Основы программирования в R

Визуализация данных в R: часть 2

Алла Тамбовцева, НИУ ВШЭ

Содержание

Распределение количественных показателей и проверка распределения на нормальность

Иногда в процессе анализа данных мы сталкиваемся с необходимостью определить тип распределения данных. Решить эту задачу непросто: нет такого универсального статистического теста, который позволил бы однозначно определить тип распределения, за исключением случаев, когда оно является нормальным. Но распределение данных можно сравнить с нормальным распределением. Требование нормальности распределения данных лежит в основе некоторых статистических тестов и моделей; плюс, при визуальном сравнении с нормальным распределением удобно отмечать всякие особенности распределения (скошенность, наличие «длинных хвостов» и прочее).

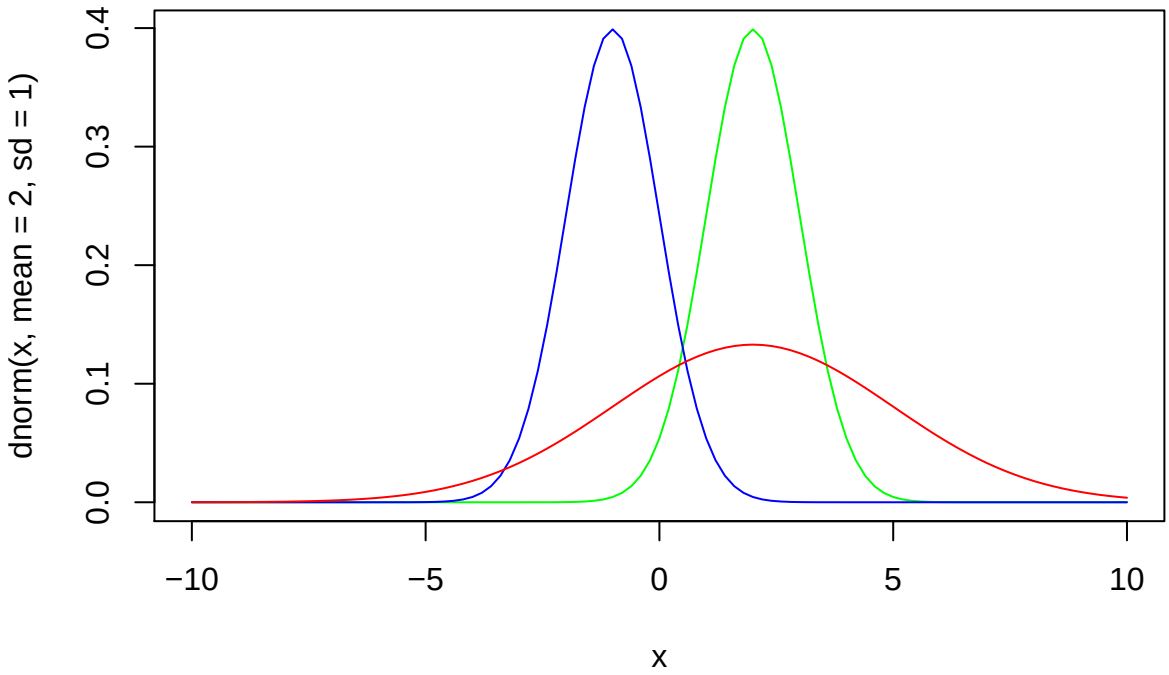
Начнем с визуального анализа. Например, наложим на гистограмму, построенную для показателя, график плотности нормального распределения с соответствующими параметрами.

Напоминание 1. О графике плотности распределения можно думать как о «сглаженной» гистограмме с большим числов столбцов.

Напоминание 2. Нормальное распределение задается двумя параметрами: математическим ожиданием и стандартным отклонением. Математическое ожидание отвечает за среднее значение распределения (значение, относительно которого симметричен график плотности распределения), стандартное отклонение — за разброс значений вокруг среднего.

Для примера построим графики плотности нормального распределения с разными параметрами. Нам потребуются две функции: `dnorm()` и `curve()`. Функция `dnorm()` возвращает значение функции плотности нормального распределения со средним `mean` и стандартным отклонением `sd` в точке, а функция `curve()` строит график функции, поданной на вход. Чтобы все графики были построены с одинаковым масштабом и были сравнимы, зафиксируем границы значений по оси Ox, записав их внутри аргумента `xlim`.

```
# add = TRUE - чтобы добавлять графики к уже нарисованным

curve(dnorm(x, mean = 2, sd = 1), xlim = c(-10, 10), col = "green" )
curve(dnorm(x, mean = -1, sd = 1), xlim = c(-10, 10), col = "blue", add = TRUE)
curve(dnorm(x, mean = 2, sd = 3), xlim = c(-10, 10), col = "red", add = TRUE)
```



Теперь попробуем совместить на графике гистограмму и график плотности нормального распределения с соответствующими параметрами. Загрузим файл с данными проекта *Comparative Political Data Set*. Полное описание данных можно найти здесь, в официальном кодбуке (*codebook*).

```
cp <- read.csv("https://allatambov.github.io/rprog/data/CPDS.csv", dec = ",")
```

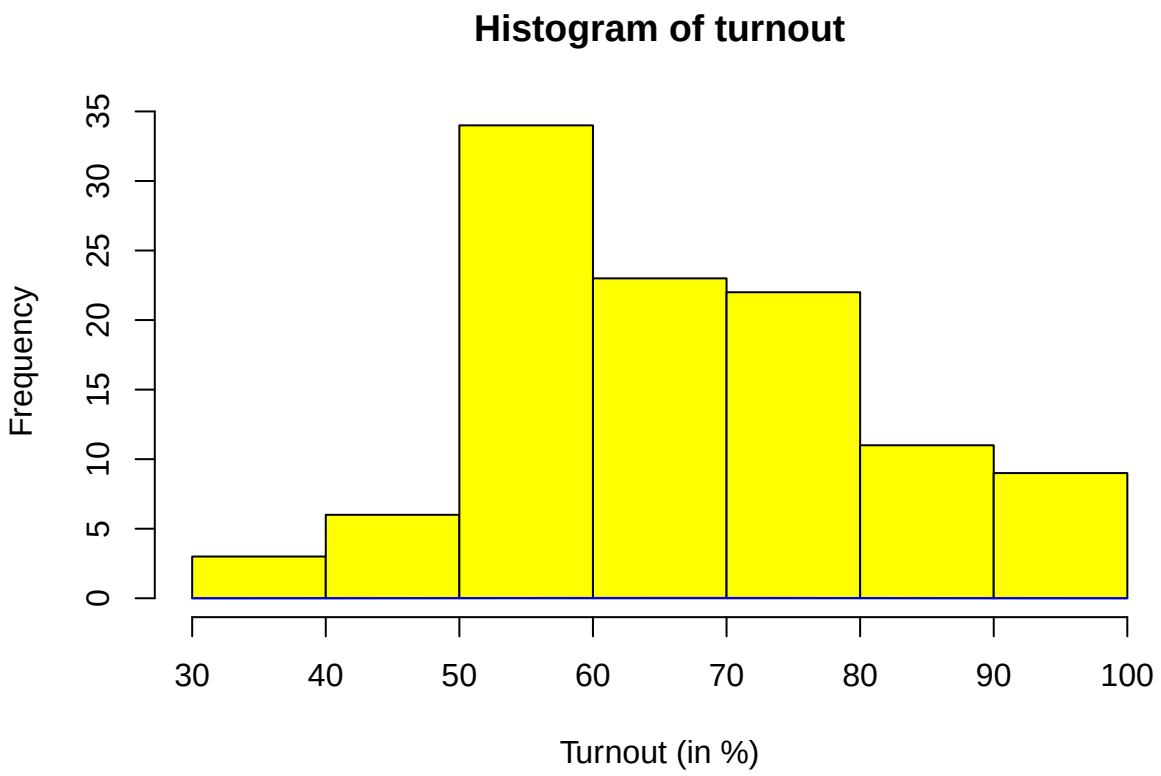
В функции выше мы добавили аргумент `dec=","`, чтобы учесть тот факт, что в качестве десятичного разделителя используется запятая.

Выберем данные, начиная с 2014 года:

```
cp <- cp[cp$year >= 2014, ]
```

Построим гистограмму для показателя `vturn` (явка на выборы) и наложим на неё график плотности нормального распределения с соответствующими параметрами. Какие параметры считать соответствующими? Среднее значение, равное среднему значению показателя `vturn`, и стандартное отклонение, равное стандартному отклонению `vturn`.

```
hist(cp$vturn,
     main = "Histogram of turnout",
     xlab = "Turnout (in %)",
     col = "yellow1")

curve(dnorm(x, mean = mean(cp$vturn),
            sd = sd(cp$vturn)),
            col = "blue", add = TRUE)
```



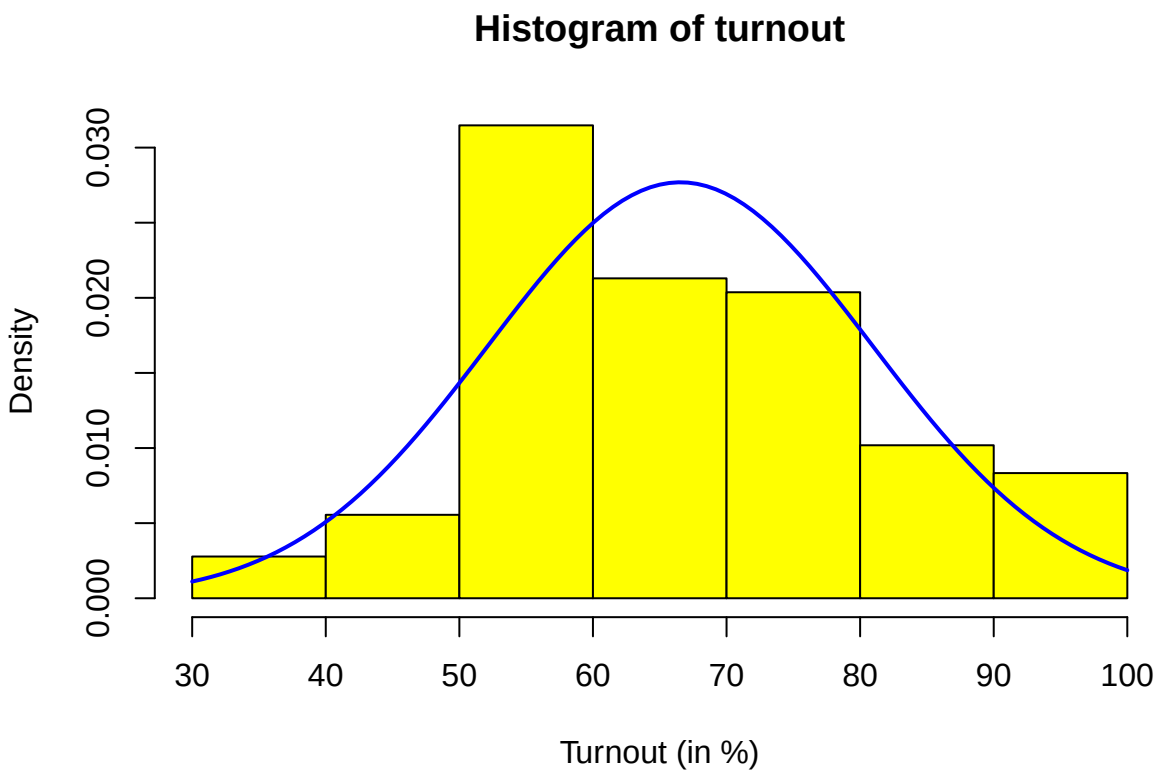
Кажется, ничего не произошло! На самом деле, кривая графика плотности на график добавилась, но её не видно из-за масштаба. Высота каждого столбика гистограммы — это частота, с которой значение в выборке попадает в определённый промежуток. А «высота» графика плотности — максимальное значение функции плотности. Чему оно равно?

```
# просто потому что максимальное значение
# достигается в точке x = mean
dnorm(mean(cp$vturn),
      mean = mean(cp$vturn),
      sd = sd(cp$vturn))
```

```
## [1] 0.02769138
```

Чтобы исправить ситуацию, добавим в функцию `hist()` аргумент `freq=FALSE`, и тогда на графике вместо абсолютных частот будут отображаться нормированные, сравнимые со значениями функции плотности:

```
hist(cp$vturn,
     main = "Histogram of turnout",
     xlab = "Turnout (in %)",
     col = "yellow1",
     freq = FALSE)

# плюс, сделаем линию кривой пожирнее
# lwd - om line width, по умолчанию  lwd=1

curve(dnorm(x, mean = mean(cp$vturn),
            sd = sd(cp$vturn)),
            col = "blue", lwd = 2, add = TRUE)
```



Пока кажется, что распределение явки не очень похоже на нормальное из-за высокого, сильно выделяющегося столбца на участке от 50 до 60. А теперь проверим формально.

Один из статистических критериев, позволяющих проверить нормальность распределения данных, это критерий Шапиро-Уилка. С помощью этого критерия проверяется нулевая гипотеза, которая состоит в том, что данные распределены нормально.

```
shapiro.test(cp$vturn)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  cp$vturn
## W = 0.96857, p-value = 0.01167
```

P-value < 0.05, следовательно, «жизнеспособность» нулевой гипотезы, оценённая на основе имеющихся данных, мала. На имеющихся данных на уровне значимости 5% (0.05) есть основания отвергнуть нулевую гипотезу о том, что данные распределены нормально. Показатель явки не распределён нормально.

Связь между качественными переменными: таблицы сопряженности и критерий хи-квадрат

С таблицами частот мы уже знакомы. Познакомимся с таблицами сопряжённости (*contingency tables*) — таблицами, которые иллюстрируют совместное распределение переменных. Построим таблицу сопряженности для двух признаков: `poco` (принадлежность к пост-коммунистическим странам) и `gov_party` (тип партийной системы).

```
table(cp$poco, cp$gov_party)
```

```
##
##      1  2  3  4  5
```

```
##   0 35 11 15  6  8
##   1  7  8  9  6  2
```

По полученной таблице сопряжённости можно определить, например, что число пост-коммунистических стран с гегемонией правых/центристских партий равно 4.

Связь между качественными переменными можно визуализировать с помощью мозаичного графика (*mosaic plot*). Подробнее о мозаичном графике см. здесь и здесь. Для этого потребуется библиотека `vcd` (от *visualising categorical data*). Установим её:

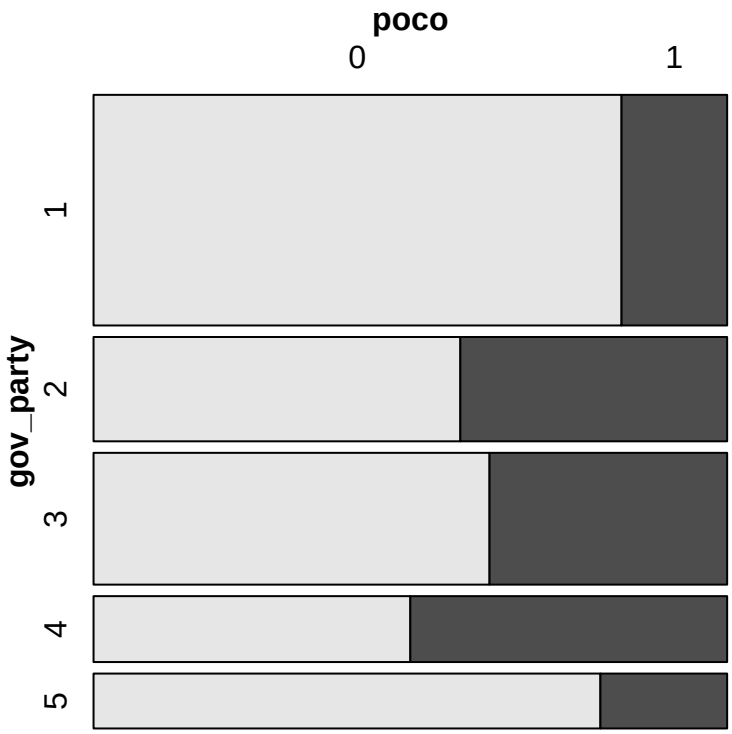
```
install.packages("vcd")
```

Обратимся к ней:

```
library(vcd)
```

Построим мозаичный график:

```
mosaic(data = cp, poco ~ gov_party)
```



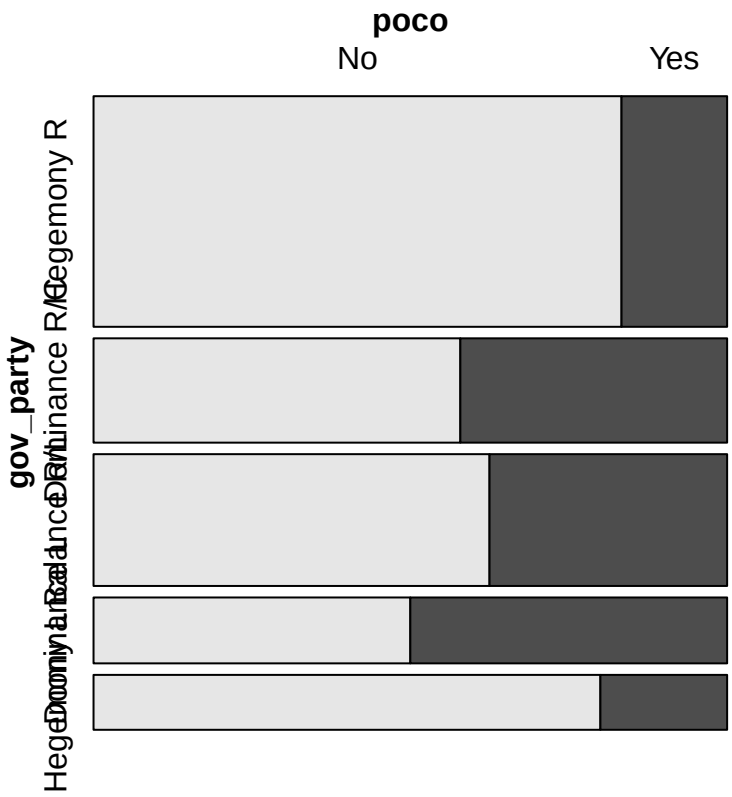
С помощью мозаичного графика мы можем визуализировать таблицу сопряжённости. Тёмно-серый цвет соответствует пост-коммунистическим странам, светло-серый — всем остальным. Разбивка на пять блоков по горизонтали — разбивка по значениям переменной `gov_party` (гегемония правых/центристских партий, доминирование левых партий и прочие).

Чтобы всё совсем стало понятно, поправим подписи по осям. Создадим список (*list*) с поименованными векторами, один для подписей к `poco`, другой — к `gov_party`.

```
args <- list(poco = c("No", "Yes"),
             gov_party = c("Hegemony R", "Dominance R/C", "Balance R/L", "Dominance L", "Hegemony L"))
```

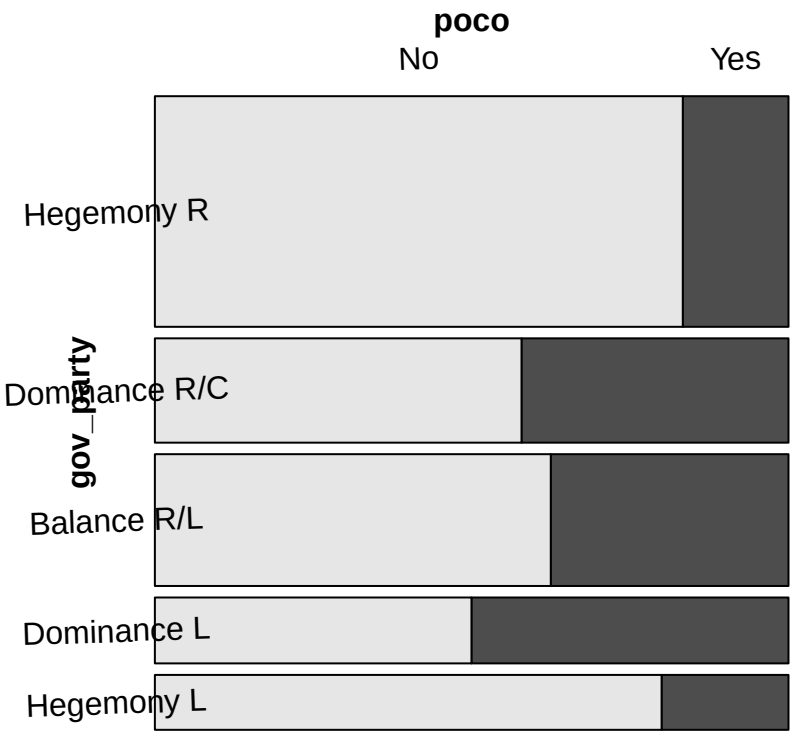
А теперь добавим полученные подписи — запишем их в аргумент `set_labels`:

```
mosaic(data = cp, poco ~ gov_party, set_labels = args)
```



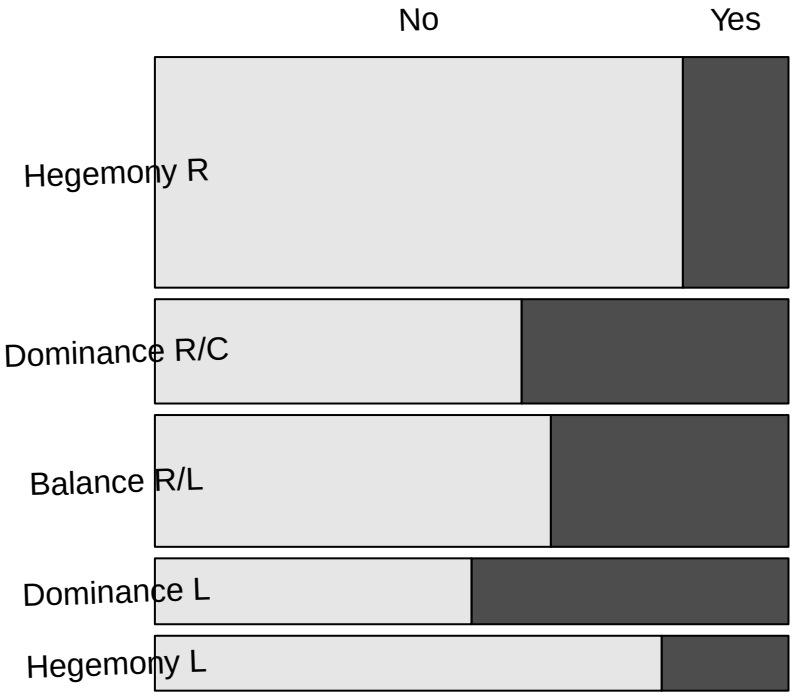
Проблему длинных подписей, которые накладываются друг на друга, можно решать по-разному. Мы пока на время воспользуемся простым, но не самым красивым: повернём подписи и сделаем их горизонтальными.

```
# rot_labels = 2 - поворот на 90 вправо
# вообще в rot_labels можно ставить градусы,
# попробуйте выставить rot_labels = 45

mosaic(data = cp, poco ~ gov_party,
       set_labels = args,
       rot_labels = 2)
```



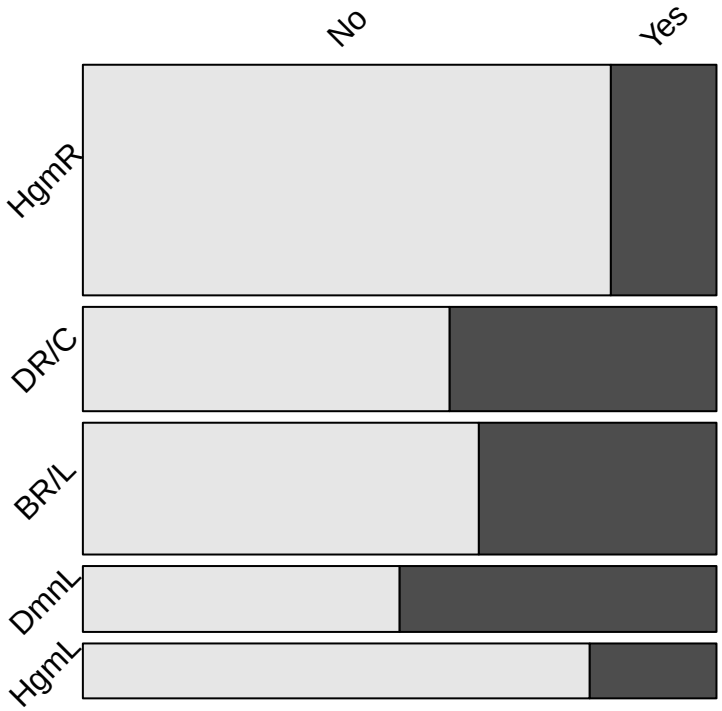
Чтобы не мешали подписи к самим осям, их можно убрать:

```
# set_varnames - подписи к самим осям x и y

mosaic(data = cp, poco ~ gov_party,
       set_labels = args,
       rot_labels = 2,
       set_varnames = c(poco = "", gov_party = ""))
```



Теперь сократим названия групп по вертикали до аббревиатур с помощью функции `abbreviate()` и повернём все подписи на 45 градусов:

```
args <- list(poco = c("No", "Yes"),
             gov_party = abbreviate(c("Hegemony R", "Dominance R/C", "Balance R/L", "Dominance L", "Hege

mosaic(data = cp, poco ~ gov_party,
       set_labels = args,
       rot_labels = 45,
       set_varnames = c(poco = "", gov_party = ""))
```



А теперь проверим формально, есть ли связь между этими признаками (принадлежность к пост-коммунистическим странам и тип партийной системы). Воспользуемся критерием хи-квадрат. Нулевая гипотеза: признаки не связаны (независимы).

$$H_0 : \text{признаки независимы (не связаны)}$$
$$H_1 : \text{признаки не независимы (связаны)}$$

```
chisq.test(table(cp$poco, cp$gov_party))
```

```
## Warning in chisq.test(table(cp$poco, cp$gov_party)): Chi-squared approximation
## may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(cp$poco, cp$gov_party)
## X-squared = 8.3005, df = 4, p-value = 0.08117
```

P-value > 0.05, следовательно, вероятность того, что мы получим результаты такие, какие получили и выше, при условии, что нулевая гипотеза верна, не мала. На имеющихся данных на уровне значимости 5% (0.05) нет оснований отвергнуть нулевую гипотезу о том, что признаки независимы. Тип партийной системы и принадлежность к пост-коммунистическим странам не связаны.

Замечание. R выдал предупреждение `Chi-squared approximation may be incorrect`. (Пояснение, возможно, будет понятно не всем, но его можно смело пропустить и посмотреть, как решается эта проблема). При расчете ожидаемых частот для расчета наблюдаемого значения статистики хи-квадрат получилось, что некоторые ожидаемые частоты в ячейках таблицы сопряженности меньше 5, и таких ячеек много. В такой ситуации p-value не может быть посчитан точно. Для решения проблемы есть два способа: объединить ячейки (укрупнить группы, если это уместно) или воспользоваться точным тестом Фишера (*Fisher's Exact test*). Мы пойдём вторым путём:

```
fisher.test(cp$poco, cp$gov_party)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  cp$poco and cp$gov_party
## p-value = 0.07206
## alternative hypothesis: two.sided
```

Логика проверки гипотезы и выводы — те же самые, что и в критерии хи-квадрат.

Связь между количественными переменными: напоминание про корреляции

Коэффициент корреляции К.Пирсона — показатель линейной связи между двумя переменными, измеренными в количественной шкале. Коэффициент корреляции принимает значения от -1 до 1. Отрицательные значения коэффициента корреляции свидетельствуют об обратной связи между переменными (с ростом значений одной переменной значения другой переменной уменьшаются), положительные значения коэффициента корреляции — о прямой связи между переменными (с ростом значений одной переменной значения другой переменной увеличиваются). Если коэффициент корреляции Пирсона между переменными равен 0, это не всегда означает, что связи между ними нет — связь между ними может просто быть нелинейной, например, квадратичной. Коэффициент корреляции показывает только связь между переменными, а не зависимость (Y зависит от X) и не влияние (X влияет на Y) и, конечно, ничего не сообщает о причинно-следственной связи.

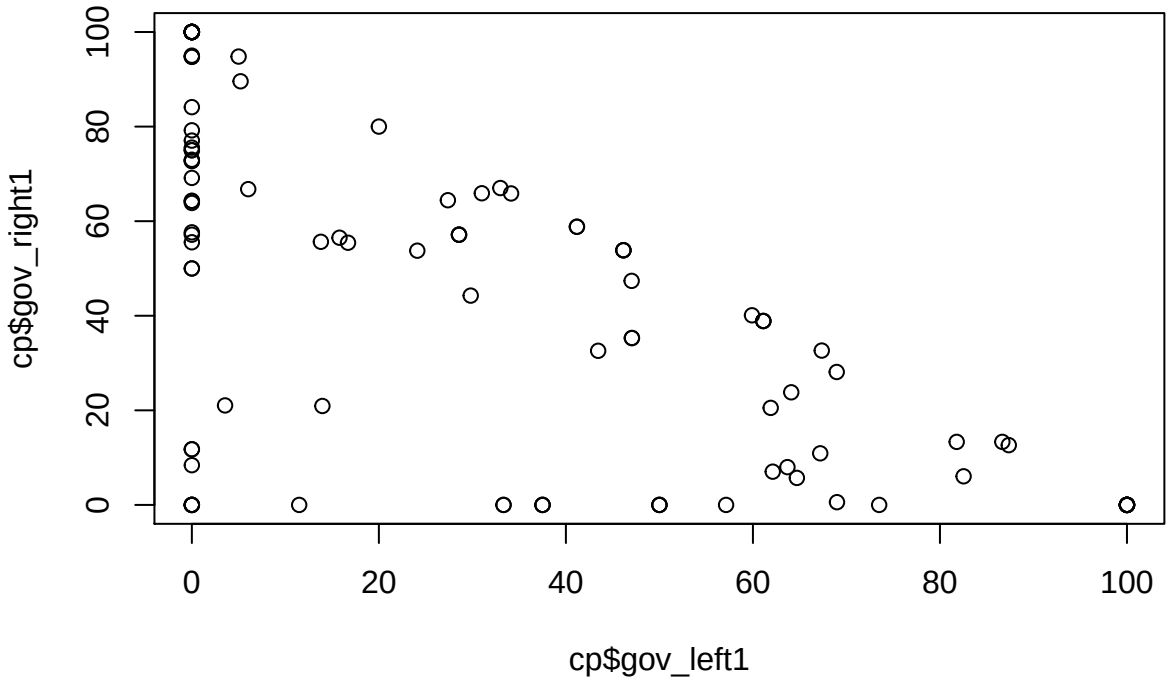
Коэффициент корреляции Ч.Спирмена также используется для измерения связи между двумя переменными, измеренными в количественной шкале, преимущественно в порядковой (ординальной). Коэффициент корреляции Спирмена, в отличие от коэффициента Пирсона, является устойчивым к наличию нетипичных значений.

Связи между количественными переменными можно представить в виде корреляционной матрицы. Корреляционная матрица всегда симметрична (коэффициент корреляции между переменными X и Y равен коэффициенту корреляции между переменными Y и X), и на главной диагонали такой матрицы стоят 1 (корреляция переменной самой с собой равна 1).

Связь между количественными переменными: диаграммы рассеяния

Допустим, мы хотим посмотреть на связь между переменными `gov_left1` и `gov_right1`. Построим диаграмму рассеяния (*scatter plot*):

```
plot(cp$gov_left1, cp$gov_right1)
```



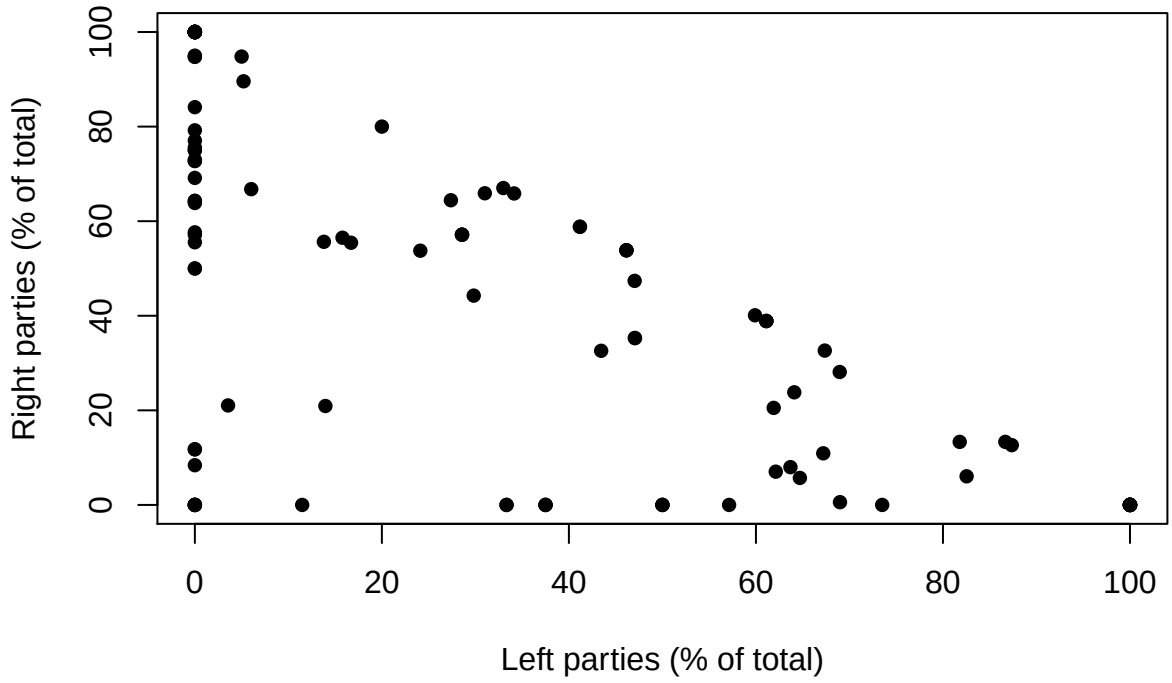
По диаграмме рассеяния видно, что связь между переменными обратная (чем больше x, тем меньше y) и, скорее всего, сильная.

Как можно заметить, особой красотой этот график не отличается. График скучный. Что мы можем сделать? Во-первых, подписать оси и изменить тип маркера для точек.

Все типы маркеров мы можем посмотреть, запросив `help` по аргументу `pch`, он как раз отвечает за тип точек (`pch` — от *point character*).

```
?pch
```

```
plot(cp$gov_left1, cp$gov_right1,
     xlab = "Left parties (% of total)",
     ylab = "Right parties (% of total)",
     pch = 16)
```



Во-вторых, мы можем добавить цвета, причем вполне содержательно. Допустим, мы хотим разделить страны на пост-коммунистические и не пост-коммунистические и отразить это на графике. То есть, точки, соответствующие пост-коммунистическим странам и точки, соответствующие всем остальным странам будут отличаться по цвету.

```
str(cp$poco) # проверим, какие значения принимает poco
```

```
##  int [1:108] 0 0 0 0 0 0 0 0 0 1 ...
```

Показатель `poco` числовой, но по смыслу он качественный, то есть факторный. Поправим тип:

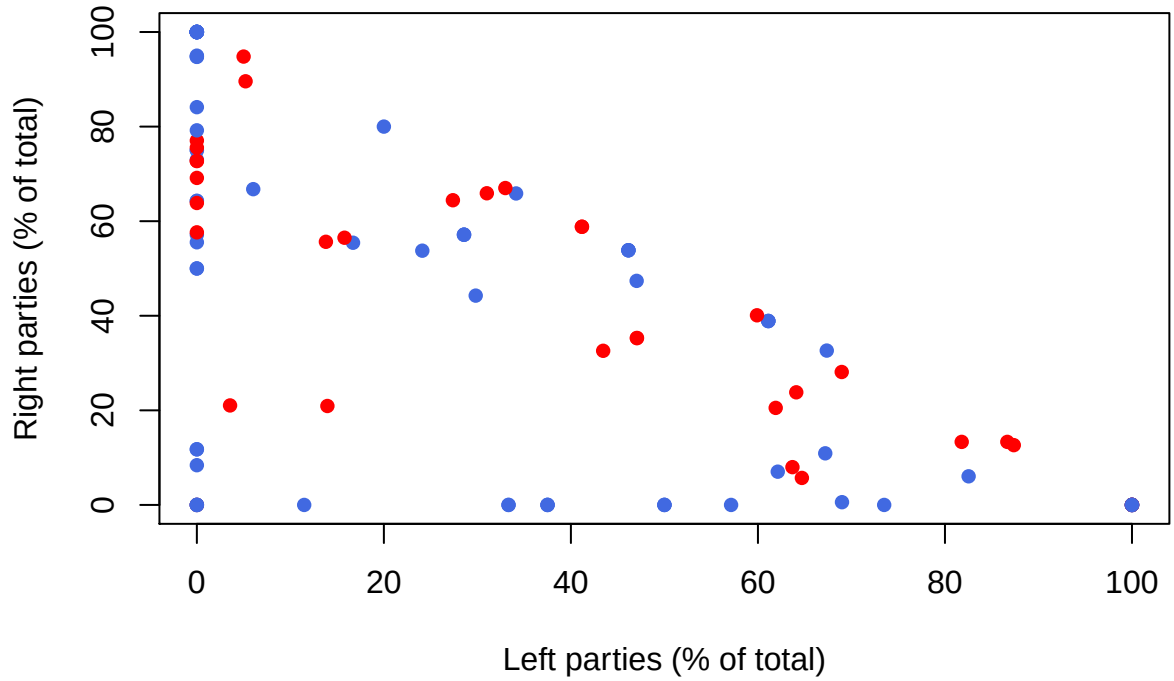
```
cp$poco <- factor(cp$poco)
```

Устанавливаем цвета по группирующей переменной `poco`:

```
# каждой строке в датафрейме - свой цвет на основе poco
colors <- c("royalblue", "red")[cp$poco]
head(colors, 10)
```

```
##  [1] "royalblue" "royalblue" "royalblue" "royalblue" "royalblue" "royalblue"
##  [7] "royalblue" "royalblue" "royalblue" "red"
```

```
plot(cp$gov_left1, cp$gov_right1,
     xlab = "Left parties (% of total)",
     ylab = "Right parties (% of total)",
     pch = 16,
     col = colors)
```



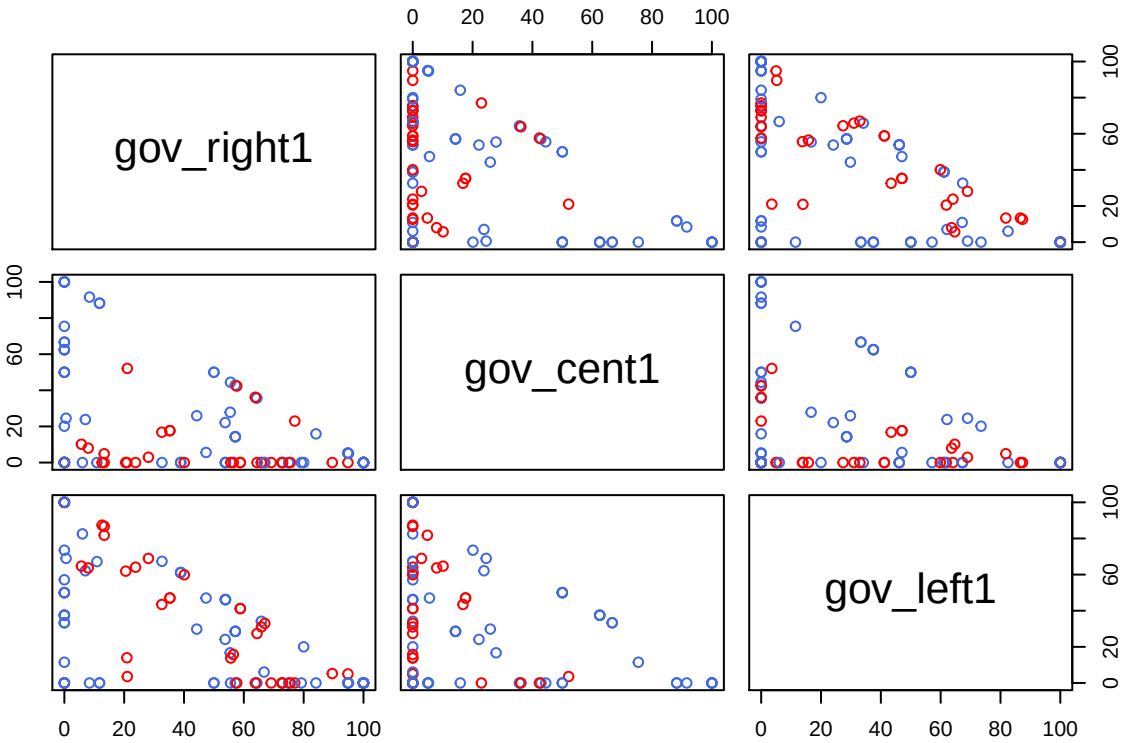
##

Связь между количественными переменными: матрицы диаграмм рассеяния

Иногда в ходе предварительного анализа бывает нужно посмотреть на связь «всего со всем». Для этого удобно использовать матрицу диаграмм рассеяния (*scatterplot matrix*).

Построим диаграммы рассеяния для процентов голосов за разные партии.

```
pairs(cp[10:12], col = colors) # выбираем столбцы 10-12
```

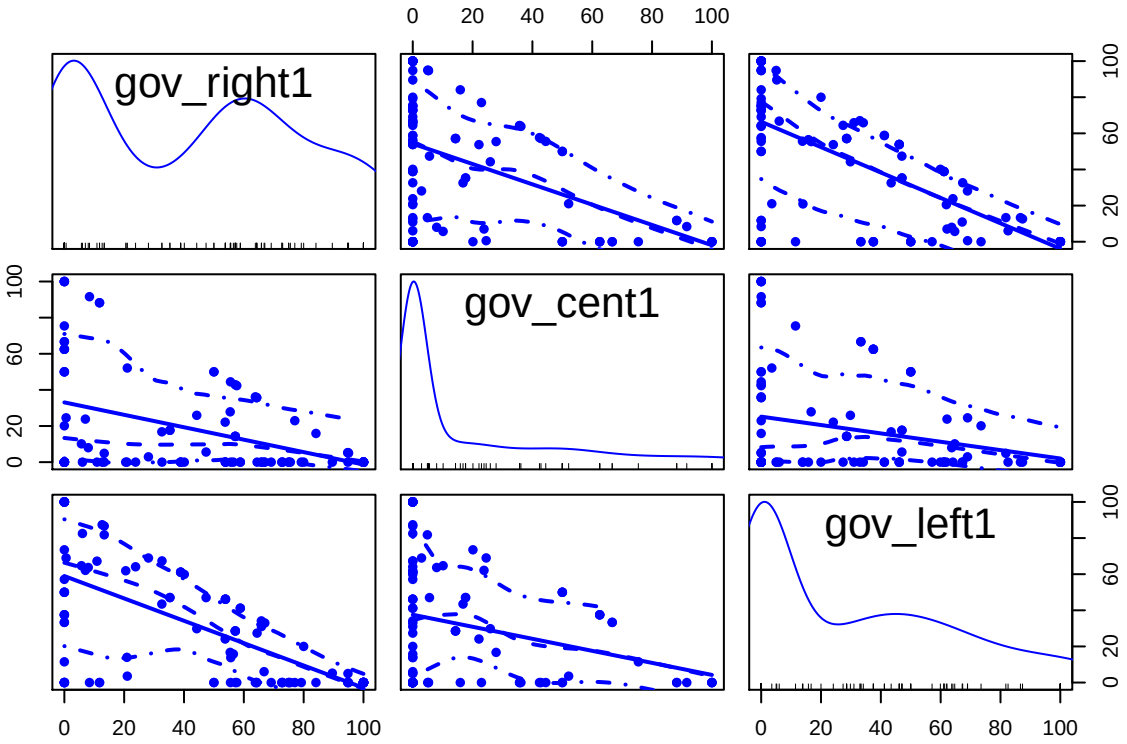


На пересечении названий переменных находятся диаграммы рассеяния, соответствующие парам показателей.

А теперь проиллюстрируем то же самое, но более красочно. Для этого потребуется библиотека `car`.

```
install.packages("car")
```

```
library(car)
scatterplotMatrix(cp[10:12], pch = 16)
```

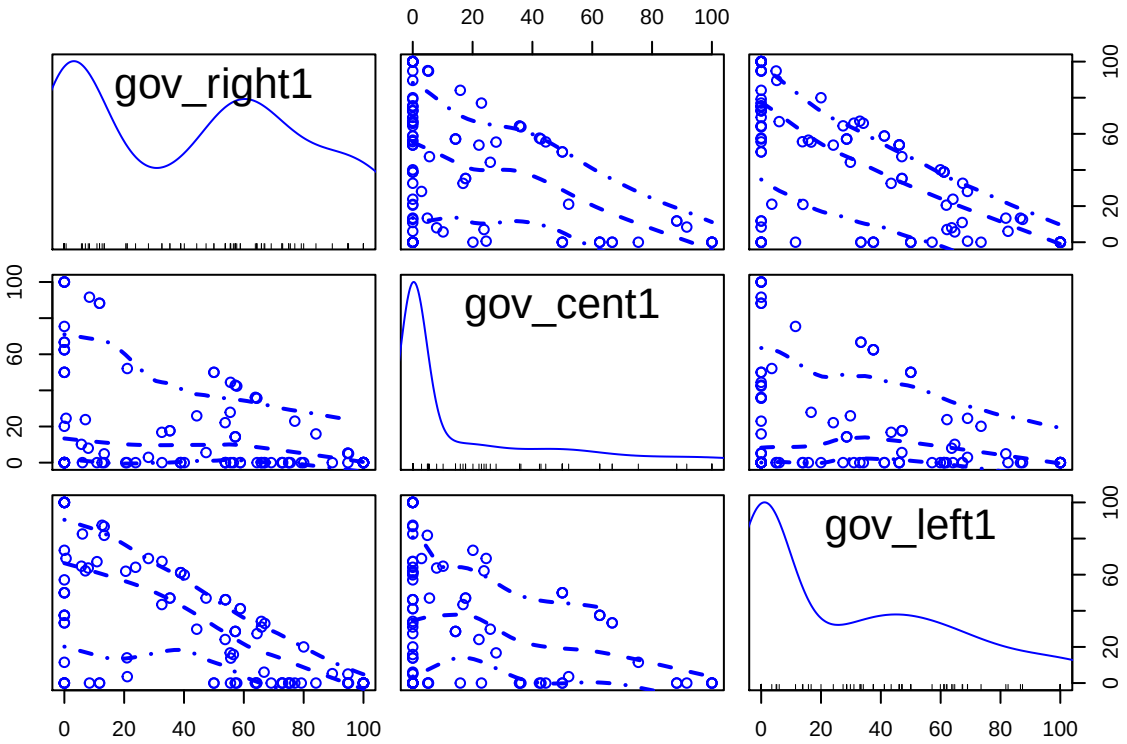


На диагонали этой матрицы диаграмм рассеяния добавляются графики плотности распределения («сглаженные» гистограммы). На сами диаграммы рассеяния добавляется регрессионная прямая (прямая вида $y = kx + b$, при $k < 0$ наклон прямой отрицательный, связь между x и y обратная, при $k > 0$ наклон прямой положительный, связь между x и y прямая). Можно также добавить кривую взвешенной регрессии (*loess regression* от *locally weigthed regression*). Логика её построения (в очень упрощённом виде) такая:

- все значения x разбиваем на много маленьких интервалов;
- на каждом интервале строим регрессионную прямую;
- «сглаживаем» получившуюся ломаную линию, чтобы получить гладкую кривую;

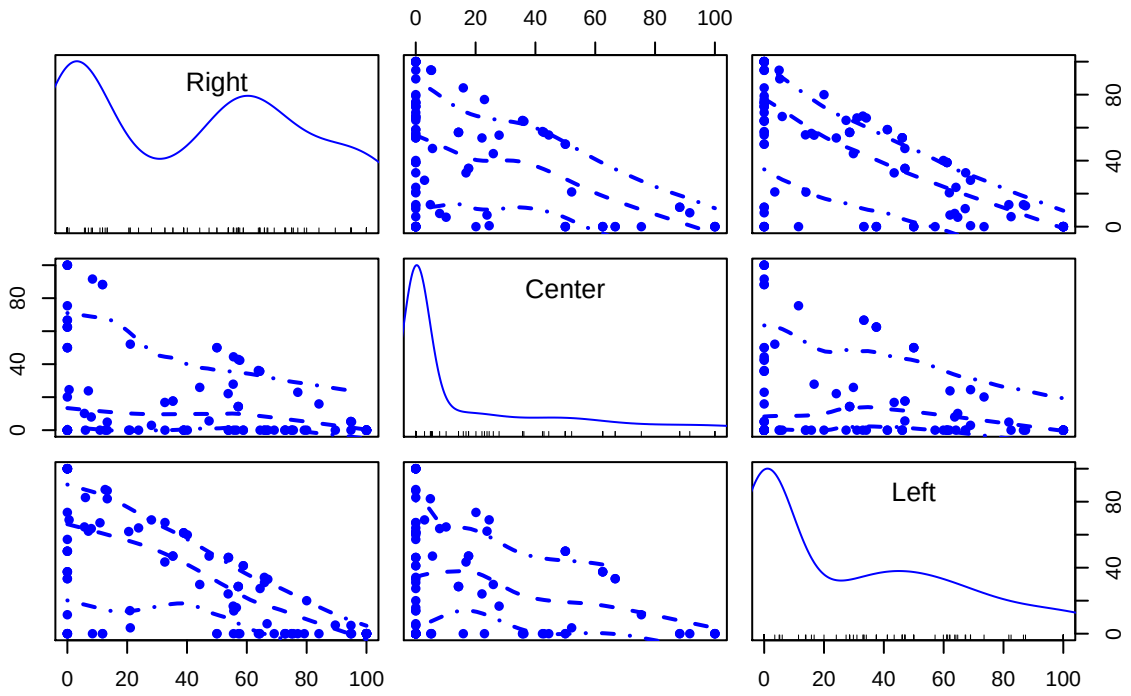
Чтобы добавить линию взвешенной регресии, нужно убрать обычную регрессионную прямую (`reg.line = FALSE`) и добавить новую сглаженную (`smooth = TRUE`):

```
scatterplotMatrix(cp[10:12], regLine = FALSE, smooth = TRUE)
```



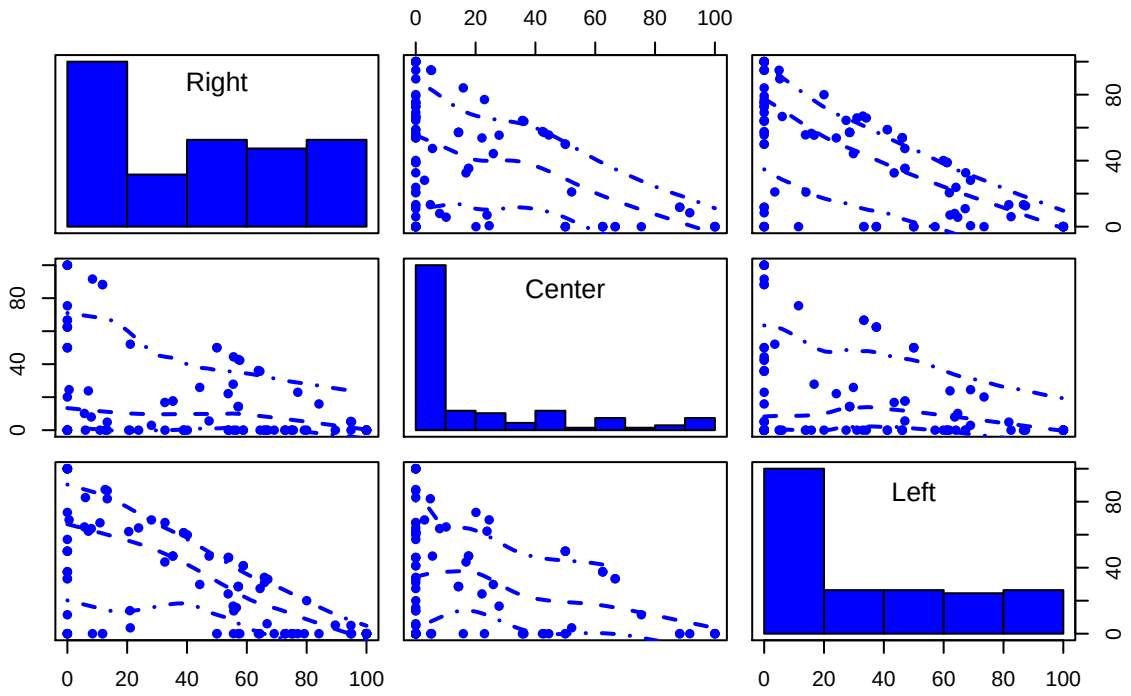
Наведем красоту на графике выше, создадим вектор с названиями переменных в более внятном виде и чуть-чуть увеличим шрифт у подписей на диагонали:

```
labs <- c("Right", "Center", "Left")

# var.labels - названия переменных на диагонали
# cex.labels - размер шрифта для labels
# main - название графика

scatterplotMatrix(cp[10:12],
                  regLine = FALSE, smooth = TRUE,
                  var.labels = labs,
                  cex.labels = 1.3,
                  pch = 16,
                  main = "Correlations of parties' share")
```



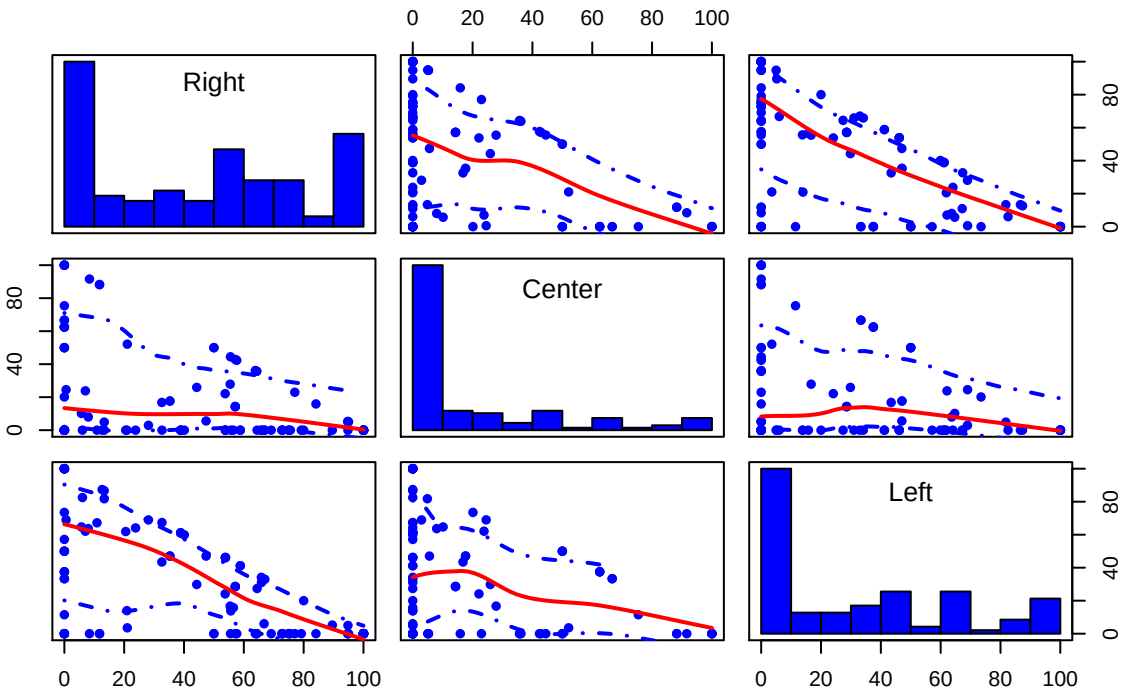
А теперь поменяем графики плотности на диагонали на гистограммы (при желании можно поменять на ящики с усами, вписав “boxplot”):

```
scatterplotMatrix(cp[10:12], regLine = FALSE, smooth = TRUE,
                  var.labels = labs,
                  cex.labels = 1.3,
                  pch = 16,
                  main = "Correlations of parties' share",
                  diagonal = list(method = "histogram"))
```



Настроим цвет и тип у линии сглаженной регрессии и число столбцов у гистограмм (кажется, цвет гистограмм в новой версии библиотеки поменять не удастся):

```
scatterplotMatrix(cp[10:12], regLine = FALSE,
                  smooth = list(col.smooth = "red", lty.smooth = 1),
                  var.labels = labs,
                  cex.labels = 1.3,
                  pch = 16,
                  main = "Correlations of parties' share",
                  diagonal = list(method = "histogram",
                                  breaks = 8))
```



Ещё один вариант симпатичного графика для корреляций — разноцветный график, созданный с помощью библиотеки `gclus`.

```
install.packages("gclus")
```

```
library(gclus)
```

Для начала получим таблицу коэффициентов корреляции (по модулю):

```
coeffs <- abs(cor(cp[10:12]))
coeffs
```

```
##             gov_right1 gov_cent1 gov_left1
## gov_right1  1.0000000 0.4397581 0.6594741
## gov_cent1   0.4397581 1.0000000 0.2777804
## gov_left1   0.6594741 0.2777804 1.0000000
```

Зададим цвета на основе таблицы `coeffs`:

```
colors <- dmat.color(coeffs)
head(colors)
```

```
##             gov_right1 gov_cent1 gov_left1
## gov_right1 NA          "#D2F4F2" "#F4BBDD"
## gov_cent1  "#D2F4F2"   NA        "#FDFFDA"
## gov_left1  "#F4BBDD"   "#FDFFDA" NA
```

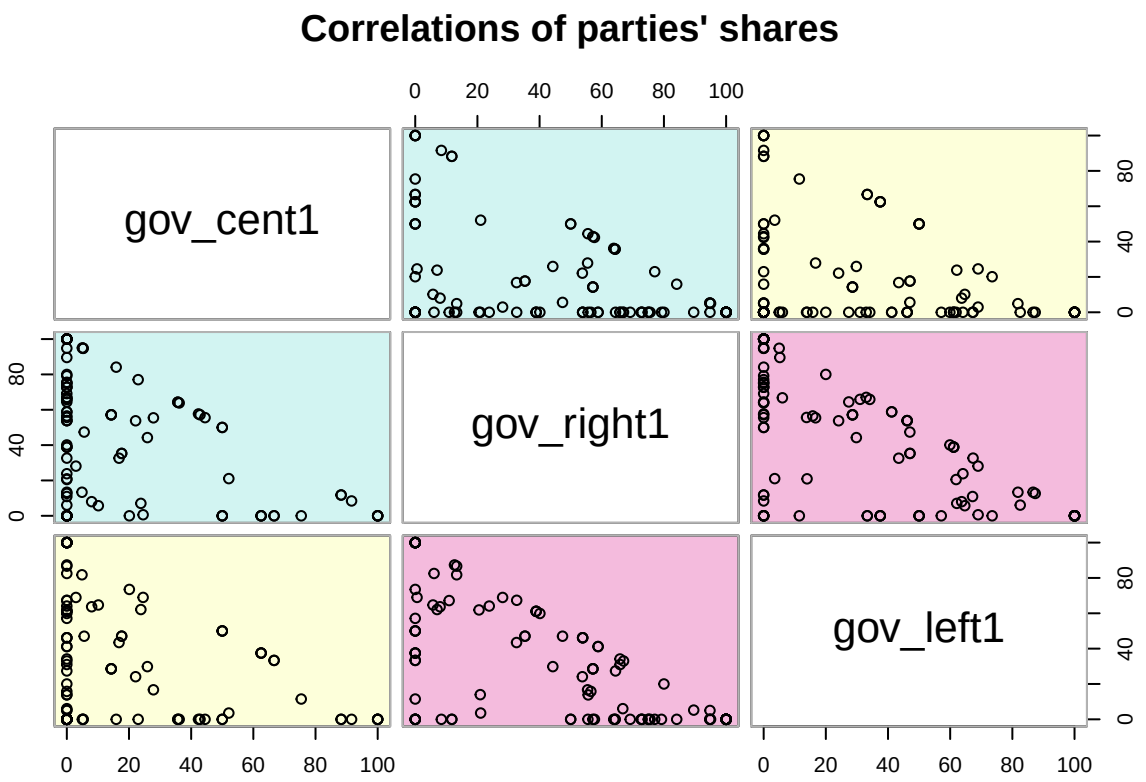
Запишем для каждого значения коэффициента корреляции по модулю в `coeffs` его номер при сортировке по возрастанию. Это нужно для того, чтобы потом графики можно было расположить так, чтобы диаграммы рассеяния, где связь переменных наибольшая, были ближе к диагонали.

```
order <- order.single(coeffs)
order
```

```
## [1] 2 1 3
```

Строим сам график:

```
# gap - расстояние между графиками в матрице
cpairs(cp[10:12], order, panel.colors = colors, gap = .5,
       main = "Correlations of parties' shares" )
```



Связь между количественными переменными: коэффициенты корреляции

Для начала посмотрим на коэффициент корреляции между какими-нибудь двумя переменными:

```
cor(cp$gov_left1, cp$gov_right1)
```

```
## [1] -0.6594741
```

Если бы в одной из переменных были пропущенные значения (NA), коэффициент корреляции бы не рассчитался. Тут можно действовать по аналогии с расчетом среднего значения:

```
# использовать всё, кроме NA (complete observations)
cor(cp$gov_left1, cp$gov_right1, use = "complete.obs")
```

```
## [1] -0.6594741
```

Как известно, существуют разные коэффициенты корреляции. Самые распространенные — линейный коэффициент корреляции Пирсона, коэффициент ранговой корреляции Спирмена и коэффициент ранговой корреляции Кендалла. По умолчанию считается коэффициент Пирсона, остальные можно получить, прописав дополнительный аргумент:

```
# коэффициент Спирмена
cor(cp$gov_left1, cp$gov_right1, method = "spearman")
```

```
## [1] -0.6544615
```

Проверить значимость коэффициента корреляции — проверить нулевую гипотезу о том, что истинный коэффициент корреляции равен 0.

$$H_0 : r = 0 \text{ (связи нет)}$$

$$H_1 : r \neq 0 \text{ (связь есть)}$$

```
corr <- cor.test(cp$gov_left1, cp$gov_right1)
corr
```

```
##
##  Pearson's product-moment correlation
##
## data:  cp$gov_left1 and cp$gov_right1
## t = -9.0321, df = 106, p-value = 8.441e-15
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.7544286 -0.5374832
## sample estimates:
##        cor
## -0.6594741
```

В выдаче R мы видим две важные вещи: значение коэффициента корреляции (`sample estimates`) и `pvalue`. В нашем случае p-value < 0.05, следовательно, на 5% уровне значимости есть основания отвергнуть нулевую гипотезу о равенстве коэффициента корреляции нулю. Раз эту гипотезу отвергаем, считаем, что коэффициент корреляции не 0, а следовательно, связь между процентом левых и правых партий действительно есть.

Выдача R представляет собой список (*list*):

```
str(corr)
```

```
## List of 9
##  $ statistic  : Named num -9.03
##   ..- attr(*, "names")= chr "t"
##  $ parameter  : Named int 106
##   ..- attr(*, "names")= chr "df"
##  $ p.value    : num 8.44e-15
##  $ estimate   : Named num -0.659
##   ..- attr(*, "names")= chr "cor"
##  $ null.value : Named num 0
##   ..- attr(*, "names")= chr "correlation"
##  $ alternative: chr "two.sided"
##  $ method     : chr "Pearson's product-moment correlation"
##  $ data.name  : chr "cp$gov_left1 and cp$gov_right1"
##  $ conf.int   : num [1:2] -0.754 -0.537
##   ..- attr(*, "conf.level")= num 0.95
##  - attr(*, "class")= chr "htest"
```

А значит, из него можно вызывать отдельные элементы.

```
coeff <- corr$estimate # коэффициент
pvalue <- corr$p.value # p-value
coeff; pvalue
```

```
##        cor
```

```
## -0.6594741
```

```
## [1] 8.440678e-15
```

Если хотим посмотреть на корреляцию «всего со всем», можем указать столбцы в базе (переменные) и получить корреляционную матрицу:

```
cor(cp[10:12])
```

```
##            gov_right1  gov_cent1  gov_left1
## gov_right1  1.0000000 -0.4397581 -0.6594741
## gov_cent1  -0.4397581  1.0000000 -0.2777804
## gov_left1  -0.6594741 -0.2777804  1.0000000
```

Для того, чтобы получить корреляционную матрицу и значимость коэффициентов в ней, нужно постараться. Загрузим библиотеку `Hmisc`.

```
install.packages("Hmisc")
```

```
library(Hmisc)
```

Внимание: функция `rcorr()` привередничает — требует матрицу, а не просто столбцы из датафрейма.

```
rcorr(as.matrix(cp[10:12]))
```

```
##            gov_right1 gov_cent1 gov_left1
## gov_right1       1.00     -0.44     -0.66
## gov_cent1       -0.44      1.00     -0.28
## gov_left1       -0.66     -0.28      1.00
##
## n= 108
##
##
## P
##            gov_right1 gov_cent1 gov_left1
## gov_right1            0.0000    0.0000
## gov_cent1  0.0000               0.0036
## gov_left1  0.0000     0.0036
```

Но то, что мы увидели, немного не похоже на то, что хотелось бы показать другим. Единой таблички с коэффициентами и значимостью нет. Действительно, в R есть некоторые проблемы с корреляционными матрицами. Позже, когда будем обсуждать красивую выгрузку описательных статистик в файл, посмотрим и на то, как выгрузить корреляционную матрицу в виде привычной «таблички со звёздочками» для значимости.