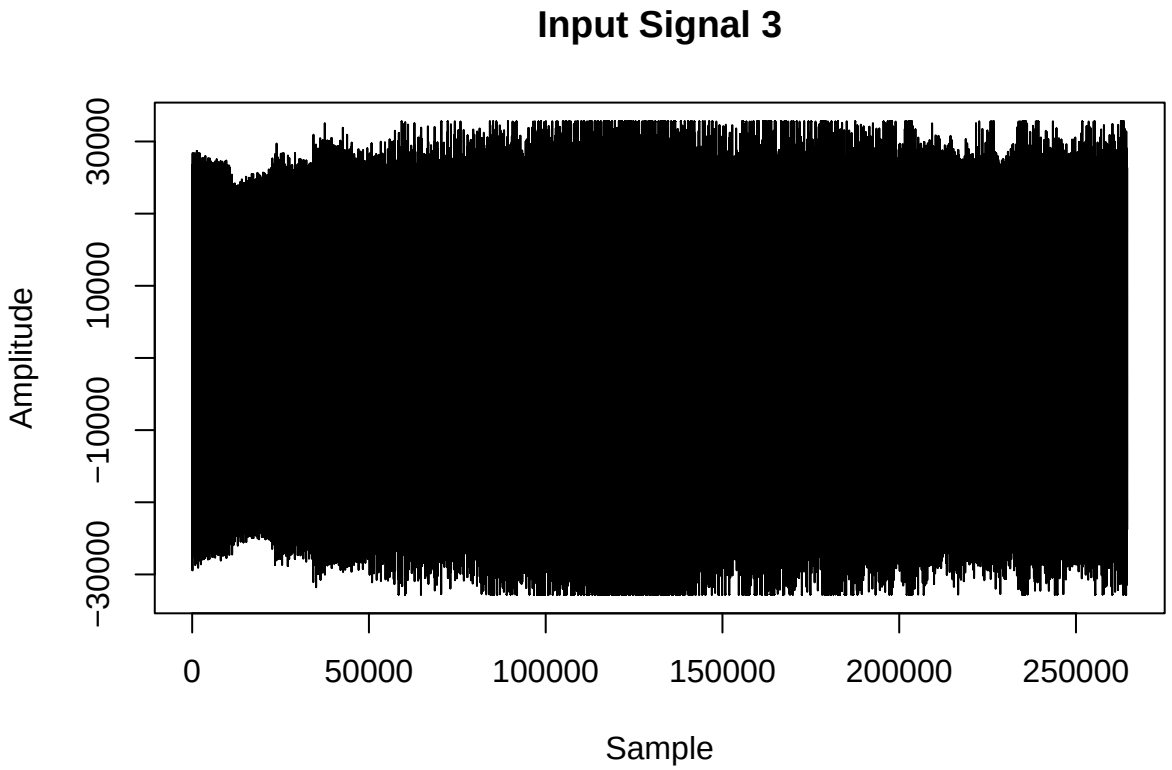
```
plot(1:length(X1), X1, type="l", xlab="Sample", ylab="Amplitude", main="Input Signal 1")
```




```
plot(1:length(X2), X2, type="l", xlab="Sample", ylab="Amplitude", main="Input Signal 2")
```




```
plot(1:length(X3), X3, type="l", xlab="Sample", ylab="Amplitude", main="Input Signal 3")
```

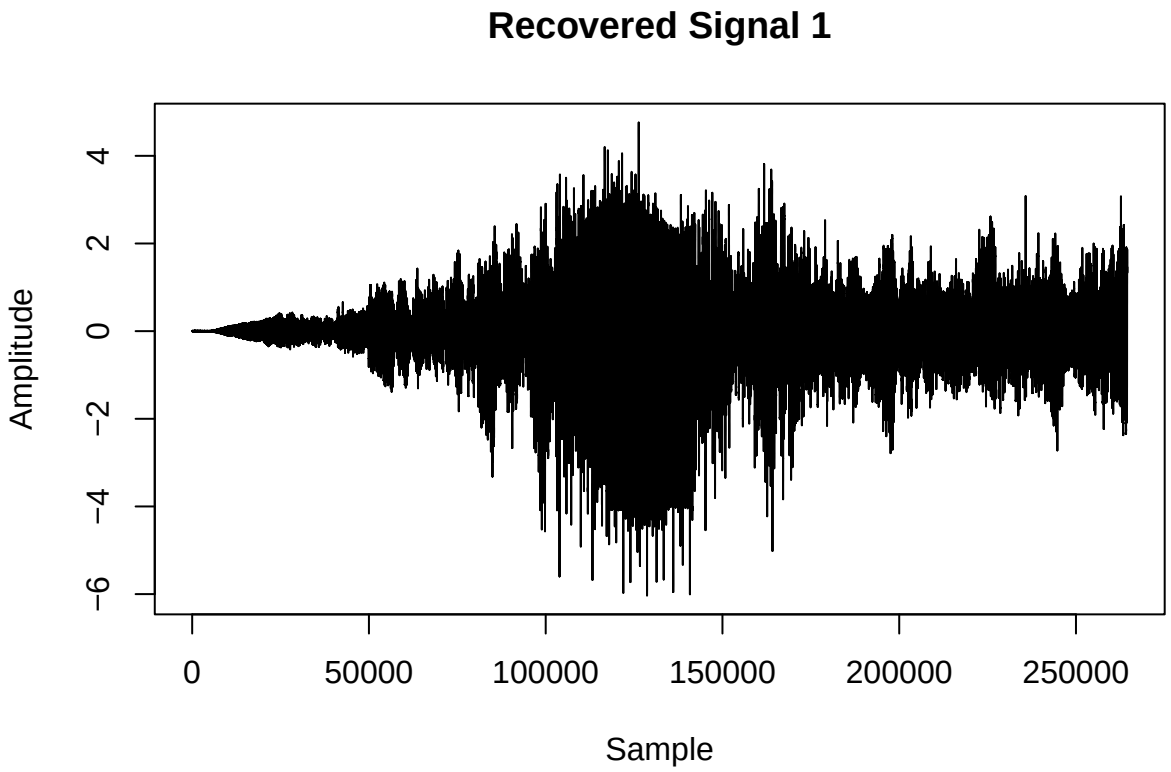


We can then use the `fastICA` package to perform independent component analysis to recover the original three signals. In doing this we specify that ICA should be performed using the function $\varphi(u) = \frac{1}{\alpha} \log \cosh(\alpha u)$ with $\alpha = 1$.

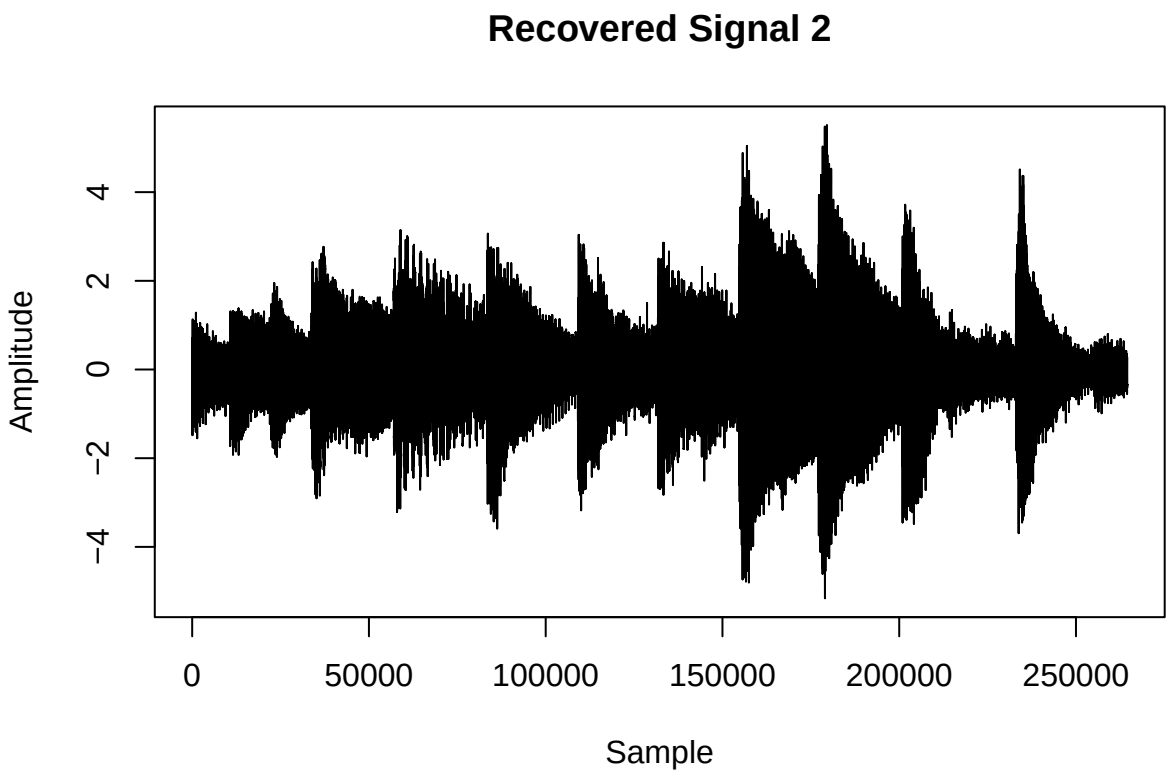
```
library(fastICA)
S = fastICA(X0, 3, fun="logcosh", alpha=1)$S
```

Visualising these recovered signals we can see that they look a lot more distinct from each other than the inputs, suggesting that the ICA has successfully separated the signals.

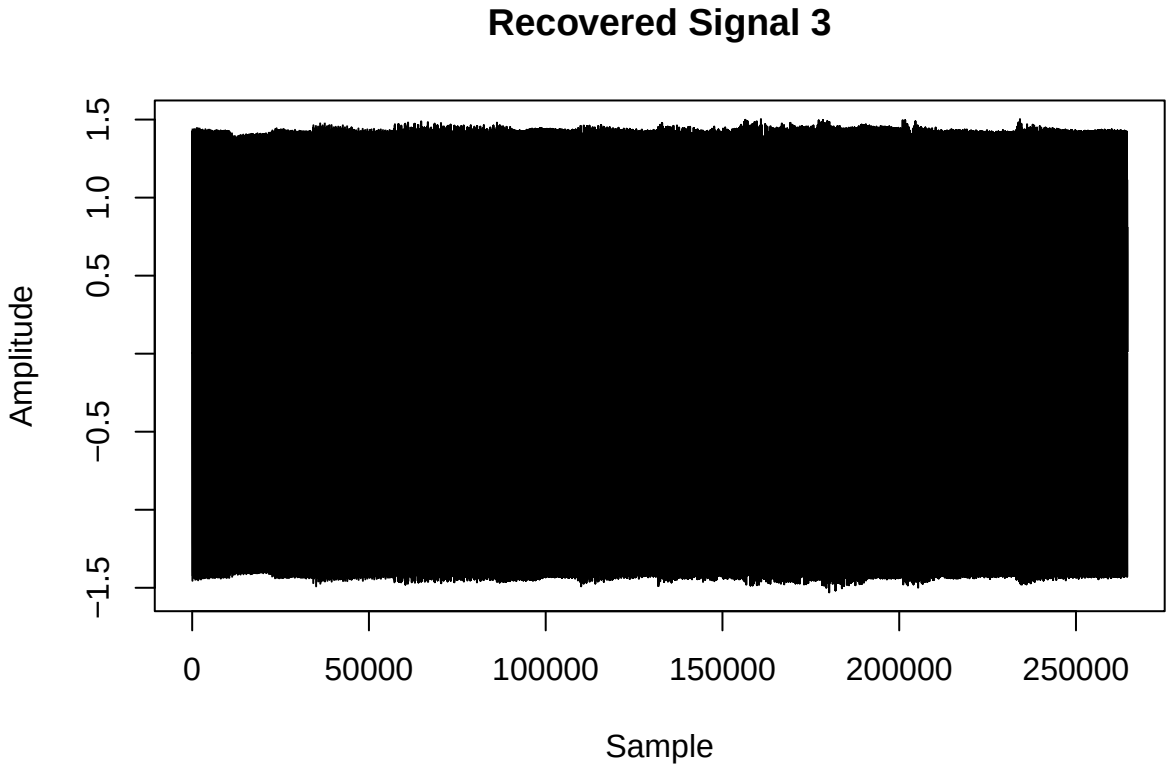
```
plot(1:length(S[,1]), S[,1], type="l", xlab="Sample", ylab="Amplitude",
     main="Recovered Signal 1")
```




```
plot(1:length(S[,2]), S[,2], type="l", xlab="Sample", ylab="Amplitude",
     main="Recovered Signal 2")
```



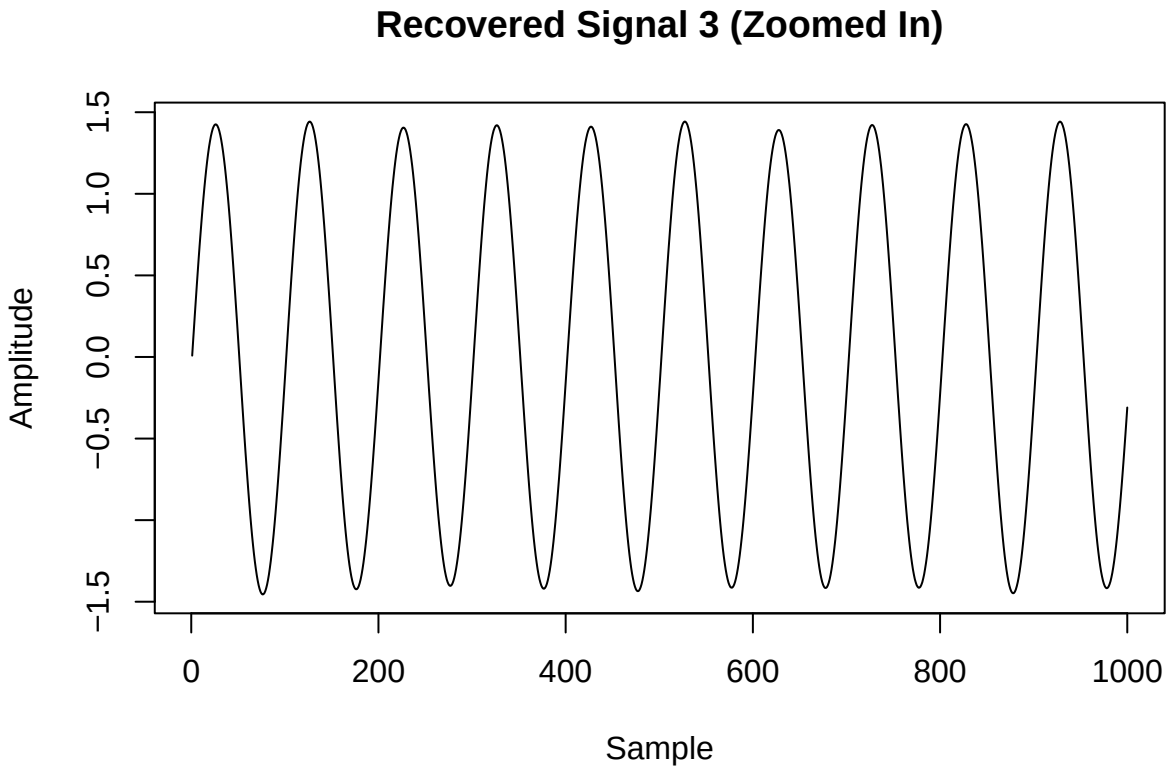

```
plot(1:length(S[,3]), S[,3], type="l", xlab="Sample", ylab="Amplitude",
     main="Recovered Signal 3")
```



The recovered signals are not perfect, however, as you can see by the non-uniformity in the third output signal which should actually be a simple sine wave of a constant frequency—this would appear to us as a

rectangular block due to the large number of samples. The (near-) constant frequency of this third signal can be seen more clearly if we zoom in.

```
plot(1:1000, S[1:1000,3], type="l", xlab="Sample", ylab="Amplitude",
     main="Recovered Signal 3 (Zoomed In)")
```



The imperfections observable when we zoom back out, which cause the jagged top and bottom edges of the 'rectangle', come from the clipping of amplitude when the three original signals were mixed together. This clipping can be heard if we write the recovered signals back as `.wav` files.

```
library(seewave)
for (i in 1:3){
  savewav(S[,i], f=f1@samp.rate, channel=1, filename = paste('signal', i, '.wav'))
}
```

Although these audio files do contain some undesired noise due to the loss of information caused by clipping, it is clear that the ICA has done an extremely good job overall of separating the signals—each of them are clearly identifiable and distinct.

References

[1] Giordano, Casey and Waller, Niels. R: Iterated Principal Axis Factor Analysis (fapa).