m21_pca

Joanna Morris

2025-06-07

This script computes the PCA for Morph21.

1. First we load the libraries we need

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
library(psych)
```

```
##
## Attaching package: 'psych'
##
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(datawizard)
```

```
##
## Attaching package: 'datawizard'
##
## The following object is masked from 'package:psych':
##
##     rescale
```

1. Set `ggplot2` parameters

Compute PCA

Following Andrews and Lo (2013) this script computes a PCA for our spelling and vocabulary measures. Because the standardised spelling and vocabulary scores were correlated, to facilitate interpretation, two orthogonal measures of individual differences were derived from a principal components analysis. Analysis based on this tutorial

First we import the data, remove missing values adn standardize the scores.

```r
df1 <- read_csv("demo_lang_vsl.csv",   # loads demographic and languag data
    col_types = cols(TestSite = col_factor(levels = c("Hampshire",
        "Providence")), `Included VSL2` = col_logical(),
        `Included LDT` = col_logical(), Date = col_date(format = "%m/%d/%Y"),
        Sex = col_factor(levels = c("Male",
            "Female", "Prefer not to say")),
        Ethnicity = col_factor(levels = c("Not Hispanic or Latino",
            "Hispanic or Latino")), Race = col_factor(levels = c("Black",
            "White", "Asian", "American Indian or Alaska Native",
            "More than one race")), read_for_pleasure = col_factor(levels = c("Not at all only for schoo
            "1-3 hours", "4-6 hours", "6+ hours"))))

describe(df1)
```

```
## Warning in FUN(newX[, i], ...): no non-missing arguments to min; returning Inf
## Warning in FUN(newX[, i], ...): no non-missing arguments to min; returning Inf
## Warning in FUN(newX[, i], ...): no non-missing arguments to min; returning Inf

## Warning in FUN(newX[, i], ...): no non-missing arguments to max; returning -Inf
## Warning in FUN(newX[, i], ...): no non-missing arguments to max; returning -Inf
## Warning in FUN(newX[, i], ...): no non-missing arguments to max; returning -Inf

##                      vars   n   mean    sd median trimmed   mad    min    max
## SubjID                  1 120 170.58 45.03 161.50  169.86 62.27 101.00 245.00
## TestSite*               2 120   1.38  0.49   1.00    1.34  0.00   1.00   2.00
## ExclReason*             3  20   5.35  2.35   6.00    5.56  1.48   1.00   8.00
## Included VSL2           4 120    NaN    NA     NA     NaN    NA    Inf   -Inf
## Included LDT            5  75    NaN    NA     NA     NaN    NA    Inf   -Inf
## Date                    6 120    NaN    NA     NA     NaN    NA    Inf   -Inf
## Sex*                    7 120   1.71  0.47   2.00    1.75  0.00   1.00   3.00
## Age                     8 104  20.24  3.80  19.00   19.58  1.48  18.00  48.00
## Ethnicity*              9 119   1.05  0.22   1.00    1.00  0.00   1.00   2.00
## Race*                  10 119   2.17  0.73   2.00    2.00  0.00   1.00   5.00
## handedness_score       11 119  42.55 42.45  11.00   40.11  2.97   5.00 100.00
## read_for_pleasure*     12 118   1.97  0.72   2.00    1.92  0.00   1.00   4.00
## born_in_us*            13 119   1.98  0.13   2.00    2.00  0.00   1.00   2.00
## first_language*        14 119   2.85  0.46   3.00    2.91  0.00   1.00   5.00
## language_disability*   15  73   1.03  0.16   1.00    1.00  0.00   1.00   2.00
## spl_cor                16 117  63.02  6.29  63.00   62.94  5.93  47.00  78.00
## spl_inc                17 117  16.98  6.29  17.00   17.06  5.93   2.00  33.00
## spl_perc               18 117  78.77  7.87  78.75   78.67  7.41  58.75  97.50
## vcb_cor                19 115  37.25  7.59  37.00   37.56  8.90  17.00  49.00
## vcb_inc                20 115  12.75  7.59  13.00   12.44  8.90   1.00  33.00
## vcb_perc               21 115  74.50 15.19  74.00   75.12 17.79  34.00  98.00
## art_cor                22 115  41.36  5.32  41.00   41.11  4.45  26.00  62.00
## art_inc                23 115  24.64  5.32  25.00   24.89  4.45   4.00  40.00
## art_diff               24 115  16.71 10.64  16.00   16.22  8.90 -14.00  58.00
## TOWRE_rank             25  30  56.20 20.41  58.00   55.29 22.98  23.00  97.00
## TOWRE_descriptor*      26  30   3.43  1.10   3.00    3.29  0.00   1.00   7.00
## hits                   27 108  18.58  5.01  18.00   18.23  4.45   9.00  32.00
## misses                 28 108  13.12  4.87  14.00   13.44  4.45   0.00  23.00
## correctRejections      29 108  18.58  5.01  18.00   18.23  4.45   9.00  32.00
## falseAlarms            30 108  13.12  4.87  14.00   13.44  4.45   0.00  23.00
## totalTrials            31 108  31.70  0.75  32.00   31.84  0.00  26.00  32.00
## hit_rate               32 108   0.59  0.15   0.56    0.57  0.14   0.28   1.00
## fa_rate                33 108   0.41  0.15   0.44    0.43  0.14   0.00   0.72
## hit_rate_z             34 108   0.01  1.00  -0.14   -0.06  0.93  -2.01   2.78
## fa_rate_z              35 108  -0.01  1.00   0.14    0.06  0.93  -2.78   2.01
## d_prime_raw            36 108   0.17  0.31   0.12    0.15  0.28  -0.44   1.00
## d_prime_zscore         37 108   0.02  2.00  -0.28   -0.11  1.85  -4.02   5.55
## sensitivity*           38 108   1.26  0.44   1.00    1.20  0.00   1.00   2.00
##                        range  skew kurtosis   se
## SubjID               144.00  0.18    -1.41 4.11
## TestSite*              1.00  0.51    -1.75 0.04
## ExclReason*            7.00 -0.68    -1.04 0.52
## Included VSL2          -Inf    NA       NA   NA
## Included LDT           -Inf    NA       NA   NA
## Date                   -Inf    NA       NA   NA
## Sex*                   2.00 -0.67    -0.95 0.04
## Age                   30.00  5.09    30.62 0.37
## Ethnicity*             1.00  4.06    14.59 0.02
## Race*                  4.00  3.00     9.13 0.07
## handedness_score      95.00  0.52    -1.71 3.89
## read_for_pleasure*     3.00  0.71     0.88 0.07
## born_in_us*            1.00 -7.42    53.55 0.01
## first_language*        4.00 -0.52     5.20 0.04
## language_disability*   1.00  5.67    30.59 0.02
## spl_cor               31.00  0.06    -0.36 0.58
## spl_inc               31.00 -0.06    -0.36 0.58
## spl_perc              38.75  0.06    -0.36 0.73
## vcb_cor               32.00 -0.34    -0.72 0.71
## vcb_inc               32.00  0.34    -0.72 0.71
## vcb_perc              64.00 -0.34    -0.72 1.42
## art_cor               36.00  0.57     1.60 0.50
## art_inc               36.00 -0.57     1.60 0.50
## art_diff              72.00  0.57     1.60 0.99
## TOWRE_rank            74.00  0.21    -0.94 3.73
## TOWRE_descriptor*      6.00  1.20     2.65 0.20
## hits                  23.00  0.65    -0.12 0.48
## misses                23.00 -0.64     0.01 0.47
## correctRejections     23.00  0.65    -0.12 0.48
## falseAlarms           23.00 -0.64     0.01 0.47
## totalTrials            6.00 -4.69    29.59 0.07
## hit_rate               0.72  0.66    -0.05 0.01
## fa_rate                0.72 -0.66    -0.05 0.01
## hit_rate_z             4.79  0.64    -0.04 0.10
## fa_rate_z              4.79 -0.64    -0.04 0.10
## d_prime_raw            1.44  0.66    -0.05 0.03
## d_prime_zscore         9.58  0.64    -0.04 0.19
## sensitivity*           1.00  1.08    -0.83 0.04
```

```r
df1_cln <- df1 |>
  filter(!(is.na(spl_cor) | is.na(vcb_cor) | is.na(art_cor )))

df1_cln_std <- mutate(df1_cln,
                       z_vcb = standardise(vcb_cor),
                       z_spl = standardise(spl_cor),
                       z_art = standardise(art_diff))
```

Now we can put the three standardized measures into a separate data frame and compute the correlations, using the `cor()` function. NB. A correlation coefficient is a standardized covariance statistic. We can run the `cov()` function on the standardized values or the `cor()` function on the unstandardized ones. Both methods will give the same results.

```r
art_vcb_spl_raw <- df1_cln_std |> select(SubjID,TestSite, vcb_cor, spl_cor, art_diff)
art_vcb_spl_z <- df1_cln_std |> select( SubjID,TestSite, z_vcb, z_spl, z_art)

cor(art_vcb_spl_raw[,3:5], use = "everything", method = "pearson")
```

```
##            vcb_cor   spl_cor  art_diff
## vcb_cor  1.0000000 0.4544972 0.6564318
## spl_cor  0.4544972 1.0000000 0.4387360
## art_diff 0.6564318 0.4387360 1.0000000
```

```r
cov(art_vcb_spl_z[,3:5], use = "everything", method = "pearson")
```

```
##           z_vcb     z_spl     z_art
## z_vcb 1.0000000 0.4544972 0.6564318
## z_spl 0.4544972 1.0000000 0.4387360
## z_art 0.6564318 0.4387360 1.0000000
```

Once we have generated the correlation coefficients we can test them for statistical significance. You can only test one correlation at a time using the `cor.test()` function, but the `corr.test()` function in the `psych` package will test a matrix of correlation coefficients.

```r
corr.test(art_vcb_spl_z[,3:5])
```

```
## Call:corr.test(x = art_vcb_spl_z[, 3:5])
## Correlation matrix
##       z_vcb z_spl z_art
## z_vcb  1.00  0.45  0.66
## z_spl  0.45  1.00  0.44
## z_art  0.66  0.44  1.00
## Sample Size
## [1] 113
## Probability values (Entries above the diagonal are adjusted for multiple tests.)
##       z_vcb z_spl z_art
## z_vcb     0     0     0
## z_spl     0     0     0
## z_art     0     0     0
##
##  To see confidence intervals of the correlations, print with the short=FALSE option
```

Now we can do the PCA. It turns out that by default, the function `PCA()` in `FactoMineR`, standardizes the data automatically, so we didn't actually need do the standardization.

Here are the arguments to the `PCA()` function:

- `X`: a data frame. Rows are individuals and columns are numeric variables
- `scale.unit`: a logical value. If TRUE, the data are scaled to unit variance before the analysis. This standardization to the same scale avoids some variables to become dominant just because of their large measurement units. It makes variables comparable.
- `ncp`: number of dimensions kept in the final results.
- `graph`: a logical value. If TRUE a graph is displayed.

The plot shows the relationships between all variables. It can be interpreted as follow:

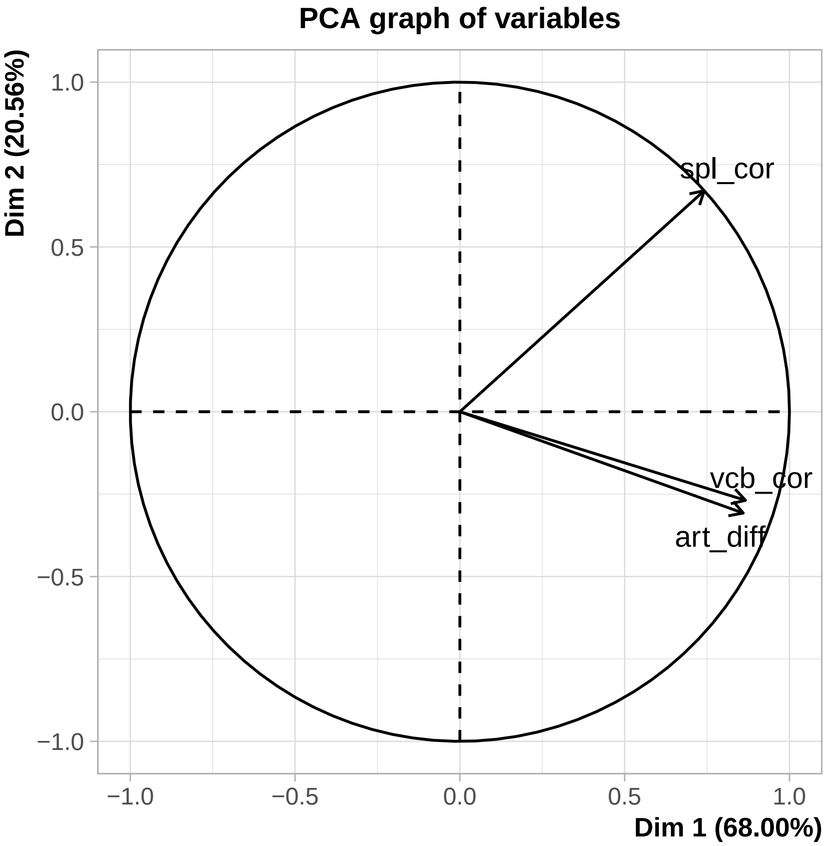
- Positively correlated variables are grouped together.

- Negatively correlated variables are positioned on opposite sides of the plot origin (opposed quadrants).
- The distance between variables and the origin measures the quality of the variables on the factor map. Variables that are away from the origin are well represented on the factor map.

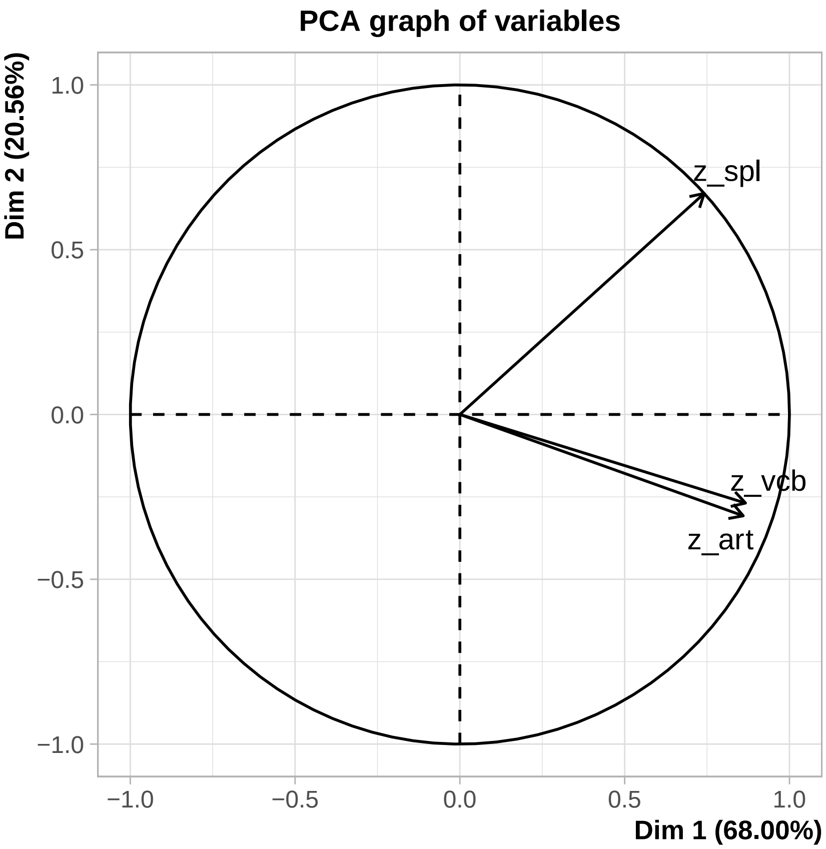
```r
library(FactoMineR)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
res.pca <- PCA(art_vcb_spl_raw[,3:5], scale.unit = TRUE, ncp = 2, graph = FALSE)
plot(res.pca, choix = "varcor", graph.type = c("ggplot"))
```



```r
res.pca <- PCA(art_vcb_spl_z[,3:5], scale.unit = TRUE, ncp = 2, graph = FALSE)
plot(res.pca, choix = "varcor", graph.type = c("ggplot"))
```



The eigenvalues measure the amount of variation retained by each principal component. Eigenvalues are large for the first PCs and small for the subsequent PCs. That is, the first PCs corresponds to the directions with the maximum amount of variation in the data set.

We examine the eigenvalues to determine the number of principal components to be considered. The sum of all the eigenvalues give a total variance of 3, the number of variables. An eigenvalue > 1 indicates that PCs account for more variance than accounted by one of the original variables in standardized data. This is commonly used as a cutoff point for which PCs are retained. This holds true only when the data are standardized.

```
(eig.val <- get_eigenvalue(res.pca))
```

```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1  2.0400351         68.00117                    68.00117
## Dim.2  0.6167486         20.55829                    88.55946
## Dim.3  0.3432163         11.44054                   100.00000
```

The quality of representation of the variables on factor map is called cos2 (square cosine, squared coordinates). *A high cos2 indicates a good representation of the variable on the principal component.* In this case the variable is positioned close to the circumference of the correlation circle. *A low cos2 indicates that the variable is not perfectly represented by the PCs.* In this case the variable is close to the center of the circle. If a variable is perfectly represented by only two principal components (Dim.1 & Dim.2), the sum of the cos2 on these two PCs is equal to one. In this case the variables will be positioned on the circle of correlations.

```
res.pca$var$cos2
```

```
##           Dim.1      Dim.2
## z_vcb 0.7511062 0.07225111
## z_spl 0.5497149 0.44994337
## z_art 0.7392139 0.09455412
```

The contributions of variables in accounting for the variability in a given principal component are expressed in percentages. Variables that are correlated with PC1 (i.e., Dim.1) and PC2 (i.e., Dim.2) are the most important in explaining the variability in the data set. The larger the value of the contribution, the more the variable contributes to the component. It's possible to use the function corrplot() [corrplot package] to highlight the most contributing variables for each dimension.

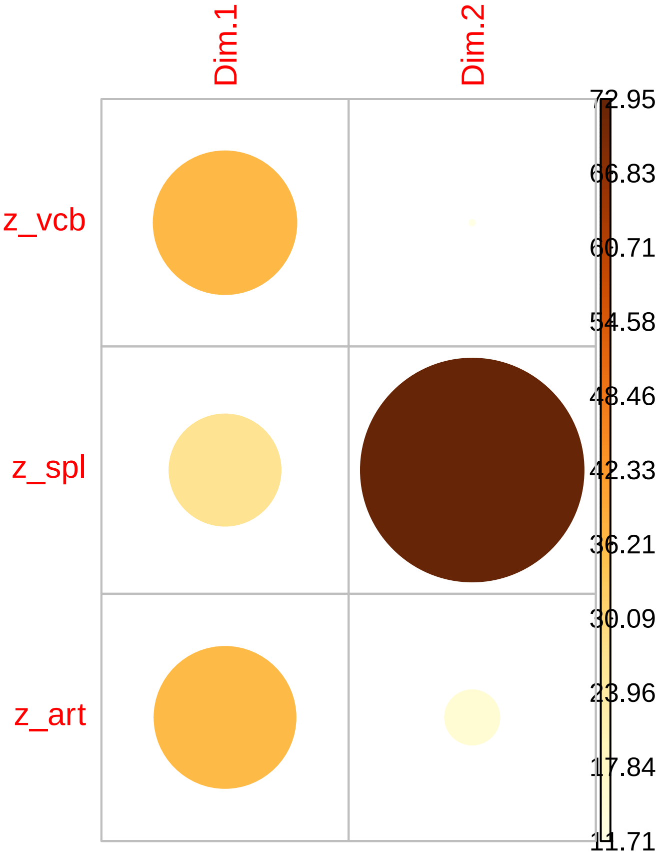
```
library('corrplot')
```

```
## corrplot 0.95 loaded
```

```
res.pca$var$contrib
```

```
##          Dim.1    Dim.2
## z_vcb 36.81830 11.71484
## z_spl 26.94635 72.95410
## z_art 36.23535 15.33106
```

```
corrplot(res.pca$var$contrib, is.corr=FALSE)
```



The correlation between a variable and a principal component (PC) is used as the coordinates of the variable on the PC.

```
(res.pca$var$coord)
```

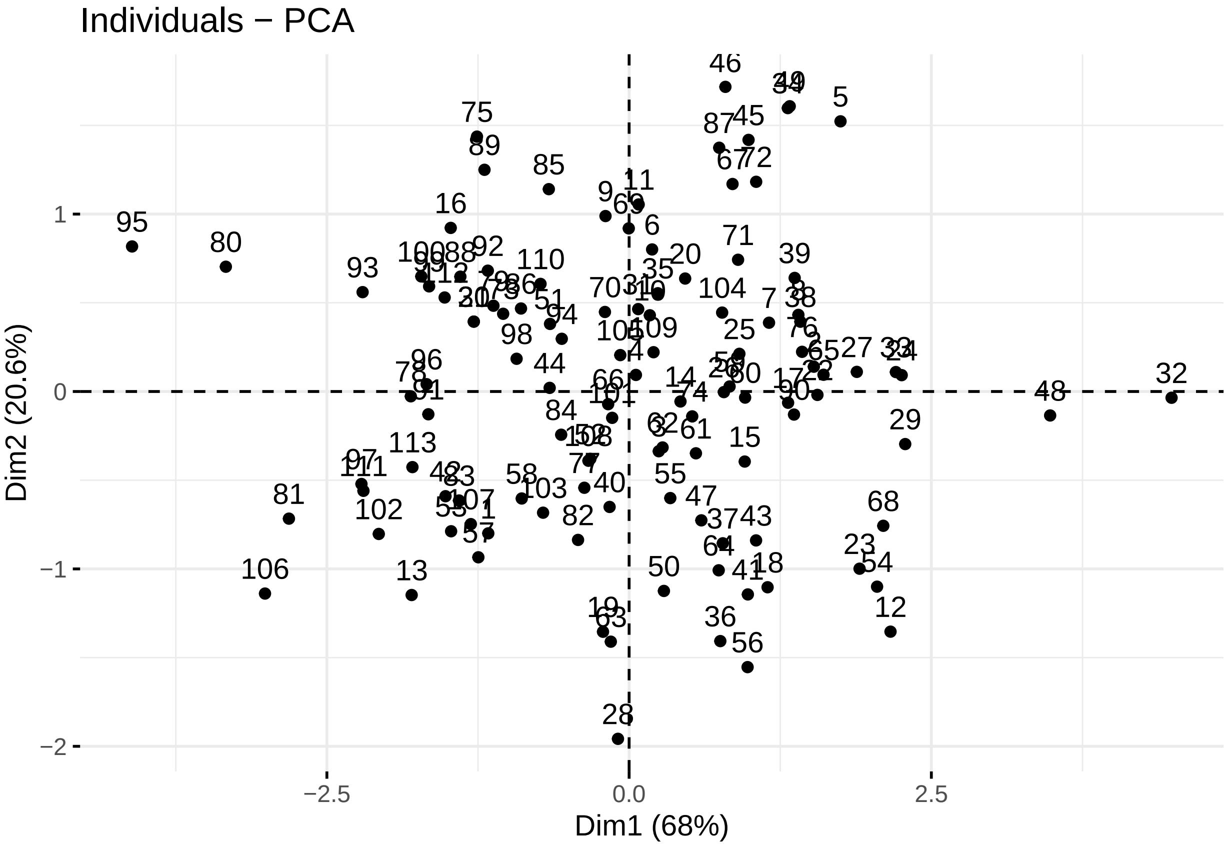
```
##           Dim.1      Dim.2
## z_vcb 0.8666638 -0.2687957
## z_spl 0.7414276  0.6707782
## z_art 0.8597755 -0.3074965
```

```
(res.desc <- dimdesc(res.pca, axes = c(1,2), proba = 0.05))
```

```
## $Dim.1
##
## Link between the variable and the continuous variables (R-square)
## =================================================================================
##       correlation      p.value
## z_vcb   0.8666638 2.617968e-35
## z_art   0.8597755 3.518474e-34
## z_spl   0.7414276 5.938765e-21
##
## $Dim.2
##
## Link between the variable and the continuous variables (R-square)
## =================================================================================
##       correlation      p.value
## z_spl   0.6707782 4.362670e-16
## z_vcb  -0.2687957 3.992603e-03
## z_art  -0.3074965 9.228738e-04
```

The fviz_pca_ind() is used to produce the graph of individuals.

```
ind <- get_pca_ind(res.pca)
fviz_pca_ind(res.pca)
```



```
df1_cln_std <- bind_cols(df1_cln_std,res.pca$ind$coord)
```

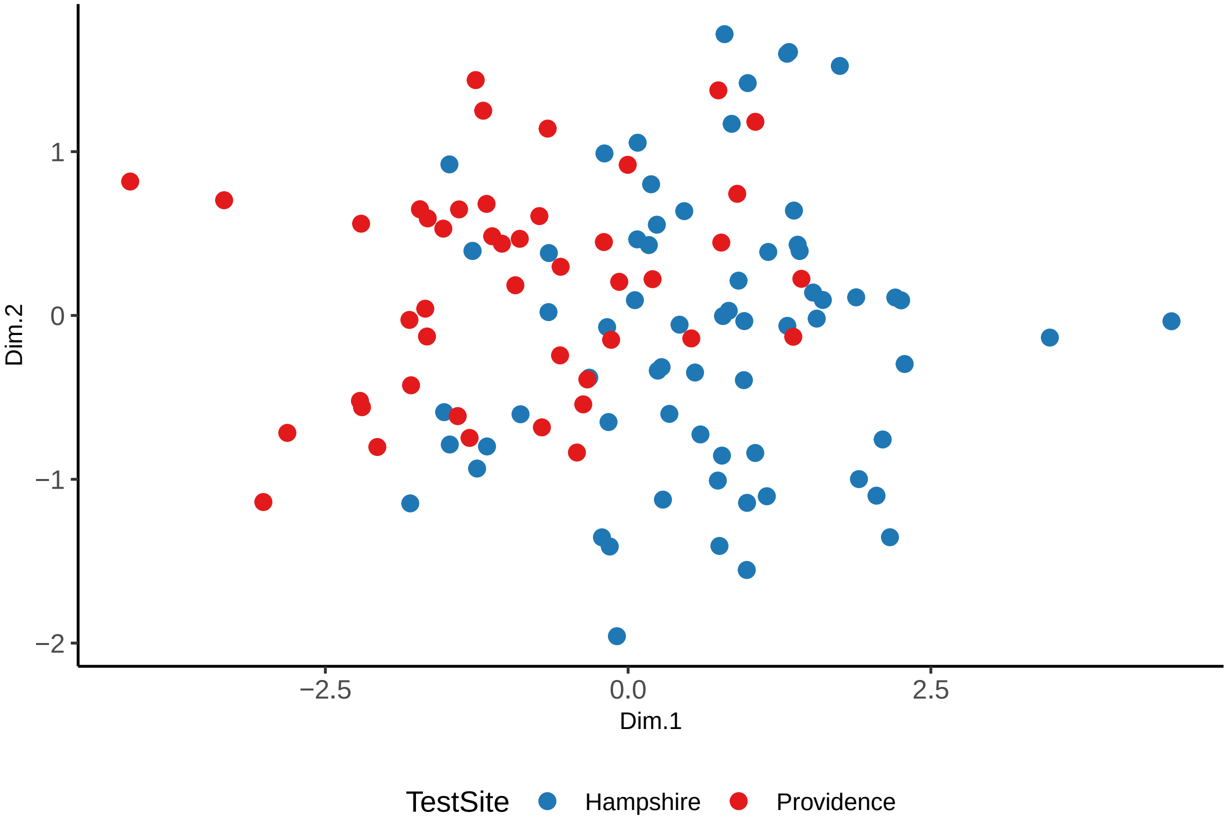
Divide participants based on median split of Dim2. Higher values on this factor indicate that spelling scores were relatively higher than vocabulary

```
df1_cln_std <- df1_cln_std |>
  mutate(lang_type_ortho = case_when(
    Dim.2 <= 0 ~ "Low Orthographic",
    Dim.2 > 0 ~ "High Orthographic"
  ))
df1_cln_std <- df1_cln_std |>
  mutate(lang_type_semantic = case_when(
    Dim.1 <= 0 ~ "Low Semantic",
    Dim.1 > 0 ~ "High Semantic"
  ))
```

We can then write the indivdiual pca values to a file

```
write_csv(df1_cln_std, "demo_lang_vsl_pca.csv")
```

```
ggplot(data = df1_cln_std,
       aes(x = Dim.1, y = Dim.2,
           colour = TestSite,
           fill = TestSite)) +
  geom_point(size = 2.5) +
  scale_color_custom() +
  scale_fill_custom()
```


Dim.2
Dim.1
TestSite
● Hampshire
● Providence