# Data Science for Actuaries (ACT6100)

Arthur Charpentier

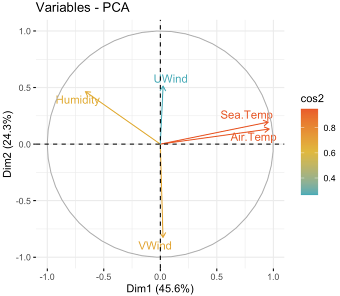
Non-Supervisé # 5 (ACP & imputation)

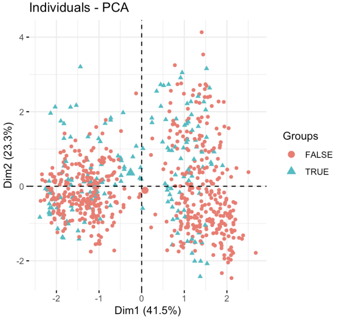
automne 2Q20

https://github.com/freakonometrics/ACT6100/

# Missing Values & PCA

```
> library(missMDA)
> library(VIM)
> names(tao)[4]="Sea.Temp"
> res.pca = PCA(tao[,4:8],graph=
    FALSE)
Warning message:
In PCA(tao[, 4:8], graph = FALSE)
  Missing values are imputed by
    the mean of the variable: you
    should use the imputePCA
    function of the missMDA
    package
> fviz_pca_var(res.pca, col.var =
    "cos2")
> miss.ind = apply(tao[,4:8],1,
    function(x) sum(is.na(x))>0)
> fviz_pca_ind(res.pca, label="
    none", habillage=miss.ind)
```



# Missing Values & PCA

The goal of PCA is maximize dispersion (inertia) of projections, or equivalently to minimize distance between observations, and their projections : we approximate our dataset $\boldsymbol{X}$ $(n \times p)$ with some lower rank matrix,

$$\min_{\boldsymbol{Z}\in\mathbb{R}^{n\times k}} \left\{\|\boldsymbol{X}-\boldsymbol{Z}\|^2\right\}$$
$$\text{subject to } \text{rank}(\boldsymbol{Z}) \leq s$$

for some $s < p$, where $\|\boldsymbol{M}\| = \text{trace}(\boldsymbol{M}\boldsymbol{M}^\top)$
From singular value decomposition,

$$\boldsymbol{Z}_s^\star = \boldsymbol{U}_{n\times s}\boldsymbol{\Delta}_{s\times s}\boldsymbol{V}_{p\times s}^\top = \underbrace{\boldsymbol{F}_{n\times s}}_{=\text{PC scores}} \underbrace{\boldsymbol{V}_{p\times s}^\top}_{\text{loadings}}$$

That was possible only with complete information (no missing data)

# Missing Values & PCA

$$\min_{\boldsymbol{Z}\in\mathbb{R}^{n\times k}} \left\{\|\boldsymbol{X}-\boldsymbol{Z}\|^2\right\}$$
$$\text{subject to } \text{rank}(\boldsymbol{Z}) \leq s$$

becomes, with missing data, $\boldsymbol{W}=(\boldsymbol{W}_{ij})$, $\boldsymbol{W}_{ij}=\mathbf{1}(\boldsymbol{X}_{ij}\text{ missing})$,

$$\min_{\boldsymbol{Z}\in\mathbb{R}^{n\times k}} \left\{\|\boldsymbol{W}\odot(\boldsymbol{X}-\boldsymbol{Z})\|^2\right\}$$
$$\text{subject to } \text{rank}(\boldsymbol{Z}) \leq s$$

This can be solved by iterative PCA, see Kiers (1997)

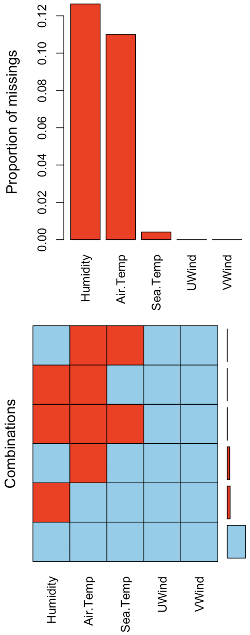
**Algorithm 1:** Iterative PCA

initialization : $\boldsymbol{X}^{(0)}$ completed by mean imputation, $s<p$;
**for** $t$=1,2,... **do**
  PCA on completed data, or SVD $\boldsymbol{U}^{(t)}_{n\times n},\boldsymbol{\Delta}^{(t)}_{n\times p},\boldsymbol{V}^{(t)\top}_{p\times p}$);
  impute values with $Y^{(t)}=\boldsymbol{U}_{n\times s},\boldsymbol{\Delta}_{s\times s},\boldsymbol{V}^{\top}_{p\times s}$);
  $\boldsymbol{X}^{(t)}\leftarrow\boldsymbol{W}\odot\boldsymbol{X}+(1-\boldsymbol{W})\odot Y^{(t)}$

# Missing Values & PCA

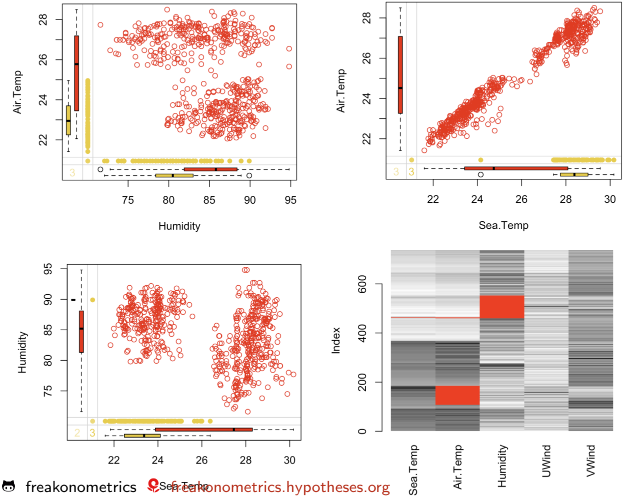
```
> res = summary(aggr(tao[,4:8],
    sortVar = TRUE),col=colrvim)
    $combinations

 Variables sorted by number of
    missings:
 Variable       Count
 Humidity 0.126358696
 Air.Temp 0.110054348
 Sea.Temp 0.004076087
    UWind 0.000000000
    VWind 0.000000000
```



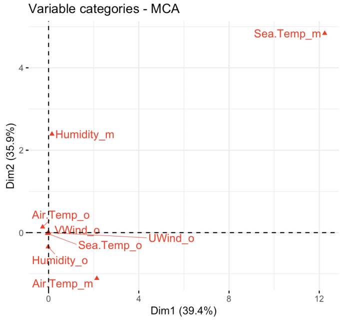
# Missing Values & PCA

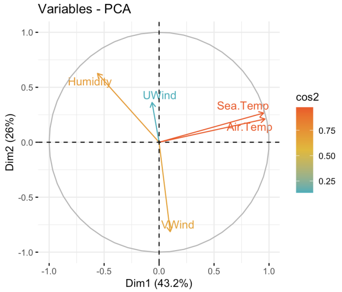
```
> marginplot(tao[ ,c("Humidity", "Air.Temp")])
> marginplot(tao[ ,c("Sea.Temp", "Air.Temp")])
> marginplot(tao[ ,c("Sea.Temp", "Humidity")])
> matrixplot(tao[,4:8])
```



# Missing Values & PCA

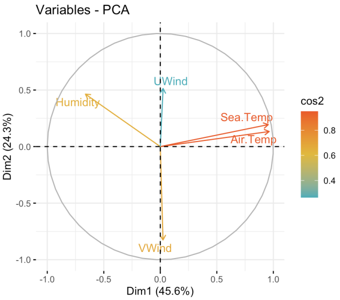
```
> mis.ind = matrix("o", nrow =
   nrow(tao[,4:8]), ncol = ncol(
   tao[,4:8]))
> mis.ind[is.na(tao[,4:8])] = "m"
> dimnames(mis.ind) = dimnames(
   tao[,4:8])
> library(FactoMineR)
> res.mca = MCA(mis.ind)
> fviz_mca_var(res.mca, repel =
   TRUE)
> res.pca = PCA(tao[!miss.ind
   ,4:8],graph=FALSE)
> fviz_pca_var(res.pca, col.var =
    "cos2")
```

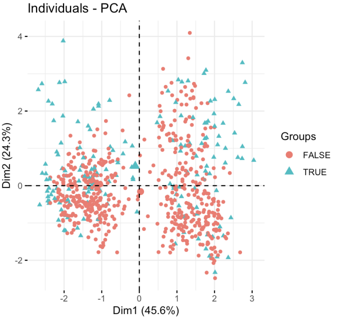


# Missing Values & PCA

```
> res.comp <- imputePCA(tao[,4:8], ncp = 2)
> res.pca = PCA(res.comp$completeObs)
> fviz_pca_var(res.pca, col.var = "cos2")
> fviz_pca_ind(res.pca, label="none", habillage=miss.ind)
```



# Missing Values & PCA

```
> tao_kNN = kNN(tao[,4:8], k = 5)
vars = c("Air.Temp","Humidity","
    Air.Temp_imp","Humidity_imp")
marginplot(tao_kNN[,vars],
    delimiter="_imp")
> X=data.frame(res.
    comp$completeObs)
names(X) = names(tao[,4:8])
W=matrix(FALSE, nrow = nrow(tao
    [,4:8]), ncol = ncol(tao
    [,4:8]))
W[is.na(tao[,4:8])] = TRUE
W=data.frame(W)
names(W) = paste(names(tao[,4:8])
    ,"_imp",sep="")
tao_ACP <- data.frame(X,W)
vars <- c("Air.Temp","Humidity","
    Air.Temp_imp","Humidity_imp")
marginplot(tao_ACP[,vars],
    delimiter="_imp"
```