PCA for big data

R for Data Science

Juan R Gonzalez
juanr.gonzalez@isglobal.org

BRGE - Bioinformatics Research Group in Epidemiology
ISGlobal - Barcelona Institute for Global Health
http://brge.isglobal.org

PCA vs SVD

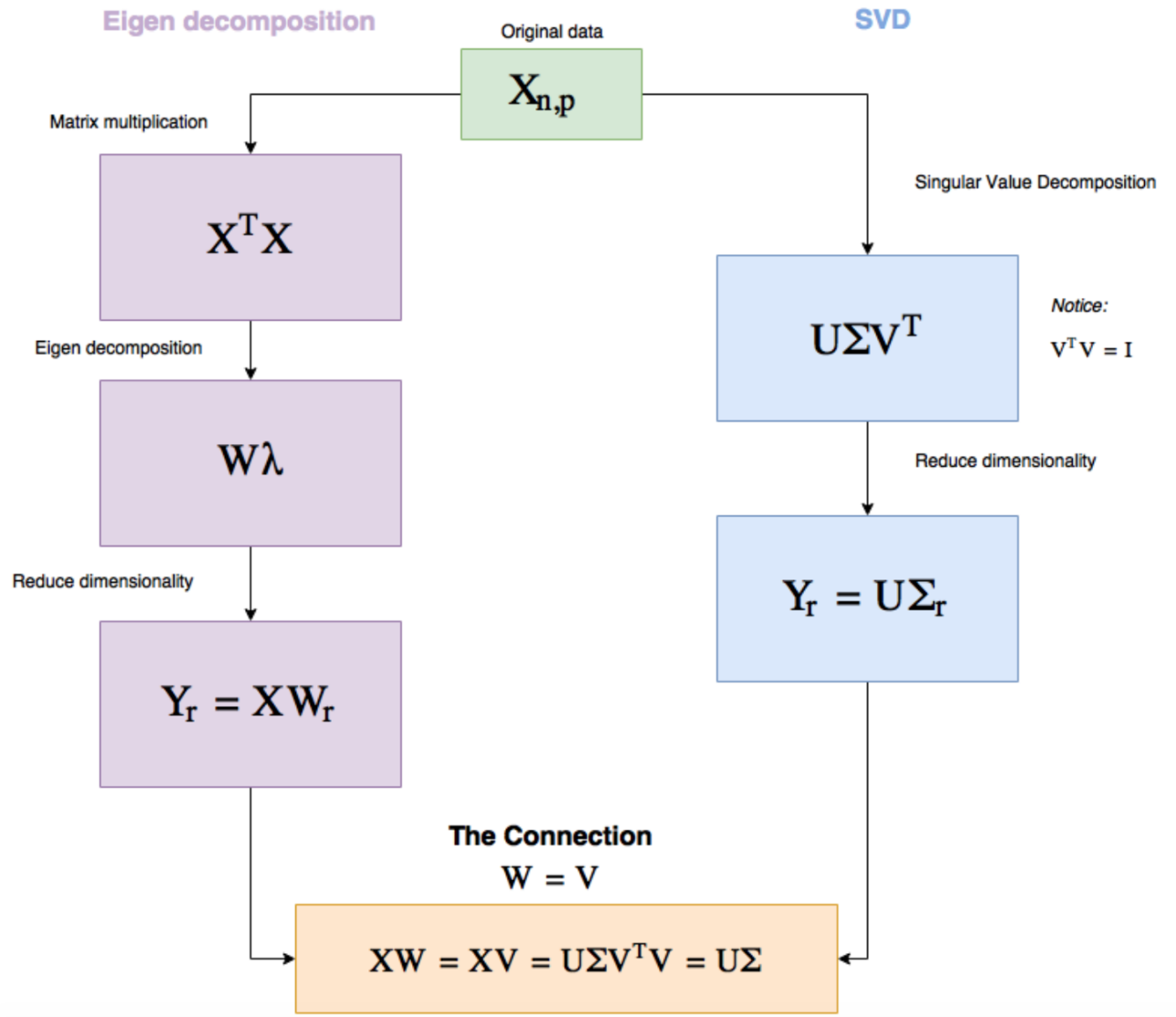


Figure 1: PCA vs SVD

Example

`musk` dataset describes a set of 102 molecules (repeated measures, in total there are 476 observations) of which 39 are judged by human experts to be musks and the remaining 63 molecules are judged to be non-musks. The goal is to learn to predict whether new molecules will be musks or non-musks. In this task we only aim to see whether the two first principal components discriminate musks and non-musks molecules. Data columns represent:

- f_1 ... f_162: distance features measured in hundredths of Angstroms.
- f163: distance of the oxygen atom in the molecule to a designated point in 3-space. This is also called OXY-DIS.
- f164: OXY-X: X-displacement from the designated point.
- f165: OXY-Y: Y-displacement from the designated point.
- f166: OXY-Z: Z-displacement from the designated point.
- musk: 0:non-musk, 1:musk

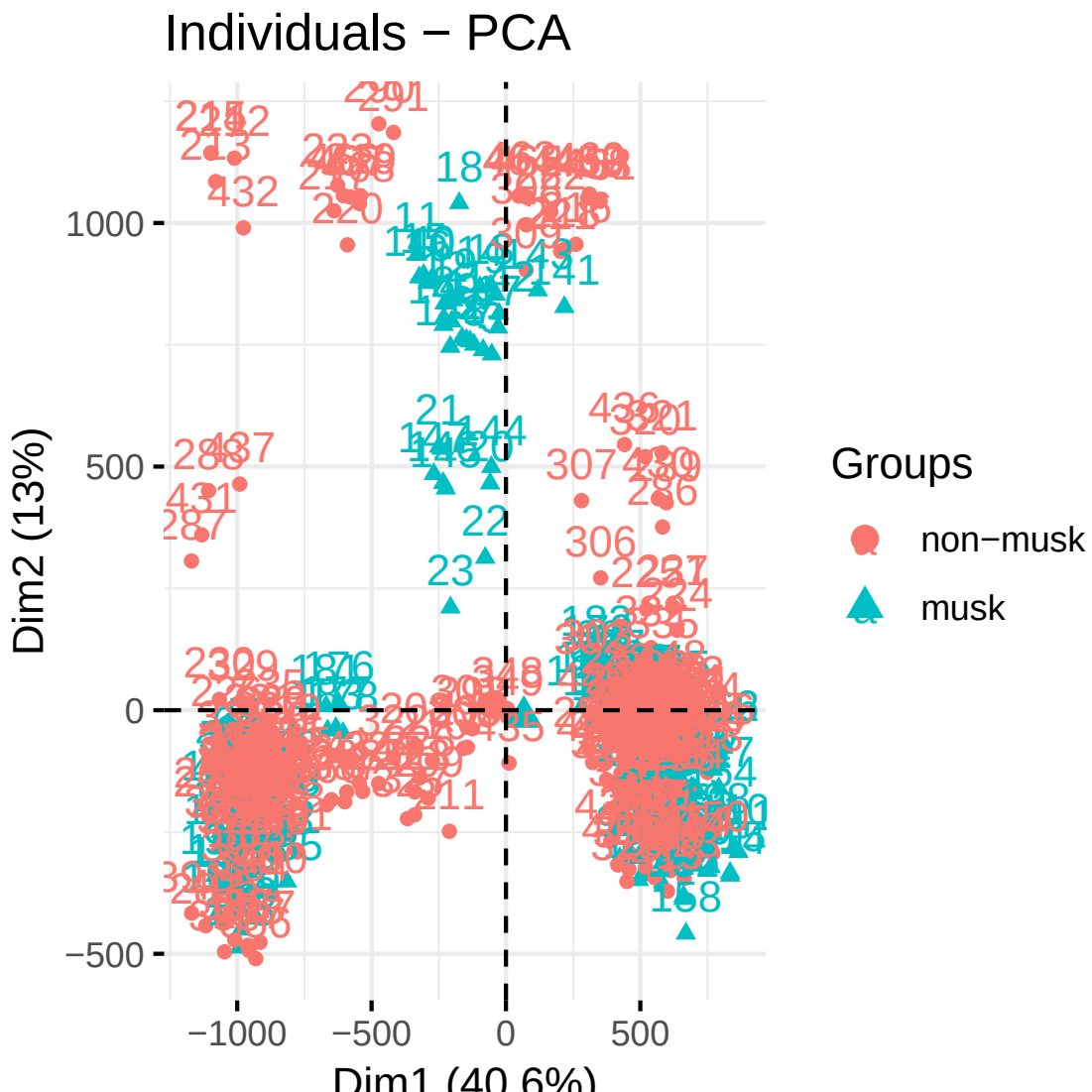
Example

Analysis with R

```
dd <- read.delim("c:/Juan/CREAL/GitHub/TeachingMaterials/Master_
names(dd)
```

```
  [1] "f_1"   "f_2"   "f_3"   "f_4"   "f_5"   "f_6"   "f_7"   "f_8"   "f_9"
 [10] "f_10"  "f_11"  "f_12"  "f_13"  "f_14"  "f_15"  "f_16"  "f_17"  "f_18"
 [19] "f_19"  "f_20"  "f_21"  "f_22"  "f_23"  "f_24"  "f_25"  "f_26"  "f_27"
 [28] "f_28"  "f_29"  "f_30"  "f_31"  "f_32"  "f_33"  "f_34"  "f_35"  "f_36"
 [37] "f_37"  "f_38"  "f_39"  "f_40"  "f_41"  "f_42"  "f_43"  "f_44"  "f_45"
 [46] "f_46"  "f_47"  "f_48"  "f_49"  "f_50"  "f_51"  "f_52"  "f_53"  "f_54"
 [55] "f_55"  "f_56"  "f_57"  "f_58"  "f_59"  "f_60"  "f_61"  "f_62"  "f_63"
 [64] "f_64"  "f_65"  "f_66"  "f_67"  "f_68"  "f_69"  "f_70"  "f_71"  "f_72"
 [73] "f_73"  "f_74"  "f_75"  "f_76"  "f_77"  "f_78"  "f_79"  "f_80"  "f_81"
 [82] "f_82"  "f_83"  "f_84"  "f_85"  "f_86"  "f_87"  "f_88"  "f_89"  "f_90"
 [91] "f_91"  "f_92"  "f_93"  "f_94"  "f_95"  "f_96"  "f_97"  "f_98"  "f_99"
[100] "f_100" "f_101" "f_102" "f_103" "f_104" "f_105" "f_106" "f_107" "f_108"
[109] "f_109" "f_110" "f_111" "f_112" "f_113" "f_114" "f_115" "f_116" "f_117"
[118] "f_118" "f_119" "f_120" "f_121" "f_122" "f_123" "f_124" "f_125" "f_126"
[127] "f_127" "f_128" "f_129" "f_130" "f_131" "f_132" "f_133" "f_134" "f_135"
[136] "f_136" "f_137" "f_138" "f_139" "f_140" "f_141" "f_142" "f_143" "f_144"
[145] "f_145" "f_146" "f_147" "f_148" "f_149" "f_150" "f_151" "f_152" "f_153"
[154] "f_154" "f_155" "f_156" "f_157" "f_158" "f_159" "f_160" "f_161" "f_162"
[163] "f_163" "f_164" "f_165" "f_166" "musk"
```

```r
library(factoextra)
o <- which(colnames(dd)=="musk")
group <- factor(dd[,o], labels = c("non-musk", "musk"))
pp <- prcomp(dd[ , -o])
fviz_pca_ind(pp, habillage=group)
```

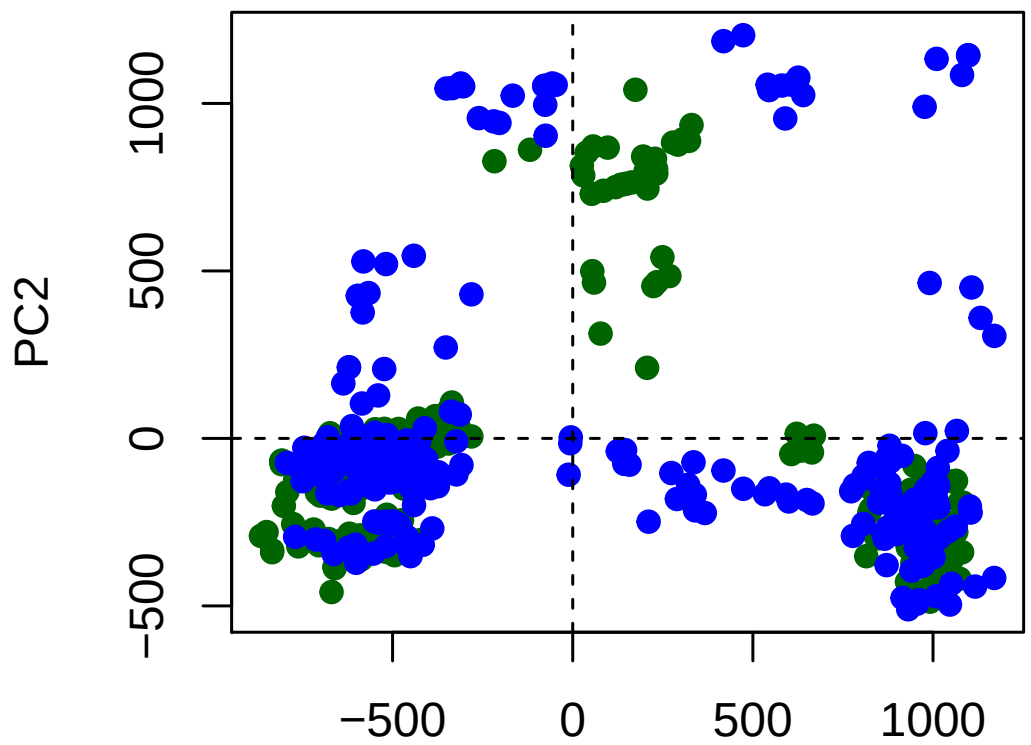


Truncated PCA and SVD (irlba)

It is a fast and memory-efficient way to compute a partial SVD, principal components, and some specialized partial eigenvalue decompositions (J. Baglama and L. Reichel. SIAM J. Sci. Comput. ,2005) implemented in an R package:

- `irlba()` partial SVD function
- `ssvd()` l1-penalized matrix decompoisition for sparse PCA (based on Shen and Huang's algorithm)–see https://bwlewis.github.io/irlba/ssvd.html for more details
- `prcomp_irlba()` PCA function similar to the `prcomp` function in stats package for computing the first few principal components of large matrices
- `svdr()` randomized SVD (alternative to truncation)
- `partial_eigen()` a very limited partial eigenvalue decomposition for symmetric matrices (see the **RSpectra** R package for more comprehensive truncated eigenvalue decomposition); see also https://bwlewis.github.io/irlba/comparison.html for more notes on RSpectra.

```r
library(irlba)
pp2 <- prcomp_irlba(dd[ , -o], n = 2)
ind.coord <- pp2$x
mycol <- ifelse(group=="musk", "darkgreen", "blue")
plot(ind.coord[,1], ind.coord[,2], pch = 19, col=mycol,
     xlab="PC1", ylab="PC2")
abline(h=0, v=0, lty = 2)
```



```
library(microbenchmark)
microbenchmark(irlba = prcomp_irlba(dd[ , -o], n = 2),
               prcomp = prcomp(dd[ , -o]))
```

```
Unit: milliseconds
   expr       min        lq      mean    median       uq      max neval cld
  irlba  5.206919  5.709438  7.142087  6.501933  8.08218 17.77345   100  a
 prcomp 50.766659 56.573082 62.807889 62.104149 67.25042 89.37054   100   b
```

Other resources

- C++ Library For Large Scale Eigenvalue Problems (Spectra): https://spectralib.org/index.html
- RSpectra: https://cran.r-project.org/web/packages/RSpectra/vignettes/introduction.html
- Benchmarck: https://spectralib.org/performance.html
- irlba vs RSpectra: https://bwlewis.github.io/irlba/comparison.html (a non-biased comparison: https://rpubs.com/koheiw/330986)

Session info

```
sessionInfo()
```

```
R version 3.5.0 (2018-04-23)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 17134)

Matrix products: default

locale:
[1] LC_COLLATE=Spanish_Spain.1252  LC_CTYPE=Spanish_Spain.1252
[3] LC_MONETARY=Spanish_Spain.1252 LC_NUMERIC=C
[5] LC_TIME=Spanish_Spain.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] factoextra_1.0.5 ggplot2_3.0.0

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.19     ggpubr_0.1.6     bindr_0.1.1      knitr_1.21
 [5] magrittr_1.5     tidyselect_0.2.4 munsell_0.5.0    colorspace_1.3-2
 [9] R6_2.2.2         rlang_0.2.2      stringr_1.3.1    plyr_1.8.4
[13] dplyr_0.7.7      tools_3.5.0      grid_3.5.0       gtable_0.2.0
[17] xfun_0.4         withr_2.1.2      htmltools_0.3.6  yaml_2.2.0
[21] lazyeval_0.2.1   digest_0.6.15    assertthat_0.2.0 tibble_1.4.2
[25] bindrcpp_0.2.2   purrr_0.2.4      codetools_0.2-15 ggrepel_0.8.0
[29] glue_1.2.0       evaluate_0.12    rmarkdown_1.11   labeling_0.3
[33] stringi_1.2.2    compiler_3.5.0   pillar_1.2.2     scales_1.0.0
[37] pkgconfig_2.0.1
```