# Лекция 6. Моделирование с Maxima

## 6.1 Общие вопросы моделирования

На сегодняшний день существует обширная литература по различным аспектам моделирования систем, природных, технических и экономических объектов.

Одно из широко применяемых универсальных программных средств для решения задач моделирования — пакет Matlab и его расширения для визуального построения блочных моделей — пакет Simulink. Подход, связанный с построением блочных моделей, развивался на протяжении многих лет, и имеет обширную теоретическую базу.

Рассматриваемый в данной книге пакет **Maxima** не рассчитан на построение блочных моделей технических систем или систем массового обслуживания, но может быть очень полезен для построения и анализа аналитических или эмпирических идентифицируемых моделей.

Построение, классификация и идентификация математических моделей — широкая область научной и практической деятельности, широко освещённая в литературе различного назначения.

Определим модель как изображение существенных сторон реальной системы (или конструируемой системы), в удобной форме отражающее информацию о ней. Модели бывают концептуальные, физические или математические (другие названия: феноменологические, эмпирические и аналитические) в зависимости от того, какая сторона явления в данном случае наиболее существенна, от методов, которые можно использовать при построении модели, от количества и качества имеющейся информации.

По мнению П. Эйкхоффа, при построении модели информация должна быть представлена в удобной форме. Это существенно, так как модель должна создать предпосылки для следующих решений. Если модель слишком сложна, её полезность становится сомнительной. Относительная простота является главной характеристикой модели. Модель представляет собой упрощённое отображение действительности. Во многих случаях, для того чтобы модель была полезной, её сложность должна находиться в определённом соотношении со сложностью описываемого объекта (пример: биологические системы).

По способу представления информации об исследуемом объекте и способу построения модели делятся на следующие группы:

- словесные или вербальные модели (описания объекта моделирования на естественном языке);
- физические модели, предполагающие представление основных свойств объекта моделирования каким-либо материальным объектом (моделью, макетом и т.п.);
- формальные модели, представляющие собой описание объекта моделирования на формальном языке, к этой группе относятся математические модели.

В свою очередь, математические модели делятся на графические, табличные, алгоритмические, аналитические.

Средствами **Maxima** можно строить довольно широкий круг различных математических моделей.

Процедуру построения модели во многих источниках называют идентификацией, при этом данный термин часто относится к построению аналитических математических моделей динамических объектов.

Обычно идентификация — многоэтапная процедура. Основные её этапы следующие:

1. Структурная идентификация заключается в определении структуры математической модели на основании теоретических соображений
2. Параметрическая идентификация включает в себя проведение идентифицирующего эксперимента и определение оценок параметров модели по экспериментальным данным.
3. Проверка адекватности — проверка качества модели в смысле выбранного критерия близости выходов модели и объекта.

### 6.1.1 Аналитические модели

Аналитические модели представляют собой отражение взаимосвязей между переменными объекта в виде математической формулы или группы таких формул. Моделирование основано на двух основополагающих признаках:

- на принципе практической ограниченности количества фундаментальных законов природы;
- на принципе подобия, означающем, что явления различной физической природы могут описываться одинаковыми математическими зависимостями.

Для аналитического моделирования характерно то, что процессы функционирования элементов системы записываются в виде некоторых функциональных соотношений (алгебраических, интегро дифференциальных, конечно-разностных и т.п.) или логических условий. Аналитическая модель может быть исследована следующими методами:

- аналитическим, когда стремятся получить в общем виде явные зависимости для искомых характеристик;
- численным, когда, не умея решать уравнений в общем виде, стремятся получить числовые результаты при конкретных начальных данных;
- качественным, когда, не имея решения в явном виде, можно найти некоторые свойства решения (например, оценить устойчивость решения).

Наиболее полное исследование процесса функционирования системы можно провести, если известны явные зависимости, связывающие искомые характеристики с начальными условиями, параметрами и переменными системы. Однако такие зависимости удаётся получить только для сравнительно простых систем. При усложнении систем исследование их аналитическим методом наталкивается на значительные трудности, которые часто бывают непреодолимыми. Поэтому, желая использовать аналитический метод, в этом случае идут на существенное упрощение первоначальной модели, чтобы иметь возможность изучить хотя бы общие свойства системы. Такое исследование на упрощённой модели аналитическим методом помогает получить ориентировочные результаты для определения более точных оценок другими методами.

Возможности чисто теоретического построения математической модели уменьшаются с ростом сложности и новизны исследуемого объекта. Впрочем, опыт показывает, что нередко даже для широко используемых на практике и, казалось бы хорошо изученных объектов и процессов, чисто аналитическим путём построить удовлетворительную модель не удаётся и это побуждает исследователя к формированию модели преимущественно на экспериментальной основе, т.е. в классе эмпирических (идентифицируемых) моделей.

По степени соответствия модели реальному объекту модели можно разделить: на состоятельные — опирающиеся на законы, характеризующие объект моделирования в области их применимости; аппроксимации — построенные на основе приближенных или эмпирических формул, характеризующих объект (их, в отличие от первых, называют несостоятельными).

### 6.1.2 Идентифицируемые модели

В основе всех ныне весьма многочисленных методов идентификации или опытного отождествления модели с объектом-оригиналом, лежит идея мысленного эксперимента с "чёрным ящиком" (Н. Винер). В предельном (теоретическом) случае "чёрный ящик" представляет собой некую систему, о структуре и внутренних свойствах которой неизвестно решительно ничего. Зато входы, т.е. внешние факторы, воздействующие на этот объект, и выходы, представляющие собой реакции на входные воздействия, доступны для наблюдений (измерений) в течение неограниченного времени. Задача заключается в том, чтобы по наблюдаемым данным о входах и выходах выявить внутренние свойства объекта или, иными словами, построить модель.

Модель чёрного ящика является начальным этапом изучения сложных систем.

Исследование объекта моделирования допускает применение двух стратегий:

1. Осуществляется активный эксперимент. На вход подаются специальные сформированные тестовые сигналы, характер и последовательность которых определена заранее разработанным планом. Преимущество: за счёт оптимального планирования эксперимента необходимая информация о свойствах и характеристиках объекта получается при минимальном объёме первичных экспериментальных данных и соответственно при минимальной трудоёмкости опытных работ. Но цена за это достаточно высока: объект выводится из его естественного состояния (или режима функционирования), что не всегда возможно.
2. Осуществляется пассивный эксперимент. Объект функционирует в своём естественном режиме, но при этом организуются систематические измерения и регистрация значений его входных и выходных переменных. Информацию получают ту же, но необходимый объём данных обычно больше, чем в первом случае.

На практике при построении идентифицируемых (эмпирических) моделей часто целесообразна смешанная стратегия эксперимента. При наличии возможности свободного манипулирования параметрами объекта моделирования проводится активный эксперимент. Его результаты дополняют данными пассивного эксперимента, охватывающего все прочие значимые переменные. "Чёрный ящик" — теоретически граничный случай. На деле есть объём исходной информации. На практике приходится иметь дело с "серым" (отчасти прозрачным) ящиком.

Построение модели сводится к следующим этапам:

1. выбор структуры модели из физических соображений;
2. подгонка параметров к имеющимся данным (оценивание);
3. проверка и подтверждение модели (диагностическая проверка);
4. использование модели но её назначению.

Исходя из перечня научных направлений и разнообразия приложений, по этим этапам нельзя дать каких-либо общих рекомендаций. Структура модели выбирается па основе исходной (априорной) информации о системе и преследуемых целях На практике отыскание подходящей модели может быть достаточно трудной задачей даже для узкой прикладной области.

Различают три основных класса постановки задачи идентификации объекта:

1. Для сложных и слабо изученных объектов системного характера достоверные исходные данные о внутренних свойствах и структурных особенностях обычно отсутствуют.

   Поэтому задача идентификации включает в себя как определение внутренней структуры объекта, так и определение зависимостей, связывающих входы и выходы (обобщённого оператора).

   На начальной стадии моделирования строятся эмпирические идентифицируемые модели (на основе статистической обработки экспериментальных результатов).

2. Второй класс задач идентификации характеризуется тем, что имеются априорные данные о структуре моделируемого объекта, в принципе имеются. Однако не определён вклад компонентов объекта в его результирующие характеристики. Задачи этого класса, связанные с уточнением структуры и оценивания параметров, часто встречаются на практике и характерны для объектов и процессов средней сложности, в частности технологических.
3. Третий класс задач связан с относительно простыми и хорошо изученными объектами, структура которых известна точно и речь идёт только о том, чтобы по экспериментальным данным оценить значения всех или некоторых входящих в исследуемую структуру параметров (параметрическая идентификация). Очевидно, что модели данного класса тесно смыкаются с требующими экспериментального доопределения аналитическими моделями и чёткой границы между ними не существует. Это наиболее массовый класс задач.

Независимо от характера решаемой на основе идентификации задачи, построение модели базируется на результатах измерений соответствующих величин переменных.

Реальные свойства подавляющего большинства сложных объектов, а также неизбежные случайные погрешности измерений, лежащих в основе идентификации, придают последней статистический характер, что влечёт за собой необходимость получения больших объёмов первичных экспериментальных данных с их последующей обработкой. Поэтому на практике построение моделей путём идентификации неизбежно связано с использованием компьютеров, как при получении первичных данных (автоматизация эксперимента), так и для их обработки и использования.

В **Maxima** включено достаточно большое количество средств для решения задач моделирования, параметрической идентификации, исследования моделей.

## 6.2 Статистические методы анализа данных

### 6.2.1 Ввод-вывод матричных данных

Для чтения и записи матричных или потоковых данных в составе **Maxima** предусмотрен пакет numericalio.

Функции пакета рассчитаны на ввод/вывод данных, каждое поле которых предполагается атомом (в смысле **Lisp**), т.е. целых чисел, чисел с плавающей точкой, строк или символов. Атомы воспринимаются numericalio так же, как при интерактивном вводе в консоли или выполнении batch-файла. Возможно использование различных символов-сепараторов для разделения полей данных (параметр $separator_flag$).

Основные функции пакета numericalio:

- $read_matrix(file_name)$ (другие формы вызова — $read_matrix(file_name, separator_flag), read_matrix(S), read_matrix(S, separator_flag)$). Функция $read_matrix$ считывает матрицу из файла. Здесь $file_name$ — имя файла, из которого считываются данные, $S$ — имя потока. Если не указан $separator_flag$, данные предполагаются разделёнными пробелами. Функция возвращает считанный объект.
- $read_list(file_name)$ (другие формы — $read_list(file_name, separator_flag), read_list(S), read_list(S, separator_flag)$) — считывает список из файла или из потока.
- $write_data(X, file_name)$ (другие формы — $write_data(object, file_name, separator_flag), write_data(X, S), write_data(object, S, separator_flag)$) — осуществляет вывод объекта $object$ (списка, матрицы, массива **Lisp** или **Maxima** и др.) в файл file_name (или объекта X в поток S). Матрицы выводятся по столбцам и строкам с использованием пробела или другого символаразделителя (см. $separator_flag$).

Наряду с указанными простыми функциями, используются более специфичные: $read_lisp_array, read_maxima_array, read_hashed_array, read_nested_list$, предназначенные для считывания массивов в формате **Lisp** или **Maxima**, особенности применения которых не рассматриваются в данной книге.

### 6.2.2 Функции Maxima для расчёта описательной статистики

Система **Maxima** содержит ряд функций для выполнения статистических расчётов (описательной статистики), объединённые в пакет descriptive. Функции, входящие в состав descriptive, позволяют выполнить расчёт дисперсии, среднеквадратичного отклонения, медианы, моды и т.п. Названия функций и краткое описание выполняемых ими действий приведены в табл.1.

Таблица 1. Функции пакета descriptive

| Функция | Выполняемые действия | Синтаксис вызова и примечания |
|---|---|---|
| $mean$ | Вычисление среднего | $mean(list)$ или $mean(matrix)$ |
| $geometric_mean$ | Вычисление среднего геометрического | $geometric_mean(list)$ или $geometric_mean(matrix)$ |
| $harmonic_mean$ | Вычисление среднего гармонического | $harmonic_mean(list)$ или $harmonic_mean(matrix)$ |
| $cor$ | Вычисляет корреляционную матрицу | $cor(matrix)$ или $cor(matrix, logical_value)logical_value$ равна $true$ или $false$ (при расчёте по ковариационной матрице) |
| $cov, cov1$ | Вычисляет ковариационную матрицу | $cov1(matrix), cov(matrix)$ |
| $median$ | Вычисляет медиану | $median(list), median(matrix)$ |
| $std, std1$ | Вычисляет среднеквадратичное отклонение (корень квадратный из $var$ или $var1$) | аналогично $var$ |
| $var, var1$ | Вычисляет дисперсию случайной величины | $var1(matrix), var(matrix), var1(list), var(list)$ |
| $central_moment$ | Вычисляет центральный момент порядка $k$ | $central_moment(list, k), central_moment(matrix, k)$ |
| $noncentral_moment$ | Вычисляет момент порядка $k$ | $noncentral_moment(list, k), noncentral_moment(matrix, k)$ |
| $skewness$ | Вычисление асимметрии | $skewness(list), skewness(matrix)$ |
| $kurtosis$ | Вычисление эксцесса | $kurtosis(list), kurtosis(matrix)$ |
| $quantile$ | Вычисление $p$ -квантиля | `quantile(list, p), quantile(matrix, p)` |
| $maxi, mini$ | Выбор наибольшего и наименьшего значения в выборке соответственно | $maxi(list), maxi(matrix), mini(list), mini(matrix)$ |
| $mean_deviation, median_deviation$ | Сумма абсолютных отклонений от среднего или медианы соответственно | Аналогично $mean, median$ |
| $range$ | Размах вариации выборки | $range(list), range(matrix)$ |
| $list_correlations$ | Возвращает список, включающий две матрицы — матрицу, обратную ковариационной, и матрицу частных коэффициентов корреляции | $list_correlations(matrix), list_correlations(matrix, logical_value)$, где $logical_value$ равна $true$ или $false$ (при расчёте по ковариационной матрице) |
| $subsample$ | Аналог функции submatrix | |
| $global_variances$ | Возвращает список, содержащий различные виды дисперсии | $global_variances(matrix)$ |

Построение графических иллюстраций производится при помощи функций $scatterplot$ (непосредственная визуализация данных), $histogram$ (строит гистограмму), $barsplot$ (также строит гистограмму, но по дискретным или нечисловым данным), $boxplot$ (график Бокса- Вискера), $piechart$ (круговая диаграмма). Синтакисис вызова и параметры функций во многом аналогичны компонентам пакета draw (см. примеры ниже).

Основные общие опции (унаследованные от draw):

- terminal — устройство, на которое выводится диаграмма (возможные значения — eps и png, по умолчанию диаграмма выводится на экран, другие варианты);
- file_name — имя файла, в который выводить гистограмму (расширение устанавливается по опции terminal)
- title — основной заготовок;
- xlabel, ylabel — названия (метки) осей,

Рассмотрим пример использования функций пакета descriptive для статистической обработки массивов данных. Данные берем из файлов, входящих в состав

пакета descriptive (файлы biomed.data, wind.data и др.). Перед началом работы загружаем необходимые пакеты descriptive и numericalio. При помощи функции $read_matrix$ считывается матрица, содержащая 100 строк и 5 столбцов.

```
(%i1)   load(descriptive)$ load(numericalio)$
        s:read_matrix (file_search ("wind.data"))$
        length(s);
(%o4)   100
(%i5)   mean(s); /* рассчитываем среднее значение. При обработке
                    матрицы получаем список средних по столбцам.*/
```

$$(\%o5)\ [9.948499999999999, 10.1607, 10.8685, 15.7166, 14.8441]$$

```
(%i6)   median(s);
```

$$(\%o6)\ [10.06, 9.855, 10.73, 15.48, 14.105]$$

```
(%i7)   var(s);
```

$$(\%o7)\ [17.22190675000001, 14.98773651000001, 15.47572875, 32.17651044000001, 24.42307619000001]$$

```
(%i8)   std(s);
```

$$(\%o8)\ [4.149928523480858, 3.871399812729242, 3.933920277534866, 5.672434260526957, 4.941970881136392]$$

```
(%i9)   mini(s);
```

$$(\%o9)\ [0.58, 0.5, 2.67, 5.25, 5.17]$$

```
(%i10)  maxi(s);
```

$$(\%o10)\ [20.25, 21.46, 20.04, 29.63, 27.63]$$

```
(%i11)  mini(%); /* При обработке списка и поиске в нём минимального
                    элемента получаем одно значение! */
```

$$(\%o11)\ 20.04$$

Для построения диаграмм разброса ($xy$-диаграмм) предназначена функция $scatterplot$. Синтаксис вызова:

$$scatterplot(list)$$
$$scatterplot(list, option_1, option_2, \dots)$$
$$scatterplot(matrix)$$
$$scatterplot(matrix, option_1, option_2, \dots)$$

Данные для функции $scatterplot$ могут представляться вектором (списком) или матрицей. Одномерные массивы рассматриваются как временные ряды с равноотстоящими точками.

Основные опции, специфичные для данной функции:

- point_size — размер точки на графике (целое положительное число);
- point_type — вид точки (отсутствие точек — none (-1), dot (0), plus (1), multiply (2), asterisk (3), square (4), filled_square (5), circle (6), filled_circle (7), up_triangle (8), filled_up_triangle (9), down_triangle (10), filled_down_triangle (11), diamant (12), filled_diamant (13));
- color — цвет точки (тот же набор цветов, что и в пакете draw);
- grid — наличие сетки на графике ($true/false$).

Для построения гистограмм используется функция $histogram$ (синтаксис вызова аналогичен $scatterplot$ и основные опции идентичны опциям $scatterplot$). Рассмотрим дополнительные опции, специфичные для $histogram$:

- nclasses (по умолчанию 10) — число классов гистограммы, или список с указанием пределов классов и их число, или только пределы;
- frequency — указывает масштаб шкалы ординат, возможные значения: абсолютный, относительный и процентный (default, absolute, persent);
- htics (default, auto) — формат промежуточных делений на гистограмме, возможные значения — auto, endpoints, intervals, или список меток;

При построении гистограммы доступны также локальные и глобальные опции пакета draw.

Для графического представления описательной статистики служит диаграмма Бокса-Уискера ("ящик-с-усами"), которая является удобным способом наглядно представить статистические данные пятью параметрами: наименьшее и наибольшее значения выборки, нижний, средний и верхний квартили. На данной диаграмме могут быть показаны и выбросы (если они есть).

Для построения диаграмм Бокса-Уискера используется функция $boxplot$. Синтаксис вызова: $boxplot(data)$ или $boxplot(data, option_1, option_2, ...)$.

Параметр $data$ — список или матрица с несколькими столбцами. Опции функции $boxplot$ идентичны опциям $scatterplot$.

Столбчатые диаграммы (обычно частотные) строятся для данных, разбитых на категории, при помощи функции $barsplot$. Эти диаграммы позволяют графически отобразить различия между данными категорий.

Достаточно распространённым способом графического изображения структуры статистических совокупностей является секторная диаграмма, так как идея целого очень наглядно выражается кругом, который представляет всю совокупность. Относительная величина каждого значения изображается в виде сектора круга, площадь которого соответствует вкладу этого значения в сумму значений. Этот вид графиков строится в **Maxima** функцией $piechart$.

Рассмотрим примеры использования графических утилит пакета descriptive.

Для дальнейшего использования считываем данные из файла wind.data (это тестовый файл в составе пакета descriptive, содержит матрицу 100x5).

```
(%i1)   load(descriptive)$ load(numericalio)$
        s:read_matrix (file_search("wind.data"))$
(%i4)   x:makelist(s[k][1],k,1,length(s))$
(%i5)   y:makelist(s[k][2],k,1,length(s))$
(%i6)   m:makelist([x[k],y[k]],k,1,100)$
(%i7)   xy:apply('matrix,m)$
```

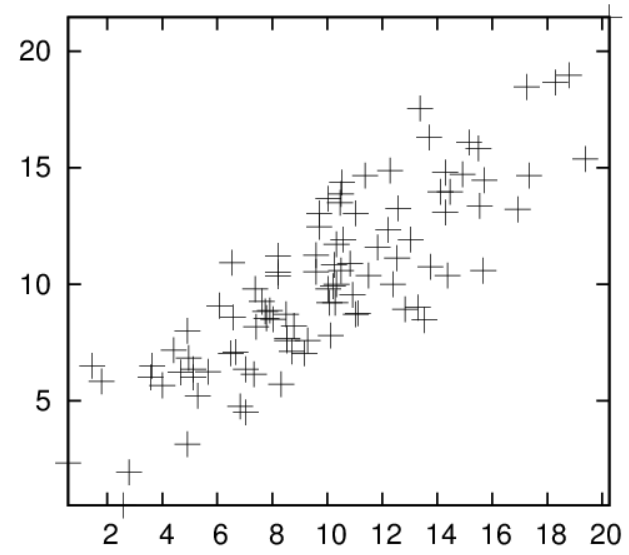


**Рис. 1.** Точечный график

Строим график (точечный) зависимости $y$ от $x$ (см. рис. 1). Результаты сохраняются в файле maxima_out.eps (имя файла — по умолчанию, он создаётся в домашнем каталоге пользователя). Необходимые команды:

```
(%i8)   scatterplot(xy,terminal=eps);
```

Считывая данные из файла pidigits.data, строим гистограмму частотного распределения десятичных знаков числа π (см. рис .2). Результаты сохраняются в файле histogram.eps (имя файла задаётся опцией file_name = "histogram", он создаётся в домашнем каталоге пользователя). Необходимые команды:

```
load (descriptive)$ s1 : read_list (file_search ("pidigits.data"))$
histogram (s1, nclasses=8, title="pi digits", xlabel="digits",
        ylabel="Absolute frequency", fill_color=grey, fill_density=0.6,
        terminal=eps, file_name="histogram")$
```

Следующий пример — график Бокса-Уискера с аннотациями по осям (см. рис. 3). Необходимые команды (результат сохраняется в файле boxwisker.eps, англоязычные наименования заменены на русские переводы):

```
(%i10)  boxplot(s,title = "Test plot", xlabel = "Seasons",
        terminal=eps,file_name="boxwisker")$
```

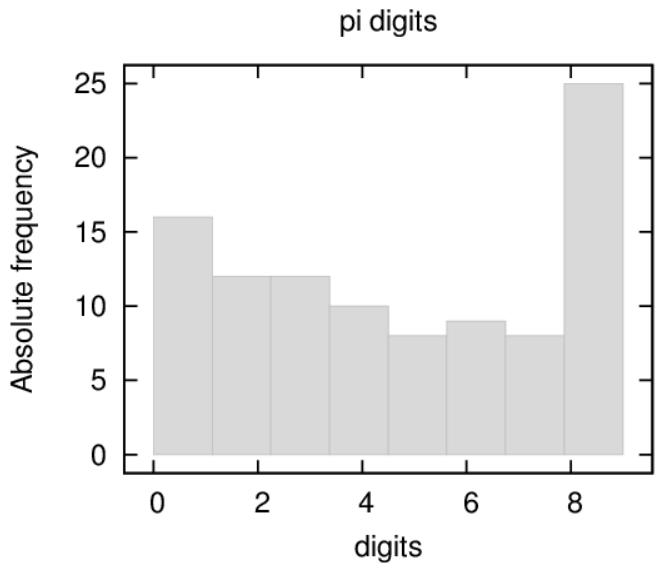


**Рис. 2.** Гистограмма

Пример построения столбчатой диаграммы с использованием функции $barsplot$ — на рис. 4. График построен командой (результат сохраняется в файл barsplot.eps):

```
load (descriptive)$
l1:makelist(random(8),k,1,50)$
l2:makelist(random(8),k,1,100)$
barsplot(l1,l2, box_width=1/2, fill_density=3/4,sample_keys=["A","B"],
        bars_colors=[grey10,grey50], terminal=eps, file_name="barsplot")$
```

Основные опции команды $barsplot$ :

- box_width — относительная ширина прямоугольников (по умолчанию 3/4, величина в пределах [0, 1]);
- grouping — индикатор, показывающий, как представляются множественные образцы (возможные значения clustered и stacked);
- groups_gap — натуральное число, представляющее разрыв между двумя соседними группами (относительная величина, по умолчанию 1);
- bars_colors — список цветов для множественных образцов (по умолчанию [ ]);
- start_at — указывает начало графика по оси $x$ (по умолчанию 0).

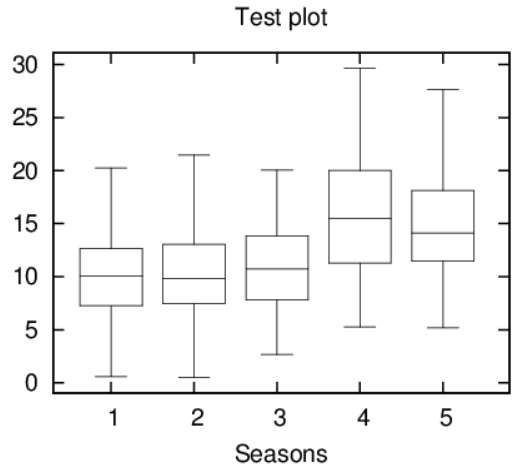


**Рис. 3.** График Бокса-Уискера

С функцией $barsplot$ можно использовать и опции пакета draw.

Для построения круговых (секторных) диаграмм используется функция $piechart$ .

**Пример** использования $piechart$ :

```
load (descriptive)$
s1 : read_list (file_search ("pidigits.data"))$
piechart(s1, xrange=[-1.1, 1.3], yrange=[-1.1, 1.1],
        title="Digit frequencies in pi")$
```

Цвета секторов и радиус диаграммы описываются опциями sector_colors и pie_radius.

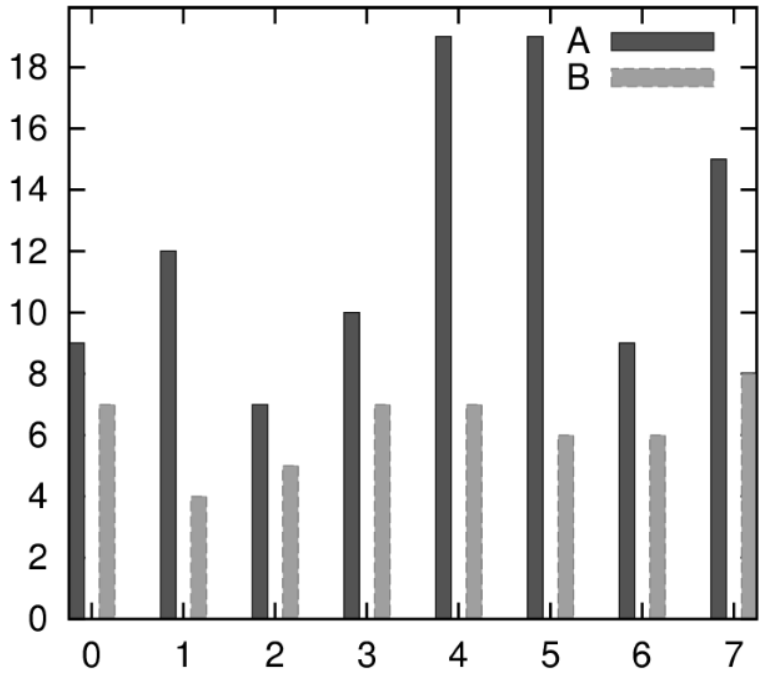


**Рис. 4.** Гистограмма распределения в группах

К сожалению, базовая программа вывода графики **Maxima** — gnuplot — написана очень давно, и воспринимает кириллические символы только в кодировках KOI8-R или KOI8-U (в последних версиях — и utf8). Возможным решением (принятым для построения графиков в этой книге) является создание файла .gnuplot, содержащего следующие команды: **set encoding koi8r или set encoding utf8.**

Однако и в этом случае может возникнуть необходимость в редактировании файла maxout.gnuplot, содержащего команды для построения последнего графика.

#### 6.2.3 Проверка статистических гипотез

Для проверки статистических гипотез в **Maxima** включён пакет stats. Он позволяет, в частности, проводить сопоставление средних или дисперсий двух выборок.

Предусмотрена и проверка нормальности распределения, а также ряд других стандартных тестов. для использования stats пакет необходимо загрузить командой load("stats"); необходимые пакеты descriptive и distrib загружаются автоматически.

Функции пакета stats возвращают данные типа $inference_result$ . Объекты этого типа содержат необходимые результаты для анализа статистических распределений и проверки гипотез.

Функция $test_mean$ позволяет оценить среднее значение и доверительный интервал по выборке. Синтаксис вызова: test_mean ( $x$ ) или test_mean ( $x, option_1, option_2, ...$).

Функция $test_mean$ использует проверку по критерию Стьюдента. Аргумент $x$ — список или одномерная матрица с тестируемой выборкой. Возможно также использование центральной предельной теоремы (опция $asymptotic$). Опции $test_mean$ :

- 'mean, по умолчанию 0, ожидаемое среднее значение;
- 'alternative, по умолчанию 'twosided, вид проверяемой гипотезы (возможные значения 'twosided, 'greater и 'less);
- 'dev, по умолчанию 'unknown, величина среднеквадратичного отклонения, если оно известно ('unknown или положительное выражение);
- 'conflevel, по умолчанию 95/100, уровень значимости для доверительного интервала (величина в пределах от 0 до 1);
- 'asymptotic, по умолчанию $false$ , указывает какой критерий использовать ($t$ -критерий или центральную предельную теорему).

Результаты, которые возвращает функция:

- 'mean_estimate — среднее по выборке;
- 'conf_level — уровень значимости, выбранный пользователем;
- 'conf_interval — оценка доверительного интервала;
- 'method — использованная процедура;
- 'hypotheses — проверяемые статистические гипотезы (нулевая $H_0$ и альтернативная $H_1$);
- 'statistic — число степеней свободы для проверки нулевой гипотезы;
- 'distribution — оценка распределения распределения выборки;
- 'p_value — вероятность ошибочного выбора гипотезы $H_1$, если выполняется $H_0$.

**Примеры** использования $test_mean$ :

Выполняется $t$ -тест с неизвестной дисперсией. Нулевая гипотеза $H_0$: среднее равно 50 против альтернативной гипотезы $H_1$: среднее меньше 50; в соответствии с результатами расчёта, величина вероятности $p$ слишком велика, чтобы отвергнуть $H_0$.

```
(%i1)   load("stats")$
(%i2)   data: [78,64,35,45,45,75,43,74,42,42]$
(%i3)   test_mean(data,'conflevel=0.9,'alternative='less,'mean=50);
```

$$(\%o3)\quad \begin{pmatrix} MEAN\ TEST \\ mean\_estimate = 54.3 \\ conf\_level = 0.9 \\ conf\_interval = [-\infty, 61.51314273502714] \\ method = Exact\ t-test.\ Unknown\ variance. \\ hypotheses = H0:\ mean = 50,\ H1:\ mean < 50 \\ statistic = .8244705235071678 \\ distribution = [student\_t, 9] \\ p\_value = .7845100411786887 \end{pmatrix}$$

Следующий тест — проверка гипотезы $H_0$ (среднее равно 50) против альтернативной гипотезы $H_1$ среднее по выборке отлично от 50. В соответствии с величиной $p << 1$ нулевая гипотеза. Данный тест применяется для больших выборок.

```
(%i1)   load("stats")$
(%i2)   test_mean([36,118,52,87,35,256,56,178,57,57,89,34,25,98,35,
                   98,41,45,198,54,79,63,35,45,44,75,42,75,45,45,
                   45,51,123,54,151],'asymptotic=true,'mean=50);
```

$$(\%o2)\quad \begin{pmatrix} MEAN\ TEST \\ mean\_estimate = 74.88571428571429 \\ conf\_level = 0.95 \\ conf\_interval = [57.72848600856193, 92.04294256286664] \\ method = Large\ sample\ z-test.\ Unknown\ variance. \\ hypotheses = H0:\ mean = 50,\ H1:\ mean \neq 50 \\ statistic = 2.842831192874313 \\ distribution = [normal, 0, 1] \\ p\_value = .004471474652002261 \end{pmatrix}$$

Функция $test_means_difference$ позволяет проверить, принадлежат ли выборки $x_1$ и $x_2$ к одной генеральной совокупности.

Синтаксис вызова: $test_means_difference(x_1, x_2)$ или t$est_means_difference(x_1, x_2, option_1, option_2, ...)$.

Данная функция выполняет t-тест для сравнения средних по выборкам $x_1$ и $x_2$ ($x_1, x_2$ — списки или одномерные матрицы). Сравнение выборок может проводиться также на основании центральной предельной теоремы (для больших выборок). Опции функции $test_means_difference$ такие же, как и для $test_mean$ , кроме оценок среднеквадратичных отклонений выборок (если они известны) Список опций:

- 'alternative, по умолчанию 'twosided, вид проверяемой гипотезы (возможные значения 'twosided, 'greater и 'less);
- 'dev1, 'dev2, по умолчанию 'unknown, величины среднеквадратичных отклонений для выборок $x_1$ и $x_2$, если они известны ('unknown или положительное

выражение);
- 'conflevel, по умолчанию 95/100, уровень значимости для доверительного интервала (величина в пределах от 0 до 1);
- 'asymptotic, по умолчанию $false$, указывает какой критерий использовать ($t$ -критерий или центральную предельную теорему).

Вывод результатов $test_means_difference$ не отличается от вывода результатов $test_mean$.

**Примеры** использования $test_means_difference$: Для двух малых выборок проверяется гипотеза $H_0$ от равенстве средних против альтернативной гипотезы $H_1$: различие математических ожиданий статистически значимо, т.е. выборки принадлежат к разным генеральным совокупностям.

```
(%i1)   load("stats")$
(%i2)   x: [20.4,62.5,61.3,44.2,11.1,23.7]$
(%i3)   y: [1.2,6.9,38.7,20.4,17.2]$
(%i4)    test_means_difference(x,y,'alternative='greater);
```

$$(\%o4)\ \left(\begin{array}{c} DIFFERENCE\ OF\ MEANS\ TEST \\ diff\_estimate = 20.31999999999999 \\ conf\_level = 0.95 \\ conf\_interval = [-.04597417812881588, \infty] \\ method = Exact\ t-test.\ Welch\ approx. \\ hypotheses = H0:\ mean1 = mean2,\ H1:\ mean1 > mean2 \\ statistic = 1.838004300728477 \\ distribution = [student\_t, 8.62758740184604] \\ p\_value = .05032746527991905 \end{array}\right)$$

Оценка доверительного интервала для дисперсии выборки проводится при помощи функции $test_variance$.

Синтаксис вызова: $test_variance(x)$ или $test_variance(x, option_1, option_2, ...)$

Данная функция использует тест $\chi^2$. Предполагается, что распределение выборки $x$ нормальное. Опции функции $test_variance$:

- 'mean, по умолчанию 'unknown, оценка математического ожидания (среднее по выборке), если оно известно;
- 'alternative, по умолчанию 'twosided, вид проверяемой гипотезы (возможные значения 'twosided, 'greater и 'less);
- 'variance, по умолчанию 1, это оценка дисперсии выборки для сравнения с фактической дисперсией;
- 'conflevel, по умолчанию 95/100, уровень значимости для доверительного интервала (величина в пределах от 0 до 1).

Основной результат, возвращаемый функцией — оценка дисперсии выборки $var_estimate$ и доверительный интервал для неё.

**Пример**: Проверка, отличается ли дисперсия выборки с неизвестным математическим ожиданием от значения 200.

```
(%i1)   load("stats")$
(%i2)   x: [203,229,215,220,223,233,208,228,209]$
(%i3)   test_variance(x,'alternative='greater,'variance=200);
```

$$(\%o3)\ \left(\begin{array}{c} VARIANCE\ TEST \\ var\_estimate = 110.75 \\ conf\_level = 0.95 \\ conf\_interval = [57.13433376937479, \infty] \\ method = Variance\ Chi-square\ test.\ Unknown\ mean. \\ hypotheses = H0:\ var\ =\ 200,\ H1:\ var\ >\ 200 \\ statistic = 4.430000000000001 \\ distribution = [chi2, 8] \\ p\_value = .8163948512777688 \end{array}\right)$$

Сравнение дисперсий двух выборок проводится при помощи функции $test_variance$ (синтаксис вызова $test_variance_ratio(x_1, x_2)$ или $test_variance_ratio(x_1, x_2, option_1, option_2, ...)$.

Данная функция предназначена для сопоставления дисперсий двух выборок с нормальным распределением по критерию Фишера ($F$-тест). Аргументы $x_1$ и $x_2$ — списки или одномерные матрицы, содержащие независимые выборки.

Опции функции $test_variance_ratio$:

- 'mean1, 'mean2, по умолчанию 'unknown, оценки математических ожиданий выборок $x_1$ и $x_2$, если они известны;
- 'alternative, по умолчанию 'twosided, вид проверяемой гипотезы (возможные значения 'twosided, 'greater и 'less);
- 'conflevel, по умолчанию 95/100, уровень значимости для доверительного интервала (величина в пределах от 0 до 1).

Основной результат, возвращаемый функцией $test_variance_ratio$ — отношение дисперсий выборок $ratio_estimate$.

**Пример**: проверяется гипотеза о равенстве дисперсий двух выборок по сравнению с альтернативной гипотезой о том, что дисперсия первой больше, чем дисперсия второй.

```
(%i1)   load("stats")$
(%i2)   x: [20.4,62.5,61.3,44.2,11.1,23.7]$
(%i3)   y: [1.2,6.9,38.7,20.4,17.2]$
(%i4)   test_variance_ratio(x,y,'alternative='greater);
```

$$(\%o4)\quad \left(\begin{array}{c} VARIANCE\ RATIO\ TEST \\ ratio\_estimate = 2.316933391522034 \\ conf\_level = 0.95 \\ conf\_interval = [.3703504689507263, \infty] \\ method = Variance\ ratio\ F-test.\ Unknown\ means. \\ hypotheses = H0:\ var1\ =\ var2,\ H1:\ var1\ >\ var2 \\ statistic = 2.316933391522034 \\ distribution = [f, 5, 4] \\ p\_value = .2179269692254463 \end{array}\right)$$

При отсутствии представлений о распределении выборки может использоваться непараметрический тест для сравнения средних. Оценка медианы непрерывной выборки проводится при помощи функции $test_sign$. Синтаксис вызова $test_sign(x)$ или $test_sign(x, option_1, option_2, ...)$.

Функция $test_sign$ допускает две опции: $alternative$ (аналогично $test_mean$) и $median$ (по умолчанию 0, или оценка значения медианы для проверки статистической значимости).

Результаты, которые возвращает функция:

- 'med_estimate: медиана выборки;
- 'method: использованная процедура;
- 'hypotheses: проверяемые статистические гипотезы (нулевая $H_0$ и альтернативная $H_1$);
- 'statistic: число степеней свободы для проверки нулевой гипотезы;
- 'distribution: оценка распределения распределения выборки;
- 'p_value: вероятность ошибочного выбора гипотезы $H_1$, если выполняется $H_0$.

**Пример**: проверка гипотезы $H_0$ о равенстве медианы выборки 6, против альтернативной гипотезы $H_1$: медиана больше 6.

```
(%i1)   load("stats")$
(%i2)   x: [2,0.1,7,1.8,4,2.3,5.6,7.4,5.1,6.1,6]$
(%i3)   test_sign(x,'median=6,'alternative='greater);
```

$$(\%o3)\quad \left(\begin{array}{c} SIGN\ TEST \\ med\_estimate = 5.1 \\ method = Non\ parametric\ sign\ test. \\ hypotheses = H0:\ median\ =\ 6,\ H1:\ median\ >\ 6 \\ statistic = 7 \\ distribution = [binomial, 10, 0.5] \\ p\_value = .05468749999999989 \end{array}\right)$$

Аналогичная функция — $test_signed_rank(x)$ (либо с указанием опций $test_signed_rank(x, option_1, option_2, ...)$), которая использует тест правила знаков Вилкоксона для оценки гипотезы о медиане непрерывной выборки. Опции и результаты функции $test_signed_rank$ такие же, как и для функции $test_sign$.

**Пример**: проверка гипотезы $H_0$: медиана равна 15 против альтернативной гипотезы $H_1$: медиана больше 15.

```
(%i1)   load("stats")$
(%i2)   x: [17.1,15.9,13.7,13.4,15.5,17.6]$
(%i3)   test_signed_rank(x,median=15,alternative=greater);
```

$$(\%o3)\quad \left(\begin{array}{c} SIGNED\ RANK\ TEST \\ med\_estimate = 15.7 \\ method = Exacttest \\ hypotheses = H0:\ med\ =\ 15,\ H1:\ med\ >\ 15 \\ statistic = 14 \\ distribution = [signed\_rank, 6] \\ p\_value = 0.28125 \end{array}\right)$$

Непараметрическое сравнение медиан двух выборок реализовано в одной функции — $test_rank_sum$. В данной функции используется тест Вилкоксона-Манна-Уитни. $U$-критерий Манна-Уитни — непараметрический метод проверки гипотез, часто использующийся в качестве альтернативы $t$-тесту Стьюдента. Обычно этот тест используется для сравнения медиан двух распределений $x_1$ и $x_2$, не являющихся нормальными (отсутствие нормальности не позволяет применить $t$ - тест).

Синтаксис вызова: $test_rank_sum(x_1, x_2)$ или $test_rank_sum(x_1, x_2, option_1)$.

Функция допускает лишь одну опцию: $alternative$ (аналогично $test_means_difference$).

Результаты, которые возвращает функция:

- 'method: использованная процедура;
- 'hypotheses: проверяемые статистические гипотезы (нулевая $H_0$ и альтернативная $H_1$);
- 'statistic: число степеней свободы для проверки нулевой гипотезы;
- 'distribution: оценка распределения распределения выборки;
- 'p_value: вероятность ошибочного выбора гипотезы $H_1$, если выполняется $H_0$.

**Пример**: проверка, одинаковы ли медианы выборок $x_1$ и $x_2$.

```
(%i1)   load("stats")$
(%i2)   x:[12,15,17,38,42,10,23,35,28]$
(%i3)   y:[21,18,25,14,52,65,40,43]$
(%i4)   test_rank_sum(x,y);
```

$$(\%o4)\begin{pmatrix} RANK\ SUM\ TEST \\ method = Exact\ test \\ hypotheses = H0:\ med1 = med2,\ H1:\ med1\ \#\ med2 \\ statistic = 22 \\ distribution = [rank\_sum, 9, 8] \\ p\_value = .1995886466474702 \end{pmatrix}$$

Для выборок большего объёма распределение выборок приблизительно нормальное. Сравниваем гипотезы $H_0$: медиана 1 = медиана 2 и $H_1$: медиана 1 < медиана 2.

```
(%i1)   load("stats")$
(%i2)   x: [39,42,35,13,10,23,15,20,17,27]$
(%i3)   y: [20,52,66,19,41,32,44,25,14,39,43,35,19,56,27,15]$
(%i4)   test_rank_sum(x,y,'alternative='less);
```

$$(\%o4)\begin{pmatrix} RANK\ SUM\ TEST \\ method = Asymptotic\ test.\ Ties \\ hypotheses = H0:\ med1 = med2,\ H1:\ med1\ <\ med2 \\ statistic = 48.5 \\ distribution = [normal, 79.5, 18.95419580097078] \\ p\_value = .05096985666598441 \end{pmatrix}$$

Проверка нормальности распределения осуществляется функцией $test_normality(x)$. В этой функции реализован тест Шапиро-Уилка. Выборка $x$ (список или одномерная матрица) должна быть размером не менее 2, но не более 5000 элементов (иначе выдаётся сообщение об ошибке). Функция возвращает два значения: statistic — величина $W$-статистики и величина вероятности $p$ (если $p$ больше принятого уровня значимости, нулевая гипотеза о нормальности распределения выборки $x$ не отвергается). Статистика $W$ характеризует близость выборочного распределения к нормальному (чем ближе $W$ к 1, тем меньше вероятность ошибочно принять гипотезу о нормальности распределения).

**Пример**: проверка гипотезы о нормальном распределения генеральной совокупности по заданной выборке.

```
(%i1)   load("stats")$
(%i2)   x:[12,15,17,38,42,10,23,35,28]$
(%i3)   test_normality(x);
```

$$(\%o3)\begin{pmatrix} SHAPIRO\ -\ WILK\ TEST \\ statistic = .9251055695162439 \\ p\_value = .4361763918860427 \end{pmatrix}$$

**2.4 Расчёт коэффициентов линейной регрессии**

Коэффициенты и оценка статистической значимости для простейшей линейной регрессии могут оцениваться при помощи функции $simple_linear_regression$ из пакета stats. Функция вычисляет коэффициенты и параметры линейной регрессии $y = a_0 + a_1 \cdot x$ (т.е. только простейшей).

Синтаксис вызова: $simple_linear_regression(x)$ или $simple_linear_regression(x, option_1)$.

Опции функции $simple_linear_regression : conflevel$ (уровень значимости, обычно 0.95, см. выше) и $regressor$ (по умолчанию $x$, имя независимой переменной). Рассматриваемая функция выводит большое количество статистических параметров:

1. 'model: полученное уравнение регрессии;
2. 'means: среднее;
3. 'variances: дисперсии обоих переменных;
4. 'correlation: коэффициент корреляции;
5. 'adc: коэффициент детерминации;
6. 'a_estimation: оценка параметра $a$;
7. 'a_conf_int: доверительный интервал для $a$;
8. $'b_estimation$: оценка параметра b;
9. 'b_conf_int: доверительный интервал для $b$;
10. 'hypotheses: нулевая и альтернативная гипотеза относительно параметра $b$;
11. 'statistic: статистические характеристики выборки, использованные для проверки нулевой гипотезы;
12. 'distribution: распределение выборки;
13. 'p_value: величина вероятности для проверки гипотезы о статистической значимости $b$;
14. 'v_estimation: оценка остаточной дисперсии;
15. 'v_conf_int: доверительный интервал для остаточной дисперсии
16. 'cond_mean_conf_int: доверительный интервал для среднего;
17. 'new_pred_conf_int: доверительный интервал для нового предсказания;
18. 'residuals: список, содержащий остатки.

По умолчанию на консоль выводятся только параметры 1, 4, 14, 9, 10, 11, 12, и 13 в этом списке. Остальные параметры скрыты, но доступ к ним обеспечивается при помощи функций $items_inference$ или $take_inference$.

Задаёмся исходными данными

```
(%i9)   s:[[125,140.7], [130,155.1], [135,160.3], [140,167.2], [145,169.8]]$
```

Вычисляем коэффициенты и прочие параметры регрессии

```
(%i10)  z:simple_linear_regression(s,conflevel=0.99);
```

(%o10)

$$\left(\begin{array}{c} SIMPLE\ LINEAR\ REGRESSION \\ model = 1.405999999999986\ x - 31.18999999999804 \\ correlation = 0.96116852552552 \\ v\_estimation = 13.57966666666665 \\ b\_conf\_int = [0.044696336625253, 2.767303663374718] \\ hypotheses = H0 : b = 0,\ H1 : b\#0 \\ statistic = 6.032686683658114 \\ distribution = [student\_t, 3] \\ p\_value = 0.0038059549413203 \end{array}\right)$$

```
(%i11)  z:simple_linear_regression(s,conflevel=0.95);
```

(%o11)

$$\left(\begin{array}{c} SIMPLE\ LINEAR\ REGRESSION \\ model = 1.405999999999986\ x - 31.18999999999804 \\ correlation = 0.96116852552552 \\ v\_estimation = 13.57966666666665 \\ b\_conf\_int = [0.66428743645021, 2.147712563549759] \\ hypotheses = H0 : b = 0,\ H1 : b\#0 \\ statistic = 6.032686683658114 \\ distribution = [student\_t, 3] \\ p\_value = 0.0038059549413203 \end{array}\right)$$

Некоторые дополнительные параметры:

```
(%i5)   take_inference(model,z), x=133;
```

(%o5) 155.808

```
(%i6)   take_inference(means,z);
```

(%o6) $[135.0, 158.62]$

```
(%i7)   take_inference(new_pred_conf_int,z), x=133;
```

(%o7) $[132.0728595995113, 179.5431404004887]$

Графическая иллюстрация построенной линейной зависимости см. на рис. 5. Использованная команда:

```
(%i11)  plot2d([[discrete,s], take_inference(model,z)],
        [x,120,150],[style,[points],[lines]],[gnuplot_term,ps],
        [gnuplot_out_file, "regress.eps"])$
```

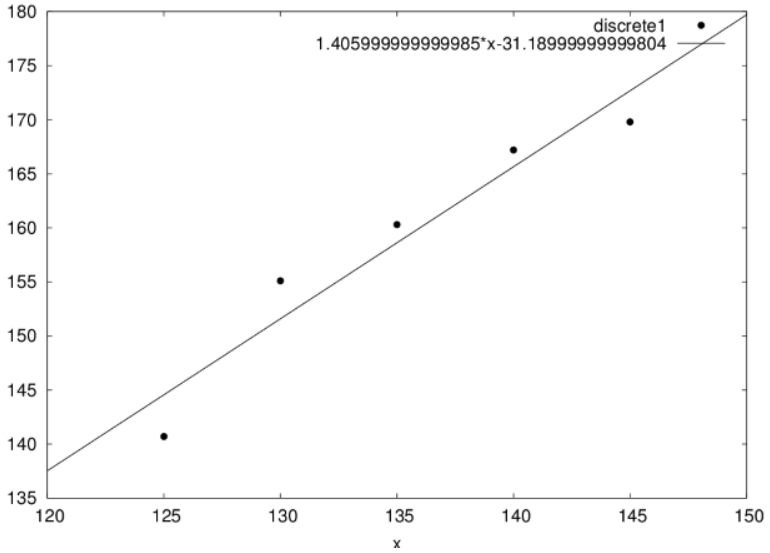


**Рис. 5.** Простая линейная регрессия

### 2.5 Использование метода наименьших квадратов

Пакет **Maxima** включает мощный модуль для линейного и нелинейного оценивания параметров различных моделей с использованием метода наименьших квадратов — пакет lsqares.

Основная функция пакета lsqares — это функция $lsquares_estimates$.

Синтаксис вызова: $lsquares_estimates(D, x, e, a)$ или $lsquares_estimates(D, x, e, a, initial = L, tol = t)$

Функция предназначена для оценки параметров, лучше всего соответствующих уравнению $e$ в переменных $x$ и $a$ по набору данных D, которые определяются методом методом наименьших квадратов. Функция $lsquares_estimates$ сначала пытается отыскать точное решение, и если это не удаётся, ищет приблизительное решение. Возвращаемое значение — список вида $[a = ..., b = ..., c = ...]$. Элементы списка обеспечивают минимум среднеквадратичной ошибки. Данные $D$ должны быть матрицей. Каждый ряд — одна запись или один случай, каждый столбец соответствует значениям некоторой переменной.

Список переменных $x$ дает название для каждого столбца $D$ (даже для столбцов, которые не входят в анализ). Список параметров содержит названия параметров, для которых отыскиваются оценки. Уравнение $e$ является выражением или уравнением в переменных $x$ и $a$; если $e$ записано не в форме уравнения, его рассматривают как уравнение $e = 0$. Если некоторое точное решение может быть найдено при помощи $solve$, данные $D$ могут содержать и нечисловые значения.

Дополнительные аргументы $lsquares_estimates$ определены как уравнения и передаются "дословно" функции $lbfgs$, которая используется, чтобы найти оценки численным методом, когда точный результат не найден. Однако, если никакое точное решение не найдено, у каждого элемента $D$ должно быть числовое значение, в том числе константы (такие как $\%pi$ и $\%e$) или числовые литералы (целые числа, рациональные, с плавающей точкой, и с плавающей точкой повышенной точности). Численные расчеты выполняются с обычной арифметикой с плавающей точкой, таким образом все другие виды чисел преобразуются к значениям с плавающей точкой. Для работы с $lsquares_estimates$ необходимо загрузить эту функцию командой load($lsquares$).

**Пример** (точное решение):

```
(%i1)   load (lsquares)$
(%i2)   M:matrix([1,1,1],[3/2,1,2],[9/4,2,1],[3,2,2],[2,2,1]);
```

$$(\%o2)\ \begin{pmatrix} 1 & 1 & 1 \\ \frac{3}{2} & 1 & 2 \\ \frac{9}{4} & 2 & 1 \\ 3 & 2 & 2 \\ 2 & 2 & 1 \end{pmatrix}$$

```
(%i3)   lsquares_estimates(M,[z,x,y],(z+D)^2=A*x+B*y+C,[A,B,C,D]);
```

$$(\%o3)\ [[A = -\frac{59}{16}, B = -\frac{27}{16}, C = \frac{10921}{1024}, D = -\frac{107}{32}]]$$

Другой пример (точное решение отсутствует, отыскивается приближенное):

```
(%i1)   load (lsquares)$
        M : matrix ([1, 1], [2, 7/4], [3, 11/4], [4, 13/4]);
```

$$(\%o2)\ \begin{pmatrix} 1 & 1 \\ 2 & \frac{7}{4} \\ 3 & \frac{11}{4} \\ 4 & \frac{13}{4} \end{pmatrix}$$

```
(%i3) lsquares_estimates(M,[x,y],y=a*x^b+c,[a,b,c],initial=[3,3,3], iprint=[-1,0]);
```

$$(\%o3)\ [[a = 1.375751433061394, b = .7148891534417651, c = -.4020908910062951]]$$

Для расчёта невязок для уравнения $e$ при подстановке в него данных, содержащихся в матрице $D$, можно использовать функцию $lsquares_residuals(D, x, e, a)$ (смысл параметров тот же, что и для функции $lsquares_estimates$).

**Пример** использования функции $lsquares_estimates$ и $lsquares_residuals$ (те же данные, что использованы для расчёта параметров простой линейной регрессии):

```
(%i1)   load (lsquares)$
(%i2)   s:[[125,140.7],[130,155.1],[135,160.3],[140,167.2],[145,169.8]];
```

$$(\%o2)\ [[125, 140.7], [130, 155.1], [135, 160.3], [140, 167.2], [145, 169.8]]$$

```
(%i3)   D:apply(matrix,s);
```

$$(\%o3)\ \begin{pmatrix} 125 & 140.7 \\ 130 & 155.1 \\ 135 & 160.3 \\ 140 & 167.2 \\ 145 & 169.8 \end{pmatrix}$$

```
(%i4)   a : lsquares_estimates(D,[y,x],y = A+B*x, [A,B]);
```

$$(\%o4)\ [[A = \frac{8231525}{267474}, B = \frac{87875}{133737}]]$$

```
(%i5)   float(%);
```

$$(\%o5)\ [[A = 30.77504729431646, B = 0.65707321085414]]$$

```
(%i6)   lsquares_residuals (D, [y,x], y = A + B*x, first(a));
```

$$(\%o6)\ [1.774751938506171, -2.687102297793416, -1.103882994234965, -0.63768814912851, 2.653921502650718]$$

Остальные функции, входящие в состав пакета lsquares, по синтаксису использования и идее реализации аналогичны приведенным (см. документацию разработчика).

### 6.3 Моделирование динамических систем

Многие модели, основанные на нелинейных дифференциальных уравнениях, демонстрируют совершенно удивительные свойства, причем решение большинства из них можно получить лишь численно.

Модели, основанные на задачах Коши для ОДУ, часто называют динамическими системами, подчёркивая, что, как правило, они содержат производные по времени $t$ и описывают динамику некоторых параметров. Проблемы, связанные с динамическими системами, на самом деле весьма разнообразны и зачастую не сводятся к простому интегрированию ОДУ.

#### 6.3.1 Моделирование системы химических реакций

Рассмотрим другой пример. Исследуем систему из трех дифференциальных уравнений, описывающих модель химической кинетики:

$$A \to B$$
$$B \to C$$

Система соответствующих дифференциальных уравнений

$$\frac{dc_A}{dt} = -k_1 c_A$$
$$\frac{dc_B}{dt} = k_1 c_A - k_2 c_B$$
$$\frac{dc_C}{dt} = k_2 c_B$$

Начальные условия:

$$c_A = 1, c_B = 0, c_C = 0$$

Результаты решения приведены на рис. 6.

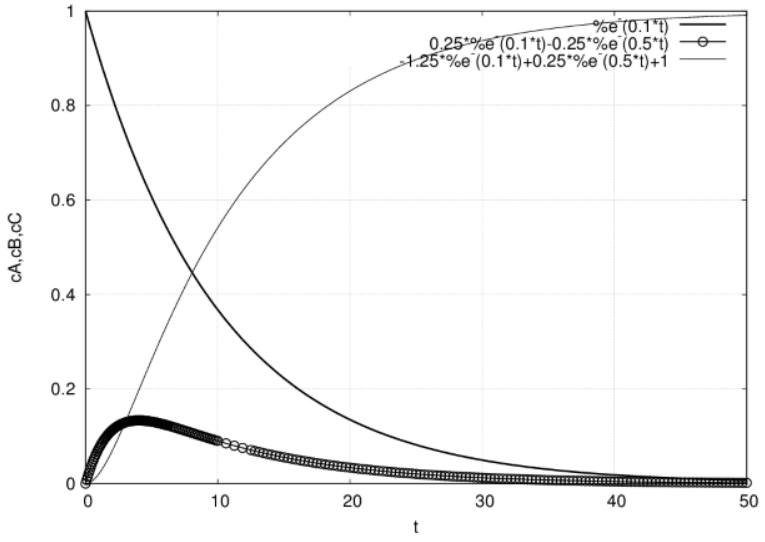


**Рис. 6.** Кинетика химических реакций

```
(%i1)   eq1:'diff(ca(t),t)=-k1*ca(t);
        eq2:'diff(cb(t),t)=k1*ca(t)-k2*cb(t);
        eq3:'diff(cc(t),t)=k2*cb(t);
```

$$(\%o1)\ \frac{d}{dt}\,\mathrm{ca}\,(t) = -k1\,\mathrm{ca}\,(t)$$
$$(\%o2)\ \frac{d}{dt}\,\mathrm{cb}\,(t) = k1\,\mathrm{ca}\,(t) - k2\,\mathrm{cb}\,(t)$$
$$(\%o3)\ \frac{d}{dt}\,\mathrm{cc}\,(t) = k2\,\mathrm{cb}\,(t)$$

```
(%i4)   atvalue(ca(t),t=0,1);
```

$$(\%o4)\ 1$$

```
(%i5)   atvalue(cb(t),t=0,0);
        atvalue(cc(t),t=0,0);
```

$$(\%o5)\ 0$$
$$(\%o6)\ 0$$

```
(%i7)   sol:desolve([eq1,eq2,eq3],[ca(t),cb(t),cc(t)]);
```

$$(\%o7)\ [\mathrm{ca}\,(t) = e^{-k1\,t}, \mathrm{cb}\,(t) = \frac{k1\,e^{-k1\,t}}{k2-k1} - \frac{k1\,e^{-k2\,t}}{k2-k1}, \mathrm{cc}\,(t) = \frac{k1\,e^{-k2\,t}}{k2-k1} - \frac{k2\,e^{-k1\,t}}{k2-k1} + 1]$$

```
(%i8)   ratsimp(sol);
```

$$(\%o8)\ [\mathrm{ca}\,(t) = e^{-k1\,t}, \mathrm{cb}\,(t) = \frac{\left(k1\,e^{k2\,t} - k1\,e^{k1\,t}\right) e^{-k2\,t-k1\,t}}{k2-k1},$$
$$\mathrm{cc}\,(t) = \frac{\left(\left((k2-k1)\,e^{k1\,t} - k2\right) e^{k2\,t} + k1\,e^{k1\,t}\right) e^{-k2\,t-k1\,t}}{k2-k1}]$$

```
(%i9)   k1:0.1; k2:0.5; ev((sol));
```

$$(\%o9)\ 0.1$$
$$(\%o10)\ 0.5$$
$$(\%o11)\ [\mathrm{ca}\,(t) = e^{-0.1\,t}, \mathrm{cb}\,(t) = 0.25\,e^{-0.1\,t} - 0.25\,e^{-0.5\,t}, \mathrm{cc}\,(t) =$$
$$-1.25\,e^{-0.1\,t} + 0.25\,e^{-0.5\,t} + 1]$$

```
(%i12)  plot2d([%e^(-0.1*t),0.25*%e^(-0.1*t)-0.25*%e^(-0.5*t),
-1.25*%e^(-0.1*t)+0.25*%e^(-0.5*t)+1],[t,0,50],
        [gnuplot_preamble,"set grid"],
        [gnuplot_term,"png size 500,500"],
        [gnuplot_out_file,"chem.png"]);
```

Несколько более сложная задача — моделирование кинетики параллельно-последовательных реакций, протекающих по схеме:

$$A \to B$$
$$B + C \to A$$
$$B \to C$$

В зависимости от констант скорости химических реакций данная система может быть довольно жёсткой.

**Пример** командного файла для решения жёсткой системы ОДУ в **Maxima** (данная система нелинейна, поэтому используем метод Рунге-Кутта, однако расчёты затрудняются жёсткостью системы):

```
load("dynamics");
load("draw");
k1:0.1; k2:100; k3:10;
eq1:-k1*ca+k3*cb*cc;
eq2:k1*ca-k3*cb*cc-k2*cb;
eq3:k2*cb;
t_range:[t,0,100,0.01];
sol: rk([eq1,eq2,eq3],[ca,cb,cc],[1,0,0],t_range)$
len:length(sol);
t:makelist(sol[k][1],k,1,len)$
ca:makelist(sol[k][2],k,1,len)$
cb:makelist(sol[k][3],k,1,len)$
cc:makelist(sol[k][4],k,1,len)$
draw2d(title="Chemical system",xlabel="ca",ylabel="cb",
        grid=true,points_joined =true,points(t,ca),
        points(t,cb),points(t,cc),terminal=eps);
```

Данная система достаточно трудно решается при помощи функции $rk$ . Увеличение констант до $k2 = 1000$ и $k3 = 100$ делает задачу практически неразрешимой средствами пакета dynamics.

Простейшим классическим примером существования автоколебаний в системе химических реакций является тримолекулярная модель "Брюсселятор", предложенная в Брюсселе Пригожиным и Лефевром (1965). Основной целью при изучении этой модели было установление качественных типов поведения, совместимых с фундаментальными законами химической и биологической кинетики. В этом смысле брюсселятор играет роль базовой модели, такую же как гармонический осциллятор в физике, или модели Вольтерра в динамике популяций.

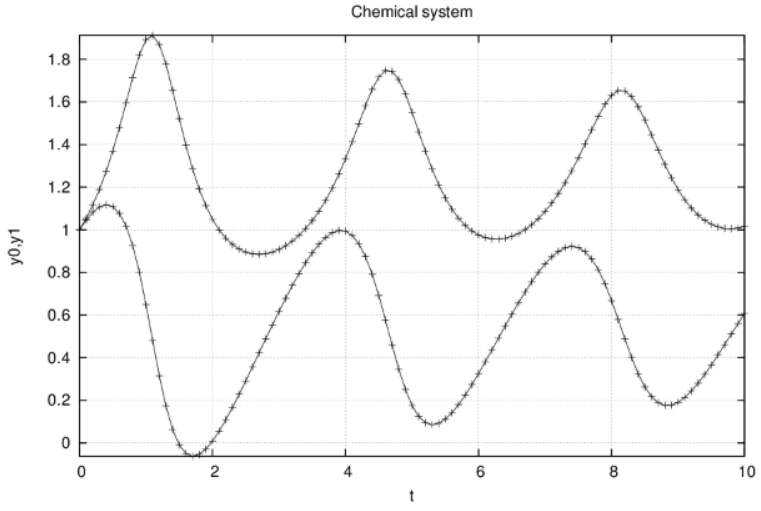


**Рис. 7.** Изменение концентраций при моделировании автоколебательной химической реакции (брюсселятора)

В рамках данной книги брюсселятор рассматривается как пример автоколебательной системы.

Описание модели брюсселятора в **Maxima** приведено в следующем командном файле:

```
load("dynamics");
load("draw");
B:0.5;
eq1:-(B+1)*y0+y0^2*y1+1;
eq2:B*y0-y0^2+1;
t_range:[t,0,10,0.1];
sol: rk([eq1,eq2],[y0,y1],[1,1],t_range)$
len:length(sol);
t:makelist(sol[k][1],k,1,len)$
```

```
y0:makelist(sol[k][2],k,1,len)$
y1:makelist(sol[k][3],k,1,len)$
draw2d(title="Brusselator",xlabel="t",ylabel="y0,y1",
grid=true,points_joined = true,
points(t,y0),points(t,y1),terminal=eps);
```

Графическая иллюстрация (автоколебательный режим в системе) — на рис. 7., а фазовые портреты — на рис. 8 и рис 9.

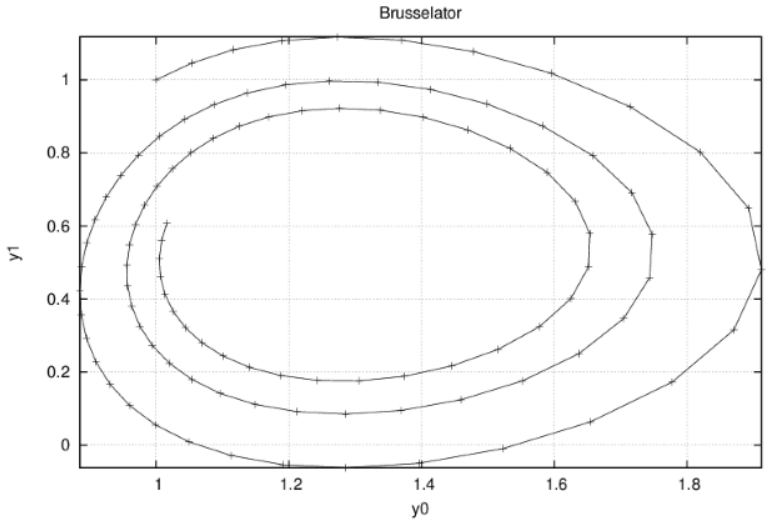


**Рис .8.** Фазовый портрет для брюсселятора (B=0.5)

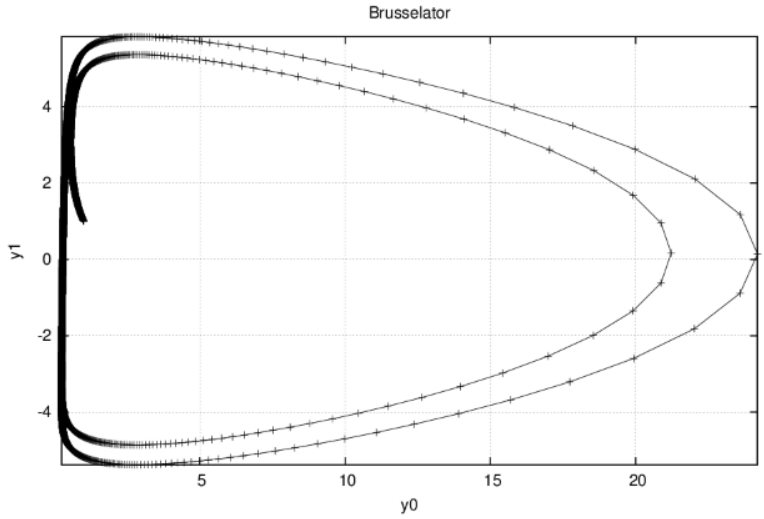


**Рис. 9.** Фазовый портрет для брюсселятора (B=2.5)

При проведении расчётов следует обратить внимание на жёсткость системы ОДУ, описывающей брюсселятор, в частности, при $B = 2.5$ для построения приведённой иллюстрации необходимо было уменьшить шаг по времени до 0.002. При очередном запуске командного файла, содержащего команды загрузки пакетов, рекомендуется перезапустить **Maxima.**

#### 6.3.2 Фазовые портреты динамических систем

Для изучения динамических систем центральным моментом является анализ фазовых портретов, т. е. решений, получающихся при выборе всевозможных начальных условий.

Решение ОДУ часто удобнее изображать не в виде графика $y0(t), y1(t), \cdots$ , а в фазовом пространстве, по каждой из осей которого откладываются значения каждой из найденных функций. При таком построении графика аргумент $t$ будет присутствовать на нем лишь параметрически.

Как правило, решение задач Коши для ОДУ и их систем — задача хорошо разработанная и с вычислительной точки зрения довольно простая. На практике чаще встречаются другие, более сложные задачи, в частности, исследование поведения динамической системы в зависимости от начальных условий. При этом в большинстве случаев бывает необходимым изучить только асимптотическое решение ОДУ, т.е. $y(t \to \infty)$, называемое аттрактором. Очень наглядным образом можно визуализировать такую информацию на фазовой плоскости, во многом благодаря тому, что существует всего несколько типов аттракторов, и для них можно построить четкую классификацию.

С одной стороны, каждое решение будет выходить из точки, координаты которой являются начальными условиями, но, оказывается, для большинства ОДУ целые семейства траекторий будут заканчиваться в одних и тех же аттракторах (стационарных точках или предельных циклах). Множество решений, вычисленное для всевозможных начальных условий, образует фазовый портрет динамической системы. С вычислительной точки зрения задача исследования фазового портрета часто сводится к обычному сканированию семейств решений ОДУ при разных начальных условиях.

Дальнейшее усложнение задач анализа фазовых портретов связано с их зависимостью от параметров, входящих в систему ОДУ. В частности, при плавном

изменении параметра модели может меняться расположение аттракторов на фазовой плоскости, а также могут возникать новые аттракторы и прекращать свое существование старые. В первом случае, при отсутствии особенностей, будет происходить простое перемещение аттракторов по фазовой плоскости (без изменения их типов и количества), а во втором — фазовый портрет динамической системы будет коренным образом перестраиваться. Критическое сочетание параметров, при которых фазовый портрет системы качественно меняется, называется в теории динамических систем точкой бифуркации.

Рассмотрим несколько наиболее известных классических примеров динамических систем, имея в виду. Это модели динамики популяций (Лотка-Вольтерры), генератора автоколебаний (Ван дер Поля), турбулентной конвекции (Лоренца) и химической реакции с диффузией (Пригожина). Для изучения динамических систем разработана специальная теория, центральным моментом которой является анализ фазовых портретов, т. е. решений, получающихся при выборе всевозможных начальных условий.

**6.3.3 Модель динамики популяций**

Модель взаимодействия "хищник—жертва" независимо предложили в 1925–1927 гг. Лотка и Вольтерра. Два дифференциальных уравнения моделируют временную динамику численности двух биологических популяций жертв $x$ и хищников $y$. Предполагается, что жертвы размножаются с постоянной скоростью, а их численность убывает вследствие поедания хищниками. Хищники же размножаются со скоростью, пропорциональной количеству пищи, и умирают естественным образом.

Модель была создана для биологических систем, но с определенными корректурами применима к конкуренции фирм, строительству финансовых пирамид, росту народонаселения, экологической проблематике и др.

Эта модель Вольтерра-Лотка с логистической поправкой описывается системой уравнений

$$\frac{d}{dt}y(t) = y(t)\,(a - b\,z(t)) - \alpha\,y(t)^2$$
$$\frac{d}{dt}z(t) = (-c + d\,y(t))\,z(t) - \alpha\,z(t)^2$$

с условиями заданной численности "жертв" и "хищников" в начальный момент $t = 0$.

Решая эту задачу при различных значениях, получаем различные фазовые портреты (обычный колебательный процесс и постепенная гибель популяций). Результаты приведены на рис 10, рис 11 и рис. .12.

```
(%i1)   a:4$ b:2.5$ c:2$ d:1$ alpha=0$
        eq1:'diff(y(t),t)=(a-b*z(t))*y(t)-alpha*y(t)^2;
        eq2:'diff(z(t),t)=(-c+d*y(t))*z(t)-alpha*y(t)^2;
        atvalue(y(t),t=0,3); atvalue(z(t),t=0,1);
```

$$(\%o6)\ \frac{d}{dt}\mathrm{y}(t) = \mathrm{y}(t)\,(4 - 2.5\,\mathrm{z}(t)) - \alpha\,\mathrm{y}(t)^2$$
$$(\%o7)\ \frac{d}{dt}\mathrm{z}(t) = (\mathrm{y}(t) - 2)\,\mathrm{z}(t) - \alpha\,\mathrm{y}(t)^2$$
$$(\%o8)\ 3$$
$$(\%o9)\ 1$$

```
(%i10)  desolve([eq1,eq2],[y(t),z(t)]);
rat: replaced -2.5 by -5/2 = -2.5
rat: replaced -2.5 by -5/2 = -2.5
rat: replaced -2.5 by -5/2 = -2.5
rat: replaced 2.5 by 5/2 = 2.5
```

$$(\%o10)$$
$$[\mathrm{y}(t) = \mathrm{ilt}\left(-\frac{5\,\mathrm{laplace}(\mathrm{y}(t)\,\mathrm{z}(t),t,g2176)+2\,\alpha\,\mathrm{laplace}\left(\mathrm{y}(t)^2,t,g2176\right)-6}{2\,g2176-8}, g2176, t\right),$$
$$\mathrm{z}(t) = \mathrm{ilt}\left(\frac{\mathrm{laplace}(\mathrm{y}(t)\,\mathrm{z}(t),t,g2176)-\alpha\,\mathrm{laplace}\left(\mathrm{y}(t)^2,t,g2176\right)+1}{g2176+2}, g2176, t\right)]$$

Очевидная проблема — неразрешимость данной системы в явном виде методом преобразования Лапласа, т.к. она нелинейна.

Используем численный метод Рунге-Кутта из пакета dynamics.

Результаты решения для значений $\alpha = 0$ и $\alpha = 0.02$ представлены на рис. 11 и рис. 12.

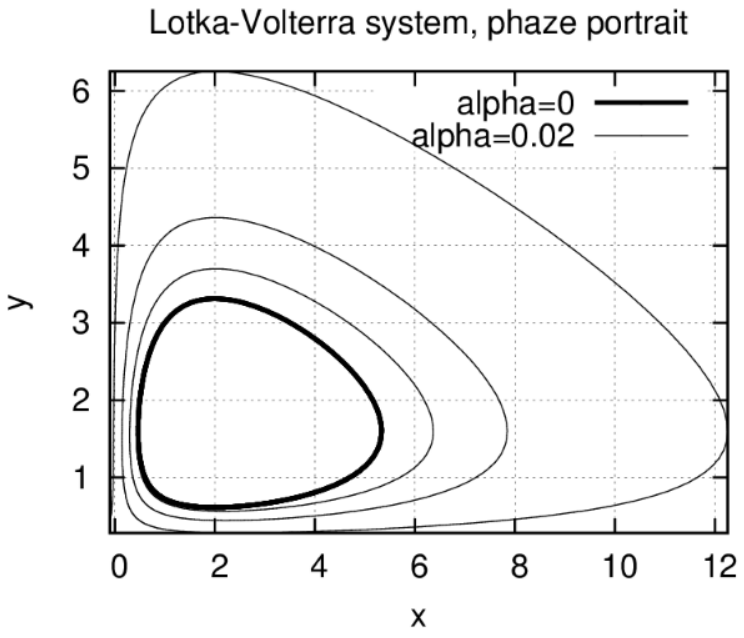

**Рис. 10.** Фазовый портрет для системы Лотка-Вольтерра

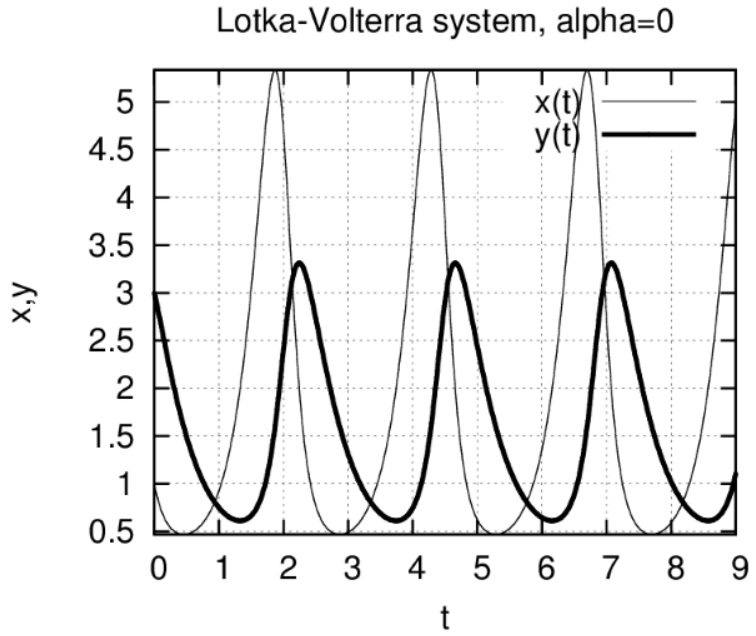


**Рис. 11.** Решения системы Лотка-Вольтерра в зависимости от времени (a = 0)

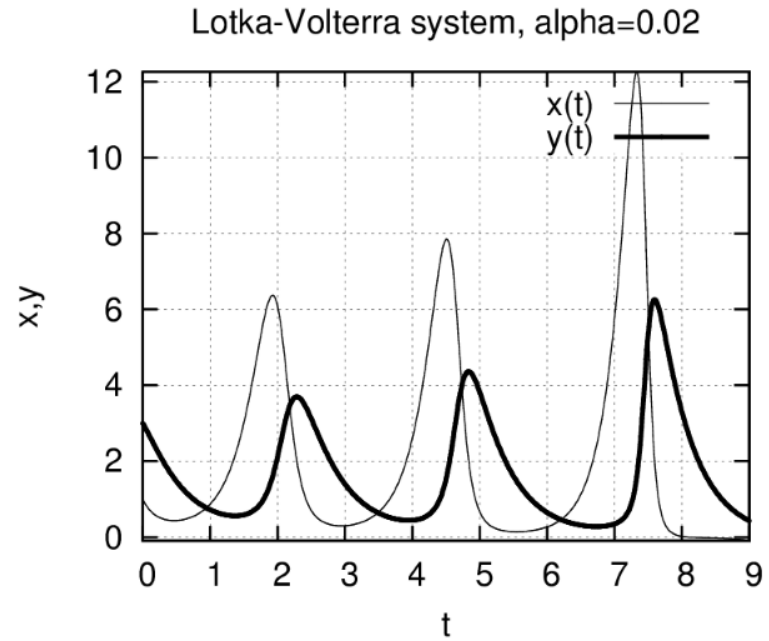


**Рис. 12.** Решения системы Лотка-Вольтерра в зависимости от времени (a = 0, 1)

Рассмотрим командный файл для задачи моделирования системы Лотка-Вольтерра в **Maxima**:

```
a:4; b:2.5; c:2; d:1; alpha1:0;
ode1:(a-b*x)*y-alpha1*x^2$ ode2:(-c+d*y)*x-alpha1*y^2$
alpha2:0.02;
ode3:(a-b*x)*y-alpha2*x^2$ ode4:(-c+d*y)*x-alpha2*y^2$
load("dynamics");
t1:[]$ xg1:[]$ yg1:[]$ t2:[]$ xg2:[]$ yg2:[]$
l1:rk([ode1,ode2],[y,x],[1,3],[t,0,9,0.01])$
l2:rk([ode3,ode4],[y,x],[1,3],[t,0,9,0.01])$
for i:1 thru length(l1) do(t1:append(t1,[l1[i][1]]),
        xg1:append(xg1,[l1[i][2]]),yg1:append(yg1,[l1[i][3]]));
for i:1 thru length(l2) do(t2:append(t2,[l2[i][1]]),
        xg2:append(xg2,[l2[i][2]]),yg2:append(yg2,[l2[i][3]]));
load("draw");
draw2d(terminal='eps, file_name="lotka1",grid=true,xlabel = "x",
        ylabel = "y", title="Lotka-Volterra system, phaze portrait",
        key= "alpha=0",points_joined = true, point_type = none,
        line_width = 4,color = black, points(xg1,yg1),
        points_joined = true, color = black,point_type = none,
        line_width = 1,key="alpha=0.02", points(xg2,yg2))$
draw2d(terminal='eps, file_name="lotka2",grid=true,xlabel = "t",
        ylabel = "x,y", title="Lotka-Volterra system, alpha=0",
        key= "x(t)",points_joined = true, line_width = 1,
        color = black,point_type = none, points(t1,xg1),
        points_joined = true, line_width = 4, point_type = none,
        color = black, key= "y(t)", points(t1,yg1))$
```

```
draw2d(terminal='eps, file_name="lotka3",grid=true,xlabel = "t",
        ylabel = "x,y", title="Lotka-Volterra system, alpha=0.02",
        key= "x(t)",points_joined = true, point_type = none,
        line_width = 1, color = black, points(t2,xg2),
        points_joined = true, line_width = 4, point_type = none,
        color = black, key= "y(t)", points(t2,yg2))$
```

Дифференциальные уравнения формируются символьными выражениями, определяющими правые части. Порядок следования выражений для расчёта правых частей ОДУ в первом списке функции $rk$ должен соответствовать друг другу. Следует обратить внимание, что результат выполнения функции $rk$ — двухуровневый список (каждый элемент списков $l1$ и $l2$ — также список, содержащий значение независимой переменной, и соответствующие значения искомых функций), поэтому оп преобразуется в векторы, используемые для построения графических иллюстраций.

**.3.4 Движение твердого тела**

Рассмотрим пример построения трехмерного фазового портрета. Находим решение задачи Эйлера свободного движения твердого тела:

$$\frac{dx_1}{dt} = x_2 x_3,$$
$$\frac{dx_2}{dt} = -x_1 x_3,$$
$$\frac{dx_3}{dt} = -0.51 x_1 x_2.$$

```
(%i1)   eq1:'diff(x1(t),t)=x2(t)*x3(t);
        eq2:'diff(x2(t),t)=-x1(t)*x3(t);
        eq3:'diff(x3(t),t)=-0.51*x1(t)*x2(t);
```

$$(\%o1)\ \frac{d}{dt}\,x1\,(t) = x2\,(t)\ x3\,(t)$$
$$(\%o2)\ \frac{d}{dt}\,x2\,(t) = -x1\,(t)\ x3\,(t)$$
$$(\%o3)\ \frac{d}{dt}\,x3\,(t) = -0.51\,x1\,(t)\ x2\,(t)$$

```
(%i4)   load("dynamics")$ l: rk([y*z, -x*z,0.51*x*y],
        [x,y,z],[1,2,3],[t,0,4,0.1])$
```

Фазовый портрет для данной динамической системы (трехмерная кривая) представлен на рис. 13.

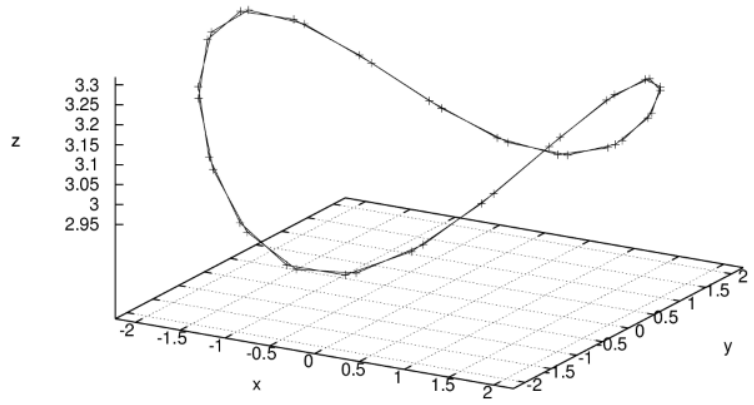


**Рис. 13.** Фазовый портрет трехмерной динамической системы

**3.5 Аттрактор Лоренца**

Одна из самых знаменитых динамических систем предложена в 1963 г. Лоренцем в качестве упрощенной модели конвективных турбулентных движений жидкости в нагреваемом сосуде тороидальной формы. Система состоит из трех ОДУ и имеет три параметра модели. Задаём правые части уравнений модели Лоренца:

```
(%i2)   eq: [s*(y-x), x*(r-z) -y, x*y - b*z];
```

$$(\%o2)\quad [s\ (y-x), x\ (r-z) - y, x\,y - b\,z]$$

Задаём временные параметры решения

```
(%i3)   t_range: [t,0,50,0.05];
```

$$(\%o3)\quad [t, 0, 50, 0.05]$$

Задаём параметры системы

```
(%i4)   s: 10.0; r: 28.0; b: 2.6667;
```

$$(\%o6)\quad 10.028.02.6667$$

Задаём начальные значения $x, y, z$

```
(%i7)   init: [1.0,0,0];
```

$$(\%o7)\quad [1.0, 0, 0]$$

Выполняем решение

```
(%i8)   sol: rk(eq, [x,y,z],init,t_range)$
(%i9)   len:length(sol);
```

(%o9) 1001

Выделяем компоненты решения и строим графические иллюстрации:

```
(%i10)  t:makelist(sol[k][1],k,1,len)$
        x:makelist(sol[k][2],k,1,len)$
        y:makelist(sol[k][3],k,1,len)$
        z:makelist(sol[k][4],k,1,len)$
        plot2d([discrete,t,x])$ plot2d([discrete,t,y])$
```

Результаты решения (хаотические колебания $x, y, z$) представлен на рис.14 и рис.15 (фазовый портрет системы). На рисунках объединены в одних осях кривые $x(t), y(t), z(t)$.

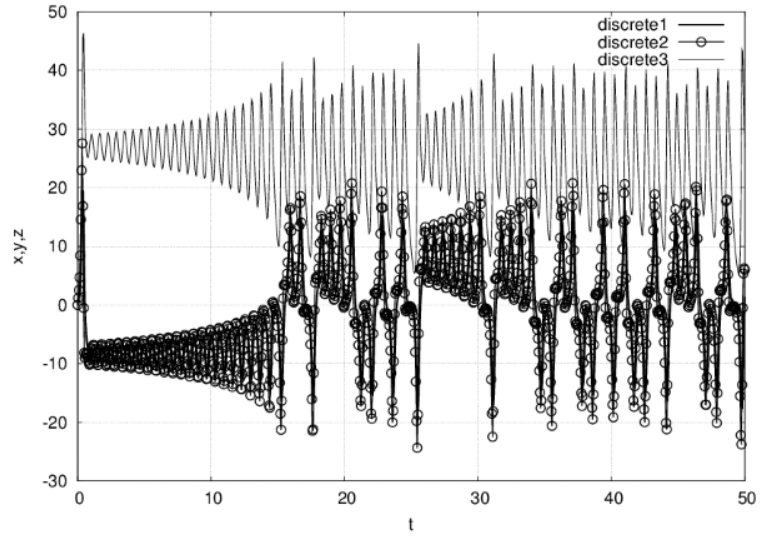


**Рис.14.** Пример формирования динамического хаоса (аттрактор Лоренца)

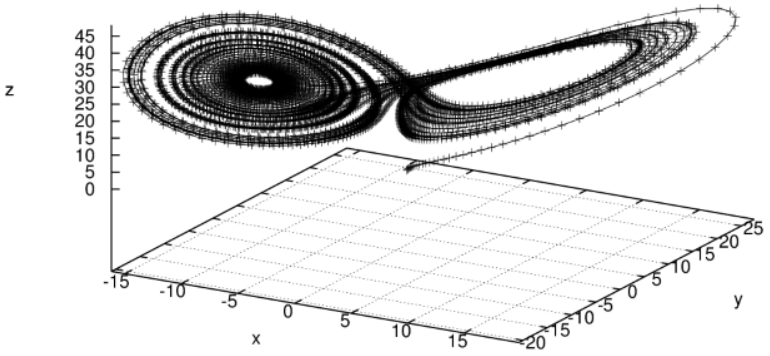


**Рис. 15.** Трехмерный фазовый портрет (аттрактор Лоренца)

Решением системы Лоренца при определенном сочетании параметров является странный аттрактор (или аттрактор Лоренца) — притягивающее множество траекторий на фазовом пространстве, которое по виду идентично случайному процессу. В некотором смысле аттрактор Лоренца является стохастическими автоколебаниями, которые поддерживаются в динамической системе за счет внешнего источника.

Решение в виде странного аттрактора появляется только при некоторых сочетаниях параметров. Перестройка типа фазового портрета происходит в области промежуточных значениях параметра r. Критическое сочетание параметров, при которых фазовый портрет системы качественно меняется, называется в теории динамических систем точкой бифуркации. Физический смысл бифуркации в модели Лоренца, согласно современным представлениям, описывает переход ламинарного движения жидкости к турбулентному.

**6.3.6 Модель автоколебательной системы: уравнение Ван дер Поля**

Рассмотрим решение уравнения Ван дер Поля, описывающего электрические колебания в замкнутом контуре, состоящем из соединенных последовательно конденсатора, индуктивности, нелинейного сопротивления и элементов, обеспечивающих подкачку энергии извне. Неизвестная функция времени $y(t)$ имеет смысл электрического тока, а в параметре $\mu$ заложены количественные соотношения между составляющими электрической цепи, в том числе и нелинейной компонентой сопротивления:

$$\frac{d^2y(t)}{dx^2} - \mu(1 - y(t)^2)\frac{dy(t)}{dt} + y(t) = 0$$

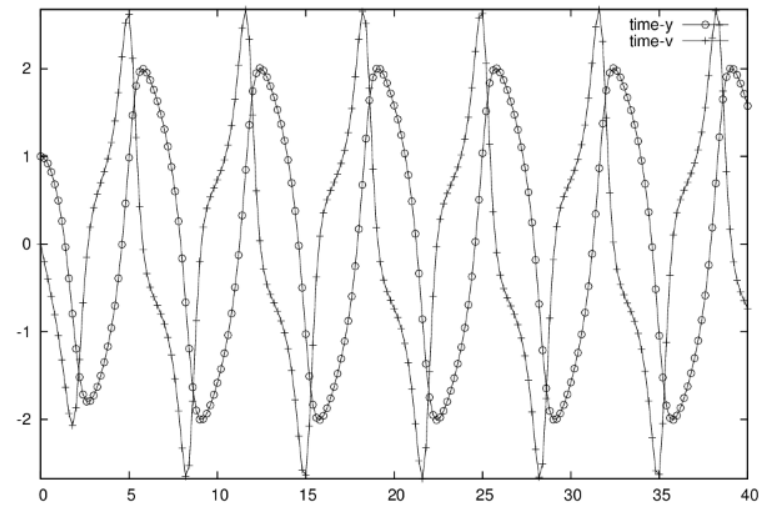


**Рис 16.** Решение уравнения Ван дер Поля

Решением уравнения Ван дер Поля являются колебания, вид которых для $\mu = 1$ показан на рис. 16. Они называются автоколебаниями и принципиально отличаются от рассмотренных ранее (например, численности популяций в модели Вольтерра) тем, что их характеристики (амплитуда, частота, спектр) не зависят от начальных условий, а определяются исключительно свойствами самой динамической системы. Через некоторое время расчетов после выхода из начальной точки решение выходит на один и тот же цикл колебаний, называемый предельным циклом. Аттрактор типа предельного цикла является замкнутой кривой на фазовой плоскости. К нему асимптотически притягиваются все окрестные траектории, выходящие из различных начальных точек, как изнутри (рис.6.17), так и снаружи предельного цикла.

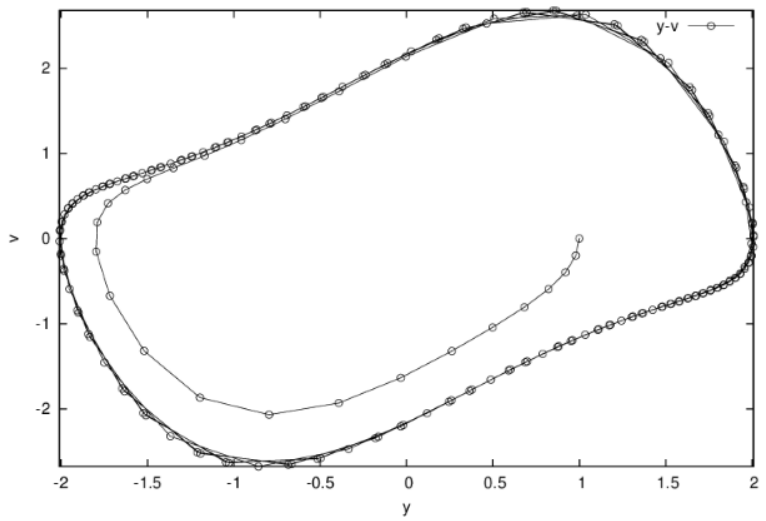


**Рис. 17.** Фазовый портрет уравнения Ван дер Поля

Использованный командный файл **Maxima** (для построения графической иллюстрации использован пакет draw):

```
load("dynamics")$ load("draw")$
mu:1$ s:rk ([v,mu*(1-y^2)*v-y],[y,v],[1,0],[t,0,40,0.2])$
time:makelist(s[k][1],k,1,length(s))$
y:makelist(s[k][2],k,1,length(s))$
v:makelist(s[k][3],k,1,length(s))$
draw2d(points_joined = true, point_type=6, key= "y-v",
        xlabel="y",ylabel="v",points(y,v), terminal=eps)$
```