Липецкий государственный технический университет

Кафедра прикладной математики

# КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ МАТЕМАТИЧЕСКИХ ИССЛЕДОВАНИЙ

Лекция 8

## 9. Методы снижения размерности моделей и обнаружение в моделях скрытой структуры.
## 10. Кластерный анализ

Составитель - Сысоев А.С., к.т.н., доц.

Липецк – 2022

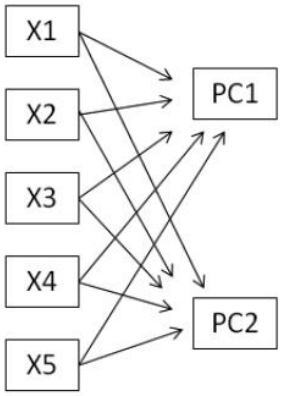
## Outline

## 9. Методы снижения размерности и обнаружения скрытой структуры

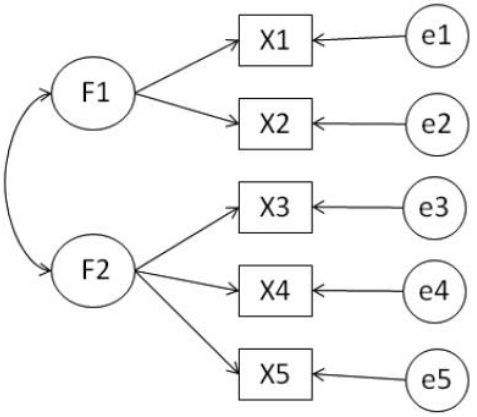
---

**Анализ главных компонент** (principal components analysis, PCA) – это способ снижения размерности данных, который преобразует большое число скоррелированных переменных в гораздо меньший набор нескоррелированных переменных, называемых *главными компонентами*.

**Факторный анализ** (exploratory factor analysis, EFA) - методы, которые обнаруживают скрытую структуру в имеющемся наборе переменных. Этот анализ позволяет найти меньший набор лежащих в основе, или латентных структурных компонентов, которые могут объяснить взаимосвязи между наблюдаемыми, или явными, переменными.




(a) Модель анализа главных компонент




(b) Модель факторного анализа

## 9.1. Анализ главных компонент и факторный анализ в R

Пакет psych

| Функции | Описание |
|---|---|
| `principal()` | Анализ главных компонент с возможностью поворота осей* |
| `fa()` | Факторный анализ методом главных осей, минимальных остатков, взвешенных наименьших квадратов или наибольшего правдоподобия |
| `fa.parallel()` | График собственных значений (scree plot) с параллельным анализом |
| `factor.plot()` | Графическое изображение результатов факторного анализа или анализа главных компонент |
| `fa.diagram()` | Графическое изображения матриц нагрузок факторного анализа или анализа главных компонент |
| `scree()` | График собственных значений для факторного анализа и анализа главных компонент |

**ШАГИ**

1. *Подготовить данные*. Результаты и PCA, и EFA выводятся из корреляций между наблюдаемыми переменными. Можно использовать в качестве аргументов функций principal() и fa() либо исходную таблицу данных, либо корреляционную матрицу.

2. *Выбрать факторную модель*. Нужно решить, что лучше подходит для исследовательских задач – PCA (снижение размерности данных) или EFA (обнаружение скрытой структуры).

3. *Решить, сколько компонент/факторов выделять.*

4. *Выделить компоненты/факторы.*

5. *Повернуть компоненты/факторы.*

6. *Интерпретировать результаты.*

7. *Вычислить значения компонент или факторов.*

## 9.2. Главные компоненты

***Главные компоненты*** – это линейные комбинации наблюдаемых переменных.

*Первая главная компонента* $PC_1 = a_1X_1 + a_2X_2 + \ldots + a_kX_k$ - это взвешенная комбинация $k$ наблюдаемых переменных, которая учитывает наибольшую дисперсию (долю изменчивости) исходного набора переменных.

*Вторая главная компонента* – это линейная комбинация, которая учитывает наибольшую дисперсию исходного набора переменных при условии, что она ортогональна первой главной компоненте (то есть не коррелирует с ней).

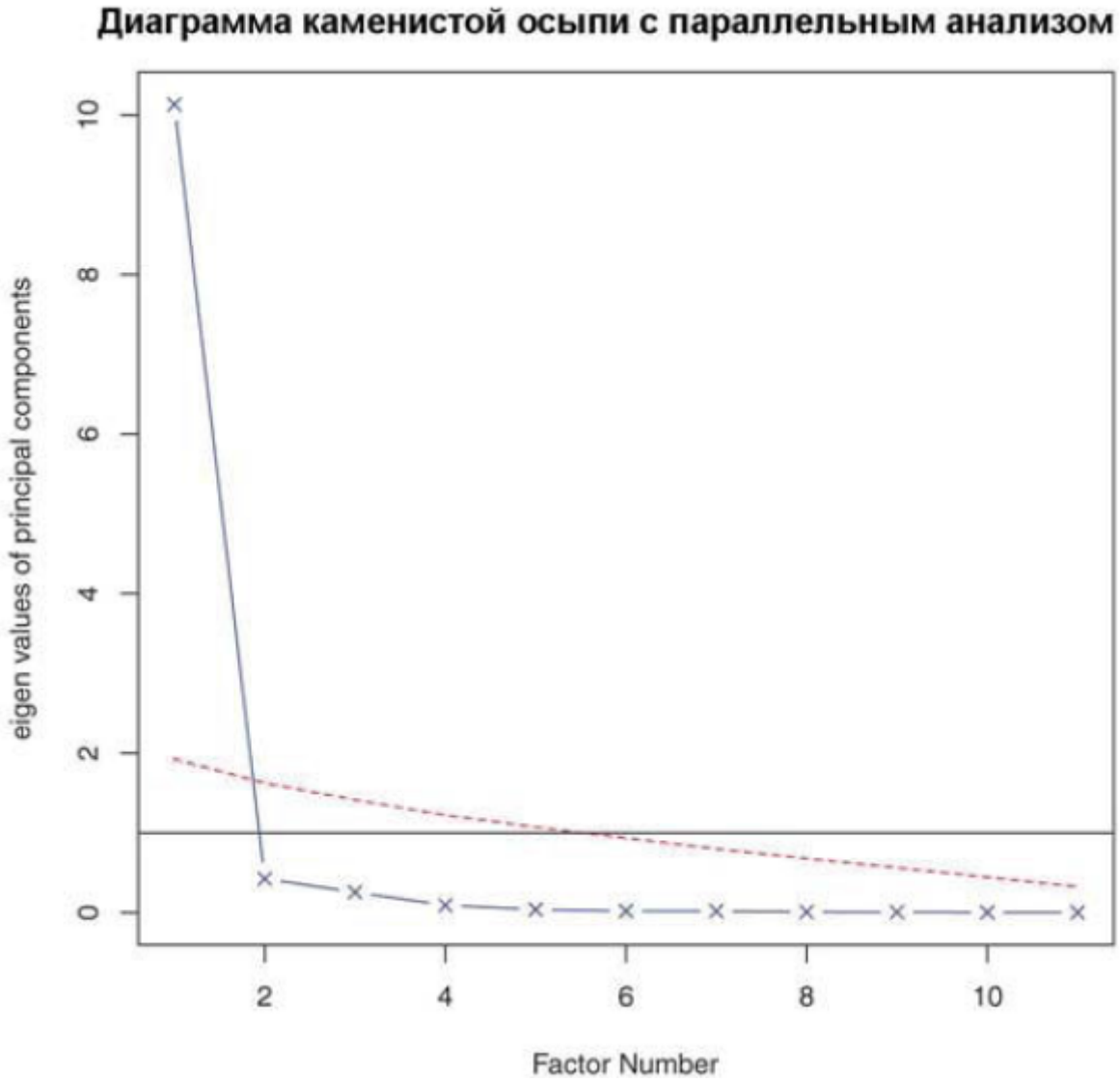
**ВЫБОР НЕОБХОДИМОГО ЧИСЛА КОМПОНЕНТ**

*Существует несколько критериев для определения числа компонент в PCA:*

- имеющийся опыт и теоретические соображения;
- объяснение заданной доли дисперсии исходных переменных (например, 80%);
- изучение собственных значений матрицы корреляций между всеми переменными.

*Каждая компонента связана с собственным значением корреляционной матрицы.* Первой главной компоненте (principal component, PC) соответствует наибольшее собственное значение, второй компоненте – второе по величине собственное значение и т. д. Согласно *критерию Кайзера-Харриса* (Kaiser-Harris), следует использовать компоненты, у которых собственные значения превышают единицу.

## 9.2. Главные компоненты



## 9.2. Главные компоненты

### ВЫДЕЛЕНИЕ ГЛАВНЫХ КОМПОНЕНТ

`principal(r, nfactors=, rotate=, scores=),`

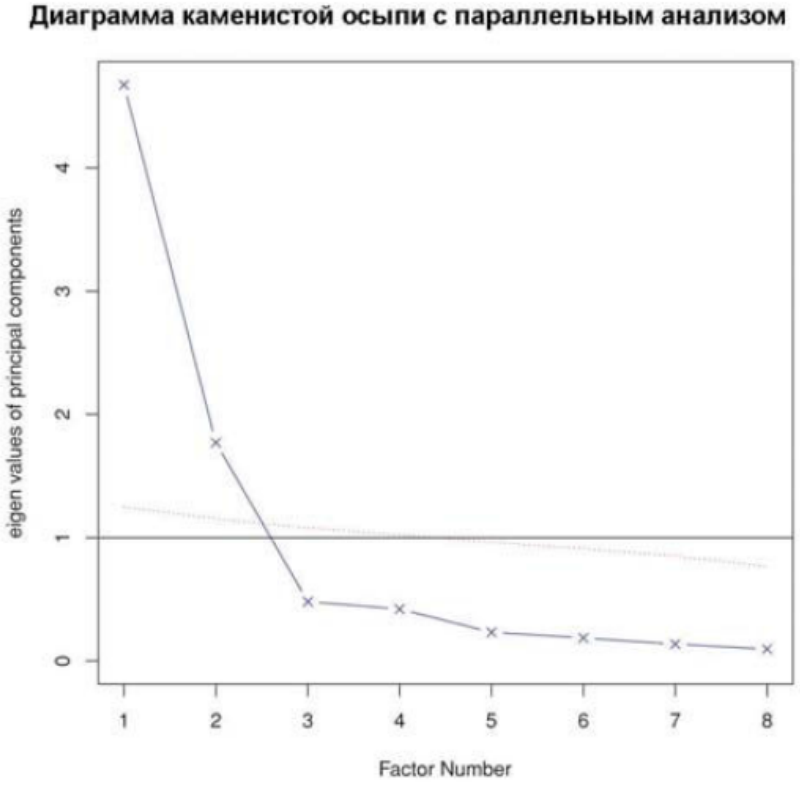
где `r` – корреляционная матрица, или исходная таблица данных; `nfactors` – определяет число главных компонент, которые нужно выделить (по умолчанию одна); `rotate` – указывает, какой тип вращения нужно применить (по умолчанию варимакс); `scores` – определяет, нужно ли рассчитывать значения главных компонент (по умолчанию – нет).

```
> pc <- principal(USJudgeRatings[,-1], nfactors=1)
> pc
Principal Components Analysis
Call: principal(r = USJudgeRatings[, -1], nfactors=1)
Standardized loadings based upon correlation matrix
      PC1   h2    u2
INTG 0.92 0.84 0.157
DMNR 0.91 0.83 0.166
DILG 0.97 0.94 0.061
CFMG 0.96 0.93 0.072
DECI 0.96 0.92 0.076
PREP 0.98 0.97 0.030
FAMI 0.98 0.95 0.047
ORAL 1.00 0.99 0.009
WRIT 0.99 0.98 0.020
PHYS 0.89 0.80 0.201
RTEN 0.99 0.97 0.028
                 PC1
SS loadings    10.13
Proportion Var  0.92
```

## 9.2. Главные компоненты

Промеры тела у девочек

```
library(psych)
fa.parallel(Harman23.cor$cov, n.obs=302, fa="pc", ntrials=100,
            show.legend=FALSE, main="Диаграмма каменистой осыпи
➥с параллельным анализом")
```



## 9.2. Главные компоненты

```
> library(psych)
> PC <- principal(Harman23.cor$cov, nfactors=2, rotate="none")
> PC
Principal Components Analysis
Call: principal(r = Harman23.cor$cov, nfactors = 2, rotate = "none")
Standardized loadings based upon correlation matrix
                 PC1   PC2   h2    u2
height          0.86 -0.37 0.88 0.123
arm.span        0.84 -0.44 0.90 0.097
forearm         0.81 -0.46 0.87 0.128
lower.leg       0.84 -0.40 0.86 0.139
weight          0.76  0.52 0.85 0.150
bitro.diameter 0.67  0.53 0.74 0.261
chest.girth     0.62  0.58 0.72 0.283
chest.width     0.67  0.42 0.62 0.375
                 PC1  PC2
SS loadings     4.67 1.77
Proportion Var 0.58 0.22
Cumulative Var 0.58 0.81
```

## ВРАЩЕНИЕ ГЛАВНЫХ КОМПОНЕНТ

Вращение – это набор математических приемов трансформации матрицы нагрузок компонент в другую, более легко интерпретируемую.

Способы вращения различаются по тому, остаются ли получившиеся компоненты нескоррелированными (*ортогональное вращение*) или же допускается их корреляция (*наклонное вращение*).

Наиболее распространенный тип ортогонального вращения – это ***варимакс*** (varimax), при котором делается попытка очистить столбцы матрицы нагрузок так, чтобы каждая компонента была определена ограниченным набором переменных (то есть в каждом столбце будет лишь несколько больших нагрузок и много очень малых).

## 9.2. Главные компоненты

```
> rc <- principal(Harman23.cor$cov, nfactors=2, rotate="varimax")
> rc
Principal Components Analysis
Call: principal(r = Harman23.cor$cov, nfactors = 2, rotate = "varimax")
Standardized loadings based upon correlation matrix
                 RC1  RC2   h2    u2
height          0.90 0.25 0.88 0.123
arm.span        0.93 0.19 0.90 0.097
forearm         0.92 0.16 0.87 0.128
lower.leg       0.90 0.22 0.86 0.139
weight          0.26 0.88 0.85 0.150
bitro.diameter 0.19 0.84 0.74 0.261
chest.girth     0.11 0.84 0.72 0.283
chest.width     0.26 0.75 0.62 0.375
                 RC1  RC2
SS loadings     3.52 2.92
Proportion Var 0.44 0.37
Cumulative Var 0.44 0.81
```

## 9.2. Главные компоненты

### ВЫЧИСЛЕНИЕ ЗНАЧЕНИЙ ГЛАВНЫХ КОМПОНЕНТ

```
> library(psych)
> rc <- principal(Harman23.cor$cov, nfactors=2, rotate="varimax")
> round(unclass(rc$weights), 2)
                 RC1   RC2
height          0.28 -0.05
arm.span        0.30 -0.08
forearm         0.30 -0.09
lower.leg       0.28 -0.06
weight         -0.06  0.33
bitro.diameter -0.08  0.32
chest.girth    -0.10  0.34
chest.width    -0.04  0.27
```

```
PC1 = 0.28*height + 0.30*arm.span + 0.30*forearm + 0.29*lower.leg -
      0.06*weight - 0.08*bitro.diameter - 0.10*chest.girth -
      0.04*chest.width
```

```
PC2 = -0.05*height - 0.08*arm.span - 0.09*forearm - 0.06*lower.leg +
       0.33*weight + 0.32*bitro.diameter + 0.34*chest.girth +
       0.27*chest.width
```

## 9.3. Разведочный факторный анализ

$$X_i = a_1F_1 + a_2F_2 + \ldots + a_pF_p + U_i,$$

где $X_i$ – это $i$-ая наблюдаемая переменная ($i$ = 1…$k$), $F_j$ – это общие факторы ($j$ = 1…$p$), и $p < k$. $U_i$ – это уникальная составляющая переменной $X_i$ (не объясненная общими факторами). Параметр $a_1$ можно интерпретировать как степень вклада каждого фактора в наблюдаемую переменную.

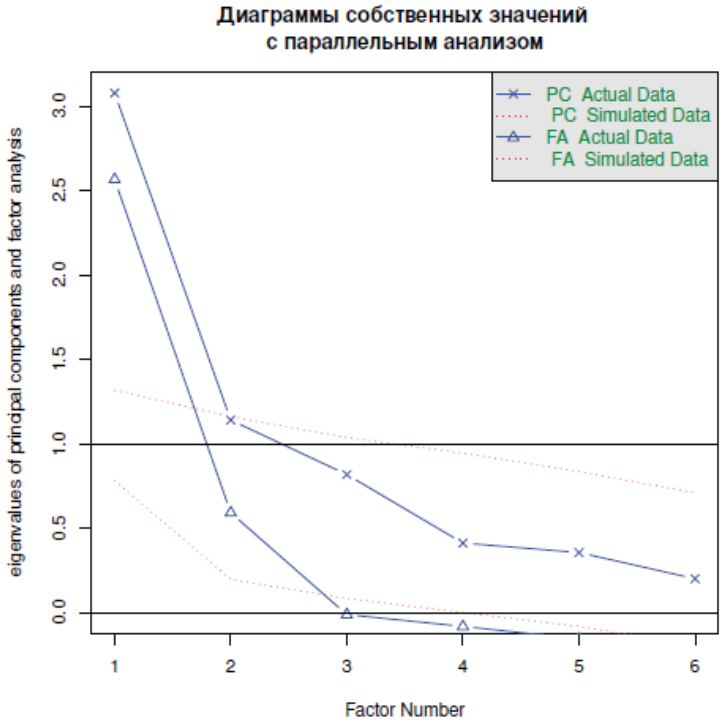
**Пример:** невербальная оценка общего умственного развития (general), тест на завершение фигур (picture), тест блочных конструкций (blocks), тест с лабиринтом (maze), тест на понимание прочитанного (reading) и тест на словарный запас (vocab).

```
> options(digits=2)
> covariances <- ability.cov$cov
> correlations <- cov2cor(covariances)
> correlations
        general picture blocks maze reading vocab
general    1.00    0.47   0.55 0.34    0.58  0.51
picture    0.47    1.00   0.57 0.19    0.26  0.24
blocks     0.55    0.57   1.00 0.45    0.35  0.36
maze       0.34    0.19   0.45 1.00    0.18  0.22
reading    0.58    0.26   0.35 0.18    1.00  0.79
vocab      0.51    0.24   0.36 0.22    0.79  1.00
```

## 9.3. Разведочный факторный анализ

### ОПРЕДЕЛЕНИЕ ЧИСЛА ИЗВЛЕКАЕМЫХ ФАКТОРОВ

```
> library(psych)
> covariances <- ability.cov$cov
> correlations <- cov2cor(covariances)
> fa.parallel(correlations, n.obs=112, fa="both", n.iter=100,
              main=" Диаграммы собственных значений с параллельным
➥анализом")
```



## 9.3. Разведочный факторный анализ

### ВЫДЕЛЕНИЕ ОБЩИХ ФАКТОРОВ

```
fa(r, nfactors=, n.obs=, rotate=, scores=, fm=),
```

где `r` – это корреляционная матрица или таблица исходных данных; `nfactors` определяет число факторов, которое нужно выделить (1 по умолчанию); `n.obs` – число наблюдений (если анализируется корреляционная матрица); `rotate` определяет тип вращения факторов (по умолчанию облимин, `oblimin`); `scores` указывает, нужно ли вычислять значения факторов (по умолчанию – нет); `fm` задает метод факторного анализа (по умолчанию минрез, `minres`).

```
> fa <- fa(correlations, nfactors=2, rotate="none", fm="pa")
> fa
Factor Analysis using method =  pa
Call: fa(r = correlations, nfactors = 2, rotate = "none", fm = "pa")
Standardized loadings based upon correlation matrix
          PA1   PA2   h2   u2
general 0.75  0.07 0.57 0.43
picture 0.52  0.32 0.38 0.62
blocks  0.75  0.52 0.83 0.17
maze    0.39  0.22 0.20 0.80
reading 0.81 -0.51 0.91 0.09
vocab   0.73 -0.39 0.69 0.31
                PA1  PA2
SS loadings    2.75 0.83
Proportion Var 0.46 0.14
Cumulative Var 0.46 0.60
```

## 9.3. Разведочный факторный анализ

### ВРАЩЕНИЕ ФАКТОРОВ

Ортогональное

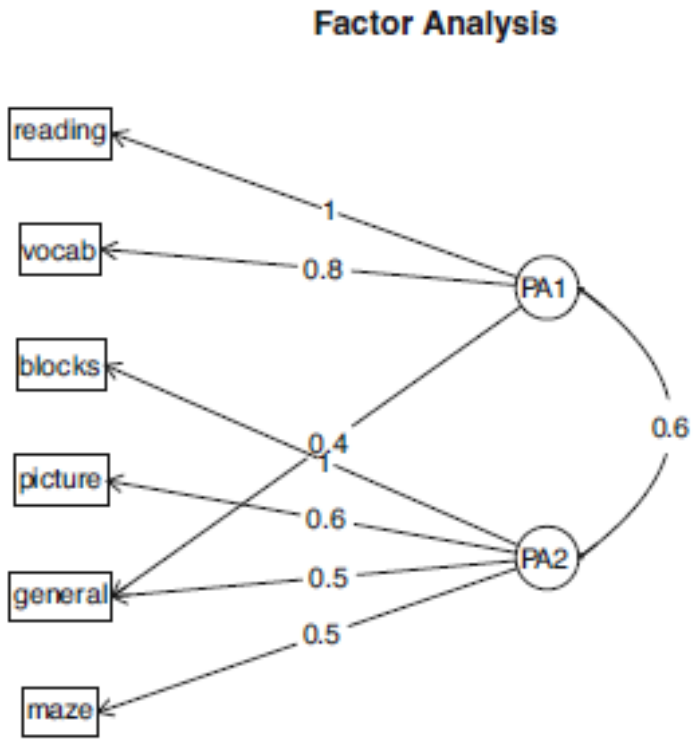
```
> fa.varimax <- fa(correlations, nfactors=2, rotate="varimax", fm="pa")
> fa.varimax
Factor Analysis using method =  pa
Call: fa(r = correlations, nfactors = 2, rotate = "varimax", fm = "pa")
Standardized loadings based upon correlation matrix
          PA1  PA2   h2   u2
general 0.49 0.57 0.57 0.43
picture 0.16 0.59 0.38 0.62
blocks  0.18 0.89 0.83 0.17
maze    0.13 0.43 0.20 0.80
reading 0.93 0.20 0.91 0.09
vocab   0.80 0.23 0.69 0.31
                PA1  PA2
SS loadings    1.83 1.75

Proportion Var 0.30 0.29
Cumulative Var 0.30 0.60
```

Наклонное

```
> fa.promax <- fa(correlations, nfactors=2, rotate="promax", fm="pa")
> fa.promax
Factor Analysis using method =  pa
Call: fa(r = correlations, nfactors = 2, rotate = "promax", fm = "pa")
Standardized loadings based upon correlation matrix
           PA1   PA2   h2   u2
general  0.36  0.49 0.57 0.43
picture -0.04  0.64 0.38 0.62
blocks  -0.12  0.98 0.83 0.17
maze    -0.01  0.45 0.20 0.80
reading  1.01 -0.11 0.91 0.09
vocab    0.84 -0.02 0.69 0.31
                PA1  PA2
SS loadings    1.82 1.76
Proportion Var 0.30 0.29
Cumulative Var 0.30 0.60
 With factor correlations of
     PA1  PA2
PA1 1.00 0.57
PA2 0.57 1.00
```

## 9.3. Разведочный факторный анализ

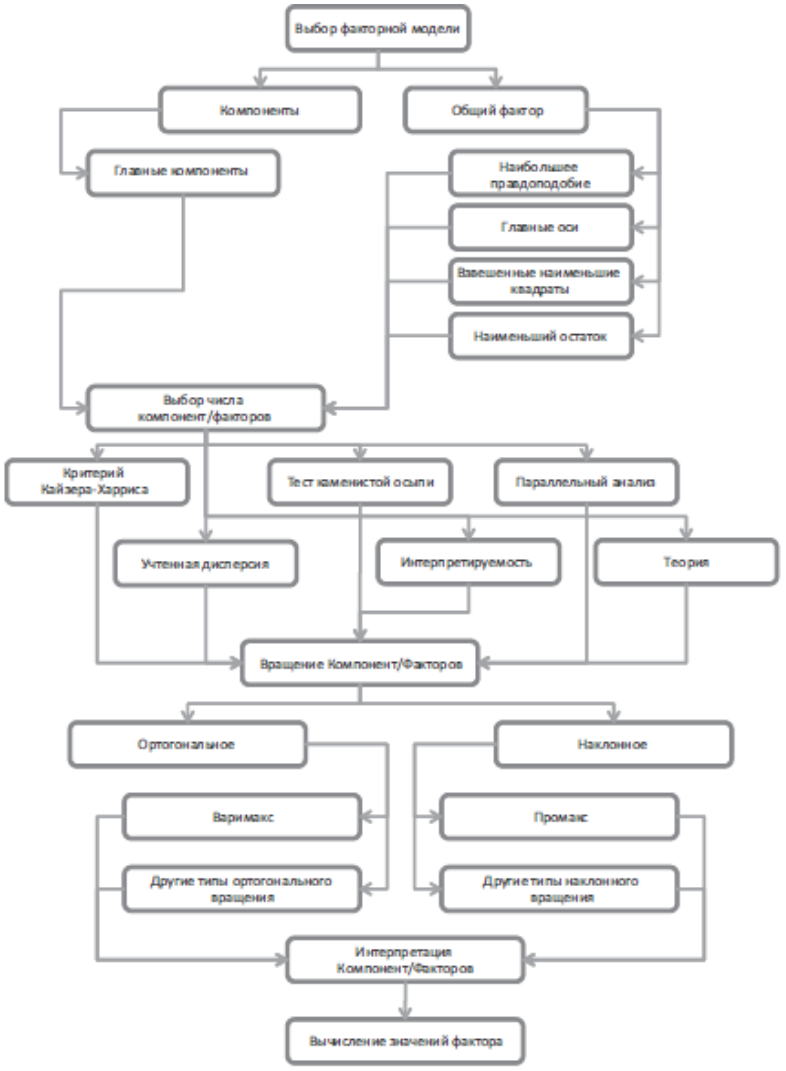
```
fa.diagram(fa.promax, simple=FALSE)
```



# 9.3. Разведочный факторный анализ

## ЗНАЧЕНИЯ ФАКТОРОВ

```
> fa.promax$weights
           [,1]  [,2]
general 0.080 0.210
picture 0.021 0.090
blocks  0.044 0.695
maze    0.027 0.035
reading 0.739 0.044
vocab   0.176 0.039
```



## 10.1. Задачи кластерного анализа

Задача кластеризации (обучения без учителя) заключается в следующем. Имеется обучающая выборка $\mathbf{X}^\ell = \{x_1, \ldots, x_\ell\} \subset \mathbf{X}$ и функция расстояния между объектами $\rho(x, x')$. Требуется разбить выборку на непересекающиеся подмножества, называемые *кластерами*, так, чтобы каждый кластер состоял из объектов, близких по метрике $\rho$, а объекты разных кластеров существенно отличались. При этом каждому объекту $x_i \in \mathbf{X}^\ell$ приписывается метка (номер) кластера $y_i$.

Алгоритм кластеризации — это функция $a: \mathbf{X} \to \mathbf{Y}$, которая любому объекту $x \in \mathbf{X}$ ставит в соответствие метку кластера $y \in \mathbf{Y}$. Множество меток $\mathbf{Y}$ в некоторых случаях известно заранее, однако чаще ставится задача определить оптимальное число кластеров, с точки зрения того или иного критерия качества кластеризации.

Решение задачи кластеризации принципиально неоднозначно:

- не существует однозначно наилучшего критерия качества кластеризации,
- число кластеров неизвестно заранее и устанавливается в соответствии с некоторым субъективным критерием,
- результат кластеризации существенно зависит от метрики $\rho$.

## 10.1. Задачи кластерного анализа

Цели кластеризации могут быть различными в зависимости от особенностей конкретной прикладной задачи:

- Понять структуру множества объектов $\mathbf{X}^\ell$, разбив его на группы схожих объектов. Упростить дальнейшую обработку данных и принятия решений, работая с каждым кластером по отдельности (стратегия «разделяй и властвуй»).
- Сократить объём хранимых данных в случае сверхбольшой выборки $\mathbf{X}^\ell$, оставив по одному наиболее типичному представителю от каждого кластера.
- Выделить нетипичные объекты, которые не подходят ни к одному из кластеров. Эту задачу называют одноклассовой классификацией, обнаружением нетипичности или новизны (novelty detection).

## 10.2. Эвристические графовые алгоритмы

Вершинам графа соответствуют объекты выборки, а рёбрам — попарные расстояния между объектами $\rho_{ij} = \rho(x_i, x_j)$.

**Алгоритм выделения связных компонент.** Задаётся параметр **R** и в графе удаляются все рёбра **(i, j)**, для которых $\rho_{ij} > R$. Соединёнными остаются только наиболее близкие пары объектов. Идея алгоритма заключается в том, чтобы подобрать такое значение $R \in [\min \rho_{ij}, \max \rho_{ij}]$, при котором граф развалится на несколько связных компонент. Найденные связные компоненты — и есть кластеры.

*Связной компонентой графа* называется подмножество его вершин, в котором любые две вершины можно соединить путём, целиком лежащим в этом подмножестве. Для поиска связных компонент можно использовать стандартные алгоритмы поиска в ширину (алгоритм Дейкстры) или поиска в глубину.

Недостатки:

- ограниченная применимость,
- плохая управляемость числом кластеров.

## 10.2. Эвристические графовые алгоритмы

**Алгоритм кратчайшего незамкнутого пути** строит граф из **ℓ−1** рёбер так, чтобы они соединяли все **ℓ** точек и обладали минимальной суммарной длиной. Такой граф называется *кратчайшим незамкнутым путём*, минимальным покрывающим деревом или каркасом.

**Алгоритм 1.1.** Алгоритм кратчайшего незамкнутого пути (КНП)

1: Найти пару точек $(i, j)$ с наименьшим $\rho_{ij}$ и соединить их ребром;
2: **пока** в выборке остаются изолированные точки
3: найти изолированную точку, ближайшую к некоторой неизолированной;
4: соединить эти две точки ребром;
5: удалить $K - 1$ самых длинных рёбер;

Недостатки:

- ограниченная применимость,
- высокая трудоёмкость.

## Алгоритм FOREL (ФОРмальный ЭЛемент)

**Алгоритм 1.2.** Алгоритм FOREL

1: Инициализировать множество некластеризованных точек:
$U := X^\ell$;
2: **пока** в выборке есть некластеризованные точки, $U \neq \varnothing$:
3: взять произвольную точку $x_0 \in U$ случайным образом;
4: **повторять**
5: образовать кластер — сферу с центром в $x_0$ и радиусом $R$:
$K_0 := \{x_i \in U \mid \rho(x_i, x_0) \leqslant R\}$;
6: поместить центр сферы в центр масс кластера:
$x_0 := \frac{1}{|K_0|} \sum_{x_i \in K_0} x_i$;
7: **пока** центр $x_0$ не стабилизируется;
8: пометить все точки $K_0$ как кластеризованные:
$U := U \setminus K_0$;
9: применить алгоритм КНП к множеству центров всех найденных кластеров;
10: каждый объект $x_i \in X^\ell$ приписать кластеру с ближайшим центром;

Недостатки:

- чувствителен к выбору начального положения точки $\mathbf{x_0}$ для каждого нового кластера.

## 10.3. Функционалы качества кластеризации

Задачу кластеризации можно ставить как задачу дискретной оптимизации: необходимо так приписать номера кластеров $\mathbf{y_i}$ объектам $\mathbf{x_i}$, чтобы значение выбранного функционала качества приняло наилучшее значение.

- **Среднее внутрикластерное расстояние должно быть как можно меньше:**

$$F_0 = \frac{\sum_{i<j}[y_i = y_j]\rho(x_i, x_j)}{\sum_{i<j}[y_i = y_j]} \to \min .$$

- **Среднее межкластерное расстояние должно быть как можно больше:**

$$F_1 = \frac{\sum_{i<j}[y_i \neq y_j]\rho(x_i, x_j)}{\sum_{i<j}[y_i \neq y_j]} \to \max .$$

- **Сумма средних внутрикластерных расстояний должна быть как можно меньше:**

$$\Phi_0 = \sum_{y \in Y} \frac{1}{|K_y|} \sum_{i:\, y_i = y} \rho^2(x_i, \mu_y) \to \min$$

- **Сумма межкластерных расстояний должна быть как можно больше:**

$$\Phi_1 = \sum_{y \in Y} \rho^2(\mu_y, \mu) \to \max$$

## 10.4. Статистические алгоритмы

Статистические алгоритмы основаны на предположении, что кластеры неплохо описываются некоторым семейством вероятностных распределений. Тогда задача кластеризации сводится к разделению смеси распределений по конечной выборке.

**EM-алгоритм**

*Гипотеза о вероятностной природе данных.* Объекты выборки $\mathbf{X^\ell}$ появляются случайно и независимо согласно вероятностному распределению, представляющему собой смесь распределений

$$p(x) = \sum_{y \in Y} w_y p_y(x), \qquad \sum_{y \in Y} w_y = 1,$$

где $\mathbf{p_y(x)}$ – функция плотности распределения кластера $\mathbf{y}$, $\mathbf{w_y}$ – неизвестная априорная вероятность появления объектов из кластера $\mathbf{y}$.

*Гипотеза о форме кластеров.* Объекты описываются $\mathbf{n}$ числовыми признаками $\mathbf{f_1(x),...,f_n(x)}$, $\mathbf{X \in R^n}$. Каждый кластер $\mathbf{y \in Y}$ описывается $\mathbf{n}$-мерной гауссовской плотностью $\mathbf{p_y(x) = N(x; \mu_y, \Sigma_y)}$ с центром $\mathbf{\mu_y = (\mu_{y1},...,\mu_{yn})}$ и диагональной ковариационной матрицей $\mathbf{\Sigma_y = diag(\sigma^2_{y1},..., \sigma^2_{yn})}$.

На E-шаге по формуле Байеса вычисляются скрытые переменные $\mathbf{g_{iy}}$. Значение $\mathbf{g_{iy}}$ равно вероятности того, что объект $\mathbf{x_i \in X^\ell}$ принадлежит кластеру $\mathbf{y \in Y}$. На M-шаге уточняются параметры каждого кластера $\mathbf{(\mu y, \Sigma y)}$, при этом существенно используются скрытые переменные $\mathbf{g_{iy}}$.

## 10.4. Статистические алгоритмы

**Алгоритм 1.3.** Кластеризация с помощью EM-алгоритма

начальное приближение для всех кластеров $y \in Y$:
$w_y := 1/|Y|$;
$\mu_y :=$ случайный объект выборки;
$\sigma^2_{yj} := \frac{1}{\ell|Y|} \sum_{i=1}^{\ell} (f_j(x_i) - \mu_{yj})^2, \;\; j = 1, \ldots, n$;
**повторять**
  E-шаг (expectation):
  $$g_{iy} := \frac{w_y p_y(x_i)}{\sum_{z \in Y} w_z p_z(x_i)}, \;\; y \in Y, \;\; i = 1, \ldots, \ell;$$
  M-шаг (maximization):
  $$w_y := \frac{1}{\ell} \sum_{i=1}^{\ell} g_{iy}, \;\; y \in Y;$$
  $$\mu_{yj} := \frac{1}{\ell w_y} \sum_{i=1}^{\ell} g_{iy} f_j(x_i), \;\; y \in Y, \;\; j = 1, \ldots, n;$$
  $$\sigma^2_{yj} := \frac{1}{\ell w_y} \sum_{i=1}^{\ell} g_{iy} (f_j(x_i) - \mu_{yj})^2, \;\; y \in Y, \;\; j = 1, \ldots, n;$$
  Отнести объекты к кластерам по байесовскому решающему правилу:
  $y_i := \arg\max_{y \in Y} g_{iy}, \;\; i = 1, \ldots, \ell$;
**пока** $y_i$ не перестанут изменяться;

## 10.4. Статистические алгоритмы

**Алгоритм 1.4.** Кластеризация с помощью алгоритма $k$-средних

1: сформировать начальное приближение центров всех кластеров $y \in Y$:
$\mu_y$ — наиболее удалённые друг от друга объекты выборки;
2: **повторять**
3: отнести каждый объект к ближайшему центру (аналог E-шага):
$y_i := \arg\min_{y \in Y} \rho(x_i, \mu_y), \quad i = 1, \ldots, \ell;$
4: вычислить новое положение центров (аналог M-шага):
$\mu_{yj} := \dfrac{\sum_{i=1}^{\ell} [y_i = y] f_j(x_i)}{\sum_{i=1}^{\ell} [y_i = y]}, \quad y \in Y, \quad j = 1, \ldots, n;$
5: **пока** $y_i$ не перестанут изменяться;

Алгоритм k-means крайне чувствителен к выбору начальных приближений центров. Случайная инициализация центров на шаге 1 может приводить к плохим кластеризациям. Для формирования начального приближения лучше выделить k наиболее удалённых точек выборки: первые две точки выделяются по максимуму всех попарных расстояний; каждая следующая точка выбирается так, чтобы расстояние от неё до ближайшей уже выделенной было максимально.

## 10.5. Иерархическая кластеризация

---

Иерархические алгоритмы кластеризации, называемые также алгоритмами *таксономии*, строят не одно разбиение выборки на непересекающиеся классы, а систему вложенных разбиений. Результат таксономии обычно представляется в виде таксономического дерева - *дендрограммы*.

- ***Дивизимные*** или нисходящие алгоритмы разбивают выборку на всё более и более мелкие кластеры.
- ***Агломеративные*** или восходящие алгоритмы, в которых объекты объединяются во всё более и более крупные кластеры.

Сначала каждый объект считается отдельным кластером. Для одноэлементных кластеров естественным образом определяется функция расстояния

$$R(\{x\},\{x'\}) = \rho(x,x').$$

Затем запускается процесс слияний. На каждой итерации вместо пары самых близких кластеров **U** и **V** образуется новый кластер **W = U ∪ V**. Расстояние от нового кластера **W** до любого другого кластера S вычисляется по расстояниям **R(U, V ), R(U, S)** и **R(V, S),** которые к этому моменту уже должны быть известны:

$$R(U \cup V, S) = \alpha_U R(U,S) + \alpha_V R(V,S) + \beta R(U,V) + \gamma |R(U,S) - R(V,S)|$$

## 10.5. Иерархическая кластеризация

---

*Расстояние ближнего соседа:*

$$R^{\text{б}}(W,S) = \min_{w\in W, s\in S} \rho(w,s); \qquad \alpha_U = \alpha_V = \tfrac{1}{2},\ \beta = 0,\ \gamma = -\tfrac{1}{2}.$$

*Расстояние дальнего соседа:*

$$R^{\text{д}}(W,S) = \max_{w\in W, s\in S} \rho(w,s); \qquad \alpha_U = \alpha_V = \tfrac{1}{2},\ \beta = 0,\ \gamma = \tfrac{1}{2}.$$

*Среднее расстояние:*

$$R^{\text{с}}(W,S) = \frac{1}{|W||S|}\sum_{w\in W}\sum_{s\in S}\rho(w,s); \qquad \alpha_U = \frac{|U|}{|W|},\ \alpha_V = \frac{|V|}{|W|},\ \beta = \gamma = 0.$$

*Расстояние между центрами:*

$$R^{\text{ц}}(W,S) = \rho^2\Big(\sum_{w\in W}\frac{w}{|W|}, \sum_{s\in S}\frac{s}{|S|}\Big); \qquad \alpha_U = \frac{|U|}{|W|},\ \alpha_V = \frac{|V|}{|W|},\ \beta = -\alpha_U\alpha_V,\ \gamma = 0.$$

*Расстояние Уорда:*

$$R^{\text{у}}(W,S) = \frac{|S||W|}{|S|+|W|}\rho^2\Big(\sum_{w\in W}\frac{w}{|W|}, \sum_{s\in S}\frac{s}{|S|}\Big); \quad \alpha_U = \frac{|S|+|U|}{|S|+|W|},\ \alpha_V = \frac{|S|+|V|}{|S|+|W|},\ \beta = \frac{-|S|}{|S|+|W|},\ \gamma = 0.$$

**Алгоритм 1.5.** Агломеративная кластеризация Ланса-Уильямса

1: инициализировать множество кластеров $C_1$:
$t := 1; \quad C_t = \{\{x_1\}, \ldots, \{x_\ell\}\};$
2: **для всех** $t = 2, \ldots, \ell$ ($t$ — номер итерации):
3: найти в $C_{t-1}$ два ближайших кластера:
$(U, V) := \arg\min_{U \neq V} R(U, V);$
$R_t := R(U, V);$
4: изъять кластеры $U$ и $V$, добавить слитый кластер $W = U \cup V$:
$C_t := C_{t-1} \cup \{W\} \setminus \{U, V\};$
5: **для всех** $S \in C_t$
6: вычислить расстояние $R(W, S)$ по формуле Ланса-Уильямса;

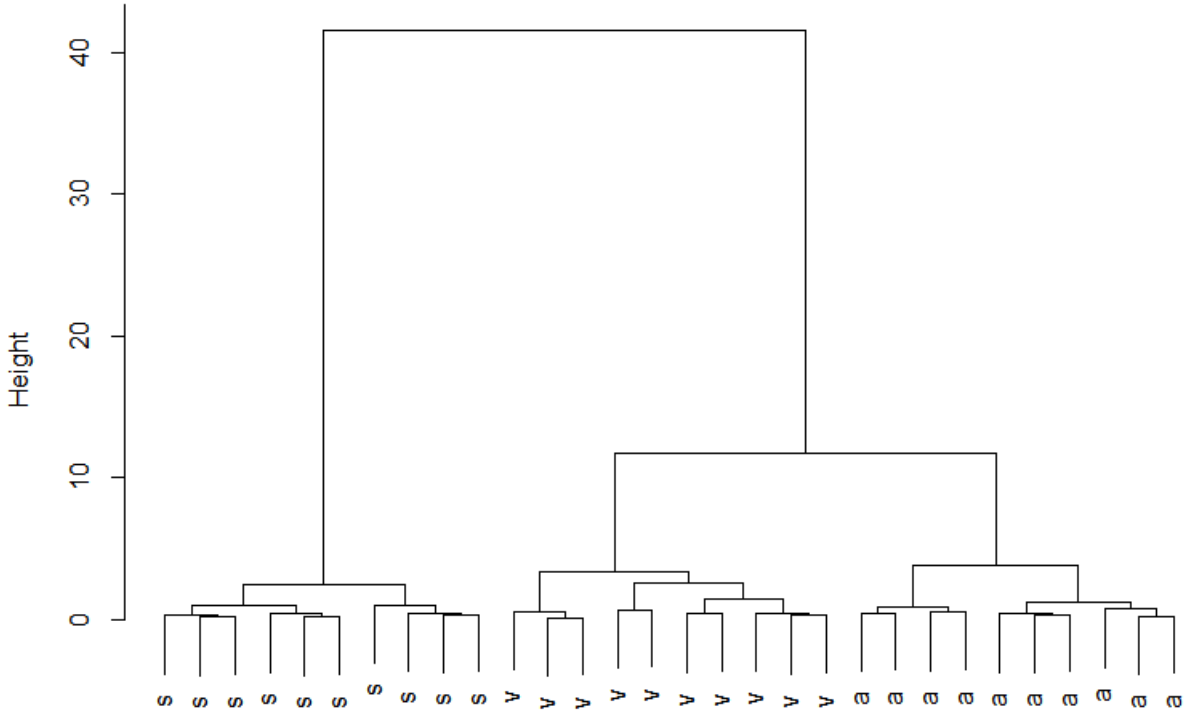
## 10.6. Определение числа кластеров

На горизонтальной оси находится интервал максимальной длины $|R_{t+1}-R_t|$, и в качестве результирующей кластеризации выдаётся множество кластеров $C_t$. Число кластеров равно $K = \ell - t + 1$. При необходимости можно задать ограничение на минимальное и максимальное число кластеров $K_0 \le K \le K_1$ и выбирать $t$, удовлетворяющие ограничениям $\ell - K1 + 1 \le t \le \ell - K0 + 1$.

Во многих прикладных задачах интерес представляет таксономическое дерево целиком, и определять оптимальное число кластеров не имеет особого смысла.

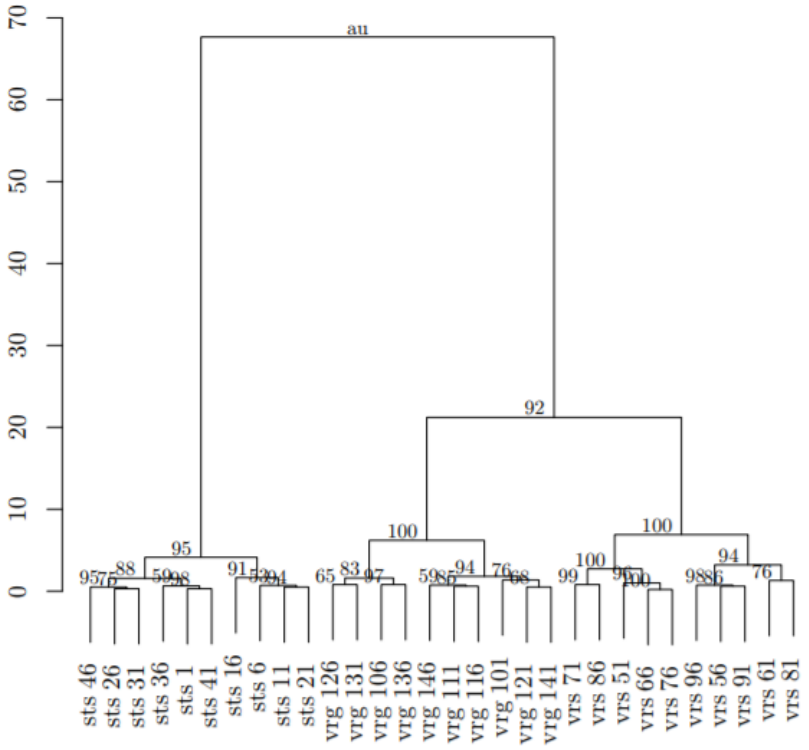
## 10.7. Кластеризация в R

```
> library(cluster)
> iriss <- iris[seq(1,nrow(iris),5),]
> iriss.dist <- daisy(iriss[,1:4])
> iriss.h <- hclust(iriss.dist, method="ward")
> plot(iriss.h, labels=abbreviate(iriss[,5],1, method="both.sides"), main="")
```



## 10.7. Кластеризация в R

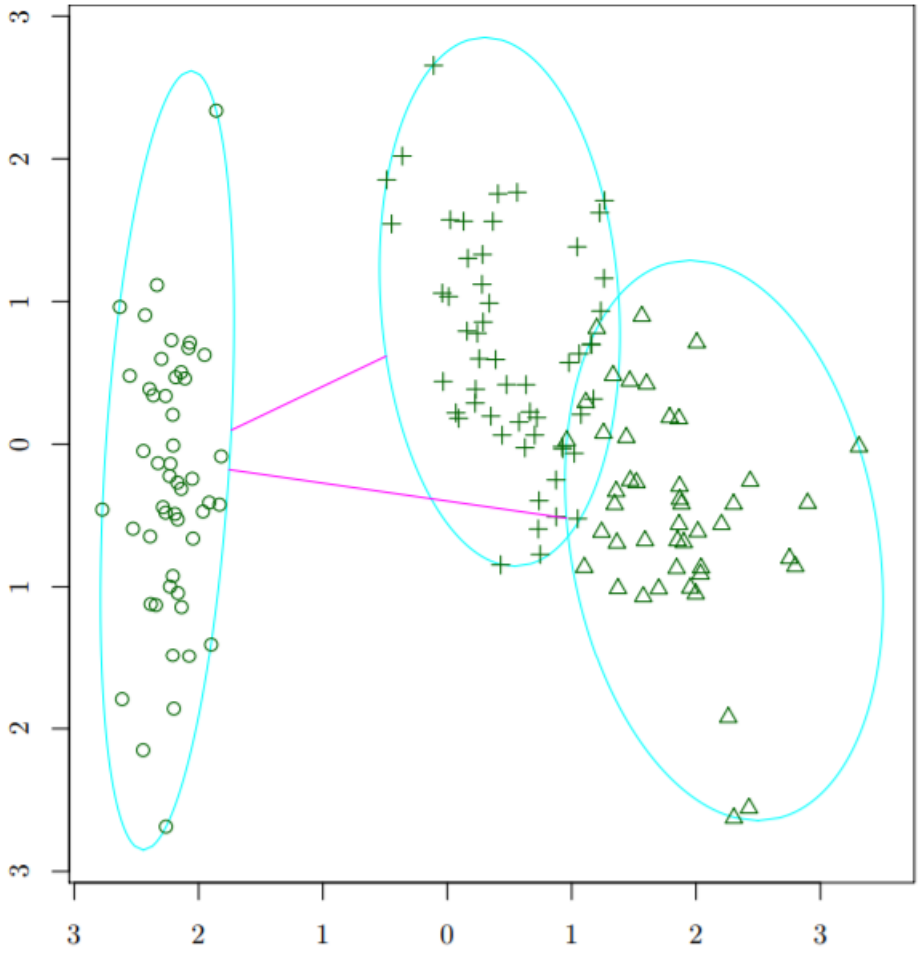
```
> library(pvclust)
> irisst <- t(iriss[,1:4])
> colnames(irisst) <- paste(abbreviate(iriss[,5], 3),  colnames(irisst))
> iriss.pv<- pvclust(irisst, method.dist="manhattan", method.hclust="ward", nboot=100)
```



## 10.7. Кластеризация в R

```
> eq <- read.table("data/eq.txt", h=TRUE)
> eq.k <- kmeans(eq[,-1], 2)
> table(eq.k$cluster, eq$SPECIES)
        arvense         fluviatile
1       37              5
2       1               41

> iris.f <- fanny(iris[,1:4], 3)
> plot(iris.f, which=1, main="")
> head(data.frame(sp=iris[,5], iris.f$membership))
        sp              X1              X2              X3
1       setosa          0.9142273       0.03603116      0.04974153
2       setosa          0.8594576       0.05854637      0.08199602
3       setosa          0.8700857       0.05463714      0.07527719
4       setosa          0.8426296       0.06555926      0.09181118
5       setosa          0.9044503       0.04025288      0.05529687
6       setosa          0.7680227       0.09717445      0.13480286
```

## 10.7. Кластеризация в R



## Библиографический список

**Кабаков Р. К.** (2014) R в действии. Анализ и визуализация данных на языке R Издательство: ДМК Пресс, 580 с.

**Шипунов А. Б., Балдин Е. М., Волкова П. А., Коробейников А. И., Назарова С. А., Петров С. В., Суфиянов В. Г.** (2012) Наглядная статистика. Используем R! - М.: ДМК Пресс, 298 с.

**Мастицкий С. Э., Шитиков В. К.** (2014) Статистический анализ и визуализация данных с помощью R. - Электронная книга, 400 с

**Лекции К.В. Воронцова по алгоритмам кластеризации и многомерному шкалированию**