Знакомство с системой *Mathematica*

Структура систем Mathematica

Общая структура систем Mathematica (всех версий) представлена на рисунке 1.1.

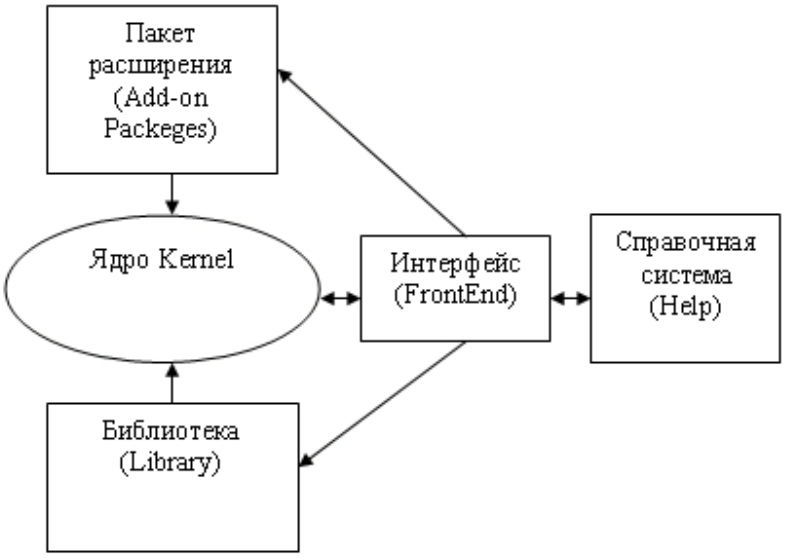

Рисунок 1.1 - *Структура системы Mathematica*

Центральное место в системах класса Mathematica занимает машинно-независимое ядро математических операций — Kernel. Для ориентации системы на конкретную машинную платформу служит программный интерфейсный процессор Front End. Именно он определяет, какой вид имеет пользовательский интерфейс системы.

Ядро сделано достаточно компактным с тем, чтобы любая функция из него вызывалась достаточно быстро. Для расширения набора функций служит библиотека (Library) и набор пакетов расширения (Add-on Packages). Пакеты расширений готовятся на собственном языке программирования систем *Mathematica* и являются главным средством расширения возможностей системы и их адаптации к решению конкретных классов задач пользователя. Кроме того, системы имеют встроенную электронную справочную систему — Help. Она содержит шесть электронных книг с «живыми» примерами.

Идеология систем Mathematica

Идеология систем Mathematica базируется на двух, казалось бы, взаимно исключающих друг друга положениях:

- решение большинства математических задач в системе может производиться в диалоговом режиме без традиционного программирования;
- входной язык общения системы является одним из самых мощных языков функционального программирования, ориентированных на решение различных задач (в том числе математических).

Противоречивость этих положений кажущаяся. На самом деле Mathematica — типичная система программирования с проблемно-ориентированным языком программирования *сверхвысокого уровня.* Его можно отнести к классу *интерпретаторов.* Как известно, языки такого типа последовательно анализируют (интерпретируют) каждое выражение и тут же исполняют его. Таким образом, работа с системой происходит явно, в *диалоговом режиме* — пользователь задает системе задание, а она тут же выполняет его. При этом язык системы Mathematica содержит достаточный набор управляющих структур для создания условных выражений, ветвления в программах, циклов и т. д.

К идеологии систем Mathematica надо отнести и комплексную визуализацию всех этапов вычислений, начиная с легко понятного и естественного ввода текстов и формул и кончая наглядным выводом результатов в разнообразных формах представления. Особое место при этом играет полная визуализация результатов вычислений, включающая в себя построение огромного числа графиков самого различного вида, в том числе средства анимации изображений и синтеза звуков.

Символьные вычисления

Особенности систем компьютерной математики

Недостатки численных расчетов

Большинство первых СКМ (Eureka, Mercury, Excel, Lotus-123, Mathcad для MS-DOS, PC MATLAB и др.) предназначались для численных расчетов. Они как бы превращали компьютер в большой программируемый калькулятор, способный быстро и автоматически (по введенной программе) выполнять арифметические и логические операции над числами или массивами чисел. Их результат всегда конкретен — это или число, или набор чисел, представляющих таблицы, матрицы или точки графиков. Разумеется, компьютер позволяет выполнять такие вычисления

с немыслимой ранее скоростью, педантичностью и даже точностью, выводя результаты в виде хорошо оформленных таблиц или графиков.

Однако результаты вычислений редко бывают абсолютно точными в математическом смысле: как правило, при операциях с вещественными числами происходит их *округление,* обусловленное принципиальным ограничением разрядной сетки компьютера при хранении чисел в памяти. Реализация большинства численных методов (например, решения нелинейных или дифференциальных уравнений) также базируется на заведомо приближенных алгоритмах. Часто из-за накопления погрешностей эти методы теряют вычислительную устойчивость и расходятся, давая неверные решения или даже ведя к полному краху работы вычислительной системы — вплоть до злополучного «зависания».

Условия, при которых это наступает, не всегда известны — их оценка довольно сложна в теоретическом отношении и трудоемка на практике. Поэтому рядовой пользователь, сталкиваясь с такой ситуацией, зачастую становится в тупик или, что намного хуже, неверно истолковывает явно ошибочные результаты вычислений, «любезно» предоставленные ему компьютером. Трудно подсчитать, сколько «открытий» на компьютере было отвергнуто из-за того, что наблюдаемые колебания, выбросы на графиках или асимптоты ошибочно вычисленных функций неверно истолковывались как новые физические закономерности моделируемых устройств и систем, тогда как на деле были лишь грубыми погрешностями численных методов решения вычислительных задач.

Многие ученые справедливо критиковали численные математические системы и программы реализации численных методов за *частный* характер получаемых с их помощью результатов. Они не давали возможности получить *общие* формулы, описывающие решение задач. Как правило, из результатов численных вычислений невозможно было сделать какие-либо общие теоретические, а подчас и практические выводы. Поэтому, прежде чем использовать такие системы в реализации серьезных научных проектов, приходилось прибегать к дорогой и недостаточно оперативной помощи математиков-аналитиков. Именно они решали нужные задачи в аналитическом виде и предлагали более или менее приемлемые методы их численного решения на компьютерах.

Понятие о символьных (аналитических) вычислениях

Символьные операции — это как раз то, что кардинально отличает систему Mathematica (и подобные ей символьные математические системы) от систем для выполнения численных расчетов. При символьных операциях, называемых также

аналитическими, задания на вычисление составляются в виде символьных (формульных) выражений, и результаты вычислений также получаются в символьном виде. Численные результаты при этом являются частными случаями символьных.

Выражения, представленные в символьном виде, отличаются высокой степенью общности. К примеру, тождество $\sin(x)^2 + \cos(x)^2 - 1$ справедливо при любых значениях аргумента *x*. Если результат символьной операции равен, к примеру, sin(l), то он и будет выведен как sin(l) — конкретное вещественное число, приближенно представляющее или аппроксимирующее sin(l), вычисляться не будет, ибо носит частный характер.

Результат вычисления $\sin(x)^2 + \cos(x)^2$ можно проверить с помощью систем для численных расчетов, задав ряд конкретных значений *x* и вычислив сумму квадратов синуса и косинуса. Однако всякий раз мы будем получать *частный* результат, не имея никакой гарантии того, что он действительно справедлив при любом значении *x*. К тому же этот результат нередко может оказаться равным 0,9999999 или 1,0000001, так что лишь наша фантазия округляет его до точной единицы.

Попытка вычислить в общем виде выражение $\sin(x)^2 + \cos(x)^2$ с помощью численных математических систем или программ на обычных языках программирования к успеху не приведет. Вместо ожидаемого результата появится сообщение об ошибке вида: «Переменная *x* не определена!». Компьютер будет ждать ввода конкретного значения для *x*. Так будет независимо от того, запрограммировали вы вычисления на простеньком Бейсике или на языке профессионалов-программистов C++. И лишь системы символьной математики при вычислениях дадут долгожданное и абсолютно точное значение 1 (рисунок 1.2).

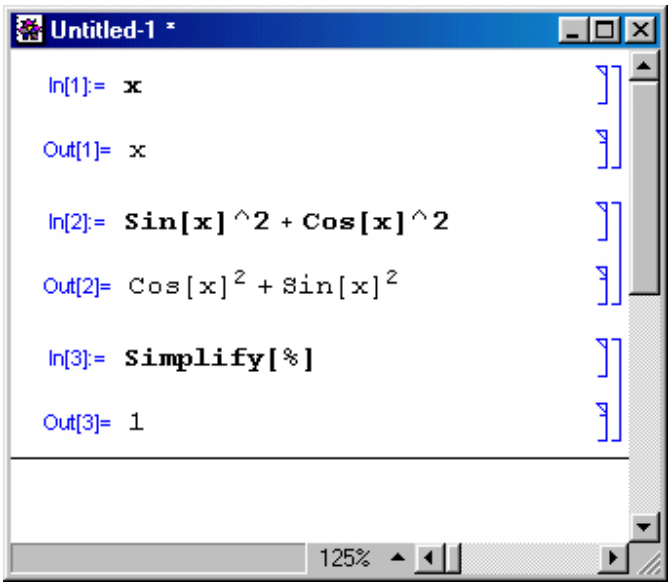

Рисунок 1.2 - *Система Mathematical вычисляет значение* $sin(x)^2+cos(x)^2$

Пока не стоит обращать внимание на то, как получен рисунок 1.2 — это окно реально работающей системы Mathematica. Уже при рассмотрении простейшего примера, представленного на этом рисунке, можно сделать несколько характерных выводов. Прежде всего видно, что при выводе неопределенной переменной *x* мы получаем просто имя этой переменной. Функции sin(x) и cos(x) в системе Mathematica обозначаются как Sin [x] и Cos [x]. Само по себе выражение $\sin(x)^2 + \cos(x)^2$ просто повторяется, а для его вычисления используется функция Simplify (упростить), аргументом которой является знак %, означающий подстановку предшествующего выражения. Два знака % можно использовать для подстановки предшествующего предшествующему выражению и т. д. Для *вычисления* строки ввода надо нажимать клавиши Shift+Enter, нажатие же одной клавиши Enter просто переводит строку в области ввода, именуемой также *ячейкой ввода.*

Любопытно, что в начале запуска Mathematica выводит чистое окно редактирования документа, в котором нет даже маркера ввода — характерной вертикальной черточки. Этот маркер появится, как только вы введете какой-то первый символ. После получения первого результата появляется и длинная горизонтальная черта, отделяющая выведенные ячейки от свободного поля окна редактирования под ними. Эта черта является признаком возможности ввода очередной ячейки. Ее можно перевести в уже созданную область документа, если вы захотите создать новую ячейку среди уже существующих ячеек ввода.

Обратите внимание на то, что система выделяет ячейки ввода определителем In [N], а ячейки вывода — определителем Out [N], где N — автоматически проставляемый номер строки. Кроме того, в левой части отображаются квадратные скобки с особыми признаками, которые будут описаны в позже. Далее мы, как правило, будем опускать определители ячеек и квадратные скобки и представлять документы в упрощенной и более компактной форме. Например, представленный на рисуноке 1.2 документ может быть записан в следующем виде:

```
x

x

Sin[x]^2+Cos[x]^2

Cos[x]^2+Sin[x]^2

Simplify[%]

1
```

Здесь входные выражения задаются жирным прямым шрифтом, а выходные — прямым шрифтом обычной насыщенности, то есть именно так, как они выглядят при настройке системы по умолчанию. При этом выходные выражения имеют обычный (в терминах системы Mathematica — стандартный) вид, присущий математическим формулам.

Ячейки нумеруются по мере их использования. При этом можно с конца документа вернуться к его началу или середине и, изменив содержимое ранее использованных ячеек, снова выполнить вычисления. При этом ячейки меняют номера. При загрузке файла ячейки перенумеруются в строго последовательном порядке. Таким образом, номера ячеек не являются жестко фиксированными, они представляют собой сугубо техническое средство, отражающее работу системы в текущем сеансе — *сессии.* Это говорит в пользу отказа от вывода определителей ячеек при записи большинства примеров.