Statistiques multivariées avec Mathematica

Dans ce notebook nous traitons de statistiques multivariées, donc de l'analyse d'un tableau de données.
Les questions qui se posent et les méthodes pour y répondre sont les suivantes :
Une variable est-elle fonction d'une ou de plusieurs variables?, Régressions simples ou multiples
Classer des objets en fonction d'une ou plusieurs variables?, Classifications
Situer un objet dans une classe déjà reconnue ?, Classements
Comparer la structure des variables et celle des objets?, Analyses factorielles
Plusieurs variables sont-elles fonction d'autres variables?, Corrélations canoniques

Création de tableaux de données

Pour répondre à ces questions nous créons deux tableaux. Le premier (datapop) issu des CountryData donne pour les pays du G8 (Canada, France, Allemagne, Italie, Japon, Russie,Grande_Bretagne, USA)), les valeurs des variables démographiques suivantes : naissances, décès, adultes_femme, adultes_homme, PNB, Population), puis un second tableau (dataaleatoire) de 15 lignes et 6 variables rempli de données aléatoires comprises entre 0 et 25. Toutes les fonctions se terminent par un ; et donc les résultats ne sont pas affichés à l'écran.

In[988]:=
```
ClearAll["Global`*"]
```
efface tout

```
datapop =
  DeleteCases[{CountryData[#, "AnnualBirths"], CountryData[#, "AnnualDeaths"],
  supprime cas  données de pays  données de pays
      CountryData[#, "FemaleAdultPopulation"],
      données de pays
      CountryData[#, "MaleAdultPopulation"],
      données de pays
      CountryData[#, "GDPPerCapita"], CountryData[#, "Population"]} & /@
      données de pays  données de pays
    CountryData["G8"], _Missing] // QuantityMagnitude;
    données de pays  ampleur de quantité
nomcolonne = {naissances, décès, adult_fe, adult_hom, pnb, population};
nomligne = {Canada, France, Allemagne, Italie, Japon, Russie, Grande_Bretagne, USA};
```

In[992]:=
```
dataaleatoire = RandomInteger[25, {15, 6}];
                entier aléatoire
dimension = Dimensions[dataaleatoire];
            dimensions
nbligne = dimension[[1]];
nbcolonne = dimension[[2]];
nomcolonne = {v1, v2, v3, v4, v5, v6};
```

Tester et mesurer les corrélations

Avant d'aborder divers outils pour répondre aux questionnements exposés en introduction, la première approche consiste à tester la présence de corrélation entre les variables, puis à mesurer leur intensité.

Tester la présence de corrélations

In[997]:=
```
testscorel = Table[IndependenceTest[dataaleatoire[[i]], dataaleatoire[[j]]],
                   table  test d'indépendance
    {i, 1, nbcolonne, 1}, {j, 1, nbcolonne, 1}];
TableForm[testscorel, TableHeadings → {nomcolonne, nomcolonne}]
 forme de table        en-têtes de table
```

Out[998]//TableForm=

	v1	v2	v3	v4	v5	v6
v1	0.	0.56670733	0.45790645	0.054543525	0.23066083	0.89(
v2	0.56670733	0.	0.66736406	0.34678351	0.43914586	0.49(
v3	0.45790645	0.66736406	0.	0.8970247	0.83317134	0.50(
v4	0.054543525	0.34678351	0.8970247	0.	0.54541842	0.95(
v5	0.23066083	0.43914586	0.83317134	0.54541842	0.	0.52(
v6	0.89058403	0.49920103	0.50643519	0.95873406	0.52970924	0.

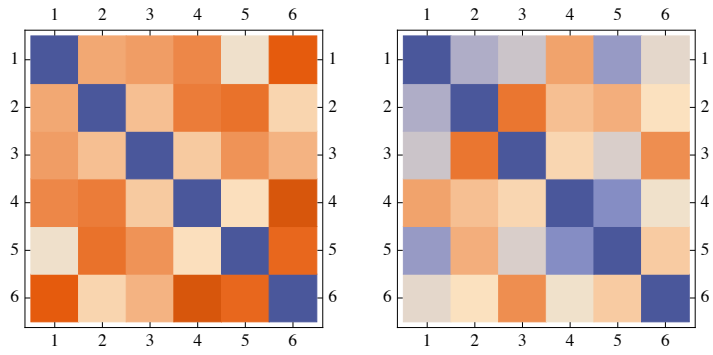
Quand les variables sont indépendantes, la valeur de p est élevée. Cette valeur de p est nulle ou proche de 0 quand une relation existe entre les variables, donc dans la diagonale du tableau. Après ce premier examen, il est possible de calculer les corrélations entre les variables, et d'en fournir une représentation visuelle.

Calcul et visualisation des corrélations entre les variables

Nous calculons donc les corrélations linéaires de Pearson et les corrélations plus robustes de Hoeffding. Il serait possible de calculer bien d'autres corrélations, notamment celle de Spearman ou de Kendall.

In[999]:=
```
mh = HoeffdingD[dataaleatoire] // N;
     Hoeffding dépendance D          valeur numérique
mp = Correlation[dataaleatoire] // N;
     corrélation                     valeur numérique
arraymh = MatrixPlot[mh, ImageSize → 200, PlotTheme → "Scientific"];
          tracé de matrice  taille d'image  thème de tracé
arraymp = MatrixPlot[mp, ImageSize → 200, PlotTheme → "Scientific"];
          tracé de matrice  taille d'image  thème de tracé
GraphicsRow[{arraymh, arraymp}]
 rangée de graphiques
```

Out[1003]=



La couleur bleue indique les valeurs des corrélations positives et la couleur rouge les valeurs de corrélation négatives.

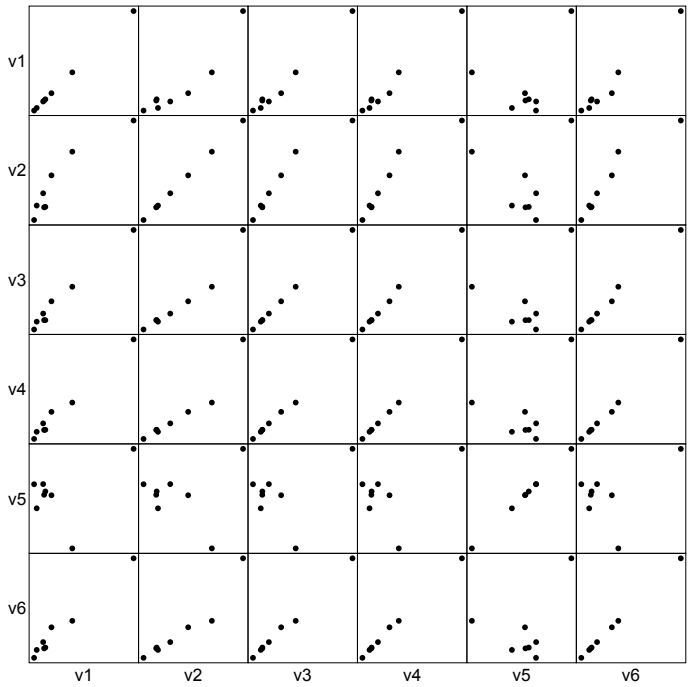
De la régression simple à la régression multiple

Une première approche visuelle est utile pour observer la forme des relations entre les variables.

In[1004]:=
```
Needs["StatisticalPlots`"]
```
nécessite

```
PairwiseScatterPlot[datapop, DataLabels → nomcolonne]
```

Out[1005]=



Pour le tableau relatif à la population du G8, la plupart des relations sont de type linéaire.

Un exemple de régression linéaire simple

Étudions d'abord la relation entre deux variables, par exemple les variables 1 (le nombre de naissances) et 4 (le nombre d'adultes homme) du tableau dataaleatoire. On récupère les données des deux colonnes du tableau pour créer le fichier dataxy. Repérez les doubles crochets.

In[1006]:=
```
dataxy = Transpose[{dataaleatoire[[All, 1]], dataaleatoire[[All, 4]]}];
```
transposée tout tout

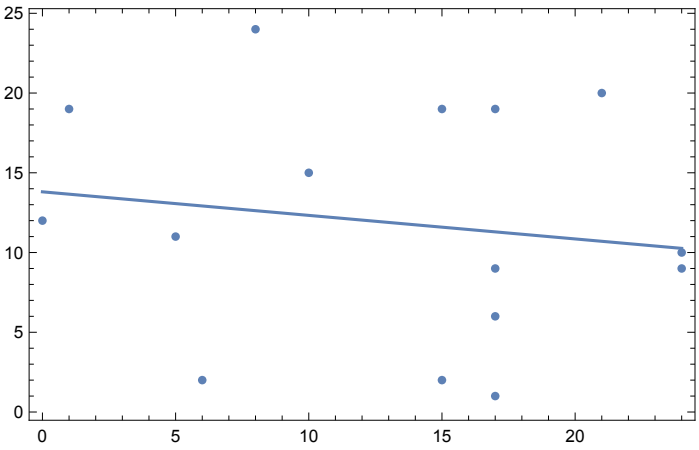
Puis, analysons comment la variable 4 varie en fonction de la variable 1 avec les fonction LinearModelFit[] et NonLinearModelFit[] :

In[1007]:=
```
model = LinearModelFit[dataxy, x, x]
        ajuste modèle linéaire
Normal[model]
forme normale
Show[ListPlot[dataxy], Plot[model[x], {x, 0, Max[dataxy]}], Frame → True]
mon··· tracé de liste   tracé de courbes   maximum   cadre   vrai
model["RSquared"]
```

Out[1007]= FittedModel[13.806462 − 0.14770014 x]

Out[1008]= 13.806462 − 0.14770014 x

Out[1009]=



Out[1010]= 0.024572559

Les données et la droite d'ajustement sont illustrés. Et le coefficient de détermination, égal à 0.16, confirme ce que montre la figure et signifie que ce modèle n'est pas significatif. Nous testons alors un modèle non linéaire, ce qui donne :

In[1011]:=
```
model2 = NonlinearModelFit[dataxy, Log[a + b x^2], {a, b}, x]
         ajustement du modèle non linéaire   logarithme
model2["RSquared"]
```

Out[1011]= FittedModel[Log[287 337.82 − 466.02838 x^2]]

Out[1012]= 0.74308509

Sans être parfait, ce second modèle est bien meilleur, car le coefficient de détermination est égal à 0.65. Bien d'autres indicateurs, utilisés dans le programme suivant sont à votre disposition pour attester la qualité d'un modèle de régression (voir ci-dessous). ***Attention pour cet exercice vous pouvez obtenir des résultats différents, car les données sont modifiées à chaque ouverture du notebook.***

Construire un modèle de régression multiple et apprécier sa validité

Le programme ci-dessous est plus complet. Pour être directement applicable, il suffirait d'importer les données (voir le notebook importer les données). Dans cet exemple, nous utilisons le tableau des données de population du G8. La variable à expliquer est toujours la dernière colonne du tableau (population). Après avoir calculé les paramètres du modèle, de nombreux tests sont réalisés pour en apprécier la validité et la robustesse.

In[1013]:=
```
(*modèle de régression linéaire*)
model = LinearModelFit[datapop, {u, v, w, x, y}, {u, v, w, x, y}];
        ajuste modèle linéaire
Print["Modèle de régression linéaire"]
imprime
Normal[model]
forme normale
dmax = Max[datapop[[All, All]]]
       maximum          tout  tout
(*Validité et robustesse du modèle*)
Print["Analyse de variance"]
imprime
anov = model["ANOVATable"]
dw = model["DurbinWatsonD"];
Print["test de Durbin Watson = ", dw]
imprime
Print["Coefficients de détermination, Akaiké et Bayes"]
imprime
Grid[Transpose[{#, model[#]} &[{"RSquared", "AIC", "BIC"}]],
grille  transposée
  Frame → All, Alignment → "."]
  cadre      tout   alignement
Print[]
imprime
Print["Graphique des résidus"]
imprime
ListPlot[model["FitResiduals"],
tracé de liste
  Filling → Axis, AxesLabel → {"Espaces", "résidus"}]
  remplissage  axe     titre d'axe
Print[]
imprime
Print["Graphique de l'influence de chaque observation"]
imprime
ListPlot[model["CovarianceRatios"], Filling → Axis,
tracé de liste                              remplissage  axe
  AxesLabel → {"Espaces", "taux_de_covariance"}]
  titre d'axe
Print[]
imprime
Print["Graphique des distances de Cook"]
imprime
ListPlot[model["CookDistances"], PlotRange → {0, All},
tracé de liste                          zone de tracé      tout
  Filling → 0, AxesLabel → {"Espaces", "DistanceCook"}]
  remplissage      titre d'axe
Print[]
imprime
Print["Graphique des valeurs prédites par rapport aux valeurs observées"]
imprime
ListPlot[Transpose[model[{"Response", "PredictedResponse"}]],
tracé de liste  transposée
  FrameLabel → {"observed", "predicted"}, Frame → True, Axes → False]
  titre de cadre                                cadre  vrai  axes  faux
```

```
Print[]
imprime
```

```
Print["Graphique des résidus par rapport à chaque variable"]
imprime
```

```
Table[ListPlot[Transpose[{datapop[[All, i]], model["FitResiduals"]}],
table  tracé de liste  transposée  tout
    Frame → True, FrameLabel → {{u, v, w, x, y,}[[i]], "Residual"},
    cadre  vrai  titre de cadre
    ImageSize → Medium], {i, 5}] // Column
    taille d'image  taille moyenne  colonne
```

Modèle de régression linéaire

Out[1015]= $1.2632564 \times 10^7 + 6.4003049\ u + 29.821526\ v - 5.5104874\ w + 7.8094959\ x - 313.60865\ y$

Out[1016]= 329 064 917

Analyse de variance

Out[1018]=

	DF	SS	MS	F-Statistic	P-Value
u	1	6.0128853×10^{16}	6.0128853×10^{16}	4418.4498	0.00022624692
v	1	4.4409296×10^{14}	4.4409296×10^{14}	32.633292	0.029303293
w	1	9.6754851×10^{14}	9.6754851×10^{14}	71.098387	0.013775069
x	1	5.78904×10^{13}	5.78904×10^{13}	4.2539615	0.17525598
y	1	6.330145×10^{12}	6.330145×10^{12}	0.46515818	0.56561189
Error	2	2.7217172×10^{13}	1.3608586×10^{13}		
Total	7	6.1631932×10^{16}			

test de Durbin Watson = 1.8411076

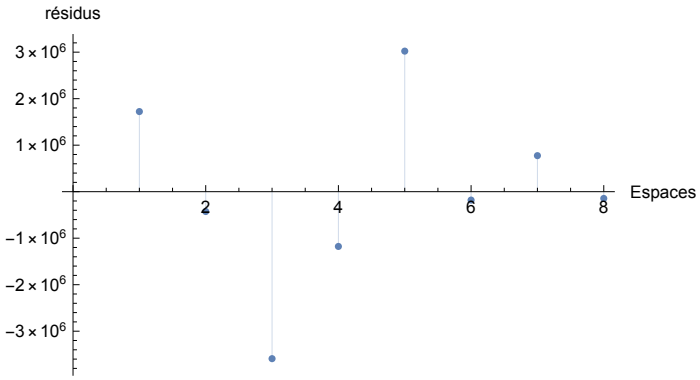
Coefficients de détermination, Akaiké et Bayes

Out[1022]=

RSquared	0.99955839
AIC	272.63679
BIC	273.19288

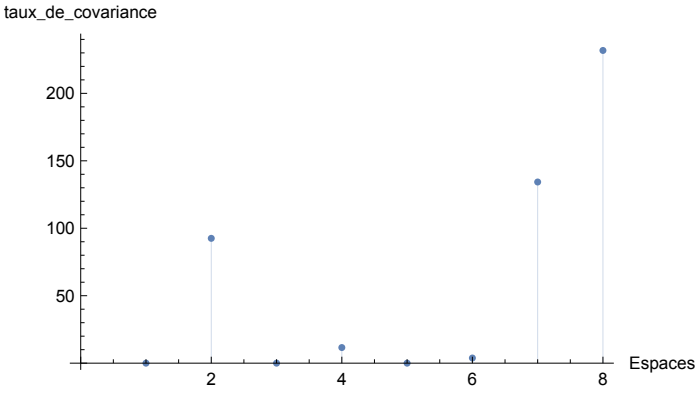
Graphique des résidus

Out[1025]=



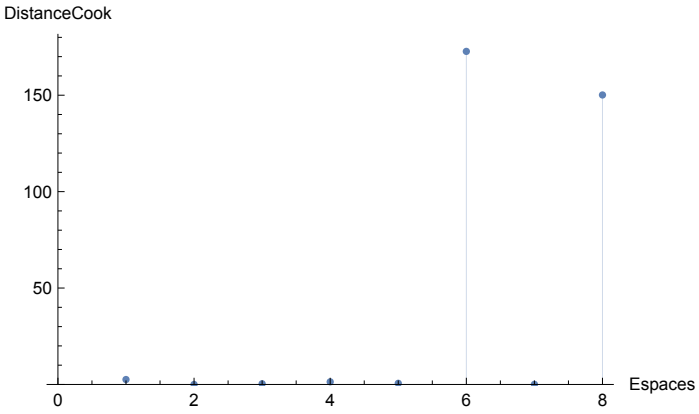
Graphique de l'influence de chaque observation

Out[1028]=



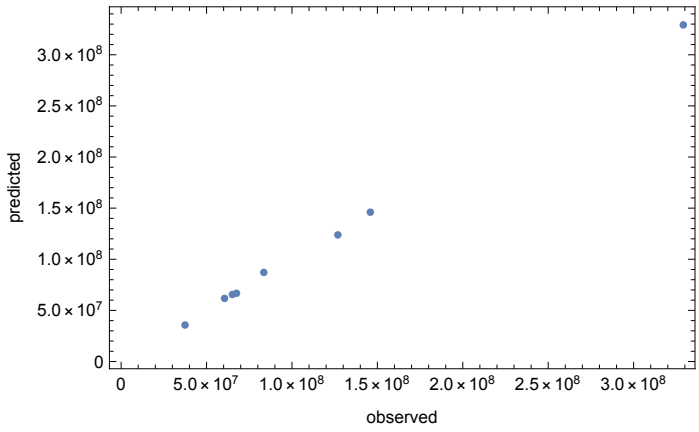
Graphique des distances de Cook

Out[1031]=



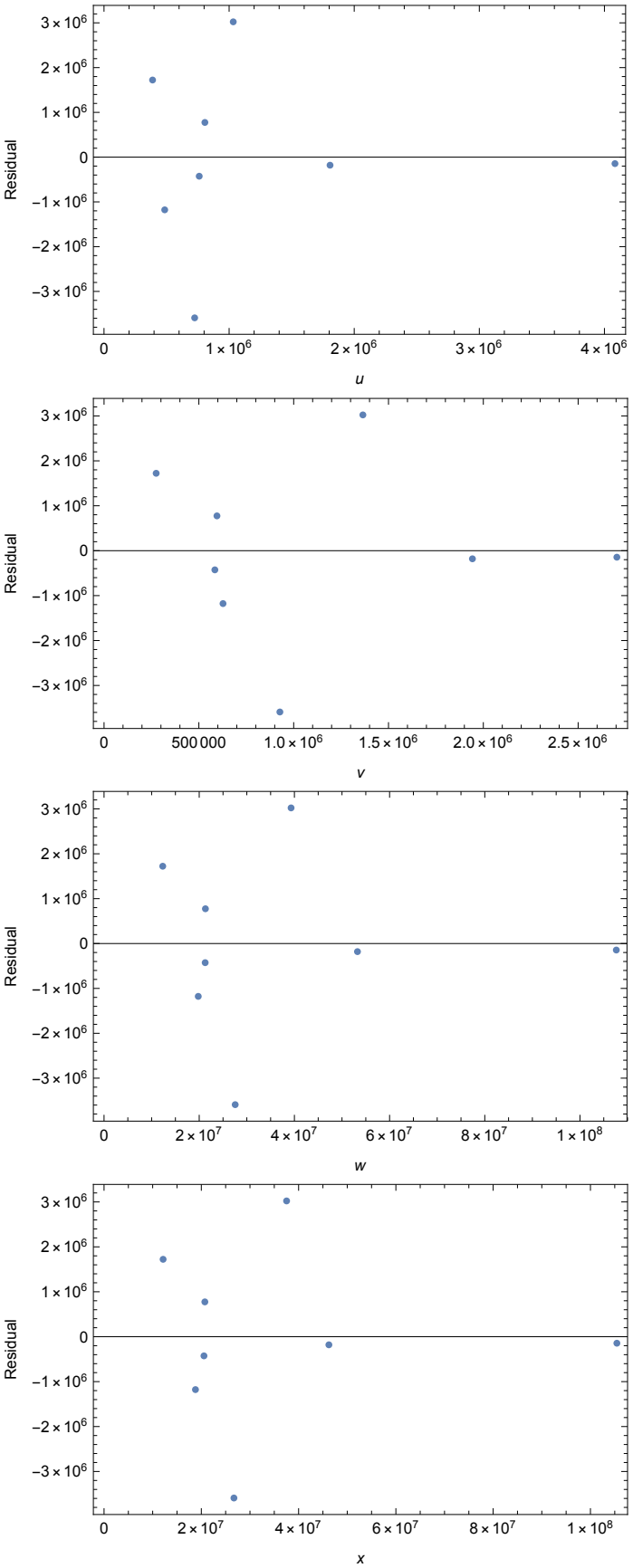
Graphique des valeurs prédites par rapport aux valeurs observées

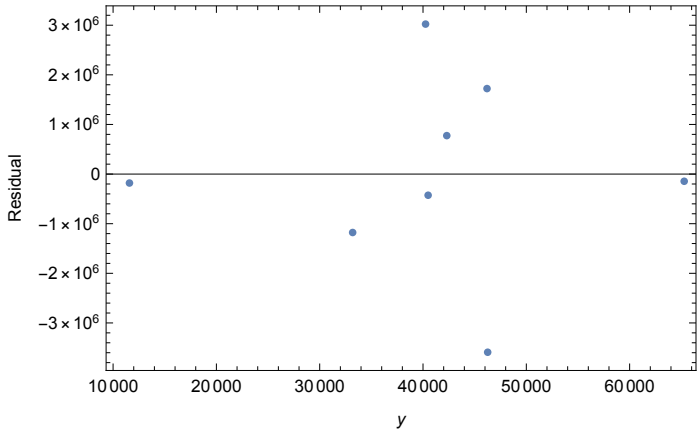
Out[1034]=



Graphique des résidus par rapport à chaque variable

Out[1037]=





Bien d'autres propriétés permettent de vérifier la qualité du modèle de régression. La liste complète de ces propriétés s'obtient avec l'instruction ci-dessous :

In[1038]:=
```
model["Properties"]
```
propriétés

Out[1038]=
```
{AdjustedRSquared, AIC, AICc, ANOVATable, ANOVATableDegreesOfFreedom,
 ANOVATableEntries, ANOVATableFStatistics, ANOVATableMeanSquares,
 ANOVATablePValues, ANOVATableSumsOfSquares, BasisFunctions,
 BetaDifferences, BestFit, BestFitParameters, BIC, CatcherMatrix,
 CoefficientOfVariation, CookDistances, CorrelationMatrix,
 CovarianceMatrix, CovarianceRatios, Data, DesignMatrix,
 DurbinWatsonD, EigenstructureTable, EigenstructureTableEigenvalues,
 EigenstructureTableEntries, EigenstructureTableIndexes,
 EigenstructureTablePartitions, EstimatedVariance, FitDifferences,
 FitResiduals, Function, FVarianceRatios, HatDiagonal, MeanPredictionBands,
 MeanPredictionConfidenceIntervals, MeanPredictionConfidenceIntervalTable,
 MeanPredictionConfidenceIntervalTableEntries, MeanPredictionErrors,
 ParameterConfidenceIntervals, ParameterConfidenceIntervalTable,
 ParameterConfidenceIntervalTableEntries, ParameterConfidenceRegion,
 ParameterErrors, ParameterPValues, ParameterTable, ParameterTableEntries,
 ParameterTStatistics, PartialSumOfSquares, PredictedResponse,
 Properties, Response, RSquared, SequentialSumOfSquares,
 SingleDeletionVariances, SinglePredictionBands,
 SinglePredictionConfidenceIntervals, SinglePredictionConfidenceIntervalTable,
 SinglePredictionConfidenceIntervalTableEntries, SinglePredictionErrors,
 StandardizedResiduals, StudentizedResiduals, VarianceInflationFactors}
```

Les résultats sont très robustes et le modèle très significatif, ce qui n'est pas surprenant vu les variables retenues. Nous laissons l'interprétation complète au lecteur.
Intégrées dans le logiciel, le chercheur peut recourir à d'autres fonctions : NonLinearModelFit[], déjà utilisée dans le précédent exercice, mais aussi GeneralizedLinearModelFit[], LogitModelFit[],ProbitModelFit[]. De plus, d'autres modèle de régression sont proposés dans les Resource functions, notamment ceux de la régression des quantiles et de la régression de Bayes.

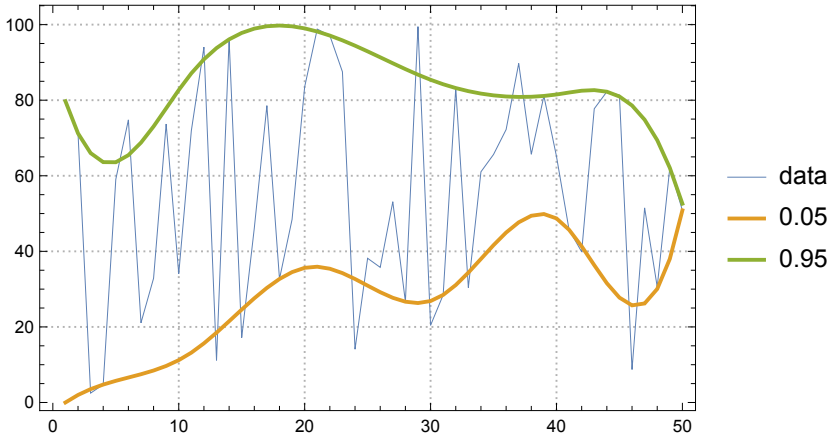
De très nombreux modèles de régression

Ci-dessous le petit programme procède à une modélisation sous la forme d'une régression quantile

pour une série de données aléatoires :

In[1039]:=
```
n = 50;
datadeux = Transpose[{Range[n], RandomReal[{0, 100.}, n]}];
           transposée   plage    nombre réel aléatoire
result = ResourceFunction["QuantileRegression"][datadeux, 5, {0.20, 0.80}];
         fonction ressource
ListLinePlot[{datadeux, result[[1]] /@ datadeux[[All, 1]],
tracé de liste de ligne                          tout
  result[[2]] /@ datadeux[[All, 1]]}, PlotLegends → {"data", 0.05`, 0.95`},
                                  tout          légendes de tracé
 PlotStyle → {Thin, Thick, Thick}, PlotTheme → "Detailed"]
 style de tracé  fin   épais  épais  thème de tracé
```

Out[1042]=



Ce type de régression, qui procède à un ajustement par des fonctions spline, est très utilisé pour l'analyse des séries temporelles.

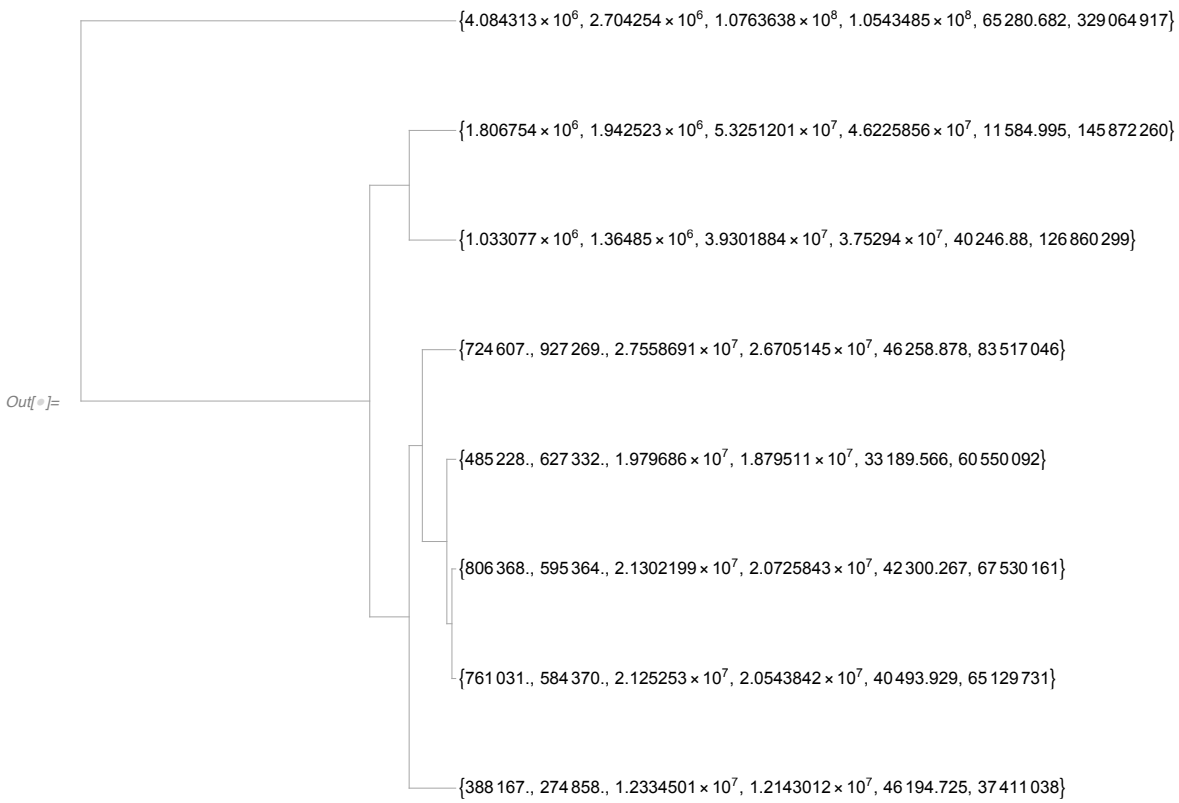
Une gamme complète de classifications

Une première approche visuelle : le Dendrogram

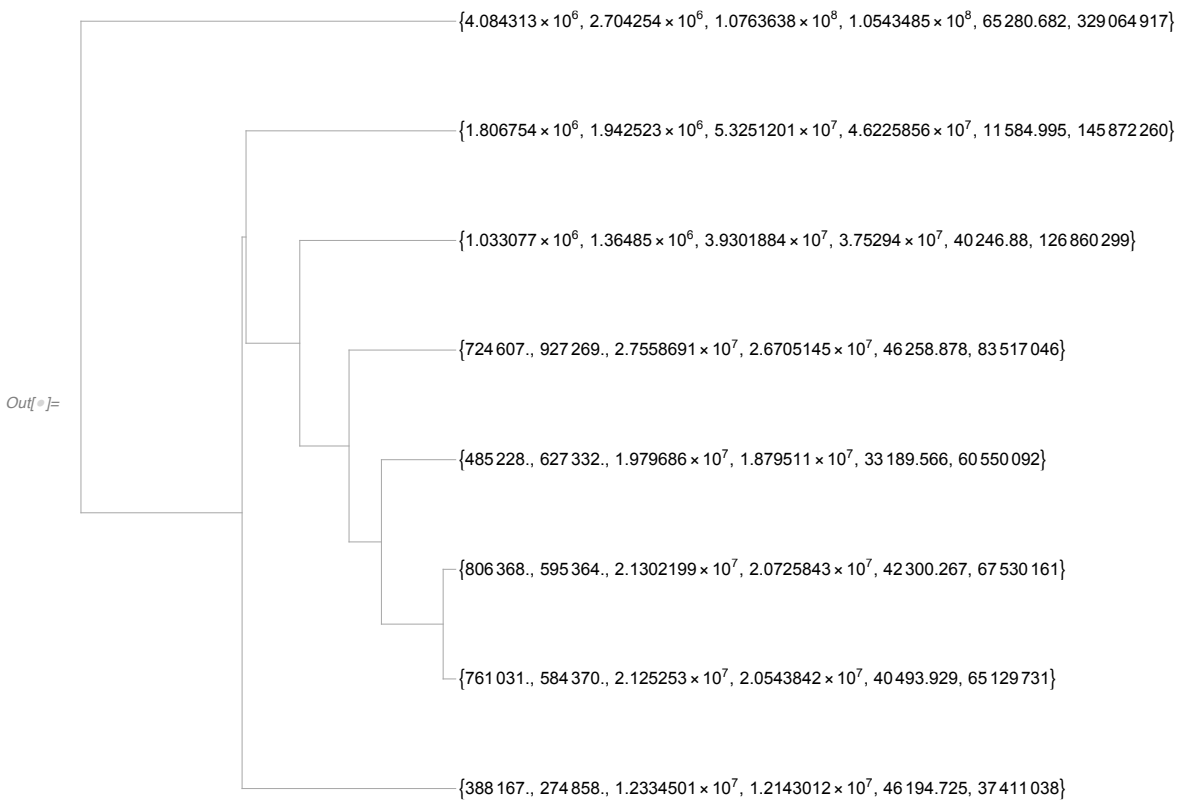
Pour obtenir une première idée de classification d'objets, il est utile d'adopter une démarche visuelle avec la fonction **Dendrogram**[]. Comme toute fonction, le chercheur peut utiliser diverses options, notamment l'orientation de l'arbre de classification, et surtout le choix d'une distance ou d'un indice de dissimilarité. Voici deux dendrogrammes issus du tableau des populations du G8, avec deux distances différentes, sans habillage graphique pour mieux les lire.

In[ ]:=
```
Dendrogram[datapop, Left,
dendogramme        gauche
 DistanceFunction → EuclideanDistance, ImageSize → Large]
 fonction de distance  distance euclidienne  taille d'image  taille grande
Dendrogram[datapop, Left, DistanceFunction → CanberraDistance, ImageSize → Large]
dendogramme        gauche  fonction de distance  distance Canberra  taille d'image  taille gran
```

Out[ ]=



Out[ ]=



Comme beaucoup d'autres fonctions, **Dendrogram**[] s'applique à tous les objets Mathematica, notamment des images ou des cartes.

Deux fonctions pour classer des objets : FindClusters[] et ClusteringComponents[]

Les deux fonctions **FindClusters**[] et **ClusteringComponents**[] élaborent des classifications plus

fines. Elles possèdent pratiquement les mêmes options, mais elles donnent un affichage différent comme le montre les deux lignes de programmes ci-dessous, appliquées à la même liste de nombre, et retenant le même nombre de classes, à savoir 3.

```
In[•]:= FindClusters[{1, 2, 10, 12, 3, 1, 13, 25}, 3]
        trouve cumuls
        ClusteringComponents[{1, 2, 10, 12, 3, 1, 13, 25}, 3]
        composants par regroupement
Out[•]= {{1, 2, 3, 1}, {10, 12, 13}, {25}}
Out[•]= {1, 1, 2, 2, 1, 1, 2, 3}
```

FindClusters[] donne les valeurs des données classées en trois groupes. C'est une liste de liste de trois listes, la dernière sous-liste donc la troisième classe comprenant le seul nombre 25. **Clustering-Components**[] donne un résultat similaire, mais ce sont leur étiquette de classe et non pas les valeurs qui est fournie. Le nombre 25, toujours seul est dans la classe 3, et le nombre 10, le troisième dans la liste est défini par l'étiquette 2. Il est bien dans la classe 2.
Ci-dessous 4 classifications réalisées avec 4 distances différentes sur le tableau datapop. L'affichage des résultats est rendu plus lisible à l'aide d'options. Dans la fenêtre qui apparaît d'abord à l'écran le chercheur tape le nombre de classes qu'il souhaite, 3 dans notre exercice.

```
(*Dialogue pour entrer le nombre de classes *)
ny = ToExpression[DialogInput[DynamicModule[{name = ""},
      convertis en une ··· entrée de dialo··· module dynamique
       Column[{"Choisir le nombre de classes", InputField[Dynamic[name], String],
        colonne                                champ d'entrée dynamique chaîne de ca
          ChoiceButtons[{DialogReturn[name], DialogReturn[]}]]]]];
          boutons de sélection retour de dialogue retour de dialogue
(*Quatre classification avec des fonctions distance différentes *)
class1 = ClusteringComponents[datapop, ny, 1];
         composants par regroupement
class2 =
  ClusteringComponents[datapop, ny, 1, DistanceFunction → ManhattanDistance];
  composants par regroupement fonction de distance distance Manhattan
class3 = ClusteringComponents[datapop, ny, 1,
         composants par regroupement
   DistanceFunction → CorrelationDistance];
   fonction de distance distance de corrélation
class4 = ClusteringComponents[datapop, ny, 1,
         composants par regroupement
   DistanceFunction → NormalizedSquaredEuclideanDistance];
   fonction de distance distance euclidienne normalisée au carré
(*Affichage des résultats dans un tableau *)
nom = {distances, euclidienne, de_Manhattan, de_correlation, euclid_normalisee};
entete = {nomligne, nom};
ad5 = Transpose[{class1, class2, class3, class4}];
      transposée
Grid[Prepend[Flatten /@ Transpose[{entete[[1]], ad5}],
grille appose aplatis transposée
  PadLeft[entete[[2]], Length@ad5[[1]] + 1, ""]], Frame → All]
  rembourre à gauche longueur cadre tout
```

Out[•]=

distances	euclidienne	de_Manhattan	de_correlation	euclid_normalisee
Canada	1	1	1	1
France	1	1	1	2
Allemagne	1	1	1	2
Italie	1	1	1	2
Japon	1	2	2	2
Russie	2	2	3	2
Grande_Bretagne	1	1	2	2
USA	3	3	1	3

En outre, le chercheur peut choisir sa méthode. Sauf indication du chercheur, ce choix est fait automatiquement en fonction de la nature des données (nombres, images, autres objets). En voici un exemple :

In[•]:=
```
class5 = ClusteringComponents[datapop, 3, 1, Method → "KMeans"]
         composants par regroupement méthode
```

Out[•]=
```
{1, 1, 1, 1, 2, 2, 1, 3}
```

Enfin de nombreux outils spécifiques de classification sont adaptés au traitement d'images et de réseaux modélisés par la théorie des graphes. Ce que nous traiterons ultérieurement.

Analyse des structures avec les analyses factorielles

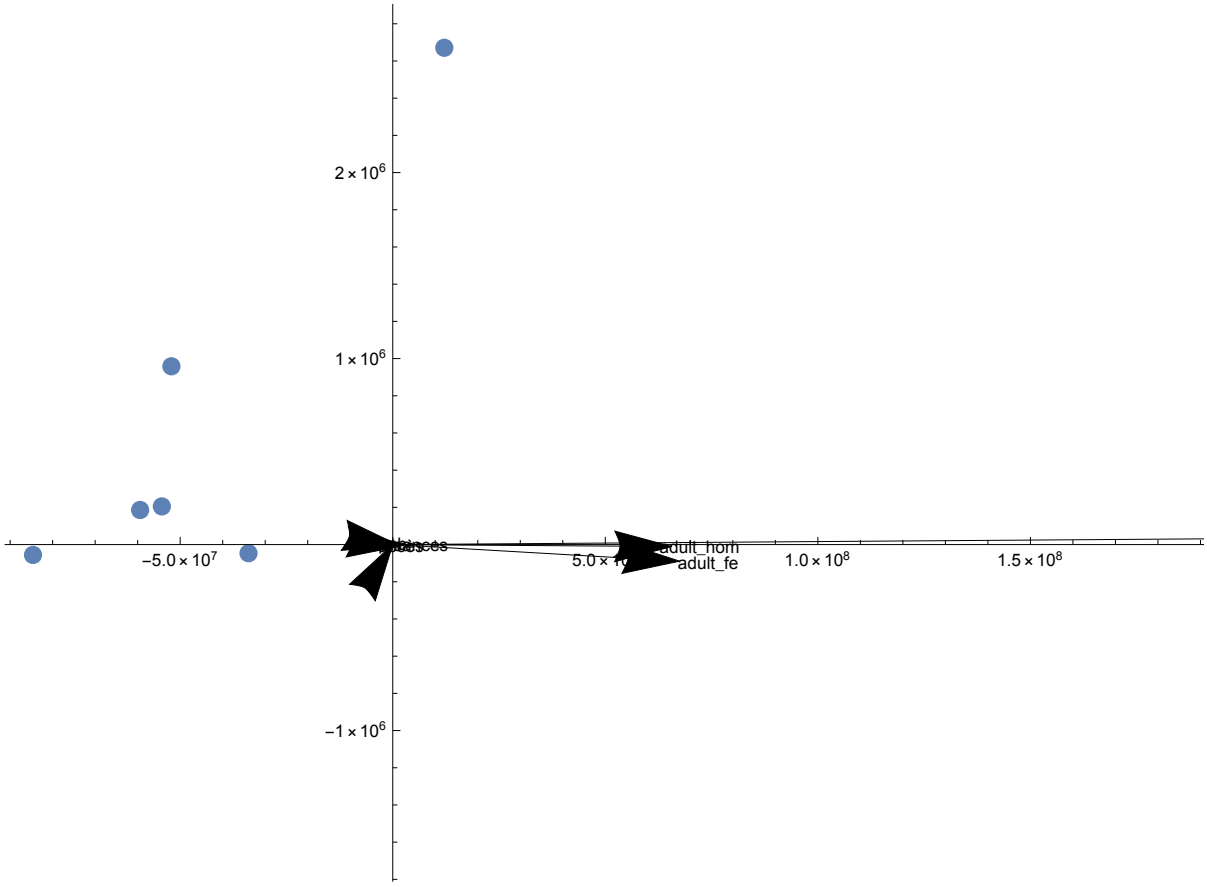
Le langage Mathematica possède de nombreuses fonctions pour réaliser des analyses factorielles. Il est possible de commencer en premier lieu avec une approche visuelle, incluse dans les Resource functions.

Une approche visuelle de la décomposition en composantes principales

```
In[•]:= ResourceFunction["BiPlot"][datapop, "ColumnNames" → nomcolonne]
```

fonction ressource

Out[•]=

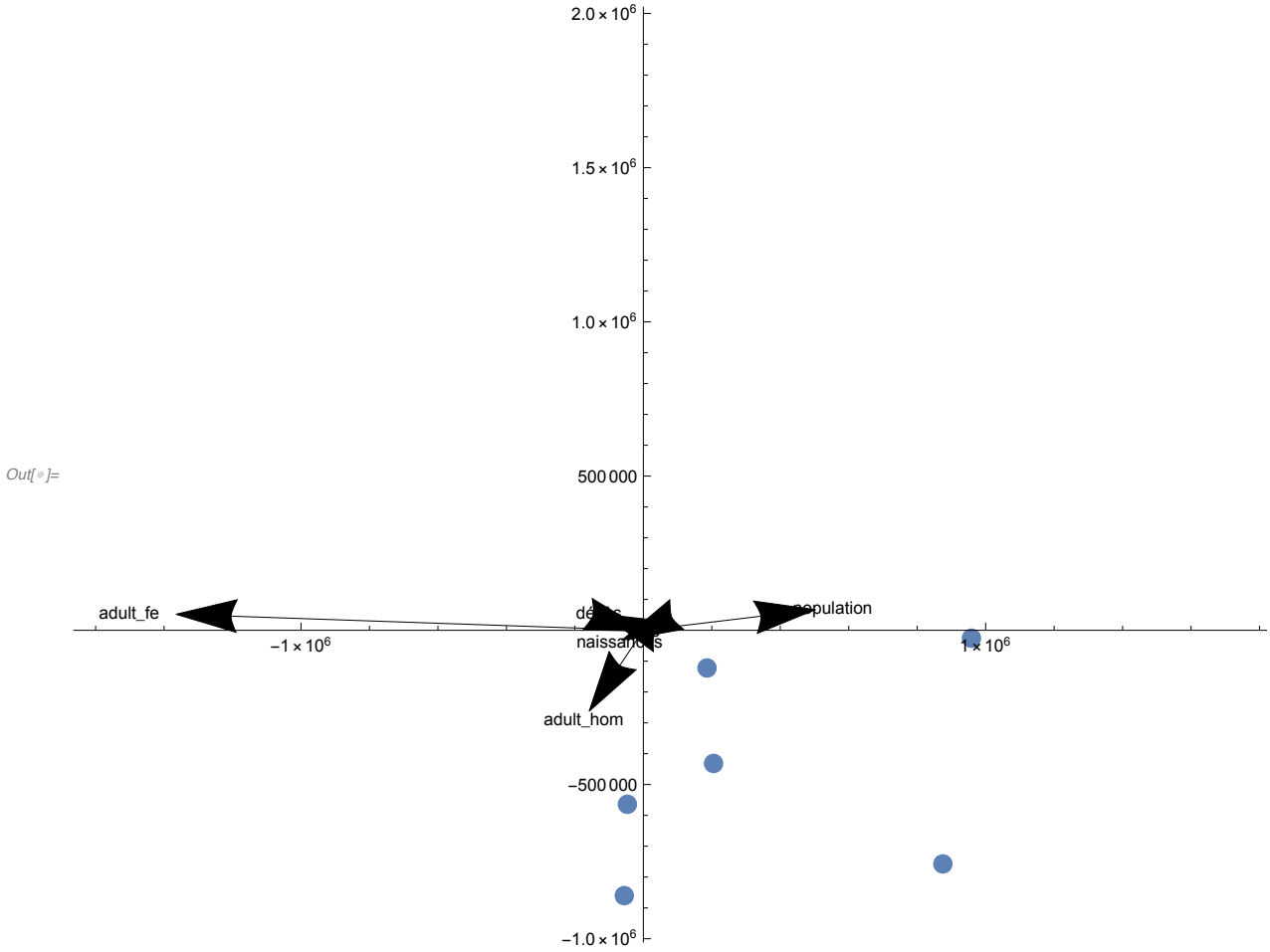


En agrandissant la figure, on perçoit immédiatement la structure des variables imposée par l'effet de taille des trois variables : la population, le nombre d'adultes femmes et d'adultes hommes. Il est possible d'obtenir un graphique avec 3 axes ou choisir deux axes, pas obligatoirement le premier et le deuxième qui sortent automatiquement. Le même exemple avec les axes 2 et 4 :

In[•]:= `ResourceFunction["BiPlot"][datapop, {2, 3}, "ColumnNames" → nomcolonne]`

fonction ressource

Out[•]=



Des fonctions pour élaborer des modèles d'analyses factorielles

Pour réaliser des programmes plus développés, il est nécessaire d'utiliser les fonctions **Eigenvalues**[], **Eigenvector**[], **Eigensystem**[] et/ou **PrincipalComponents**[]. Mais seule cette dernière fonction s'applique directement à la matrice de départ. Les autres fonctions s'appliquent à des matrices carrées, donc généralement à la matrice des covariances ou celle des corrélations. Ci-dessous un programme élémentaire d'analyse en composantes principales appliquée au tableau datapop. Le lecteur choisit les deux axes à retenir pour les représentations graphiques.

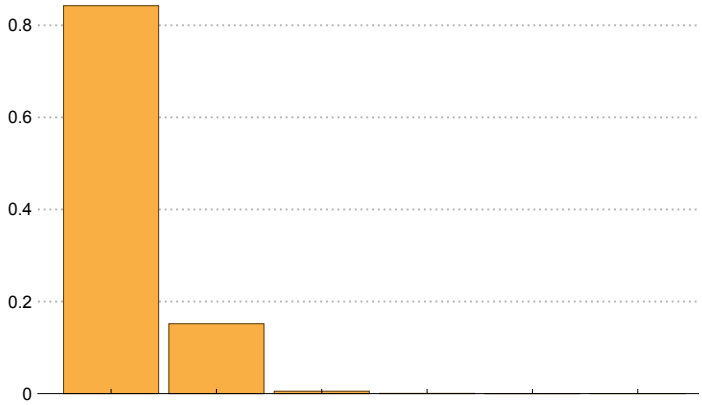
In[971]:=
```
mp = Correlation[datapop] // N;
axe1 = ToExpression[Input["choisir le premier axe"]]
axe2 = ToExpression[Input["choisir le second axe"]]
{vals, vecs} = Eigensystem[mp];
Print["Daigramme des valeurs propres "]
BarChart[vals / Total[vals], PlotTheme → "Business"]
cp = PrincipalComponents[datapop, Method → "Correlation"];
Print["Saturations composantes principales objets "]
TableForm[cp, TableHeadings → {nomligne, Range[Length[nomcolonne]]}]
Print["graphique des deux composantes principales choisies"]
ListPlot[Transpose[{cp[[All, axe1]], cp[[All, axe2]]}] → nomligne,
 PlotRange → All, PlotTheme → "Scientific"]
Print["Saturations composantes principales variables"]
vec3 = vecs / Sqrt[Total[vecs^2]];
saturationcv = -Transpose[Sqrt[vals] * vec3];
TableForm[saturationcv,
 TableHeadings → {nomcolonne, Range[Length[nomcolonne]]}]
Print["graphique des deux saturations composantes variables choisies"]
ListPlot[Transpose[{-saturationcv[[All, axe1]], -saturationcv[[All, axe2]]}] →
  nomcolonne, PlotRange → All, PlotTheme → "Scientific"]
```

Out[972]= 1

Out[973]= 2

```
Daigramme des valeurs propres
```

Out[976]=



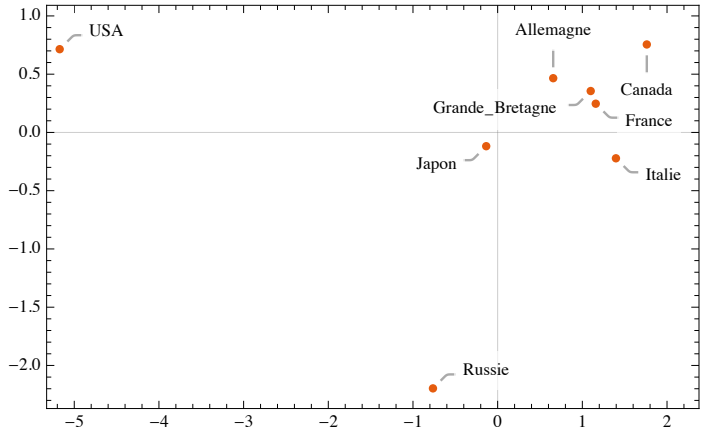
Saturations composantes principales objets

Out[979]//TableForm=

	1	2	3	4	5
Canada	1.7613223	0.75503435	0.097194606	0.023319747	0
France	1.1584138	0.24660016	0.14264277	−0.0002325345	-
Allemagne	0.65597361	0.46549121	−0.22591318	0.066574171	0
Italie	1.396812	−0.22203962	0.038919136	−0.10606882	0
Japon	−0.13509195	−0.11816736	−0.32538364	−0.036225736	-
Russie	−0.76402084	−2.1968402	0.061539929	0.035737517	0
Grande_Bretagne	1.098843	0.35540553	0.13583086	0.027663765	-
USA	−5.172252	0.7145159	0.075169525	−0.010768105	0

graphique des deux composantes principales choisies

Out[981]=



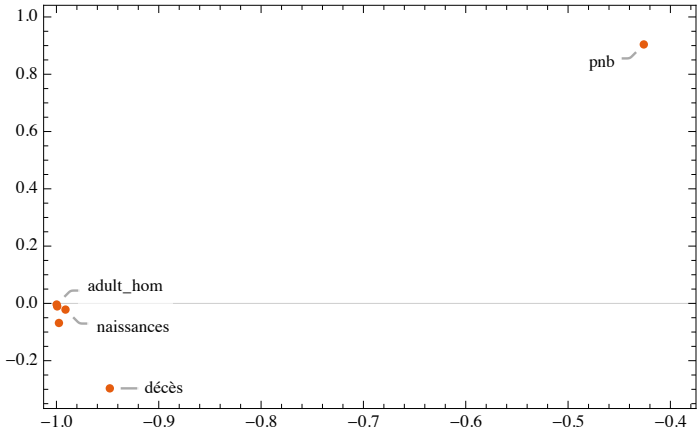
Saturations composantes principales variables

Out[985]//TableForm=

	1	2	3	4	5
naissances	0.99114684	0.021589953	0.12859661	0.024533272	−0.00
décès	0.94782679	0.29659357	−0.11328667	0.028593648	−0.00
adult_fe	0.99754747	0.068325534	0.0065059377	−0.0063220255	0.011
adult_hom	0.99976331	0.0039379657	0.0049417477	−0.018489997	0.008
pnb	0.42598573	−0.90405366	−0.033780954	0.0090536827	−0.00
population	0.99923641	0.010508839	−0.017135415	−0.030505792	−0.01

graphique des deux saturations composantes variables choisies

Out[987]=



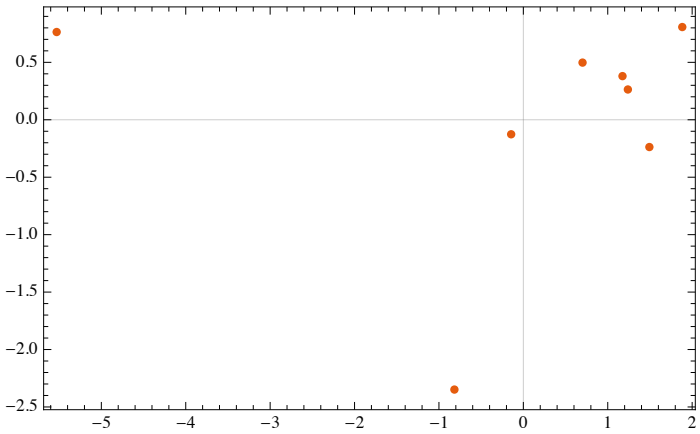
Le lecteur trouvera des programmes d'analyse des correspondances dans notre ouvrage *Modèles géographiques avec le langage Mathematica.*D'autres techniques d'analyses multivariées, proches des analyses factorielles sont disponibles dans les Resource functions, et dans la version 12.2 du logiciel livré en décembre 2020.

L'analyse en composante principale est en fait une méthode particulière d'une approche plus générale, la réduction de dimensions. Il est par exemple possible de réaliser une analyse de réduction de dimensions avec la fonction **DimensionReduce[]**, en retenant en option soit la PrincipalComponentAnalysis, soit la MultidimensionalScaling (MDS), ou même l'option TSNE comme le montre la troisième ligne de programmation ci-dessous. Le MDS a l'avantage de convenir quand les relations entre les variables ne sont pas linéaires, tout comme l'option Isomap utilisée dans la dernière ligne. On entre dans le domaine de l'apprentissage automatique traité dans un futur notebook. C'est dans ce cadre que seront abordés les problèmes relative au classement d'objets.

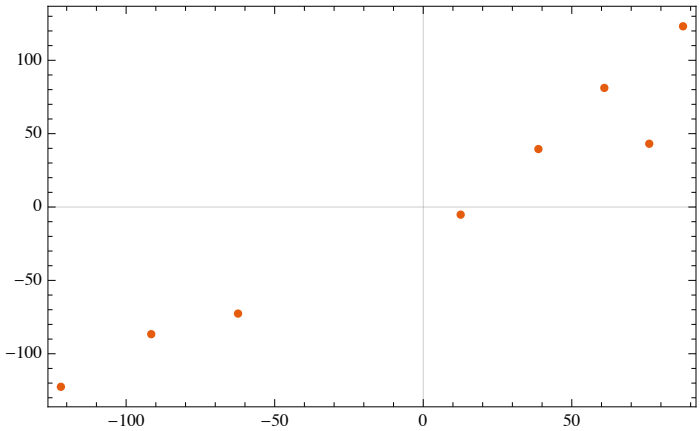
In[274]:=
```
Print["graphique des axes de réduction selon trois méthodes"]
ListPlot[DimensionReduce[datapop, 2, Method → "PrincipalComponentsAnalysis"],
  PlotRange → All, PlotTheme → "Scientific"]
ListPlot[DimensionReduce[datapop, 2, Method → "TSNE"],
  PlotRange → All, PlotTheme → "Scientific"]
ListPlot[DimensionReduce[datapop, 2, Method → "Isomap"],
  PlotRange → All, PlotTheme → "Scientific"]
```
imprime | tracé de liste | réduis dimension | méthode | zone de tracé | tout | thème de tracé

graphique des axes de réduction selon trois méthodes

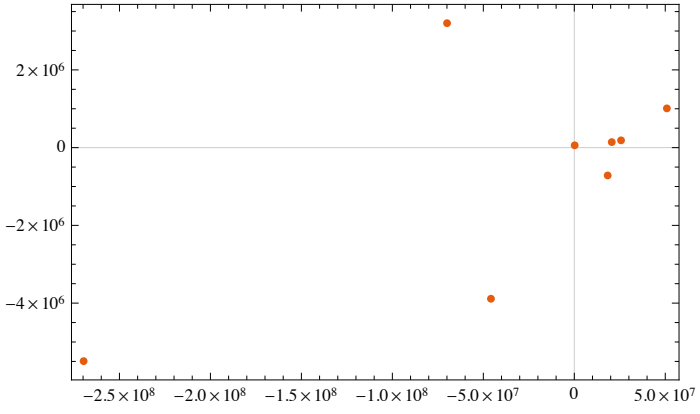
Out[275]=



Out[276]=



Out[277]=



Le lecteur attentif notera la similitude entre le premier graphique et celui issu de l'analyse en composantes principales exposé plus haut.

Pour l'approche par une technique de corrélation canonique, le lecteur se reportera à l'article de Rodrigo Loureiro Malacarne paru dans *The Mathematica Journa*l.

Conclusion

La plupart des modèles d'analyse des données sont enrichis et généralisés par les outils d'apprentissage automatique que nous aborderons dans un autre cahier.

Ouvrage recommandé :

Pierre Dumolard, 2011, *Données géographiques, analyse statistique multivariée*, Hermès.