### Исследование временных рядов с помощью среды R. Несбалансированные данные. Временные ряды.

Форматы представления даты и времени

С форматом представления дат и времени в R можно ознакомиться, выполнив команду

Sys.time()

[1] "2017-04-03 15:07:13 AST"

Как видим, формат строго иерархичен: сначала идет наиболее крупная временная единица – год, потом месяц и день, разделенные дефисом, а затем пробел, час, минуты, секунды и, после еще одного пробела, аббревиатура временной шкалы.

Отдельные элементы из этого результата можно извлечь при помощи функции substr() (от substring – часть строки), указав позиции первого и последнего элементов извлекаемой строки:

substr(as.character(Sys.time()), 1, 10)

[1] "2017-04-03"

или

substr(as.character(Sys.time()), 12, 19)

[1] "15:08:23"

Функция date() позволяет выяснить текущую дату:

"Mon Apr 03 15:13:55 2017"

Если выполнить команду

unclass(Sys.time())

[1] 1491221679

то получим время в формате POSIXct, т.е. выраженное в секундах, прошедших с 1 января 1970 г. (его еще трактуют как Unix-время, по названию операционной системы).

Такой "машинный" формат удобен для включения в таблицы данных.

Для человека более удобным является представление времени в формате класса POSIXlt. Объекты этого класса представляют собой списки, включающие такие элементы, как секунды, минуты, часы, дни, месяцы, и годы. Например, мы можем конвертировать системное время в объект POSIXlt класса следующим образом:

date <- as.POSIXlt(Sys.time())

Из списка date далее легко можно извлечь такие содержащиеся в нем элементы, как sec (секунды), min (минуты), hour (часы), mday (день месяца), mon (месяц), year (год), wday (день недели, начиная с воскресенья = 0), yday (день года, начиная с 1 января = 0), и isdst ("is daylight savings time in operation?" – логическая переменная, обозначающая, используется ли режим перехода на "зимнее" и "летнее" время: 1 если TRUE и 0 если FALSE), например:

date$wday

[1] 2

date$yday

[1] 91

Вычисления с датами и временем

В R можно выполнять следующие типы вычислительных операций с датами и временем:

° число + время;

° время – число;

° время1 – время2

° время1 "логический оператор" время2 (в качестве логического оператора могут использоваться ==, !=, <=, <, > или >=).

Важной особенностью является то, что перед выполнением любых вычислений с датами или временем необходимо конвертировать их в объекты класса POSIXlt.

Например, количество дней между 15 сентября 2011 г. и 15 сентября 2000 года можно найти следующим образом:

t1 <- as.POSIXlt("2011-09-15")

t2 <- as.POSIXlt("2000-09-15")

t1 - t2

Time difference of 4017 days

Разницу во времени, выраженную в часах, можно рассчитать так:

t3<-as.POSIXlt("2010-09-22 08:30:30")

t4<-as.POSIXlt("2010-09-22 22:25:30")

t4-t3

Time difference of 13.91667 hours

Еще проще разницу между двумя датами можно найти при помощи готовой функции difftime() (от difference – разница, и time – время):

difftime("2011-09-22", "2010-06-22")

Time difference of 457 days

Чтобы извлечь непосредственно количество дней из результата выполнения предыдущей команды используйте функцию as.numeric():

as.numeric(difftime("2011-09-22", "2010-06-22"))

[1] 457

Обратите внимание: в R отсутствует возможность для сложения двух дат.

Извлечение даты/времени из текстовых переменных

Функция strptime() (от strip – раздевать, оголять, и time – время) позволяет извлекать даты и время из различных текстовых выражений. При этом важно верно указать формат (при помощи аргумента format), в котором приведены временные величины. Приняты следующие условные обозначения для наиболее часто используемых форматов дат и времени (детали доступны по команде ?strptime):

%a – сокращенное название для недели (англ. яз.)

%A – полное название для недели (англ. яз.)

%b – сокращенное название месяца (англ. яз.)

%B – полное название месяца (англ. яз.)

%d – день месяца (01–31)

%H – часы от 00 до 23

%I – часы от 01 до 12

%j – порядковый номер дня года (001–366)

%m – порядковый номер месяца (01–12)

%M – минуты (00–59)

%S – секунды (00–61, с возможностью добавить "високосную секунду")

%U – неделя года (00–53), первое вокресенье считается первым днем первой недели

%w – порядковый номер дня недели (0–6, воскресенье – 0)

%W – неделя года (00–53), первый понедельник считается первым днем первой недели

%Y – год с указанием века

%y – год без указания века

Рассмотрим пример. Предположим, у нас имеется текстовый вектор, в котором хранятся даты в формате программы Microsoft Excel:

dates.excel <- c("25/02/2008", "24/04/2009", "14/06/2009", "25/07/2010", "04/03/2011")

Формат имеющихся Excel-дат таков, что сначала идет день месяца, затем порядковый номер самого месяца и, наконец, год с указанием века. Требуется преобразовать эти текстовые выражения в даты формата R. Используя приведенные выше обозначения форматов функции strptime(), параметр format можно представить в виде %d/%m/%Y. Тогда команда для преобразования Excel-дат в R-даты будет выглядеть следующим образом:

strptime(dates.excel, format = "%d/%m/%Y")

[1] "2008-02-25 AST" "2009-04-24 AST" "2009-06-14 AST" "2010-07-25 AST"

[5] "2011-03-04 AST"

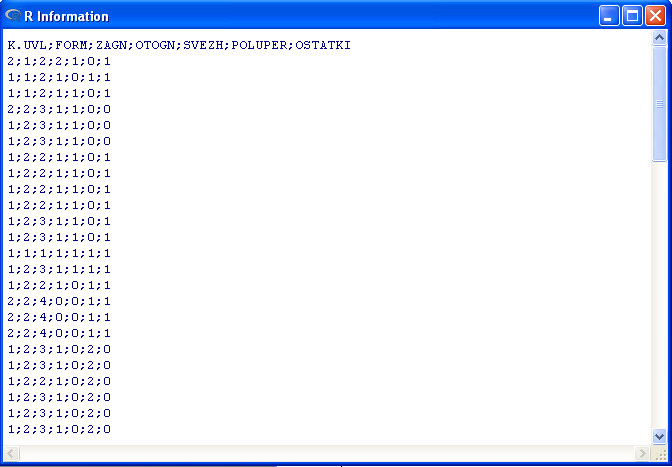
В R существует специальный класс объектов для работы с данными, представляющими собой временные ряды – ts (от time series – временной ряд). Для создания объектов этого класса служит одноименная функция ts().

**Пример 1.** В текстовом файле leaf2-4.txt записаны результаты длившихся трое суток непрерывных наблюдений над хищным растением росянкой. Состояние листа отмечали каждые 40 минут, всего в сутки делали 36 наблюдений.

Попробуем создать временной ряд из колонки FORM, в которой закодированы изменения формы пластинки листа (1 — практически плоская, 2 — вогнутая).

Просмотреть, как устроен файл данных, при помощи команды

file.show("F:/leaf2-4.txt")



Загрузить таблицу:

leaf <- read.table("F:/leaf2-4.txt", head=TRUE, as.is=TRUE, sep=";")

Проверка загрузки таблицы:

str(leaf)

'data.frame': 80 obs. of 7 variables:

$ K.UVL : int 2 1 1 2 1 1 1 1 1 1 ...

$ FORM : int 1 1 1 2 2 2 2 2 2 2 ...

$ ZAGN : int 2 2 2 3 3 3 2 2 2 2 ...

$ OTOGN : int 2 1 1 1 1 1 1 1 1 1 ...

$ SVEZH : int 1 0 1 1 1 1 1 1 1 1 ...

$ POLUPER: int 0 1 0 0 0 0 0 0 0 0 ...

$ OSTATKI: int 1 1 1 0 0 0 1 1 1 1 ...

summary(leaf)

K.UVL FORM ZAGN OTOGN SVEZH

Min. :1.000 Min. :1.0 Min. :1.0 Min. :0.00 Min. :0.0000

1st Qu.:1.000 1st Qu.:1.0 1st Qu.:2.0 1st Qu.:0.00 1st Qu.:0.0000

Median :1.000 Median :2.0 Median :2.0 Median :1.00 Median :0.0000

Mean :1.325 Mean :1.7 Mean :2.5 Mean :0.75 Mean :0.1625

3rd Qu.:2.000 3rd Qu.:2.0 3rd Qu.:3.0 3rd Qu.:1.00 3rd Qu.:0.0000

Max. :2.000 Max. :2.0 Max. :4.0 Max. :2.00 Max. :1.0000

POLUPER OSTATKI

Min. :0.000 Min. :0.0000

1st Qu.:1.000 1st Qu.:0.0000

Median :1.000 Median :1.0000

Mean :0.925 Mean :0.6125

3rd Qu.:1.000 3rd Qu.:1.0000

Max. :2.000 Max. :2.0000

Преобразуем колонку FORM во временной ряд:

forma <- ts(leaf$FORM, frequency=36)

Проверка

str(forma)

Time-Series [1:80] from 1 to 3.19: 1 1 1 2 2 2 2 2 2 2 ...

Проверка периодичности данных и тренда

acf(forma, main="")



Эта команда («auto-correlation function», ACF) выводит коэффициенты автокорреляции и рисует график автокорреляции, на котором в нашем случае можно увидеть, что значимой периодичности нет — все пики лежат внутри обозначенного пунктиром доверительного интервала, за исключением самых первых пиков, которые соответствуют автокорреляции без лага или с очень маленьким лагом. По сути, это показывает, что в пределах 0.05 суток (у нас период наблюдений — сутки), то есть около 1 часа, следующее состояние листа будет таким же, как и текущее, а вот на больших интервалах таких предсказаний сделать нельзя. То, что волнообразный график пиков как бы затухает, говорит о том, что в наших данных возможен тренд.

plot(stl(forma, s.window="periodic")$time.series, main="")



Возможный тренд изображен на среднем графике Действительно, наблюдается тенденция к уменьшению значения формы с течением времени. Мы выяснили это при помощи функции stl() (STL — «Seasonal Decomposition of Time Series by Loess»), которая вычленяет из временного ряда три компоненты: сезонную (в данном случае суточную), тренд и случайную.

**Пример 2**. Ежемесячные данные по рождаемости, собранные в период с января 1996 г. по декабрь 2009 г.

birth <-scan("F:/nybirths.dat")

Read 168 items

Объект birth представляет собой вектор со всеми 168 ежемесячными значениями рождаемости (в тыс. человек), в чем можно убедиться при помощи функции:

is.vector(birth)

[1] TRUE

Функция head() позволяет просмотреть первые несколько значений вектора birth (по умолчанию первые 6 значений):

head(birth)

[1] 26.663 23.598 26.931 24.740 25.806 24.364

Преобразуем объект birth во временной ряд:

birth.ts <- ts(birth, start = c(1996, 1), frequency = 12)

В приведенной команде аргумент start был использован для того, чтобы указать дату, с которой начинается временной ряд birth.ts (1996 год, 1-й месяц).

Дополнительный аргумент frequency (частота) позволяет задать шаг приращения последующих дат – в рассматриваемом примере год разбивается на 12 промежутков, так что шаг приращения составляет 1 месяц. Созданный таким образом объект birth.ts при просмотре внешне напоминает матрицу. При этом строкам и столбцам этой матрицы были автоматически, исходя из значений аргументов start и frequency, присвоены соответствующие имена:

birth.ts

Jan Feb Mar Apr May Jun Jul Aug Sep Oct

1996 26.663 23.598 26.931 24.740 25.806 24.364 24.477 23.901 23.175 23.227

1997 21.439 21.089 23.709 21.669 21.752 20.761 23.479 23.824 23.105 23.110

1998 21.937 20.035 23.590 21.672 22.222 22.123 23.950 23.504 22.238 23.142

1999 21.548 20.000 22.424 20.615 21.761 22.874 24.104 23.748 23.262 22.907

2000 22.604 20.894 24.677 23.673 25.320 23.583 24.671 24.454 24.122 24.252

2001 23.287 23.049 25.076 24.037 24.430 24.667 26.451 25.618 25.014 25.110

2002 23.798 22.270 24.775 22.646 23.988 24.737 26.276 25.816 25.210 25.199

2003 24.364 22.644 25.565 24.062 25.431 24.635 27.009 26.606 26.268 26.462

2004 24.657 23.304 26.982 26.199 27.210 26.122 26.706 26.878 26.152 26.379

2005 24.990 24.239 26.721 23.475 24.767 26.219 28.361 28.599 27.914 27.784

2006 26.217 24.218 27.914 26.975 28.527 27.139 28.982 28.169 28.056 29.136

2007 26.589 24.848 27.543 26.896 28.878 27.390 28.065 28.141 29.048 28.484

2008 27.132 24.924 28.963 26.589 27.931 28.009 29.229 28.759 28.405 27.945

2009 26.076 25.286 27.660 25.951 26.398 25.565 28.865 30.000 29.261 29.012

Nov Dec

1996 21.672 21.870

1997 21.759 22.073

1998 21.059 21.573

1999 21.519 22.025

2000 22.084 22.991

2001 22.964 23.981

2002 23.162 24.707

2003 25.246 25.180

2004 24.712 25.688

2005 25.693 26.881

2006 26.291 26.987

2007 26.634 27.735

2008 25.912 26.619

2009 26.992 27.897

При помощи функции plot.ts() можно быстро изобразить временной ряд графически:

plot.ts(birth)



Мы можем заметить, что есть сезонные колебания в числе рождений в месяц: есть пик каждое лето и дно каждую зиму. Опять же, похоже временной ряд может быть описан аддитивной моделью, поскольку сезонные колебания примерно постоянны с течением времени и, кажется, не зависят от уровня временного ряда, и случайные колебания также примерно постоянны по значению в течение долгого времени.

**Пример 3.** Прогноз числа абонентов интернет- провайдера. Исходные данные состоят из: 1) данных о подключениях за декабрь 2004 года; 2) помесячных данных о подключениях в 2005–2008 годах.

polzovateli <- ts(read.table("F:/data.txt")$V3, start=c(2004,12), frequency=12)

Общее количество абонентов на каждый месяц отчетного периода составило:

abonents <- ts(cumsum(polzovateli), start=c(2004,12), frequency=12)

Отобразим данные помесячного подключения и изменения коли- чества пользователей графически. Оба временных ряда показывают экспоненциальный рост, поэтому выведем их в полулогарифмических координатах

plot(polzovateli, type="b", log="y", xlab="")

plot(abonents, type="b", ylim=c(1,3000), log="y")



Линейный рост временных рядов в полулогарифмических координатах подтверждает предположение об экспоненциальном росте во времени и количества подключений в месяц, и общего количества пользователей. Попробуем теперь построить модель временного ряда общего числа подключений распространенным методом ARIMA («Autoregressive Integrated Moving Average», авторегрессия интегрированного скользящего среднего).

model01 <- arima(abonents, order=c(0,0,1))

model02 <- arima(abonents, order=c(0,0,2))

model03 <- arima(abonents, order=c(0,0,3))

model04 <- arima(abonents, order=c(0,0,4))

model05 <- arima(abonents, order=c(0,0,5))

model06 <- arima(abonents, order=c(0,0,6))

model07 <- arima(abonents, order=c(0,0,7))

model08 <- arima(abonents, order=c(0,0,8))

model09 <- arima(abonents, order=c(0,0,9))

model010 <- arima(abonents, order=c(0,0,10))

model011 <- arima(abonents, order=c(0,0,11))

model012 <- arima(abonents, order=c(0,0,12))

model013 <- arima(abonents, order=c(0,0,13))

model014 <- arima(abonents, order=c(0,0,14))

Теперь сравним модели. «Лучшая» модель будет соответствовать минимуму AIC

> plot(AIC(model01, model02, model03, model04, model05, model06, model07, model08, model09, model010, model011, model012, model013, model014), type="b")



На графике можно увидеть, что по ходу кривой первый минимум наблюдается в районе компонента, соответствующего коэффициенту = 12, а дальше ход вычислений становится нестабильным. Теперь выберем лаг авторегресии (первый элемент order)

model012 <- arima(abonents, order=c(0,0,12))

model112 <- arima(abonents, order=c(1,0,12))

model212 <- arima(abonents, order=c(2,0,12))

model312 <- arima(abonents, order=c(3,0,12))

model412 <- arima(abonents, order=c(4,0,12))

AIC(model012, model112, model212, model312, model412)

df AIC

model012 14 477.4036

model112 15 438.3171

model212 16 435.7380

model312 17 438.1186

model412 18 439.0280

AIC минимален, когда лаг авторегресии равен 2, поэтому принимаем это значение для дальнейшего анализа. Аналогично выберем второй компонент:

model2120 <- arima(abonents, order=c(2,0,12))

model2121 <- arima(abonents, order=c(2,1,12))

model2122 <- arima(abonents, order=c(2,2,12))

model2123 <- arima(abonents, order=c(2,3,12))

model2124 <- arima(abonents, order=c(2,4,12))

model2125 <- arima(abonents, order=c(2,5,12))

AIC(model2120, model2121 ,model2122, model2123, model2124)

df AIC

model2120 16 435.7380

model2121 15 421.6246

model2122 15 405.5416

model2123 15 399.1918

model2124 15 406.5416

Видно, что оптимальной моделью является model2123. Ну а теперь, зная оптимальную модель, построим прогноз изменения общего числа абонентов на 2009 год.

plot(abonents, xlim=c(2004.7,2010), ylim=c(0,6500))

lines(predict(model2123, n.ahead=12, se.fit = TRUE)$pred,col="green")

lines(predict(model2123, n.ahead=12, se.fit = TRUE)$se + predict(model2123, n.ahead=12, se.fit = TRUE)$pred, col="red")

lines(-predict(model2123, n.ahead=12, se.fit = TRUE)$se + predict(model2123, n.ahead=12, se.fit = TRUE)$pred,col="blue")



Максимальное и минимальное ожидаемое количество абонентов по месяцам 2009 года составит:

round(predict(model2123, n.ahead=12, se.fit = TRUE)$se + predict(model2123, n.ahead=12, se.fit = TRUE)$pred)

Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

2009 3158 3337 3533 3767 3992 4239 4494 4765 5050 5349 5663 5991

round(-predict(model2123, n.ahead=12, se.fit = TRUE)$se + predict(model2123, n.ahead=12, se.fit = TRUE)$pred)

Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec

2009 3134 3283 3437 3628 3817 4031 4253 4490 4728 4969 5213 5463

### Задание

1) Для примера 1. Исследовать другой признак того же листа (K.UVL, коэффициент увлажнения, отражающий степень «мокрости» листа), выяснить, имеет ли признак такую же периодичность и тренд.

2) Для примера 2. Исследовать временной ряд и выявить, имеет ли он периодичность и тренд. Для этого аналогично примеру 1 построить график ACF, STL. Можно также использовать функцию decompose(). Это функция оценивает тренд, периодическую и нерегулярную составляющие временного ряда, который может быть описан аддитивной моделью. Функиция decompose() возвращает список объектов в качестве результата, где содержатся оценки периодической составляющей, тренда и нерегулярной компоненты, хранящиеся в именованых элементах этого списка объектов, называемых «seasonal», «trend» и «random» соответственно.

birthtscomponents <- decompose(birth.ts)

plot(birthtscomponents)

3) В файле dollar.txt содержатся значения курса доллара Центрального Банка с 1 июля по 9 августа 2011 года, всего за 11 недель. Попробуйте предсказать курс доллара на две недели вперед. Чтобы проверить эффективность предсказания, возьмите для модели данные по 26 июля, а предскажите последние две недели.

4) В своих данных выберите те, которые составляют временной ряд, т.е. замеры какой-либо величины, осуществленные через одинаковые промежутки времени. Примените описанные функции к выбранному временному ряду, постройте графики изменения временного ряда, автокорреляции, попытайтесь выделить периодическую составляющую и тренд.

### Содержание отчёта

1. Полученные результаты по каждому пункту.

2. Выводы о проделанной работе.