## §2. ПОНЯТИЕ О КОМПЬЮТЕРНОМ МАТЕМАТИЧЕСКОМ МОДЕЛИРОВАНИИ

### 2.1. МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ И КОМПЬЮТЕРЫ

Математическая модель выражает существенные черты-объекта или процесса языком уравнений и других математических средств. Собственно говоря, сама математика обязана своим существованием тому, что она пытается отразить, т.е. промоделировать, на своем специфическом языке закономерности окружающего мира.

Путь математического моделирования в наше время гораздо более всеобъемлющ, нежели моделирования натурного. Огромный толчок развитию математического моделирования дало появление ЭВМ, хотя сам метод зародился одновременно с математикой тысячи лет назад.

Математическое моделирование как таковое отнюдь не всегда требует компьютерной поддержки. Каждый специалист, профессионально занимающийся математическим моделированием, делает все возможное для аналитического исследования модели. Аналитические решения (т.е. представленные формулами, выражающими результаты исследования через исходные данные) обычно удобнее и информативнее численных. Возможности аналитических методов решения сложных математических задач, однако, очень ограниченны и, как правило, эти методы гораздо сложнее численных. В данной главе доминируют численные методы, реализуемые на компьютерах. Это связано с тем, что моделирование здесь рассматривается под углом зрения компьютерных (информационных) технологий. Такой подход несколько сужает возможности метода в целом; его достоинство - некоторое снижение барьера необходимой математической подготовки (хотя, разумеется, и в численные методы при профессиональном занятии математическим моделированием приходится углубляться настолько, что при этом требуется значительное математическое образование). Наконец, отметим, что понятия «аналитическое решение» и «компьютерное решение» отнюдь не противостоят друг другу, так как

а) все чаще компьютеры при математическом моделировании используются не только для численных расчетов, но и для аналитических преобразований;

б) результат аналитического исследования математической модели часто выражен столь сложной формулой, что при взгляде на нее не складывается восприятия описываемого ей процесса. Эту формулу (хорошо еще, если просто формулу!) нужно протабулировать, представить графически, проиллюстрировать в динамике, иногда даже озвучить, т.е. проделать то, что называется «визуализацией абстракций» . При этом компьютер - незаменимое техническое средство.

### 2.2. ЭТАПЫ И ЦЕЛИ КОМПЬЮТЕРНОГО МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ

Здесь мы рассмотрим процесс компьютерного математического моделирования, включающий численный эксперимент с моделью (рис. 7.1).

Первый этап - определение целей моделирования. Основные из них таковы:

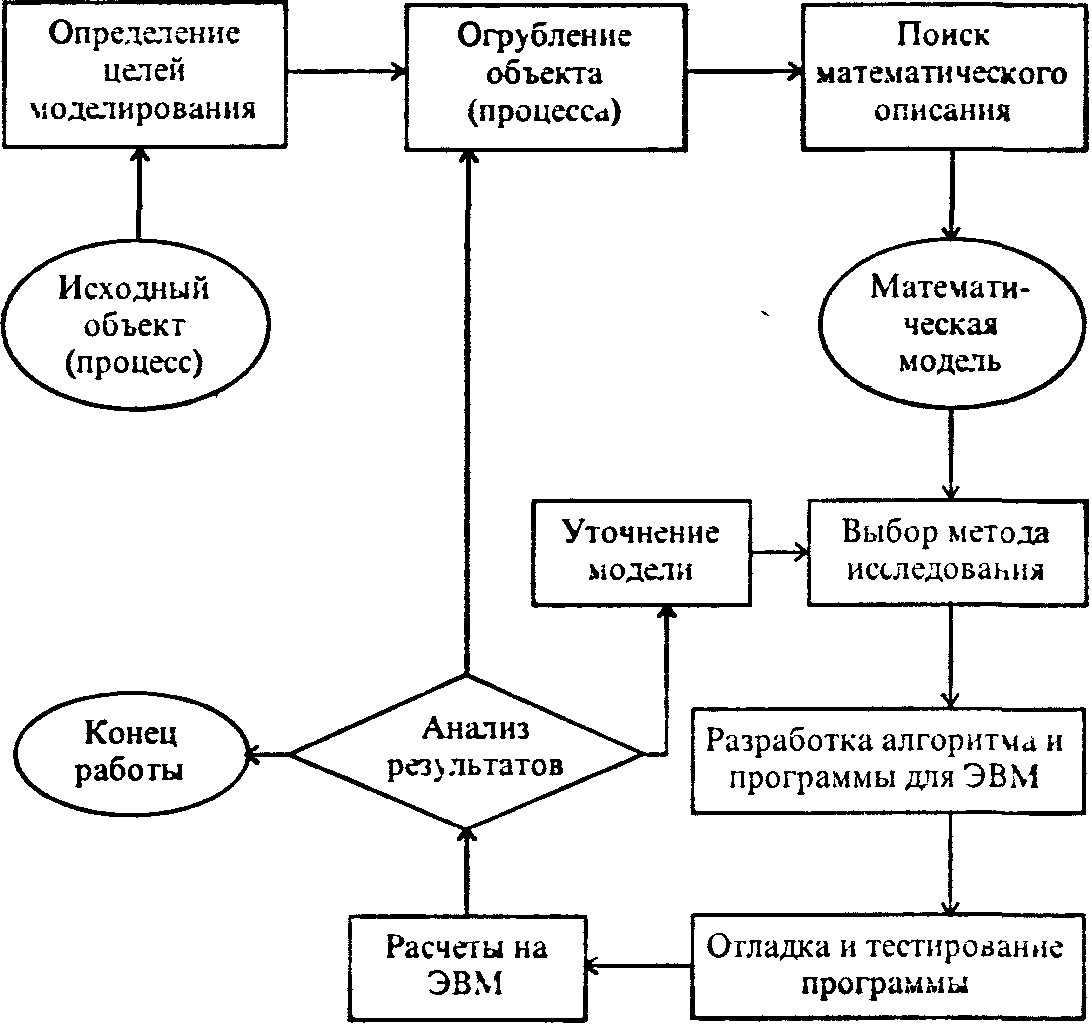
1) модель нужна для того, чтобы понять как устроен конкретный объект, какова его структура, основные свойства, законы развития и взаимодействия с окружающим миром (**понимание**);

2) модель нужна для того, чтобы научиться управлять объектом (или процессом) и определить наилучшие способы управления при заданных целях и критериях (**управление**);

3) модель нужна для того, чтобы прогнозировать прямые и косвенные последствия реализации заданных способов и форм воздействия на объект (**прогнозирование**).

Поясним это на примерах. Пусть объект исследования - взаимодействие потока жидкости или газа с телом, являющимся для этого потока препятствием. Опыт показывает, что сила сопротивления потоку со стороны тела растет с ростом скорости потока, но при некоторой достаточно высокой скорости эта сила скачком уменьшается с тем, чтобы с дальнейшим увеличением скорости снова возрасти. Что же произошло, обусловив уменьшение силы сопротивления? Математическое моделирование позволяет получить четкий ответ: в момент скачкообразного уменьшения сопротивления вихри, образующиеся в потоке жидкости или газа позади обтекаемого тела, начинают отрываться от него и уноситься потоком.

Пример совсем из другой области: мирно сосуществовавшие со стабильными численностями популяции двух видов особей, имеющих общую кормовую базу, «вдруг» начинают резко менять численность - и здесь математическое моделирование позволяет (с известной долен достоверности) установить причину (или, по крайней мере, опровергнуть определенную гипотезу).



*Рис. 7.1.* Общая схема процесса компьютерного математического моделирования

Выработка концепции управления объектом -другая возможная цель моделирования. Какой режим полета самолета выбрать для того, чтобы полет был вполне безопасным и экономически наиболее выгодным? Как составить график выполнения сотен видов работ на строительстве большого объекта, чтобы оно закончилось в максимально короткий срок? Множество таких проблем систематически возникает перед экономистами, конструкторами, учеными.

Наконец, прогнозирование последствий тех или иных воздействий на объект может быть как относительно простым делом в несложных физических системах, так и чрезвычайно сложным - на грани выполнимости - в системах биолого-экономических, социальных. Если относительно легко ответить на вопрос об изменении режима распространения тепла в тонком стержне при изменениях в составляющем его сплаве, то несравненно труднее проследить (предсказать) экологические и климатические последствия строительства крупной ГЭС или социальные последствия изменений налогового законодательства. Возможно, и здесь методы математического моделирования будут оказывать в будущем более значительною помощь.

Составим список величин, от которых зависит поведение объекта или ход процесса, а также тех величин, которые желательно получить в результате моделирования. Обозначим первые (входные) величины через *х1, x2, ....* *хn*; вторые (выходные) через *y1,y2, … ,yk.* Символически поведение объекта или процесса можно представить в виде

*уj = Fj (x1, х2,....xn) (j*=1,2,..., *k*), (7.1)

где *Fj -* те действия, которые следует произвести над входными параметрами, чтобы получить результаты. Хотя запись *F (x1,* *x2*, ..., *хn)* напоминает о функции, мы здесь используем ее в более широком смысле. Лишь в простейших ситуациях *F(x)* есть функция в том смысле, который вкладывается в это понятие в учебниках математики; чтобы это подчеркнуть, лучше использовать по отношению к *F(x)* термин «оператор».

Входные параметры *xi* могут быть известны «точно», т.е. поддаваться (по крайней мере, в принципе) измерению однозначно и с любой степенью точности - тогда они являются детерминированными величинами. Так, в классической механике, сколь сложной ни была бы моделируемая система, входные параметры детерминированы - соответственно, детерминирован, однозначно развивается во времени процесс эволюции такой системы. Однако, в природе и обществе гораздо чаще встречаются процессы иного рода, когда значения входных параметров известны лишь с определенной степенью вероятности, т.е. эти параметры являются вероятностными (стохастическими), и, соответственно, таким же является процесс эволюции системы (случайный процесс).

«Случайный» - не значит «непредсказуемый»; просто характер исследования, задаваемых вопросов резко меняется (они приобретают вид «С какой вероятностью...», «С каким математическим ожиданием...» и т.п.). Примеров случайных процессов не счесть как в науке, так и в обыденной жизни (силы, действующие на летящий самолет в ветренную погоду, переход улицы при большом потоке транспорта и т.д.).

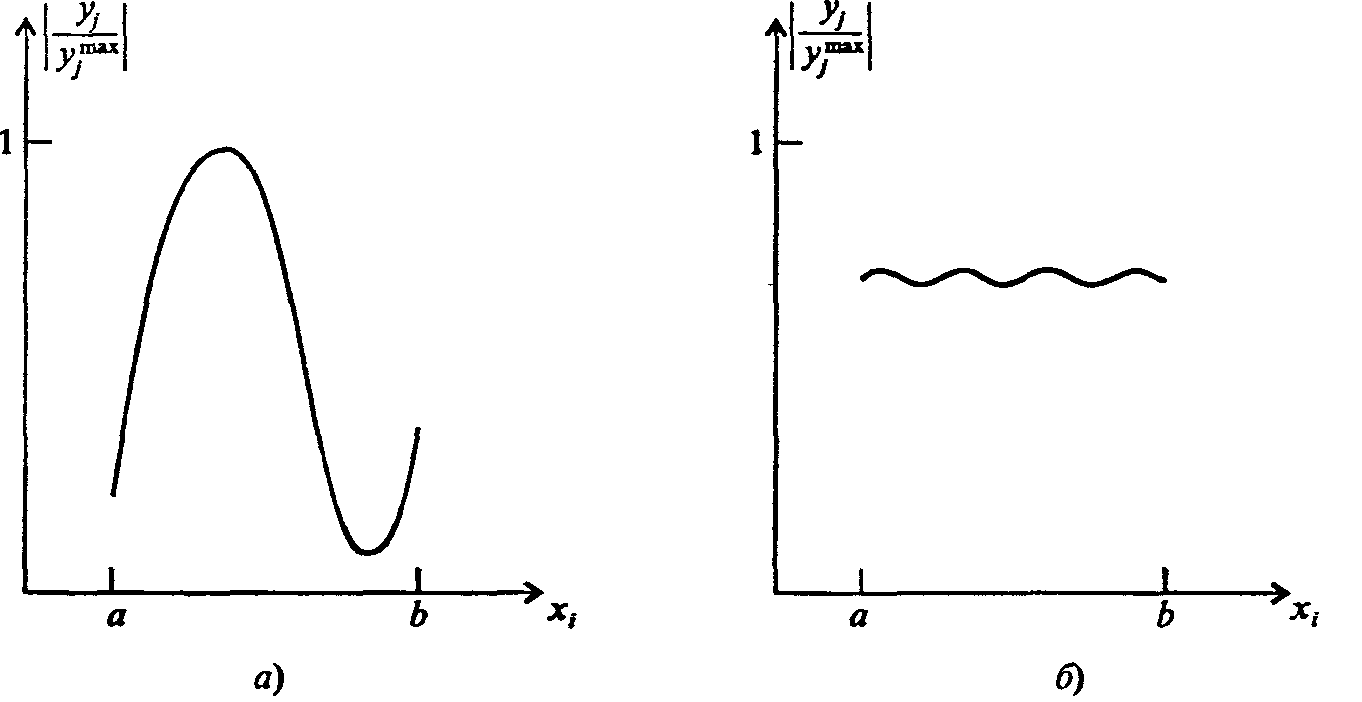
Для стохастической модели выходные параметры могут быть как величинами вероятностными, так и однозначно определяемыми. Пример последнего: на перекрестке улиц можно ожидать зеленого сигнала светофора и полминуты, и две минуты (с разной вероятностью), но среднее время ожидания есть величина вполне определенная, и именно она может быть объектом моделирования.

Важнейшим этапом моделирования является разделение входных параметров по степени важности влияния их изменений на выходные. Такой процесс называется ранжированием (разделением по рангам). Чаще всего невозможно (да и не нужно) учитывать все факторы, которые могут повлиять на значения интересующих нас величин *yj.* От того, насколько умело выделены важнейшие факторы, зависит успех моделирования, быстрота и эффективное гь достижения цели. Выделить более важные (или, как говорят, значимые) факторы и отсеять менее важные может лишь специалист в той предметной области, к которой относится модель. Так, опытный учитель знает, что на успех контрольной работы влияет степень знания предмета и психологический настрой класса; однако, влияют и другие факторы - например, каким уроком по счету идет контрольная, какова в этот момент погода и т.д. -фактически проведено ранжирование.

Отбрасывание (по крайней мере при первом подходе) менее значимых факторов огрубляет объект моделирования и способствует пониманию его главных свойств и закономерностей. Умело ранжированная модель должна быть адекватна исходному объекту или процессу в отношении целей моделирования. Обычно определить адекватна ли модель можно только в процессе экспериментов с ней, анализа результатов.

На рис. 7.2 проиллюстрированы две крайние ситуации: а) некоторый параметр *х,* очень сильно влияет на результирующую величину *yj*, б) почти не влияет на нее. Ясно, что если все представляющие интерес величины *уj* реагируют на *хi* так, как изображено на рис. 7.2, *б,* то *хi* является параметром, который при первом подходе может быть из модели исключен; если же хотя бы одна из величин *уj* реагирует на изменение *xi* так, как изображено на рис. 7.2, *а,* то *хi* нельзя исключать из числа важнейших параметров.

Следующий этап - поиск математического описания. На этом этапе необходимо перейти от абстрактной формулировки модели к формулировке, имеющей конкретное математическое наполнение. В этот момент модель предстает перед нами в виде уравнения, системы уравнений, системы неравенств, дифференциального уравнения или системы таких уравнений и т.д.



*Рис. 7.2.* Варианты степени влияния величины *х,* на результирующую величину *yi*

Когда математическая модель сформулирована, выбираем метод ее исследования. Как правило, для решения одной и той же задачи есть несколько конкретных методов, различающихся эффективностью, устойчивостью и т.д. От верного выбора метода часто зависит успех всего процесса.

Разработка алгоритма и составление программы для ЭВМ - это творческий и трудно формализуемый процесс. В настоящее время при компьютерном математическом моделировании наиболее распространенными являются приемы процедурно-ориентированного (структурного) программирования, описанные в главе 3. Из языков программирования многие профессионалы-физики, например, до сих пор предпочитают FORTRAN как в силу традиций, так и в силу непревзойденной эффективности компиляторов (для расчетных работ) и наличия написанных на нем огромных, тщательно отлаженных и оптимизированных библиотек стандартных программ математической ориентации. В ходу и такие языки, как PASCAL, BASIC, С - в зависимости от характера задачи и склонностей программиста.

После составления программы решаем с ее помощью простейшую тестовую задачу (желательно, с заранее известным ответом) с целью устранения грубых ошибок. Это -лишь начало процедуры тестирования, которую трудно описать формально исчерпывающим образом. По существу, тестирование может продолжаться долго и закончиться тогда, когда пользователь по своим профессиональным признакам сочтет программу верной. Программистский фольклор полон историй об ошибках на этом пути.

Затем следует собственно численный эксперимент, и выясняется, соответствует ли модель реальному объекту (процессу). Модель адекватна реальному процессу, если некоторые характеристики процесса, полученные на ЭВМ, совпадают с экспериментальными с заданной степенью точности. В случае несоответствия модели реальному процессу возвращаемся к одному из предыдущих этапов.

### 2.3. КЛАССИФИКАЦИЯ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ

К классификации математических моделей разные авторы подходят по-своему, положив в основу классификации различные принципы. Можно классифицировать модели по отраслям наук (математические модели в физике, биологии, социологии и т.д.) - это естественно, если к этому подходит специалист в какой-то одной науке. Можно классифицировать по применяемому математическому аппарату (модели, основанные на применении обыкновенных дифференциальных уравнений, дифференциальных уравнений в частных производных, стохастических методов, дискретных алгебраических преобразований и т.д.) - это естественно для математика, занимающегося аппаратом математического моделирования. Наконец, человек, интересующийся общими закономерностями моделирования в разных науках безотносительно к математическому аппарату, ставящий на первое место цели моделирования, скорее всего заинтересуется такой классификацией:

• дескриптивные (описательные) модели;

• оптимизационные модели;

• многокритериальные модели;

• игровые модели;

• имитационные модели.

Остановимся на этом чуть подробнее и поясним на примерах. Моделируя движение кометы, вторгшейся в Солнечную систему, мы описываем (предсказываем) траекторию ее полета, расстояние, на котором она пройдет от Земли и т. д. , т. е. ставим чисто описательные цели. У нас нет никаких возможностей повлиять на движение кометы, что-то изменить.

На другом уровне процессов мы можем воздействовать на них, пытаясь добиться какой-то цели. В этом случае в модель входит один или несколько параметров, доступных нашему влиянию. Например, меняя тепловой режим в зернохранилище, мы можем стремиться подобрать такой, чтобы достичь максимальной сохранности зерна, т. е. оптимизируем процесс.

Часто приходится оптимизировать процесс по нескольким параметрам сразу, причем цели могут быть весьма противоречивыми. Например, зная цены на продукты и потребность человека в пище, организовать питание больших групп людей (в армии, летнем лагере и др.) как можно полезнее и как можно дешевле. Ясно, что эти цели, вообще говоря, совсем не совпадают, т.е. при моделировании будет несколько критериев, между которыми надо искать баланс.

Игровые модели могут иметь отношение не только к детским играм (в том числе и компьютерным), но и к вещам весьма серьезным. Например, полководец перед сражением в условиях наличия неполной информации о противостоящей армии должен разработать план: в каком порядке вводить в бой те или иные части и т.д., учитывая и возможную реакцию противника. Есть специальный достаточно сложный раздел современной математики - теория игр, - изучающий методы принятия решений в условиях неполной информации.

Наконец, бывает, что модель в большой мере подражает реальному процессу, т.е. имитирует его. Например, моделируя изменение (динамику) численности микроорганизмов в колонии, можно рассматривать много отдельных объектов и следить за судьбой каждого из них, ставя определенные условия для его выживания, размножения и т.д. При этом иногда явное математическое описание процесса не используется, заменяясь некоторыми словесными условиями (например, по истечении некоторого отрезка времени микроорганизм делится на две части, а другого отрезка - погибает). Другой пример - моделирование движения молекул в газе, когда каждая молекула представляется в виде шарика, и задаются условия поведения этих шариков при столкновении друг с другом и со стенками (например, абсолютно упругий удар); при этом не нужно использовать никаких уравнений движения. Можно сказать, что чаще всего имитационное моделирование применяется в попытке описать свойства большой системы при условии, что поведение составляющих ее объектов очень просто и четко сформулировано. Математическое описание тогда производится на уровне статистической обработки результатов моделирования при нахождении макроскопических .характеристик системы. Такой компьютерный эксперимент фактически претендует на воспроизведение натурного эксперимента: на вопрос «зачем же это делать» можно дать следующий ответ: имитационное моделирование позволяет выделить «в чистом виде» следствия гипотез, заложенных в наши представления о микрособытнях, очистив их от неизбежного в натурном эксперименте влияния других факторов, о которых мы можем даже не подозревать. Если же, как это иногда бывает, такое моделирование включает и элементы математического описания событий на .микроуровне, и если исследователь при этом не ставит задачу поиска стратегии регулирования результатов (например, управления численностью колонии микроорганизмов), то отличие имитационной модели от дескриптивной достаточно условно; это, скорее, вопрос терминологии.

### 2.4. НЕКОТОРЫЕ ПРИЕМЫ ПРОГРАММИРОВАНИЯ

Как уже отмечалось выше, наибольшая польза при изучении вопросов, затрагиваемых в данной главе, будет в том случае, если, наряду с проникновением в общую методологию моделирования и осознанием существа рассматриваемых задач, читатели реализуют несколько моделей в виде компьютерных программ достаточно профессионального уровня, т. е. не просто заглатывающих одни числа и выводящих на экран или принтер длинные столбцы других, но и реализующих некоторый интерфейс пользователя с элементами диалога, графики и т.д.

Именно поэтому мы не будем касаться здесь вопроса об использовании специализированных пакетов программ решения математических задач и графической поддержки. Во-первых, такие пакеты различны на разных ЭВМ, постоянно появляются новые, и за этим не угнаться никакому учебнику. Во-вторых (и это главное), самостоятельно запрограммированное осознанное решение системы дифференциальных уравнений методом Эйлера и сопровождающая его простенькая самостоятельно созданная на BASIC или PASCAL иллюстрация в виде графика или движущегося по экрану предмета дают тому, кто это сделал, куда больше, чем обращение к пакету MATHEMATICA с его могучими программами. От того, что задача будет решена, скажем, программой из этого пакета по методу Рунге - Кутта - Мерсона с автоматическим выбором шага интегрирования, реальных знаний почти не прибавится. Здесь мы упираемся в фундаментальные проблемы целей образования. Разумеется, овладение возможностями одного из пакетов математической поддержки (Eureka, MathCad, MathLab, Derive и др. ) вполне желательно и может принести определенную пользу, но не заменить самостоятельно проделанной работы.

Далее остановимся лишь на отдельных вопросах программирования, наиболее существенных при разработке моделирующих программ, ограничиваясь лишь примерами и советами, призванными уточнить суть дела.

**Организация диалога человек - ЭВМ в процессе проектирования и разработки моделирующих программ.** Организация интерфейса пользователя при разработке профессиональных программ - работа не менее (а часто и более) сложная, чем реализация логической или расчетной части задачи. Многие, работая с табличными процессорами, текстовыми редакторами и другими широко распространенными программами, не склонны задумываться, сколько усилий потребовала от разработчиков организация экрана, на котором всегда есть нужная в данный момент информация, причем именно в том месте экрана, где ее легче найти, система подсказок и помощи, меню для смены режимов работ и т.д. Очень жесткие требования по оформлению диалога предъявляются к обучающим и контролирующим Программам, компьютерным дидактическим играм, многим другим классам программ. Именно по соответствию этим требованиям сегодня в первую очередь судят о профессионализме разработки.

Характерной чертой практически всех профессиональных моделирующих программ является их диалоговый характер. Диалоговые (иначе говоря, интерактивные) программы - большой шаг вперед в развитии процесса взаимодействия человека с компьютером. Обсудим коротко лишь некоторые черты подобной организации человеко-машинного интерфейса.

Диалог человека с машиной весьма далек от привычного человеческого диалога. Главная особенность - ограниченные возможности компьютера в форме восприятия и анализа смысла человеческих сообщений. Воспринимаемые компьютером в ходе диалога сообщения жестко ограниченны в своих конструкциях как синтаксически, так и семантически. Синтаксически -т. е. сообщения воспринимаются лишь в предусмотренной автором программы форме. Например, оговорено, что сообщение содержит 5 букв кириллицы - любое другое система признает неверным и отвергнет. Однако, она в этом случае сочтет верным любой текст из 5 букв кириллицы, даже совершенно бессмысленный; такой текст будет отвергнут на уровне семантического анализа, в котором исследуется, приписалли автор программы этому тексту какой-либо смысл.

Слово «семантика» означает смысловую сторону сообщения. В распознании смысла сообщений компьютерные программы пока преуспели гораздо меньше, чем в синтаксическом анализе, и могут несравненно меньше, чем человек. Работы в этом направлении - одно из самых перспективных направлений развития информатики. Подавляющее число программ, реализованных на компьютерах с первого по четвертое поколения, могут распознавать «верно - неверно» лишь по сравнению с эталонными ответами, котооые либо заранее заложены в эти программы, либо выражены в числовой форме и рассчитываются программами.

Обсудим технические вопросы ведения диалога с компьютером. В моделирующих программах, в основном, используются следующие типы элементов диалога:

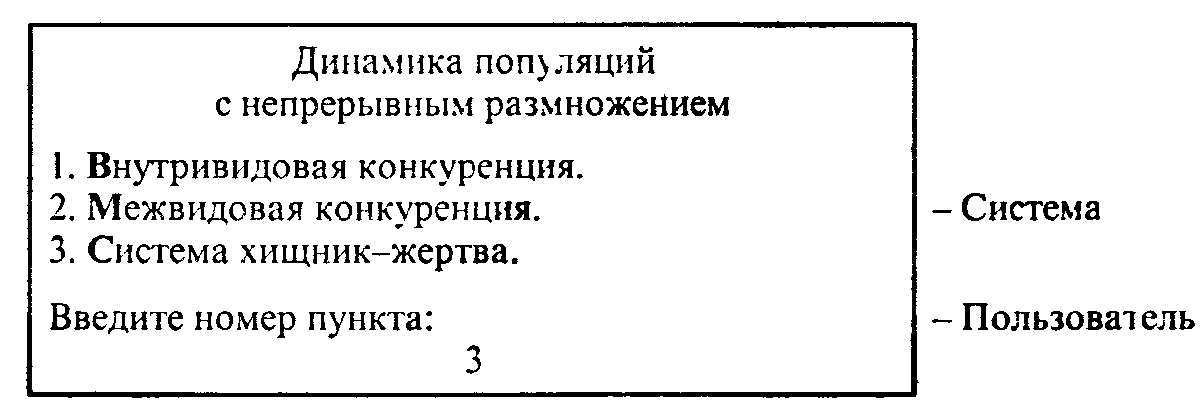
• меню;

• вопросы, требующие ответа «да/нет»;

• ответы по шаблону;

• команды.

Меню - на экране нумерованное или помеченное иным способом (например, выделением заглавных букв первых слов) множество функций системы, реализация которых возможна в настоящий момент, рис. 7. 3. Пользователь выбирает нужную функцию и тем самым начинает диалог.



*Рис. 7. 3, а.* Пример кадра меню

*Программа 146.* Фрагмент программы, реализующей меню с рис. 7. 3,а.

Program St; Uses Crt;

Var M : Byte; Logic : Boolean;

(Про\"(о}е\"(a)ypa печати меню и выбора соответствующего пункта)

Procedure Menu (Var M : Byte) ;

Begin Repeat (Выбор пункта меню с верифика\"(о}ией вво\"(а)а}

CIrScr; GotoXY (15, 7);

Write('\"(А}ИНАМИКА ПОПУЛ\ss\"{0}ИЙ С НЕПРЕРЫВНЫМ РАЗМНОЖЕНИЕМ')•;

GotoXY(22, 10); Write('1. Внутриви\"{а}овая конкурен\"(о}ия. ');

GotoXY(22, 12); Write('2. Межви\"{а}овая конкурен\"(о}ия. ');

GotoXY(22, 14); Write('3. Система хищник-жертва. ');

GotoXY(22, 16); Write('4. Коне\"{о} работы. ');

GotoXY(15, 22); Write('ВЫБЕРИТЕ НОМЕР НУЖНОГО ПУНКТА МЕНЮ ');

Read(M)

Until M In [1. . 4]

End;

Begin (OCHOBHA\ss ПРОГРАММА)

Logic := False;

Repeat

Menu(M); (меню)

Case M Of

1 : Begin (Внутриви\"{а}овая конкурен\"{о}ия} End;

2 : Begin {Межви\"{а}овая конкурен\"{о}ия)

CIrScr; WriteLn('Пункт меню "Межви\"{а}овая конкурен\"{о}ия"') ;

Repeat Until KeyPressed

End;

3 : Begin {Система хищник-жертва} End;

4 : Logic := True (Выхо\"{а} из программы)

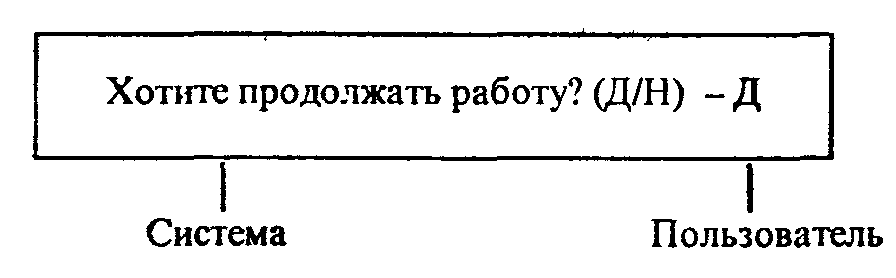
End

Until Logic

End.

Программа может предусматривать и другие способы организация реакции пользователя: вводом букв В, M. С; еще удобнее - установкой курсора в нужную строку клавишами и (или) «мышью» (если она есть). Сути дела это не меняет.

Вопрос, требующий ответа «да/нет» - частный случай меню, который, однако, целесообразно выделить.



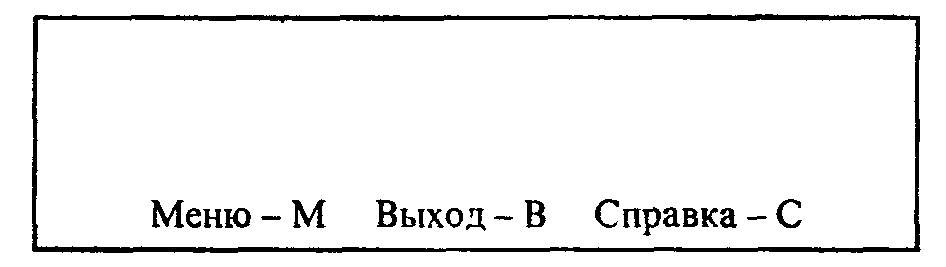
*Рис. 7. 3, б.* Выбор одного из возможных ответов

Ответы по шаблону даются чаще всего при ожидании числовой информации, когда хотят избежать длинных объяснений о формате числа, количестве значащих цифр.

Команда подается пользователем. В диалоговой моделирующей программе может быть строго фиксированный набор команд, краткий перечень которых сопровождает все или часть кадров в строке-подсказке; по этим командам пользователь может в любой момент прекратить работу или изменить ее характер. Например, в нижней части кадра, содержание которого сейчас неважно, написано:



*Рис. 7. 3, в.* Форматированный ввод данных



*Рис. 7, 3, г.* Выбор последующего режима работы

В основной части кадра-некий текст, предписывающий определенную реакцию. Но указанные внизу команды имеют приоритет, и при нажатии «М» мы попадем в основное меню независимо от того, что требовалось в верхней части кадра. При этом предусматривается возможность возврата в кадр, из которого вышли.

Разработка диалоговой моделирующей программы - достаточно сложный процесс, включающий создание сценария, программирование, разработка инструкций и документации. В нашем курсе невозможно требовать реализации всех этих элементов в каждой из выполняемых работ, поскольку тогда программистская работа отодвинет на второй план все остальное; можно лишь рекомендовать использовать элементы организации диалога. В отчетной (курсовой, дипломной) работе, напротив, уместна реализация весьма интересных диалоговых оболочек; в текущих же работах элементы разумной организации диалога, оформления экрана вполне уместны и не требуют чрезмерных усилий. Весь диалог при моделировании того уровня, который соответствует данному курсу, невелик по объему и несложен по логической организации. Согласитесь, что ситуация, когда программа требует ввода данных и извещает об этом лишь знаком «?» на совершенно пустом экране, вам, вероятно, знакома, как знакома и реакция сидящего за компьютером человека, который не понимает, что бы это значило (что особенно смешно, если он сам - автор этой программы). Так не стоит ли снабдить этот знак фразой «введите начальную скорость *v0,* звуковым сигналом, привлекающим внимание, и т.д. ? Как быть с огромным количеством чисел, проносящихся по экрану в процессе решения дифференциальных уравнений и сметающих на своем пути всю нужную информацию? В отлаженной программе такого просто не должно быть.

Спроектируем возможное оформление диалога и формы представления результатов для одной из рассмотренных ниже задач по моделированию. Уместно начать с кадра-заставки, где написаны название задачи, фамилия автора программы - это минимум. Если удастся создать несложный рисунок, иллюстрирующий задачу - очень хорошо, он уместен в том же кадре. Затем, через несколько секунд, кадр сменяется следующим. На нем - основное уравнение (т.е. сама математическая модель) и предложение ввести исходные данные - с представлением шаблонов, или, если это затруднительно, то в свободной форме, но обязательно со словесным запросом значения каждого параметра по очереди. Более чем существенно разумно спроектировать формы для вывода результатов - таблицы, графики, траектории и т.д., их очередность, взаимное расположение.

Неплохо, если в процессе счета программа выводит на экран стационарную или мерцающую надпись типа «Подождите, идут расчеты». Когда счет закончен, на экране может появиться меню с предложениями: «Вывод таблицы результатов», «Вывод графика *N(t)»,* «Запись параметров и результатов в файл», «Печать результатов», «Выполнение очередного расчета», «Окончание расчетов».

Разумеется, можно усовершенствовать это оформление, но каждый шаг здесь требует определенной программистской культуры и немалого труда.

Повсеместный переход на графические интерфейсные операционные системы уровня WINDOWS, появление сред визуального программирования типа DELPHI ведут к значительному повышению стандартизации пользовательского интерфейса. Если при разработке моделирующих программ читатели используют возможности таких сред*,* то могут получить вполне профессиональные программные продукты.

Отметим, наконец, что те несколько программ, которые приведены ниже при рассмотрении конкретных моделей, не претендуют на совершенный интерфейс. Их цель - пояснение существа рассматриваемых процессов; попытка же оформить их «по всем правилам» привела бы к большим, плохо читаемым программам, не выполняющим этой разъяснительной функции.

**Компьютерная научная графика**. С простейшей научной графикой мы встречаемся очень рано. Уже в курсе математики 6-7 классов есть достаточно абстрактные и условные рисунки, которые дети легко воспринимают - например, график линейной функции. А ведь на нем немало элементов, интерпретация которых, если задуматься, вовсе не очевидна: линии, штрихи, стрелки, масштабы и т.д. Несмотря на это, понять по графику свойства сложной функции человеку гораздо легче, чем из соответствующей формулы, хотя в ней информации, строго говоря, гораздо больше. Так уж устроено человеческое восприятие, что рисунки, пусть даже условные, гораздо легче воспринимаются рассудком, чем сложные формулы или колонки чисел.

В современной прикладной информатике этим обстоятельством очень широко пользуются, и в ней сформировалось соответствующее направление - машинная (компьютерная) графика. По определению, машинная графика - раздел информатики, в рамках которого исследуются и разрабатываются технические, математические, программные и методические средства и приемы использования ЭВМ для создания, обработки, хранения и практического применения графических изображений.

В машинной графике выделяют несколько разделов.

*Иллюстративная графика,* простейшими программными средствами которой являются всем знакомые диалоговые программы - графические редакторы, служит для создания изображений, за которыми, как правило, не стоят какие-либо математические объекты (уравнения и др.). Это - средство реализации свободного полета мысли и воображения, любимое занятие начинающих приобщаться к компьютеру.

*Деловая графика* существенно «скучнее». Когда администратору, бухгалтеру, экономисту и т.д. нужно перевести сухие колонки чисел в столбчатую диаграмму, круговую диаграмму, график, достаточно вызвать такую программу и в ходе диалога сообщить ей заголовки, подписи, разметки, цвета и т.д. и имя файла, в котором по определенным правилам записаны указанные числа. Система построит заданное изображение на экране, выведет его на принтер.

Одна из самых сложных и специализированных разновидностей систем машинной графики - *инженерная графика,* известная также под именем САПР - системы автоматизированного проектирования. Это диалоговые системы, предназначенные для автоматизации процесса проектирования технических объектов, создания полных комплектов проектных документов с учетом существующих норм стандартов.

И, наконец, *научная графика -* наиболее актуальная для изучаемого курса и наименее всех допускающая единое описание. Универсальных систем компьютерной научной графики, по-видимому, не существует из-за огромного разнообразия задач. Часто программы, реализующие наглядное изображение решения научной задачи (почти всегда по итогам математического моделирования), встраиваются внутрь основной программы, пишутся на том же самом языке программирования.

Общую цель научной графики можно сформулировать так: сделать невидимое и абстрактное «видимым». Берем последнее слово в кавычки, так как часто эта «видимость» весьма условна. Можно ли увидеть распределение температуры внутри неоднородно нагретого тела сложной формы без введения в него сотен микродатчиков, т.е. , по существу, его разрушения? - Да, если есть соответствующая математическая модель, и, что очень важно - договоренность о восприятии определенных условностей на рисунке. Можно ли увидеть распределение металлических руд под землей без раскопок? Строение поверхности чужой планеты по результатам радиолокации? На эти и множество других вопросов ответ - да, можно, с помощью машинной графики и предшествующей ей математической обработки. Изображения такого рода систематически публикуются научными и научно-популярными изданиями.

Более того, можно «увидеть» и то, что строго говоря, вообще плохо соответствует слову «видеть». Так. возникшая на стыке химии и физики наука - квантовая химия - дает нам возможность «увидеть» строение молекулы. Эти изображения -верх абстракции и системы условностей, так как в атомном мире обычные наши понятия о частицах (ядрах, электронах и т.д. ) принципиально не применимы. Однако, многоцветное «изображение» молекулы на экране компьютера для тех, кто понимает всю меру его условности, приносит большую пользу, чем тысячи чисел, являющихся плодом квантовохимического расчета.

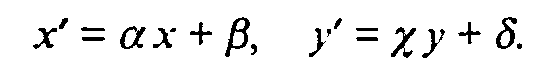
При реализации относительно несложных задач нашего курса прибегать к помощи каких-либо стандартных пакетов машинной графики - дело вовсе необязательное. Конечно, можно себе представить, что итоговое оформление экрана в конце процесса моделирования выполнено с помощью графического редактора, а содержащийся в нем график - с помощью пакета деловой графики (или еще какого-то), но возникающую при этом проблему совмещения разных систем программирования не всегда легко решить. Целесообразнее, по-видимому, ориентироваться на тот язык программирования, на котором реализуется математическая модель.

Приведем несколько конкретных примеров, привязанных к нашему курсу.

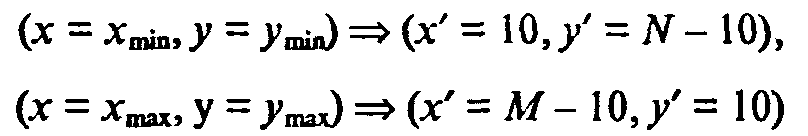
**Траектории движения тел, графики**. В ряде рассмотренных ниже задач уместно иллюстрировать процесс моделирования изображениями движущихся объектов и их траекториями. Мы сознательно ограничивались случаями плоских (двумерных) движений, которые легко отобразить на плоском экране компьютера.

Поскольку основные графические операторы языка BASIC или процедуры модуля GRAPH Turbo Pascal нашим читателям известны, опишем лишь общие моменты построения графиков и траекторий. Напомним, что в главе 3 в разделе, посвященном PASCAL, приведена программа построения графика аналитически заданной функции на произвольном отрезке; здесь мы детализируем рассмотрение. Пусть численные расчеты уже закончены и нам известны границы значений координат [*x*min, *x*max] и [*y*min, *y*mах] и есть таблица значений *х* и *у* в некоторые моменты времени, разделенные равными промежутками: 0, *t*, 2*t*, 3*t*,..., *пt.* Требуется построить графики зависимости *x(t)*, *у(t)* и траекторию. Проиллюстрируем это, используя графические процедуры PASCAL.

С помощью директивы Uses Graph и процедуры InitGraph (<параметры>) осуществляется переход в графический режим, в котором можно строить изображения. Необычная ориентация «экранной» системы координат создает определенные проблемы при построении графиков и траекторий. Мы хотим выводить их и задавать координаты точек в «естественной» системе координат *x, y,* изображенной на рис. 7.4, а графические процедуры (Circle, Line. OutText и др.) воспринимают аргументы в «экранной» системе *x'*, *у'.* Сделаем разметку так,как показанонарисунке, и произведем линейное преобразование координат

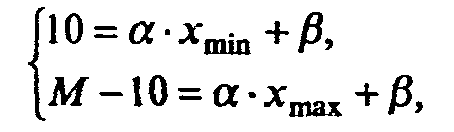


Если известны разрешающая способность экрана - *М* точек по оси *х'* и *N* точек по оси *у',* то для нахождения коэффициентов *α, β, γ, δ* достаточно связать любые две точки в разных системах координат, например

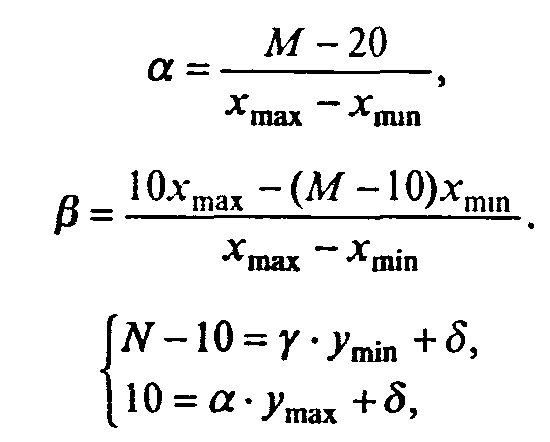


(отступ на 10 позиций от краев экрана позволит создавать подписи, разметку осей и

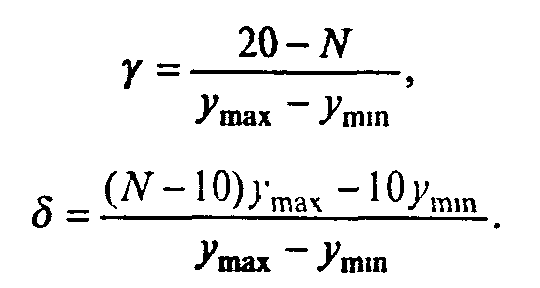
др.). Имеем



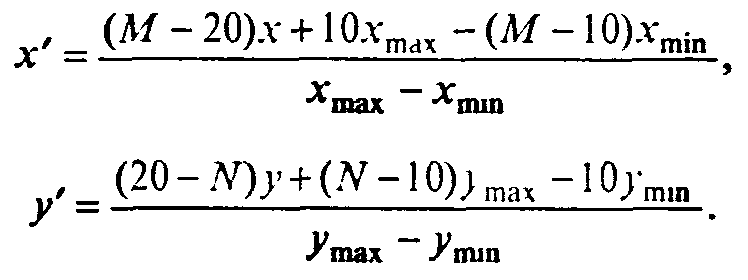
откуда

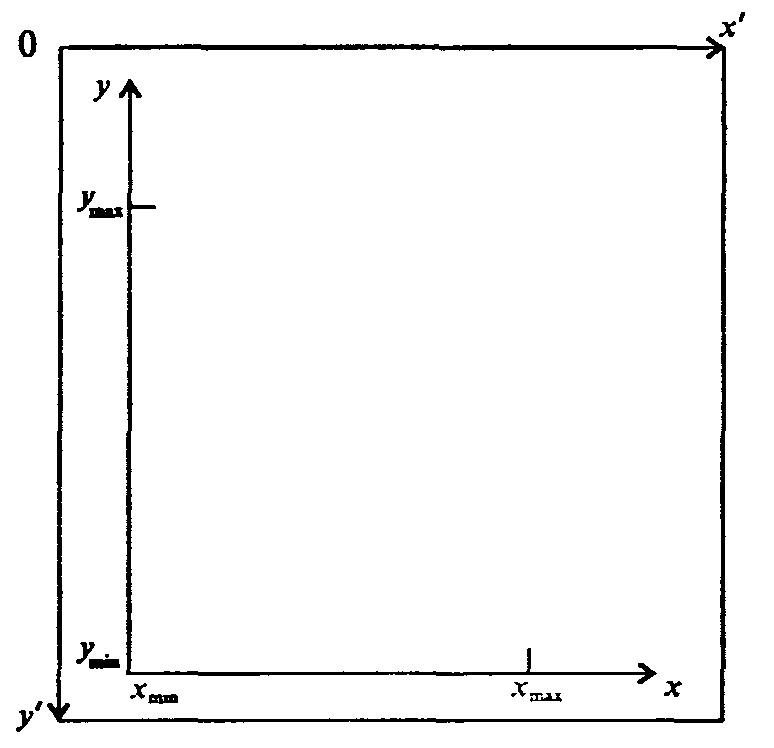


откуда



Таким образом, перевод одних координат в другие осуществляется по формулам





*Рис. 7.4.* Экранная и «естественная» системы координат

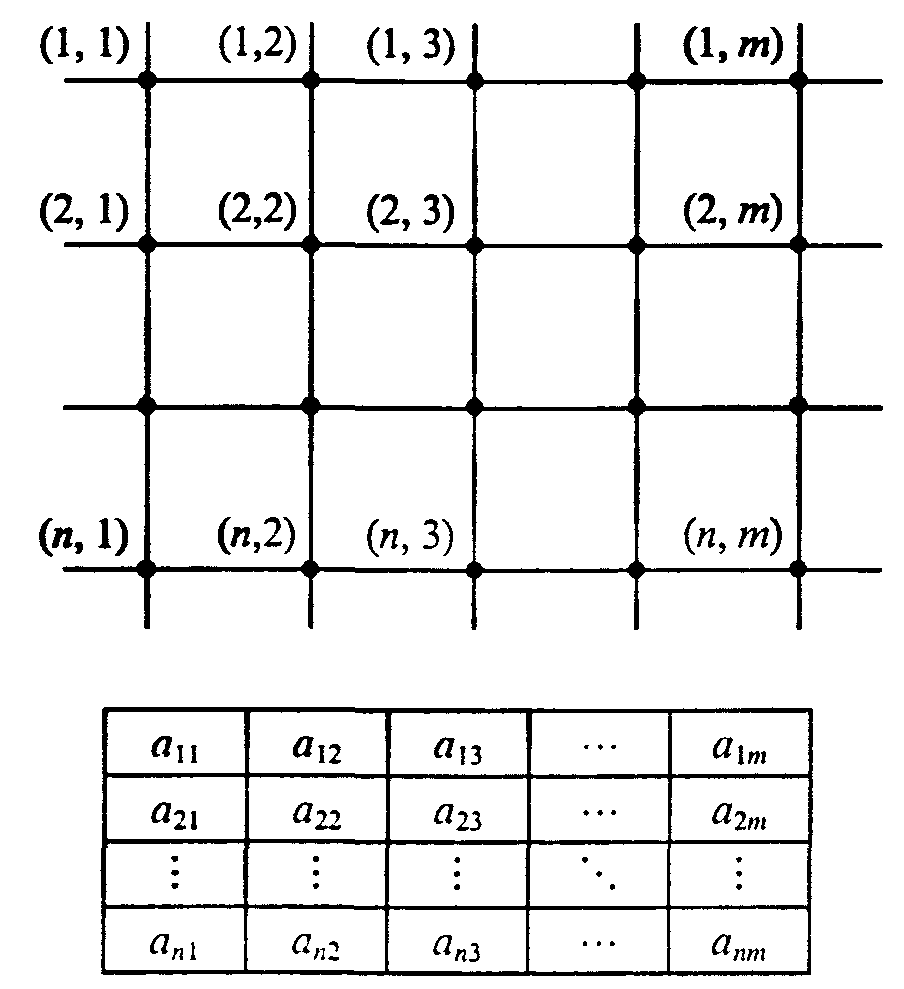
Теперь достаточно поставить точку с нужной координатой (*x*, *у)* с помощью процедуры PutPixel, а введя ее в цикл, изобразить график или траекторию. Если же требуется изобразить движение тела, то перед выводом на экран очередной точки достаточно стереть предыдущую или воспользоваться несколькими видеостраницами - соответствующие приемы программирования читателю, скорее всего, известны. Отметим, что создание на экране дисплея динамических зрительных изображений - так называемая анимация - одно из перспективных направлений искусственного интеллекта-раздела современной информатики.

**Изолинии.** В задачах моделирования достаточно стандартная проблема - построение линий (поверхностей), вдоль которых некоторая функция имеет одинаковое значение, называемых изолиниями (изоповерхностями). Это очень распространенная задача визуализации характеристик некоторого скалярного поля в приближении сплошной среды: изотермы - линии равной температуры, изобары - линии равного давления, изолинии функции тока жидкости или газа, по которым легко можно представить себе их потоки, изолинии численностей экологической популяции на местности, изолинии концентрации вредных примесей в окружающей среде и т.д.

Опишем типичную процедуру построения изолиний на экране компьютера. На старте мы имеем двумерную таблицу значений некоторой величины А, полученную в ходе математического моделирования; числа в этой таблице соответствуют значениям этой величины в узлах пространственной сетки (рис. 7.5).

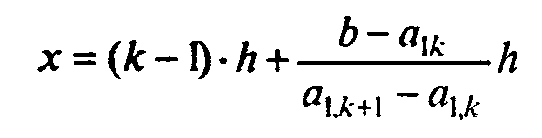
Зададим некоторый, совершенно условный, пространственный шаг *h* между соседними узлами по горизонтали и вспомогательную систему координат, в которой узел (1, 1) имеет координату (0, 0), узел (1, 2) - координату (*h*, 0), узел (1, 3) - координату *(2h, 0)* и т.д. Если шаг по вертикали *h*\*, то узел *(i, j)* в этой системе имеет координату ((*i*-1) ∙ *h,* (*y*-1) ∙ *h*\*).

Предварительно найдем в таблице наибольшее и наименьшее значения величин *аij -* допустим, это *a*min и *а*mах. Пусть *b -* некоторое промежуточное значение: *a*min *< b <* *a*max. Обсудим в общих чертах, как построить изолинию *A* = *b.* Будем для этого (в цикле) просматривать вначале все пары ближайших чисел в первой строке таблицы в поисках такой пары, для которой *b* находится «внутри». Допустим, число *b* находится между *a1k* и *a1,k+1*, т.е. либо *a1k* < *b <* *a1,k+1*, либо *a1k* > *b >* *a1,k+1*.



*Рис. 7.5.* Пространственная сетка и соответствующая ей таблица значений величины А

С помощью линейной интерполяции найдем соответствующую горизонтальную координату точки, в которой *А = b:*



(координата *у* определяется номером горизонтальной линии; в данном случае у = 0).

Найденные координаты запомним и просмотрим первую строку в таблице до конца, затем просмотрим вторую строку и т.д. Покончив с просмотром строк, мы получим часть точек, соответствующих изолинии *А = b.*

После этого займемся просмотром столбцов. Допустим, во втором столбце нашлась пара чисел, для которой число *b* находится между *аp2* и *ap+1,2*. Она дает следующую точку для изолинии. Закончив просмотр всех столбцов, мы получим максимально возможный набор координат точек, принадлежащих данной изолинии. Выведя их на экран в нужном масштабе, получим точечное изображение изолинии *А = b,* после чего можем, взяв другое значение *b,* построить следующую изолинию. Более детально эта процедура изложена ниже в пункте 3.8 на примере построения линий равного потенциала электрического поля.

**Условные цвета, условное контрастирование**. Еще один интересный прием современной научной графики - условная раскраска. Она находит широчайшее применение в самых разных приложениях науки и представляет собой набор приемов по максимально удобной, хотя и очень условной, визуализации результатов компьютерного моделирования.

Приведем примеры. В различных исследованиях температурных полейвстаетпроблема наглядного представления результатов. Самый простой (и, с точки зрения специалиста, весьма неэффективный) - привести карту (чертеж, план), в некоторых точках которой обозначены значения температуры.

Другой способ - набор изотерм - гораздо эффективнее; к нему прибегают некоторые газеты, давая состояние и прогноз погоды. Но можно добиться еще большей наглядности, учитывая, что большинству людей свойственно, сравнивая разные цвета, воспринимать красный как «горячий», голубой как «холодный», а все остальные - между ними. Допустим, что на некоторой территории температура в данный момент имеет в разных местах значения от -25°С до + 15°С. Разделим этот диапазон на участки с шагом, равным, например, 5°

[-25,-20], [-20,-15],...,[+10,+15],

и закрасим первый из них в ярко-голубой, последний - в ярко-красный, а все остальные - в промежуточные оттенки голубого и красного цветов. Получится замечательная наглядная картина температурного поля.

А что делать, если дисплей монохромный?! Или если изображение надо перенести с цветного дисплея на бумагу при отсутствии возможности цветной печати? -Тогда роль цвета может сыграть контраст. Сделаем самый «горячий» участок самым темным, самый «холодный» - прозрачным, а остальные - между ними. Эффектность, конечно, меньше, чем при цветовой раскраске, но для наметанного глаза изображение информативно.

То же самое можно делать при иллюстрации температурного поля и на поверхности обрабатываемой на станке детали, и на поверхности далекой планеты.

В нашем курсе есть несколько моделей, в которых можно (и очень полезно) прибегнуть к подобному приему визуализации. В задаче о теплопроводности в стержне это даже не очень сложно; можно делать такие условные раскраски при моделировании распределения электрических полей. Если заниматься имитационным моделированием конкурирующих популяций, то, раскрасив их в разные цвета, можно получить на экране причудливые картины, передающие ход конкурентной борьбы.

Условные раскраски бывают и гораздо более абстрактными, чем в описанных выше случаях. При моделировании сложных органических молекул компьютер может выдавать результаты в виде многоцветной картины, на которой атомы водорода изображены одним цветом, углерода - другим и т.д., причем атом представлен шариком (кружочком), в пределах которого плотность цвета меняется в соответствии с распределением электронной плотности.

При поиске полезных ископаемых методами аэрофотосъемки с самолетов или космических спутников компьютеры строят условные цветовые изображения распределений плотности под поверхностью Земли. Подобных примеров можно привести достаточно много.

Подведем итог: изображения в условных цветах и контрастах - мощнейший прием научной графики. Он позволяет понять строение не только плоских, но и объемных (трехмерных) объектов, дает в руки исследователя один из замечательных методов познания. Приведем в качестве иллюстрации фрагмент программы.

*Программа 147.* Условная раскраска неравномерно нагретого стержня в разные моменты времени (по заранее заготовленным данным).

Program Stergen;

Uses Crt, Graph

Type Mas2 = Array [0..10, 0..4] of Real;

Const (Массив распределения температуры в разные моменты времени)

U : Mas2 *=*

((3.000, 3.667, 4.333, 5.000, 3.000), (3.000, 3.628, 4.128, 3.952, 3.000), (3.000, 3.514, 3.783, 3.593, 3.000), (3.000, 3.377, 3.546, 3.396, 3.000), (3.000, 3.267, 3.381, 3.272, 3.000), (3.000, 3.187, 3.266, 3.188, 3.000), (3.000, 3.131, 3.185, 3.131, 3.000), (3.000, 3.091, 3.129, 3.091, 3.000),

(3.000, 3.064. 3.090, 3.064, 3.000), (3.000, 3.044, 3.063, 3.044, 3.000), (3.000, 3.031, 3.044, 3.031, 3.000));

Var

M, I, J, N1, Nt : Integer; MaxF, L, T, HI, Ht : Real;

Procedure Initialize; (Процедура инициализации графического режима)

Var GraphDriver, GraphMode : Integer;

Begin

DetectGraph(GraphDriver, GraphMode) ;

InitGraph(GraphDriver, GraphMode, '');

End;

(Графическая иллюстрация решения)

Procedure Postar.ovka (U : Mas2; Nt, N1 : Integer; HI, L, MaxF : Real);

Var X\_N, Shag, Y\_N, Shir, Dlin, Color, I, J, K, Y : Integer;

Flag .: Boolean; Ff : String; Col : Array [0..15] Of Byte;

Begin

Initialize; (Инициализация графического режима)

X\_N := GetMaxX Div 6;

If Nt <= 6 Then M := Nt Else M := Nt Div 2;

Y\_N := GetMaxY Div M - 20; Shir := Y\_N Div 2;

Dlin := GetMaxX -2 \* X\_N; Shag := Trunc(Dlin / N1); Str(Shag,Ff) ;

Col[0] := 0; Col[l] := 8; Col[2] := 1; (Палитра цветов)

Col[3] := 9; Col[4] := 3; Col[5] := 11;

Col[6] := 2; Col[7] := 10; Col[8] := 14;

Col[9] := 13; Col[l0] := 5; Col[ll] := 12; Col [12] := 4;

For I :== 0 To M - 1 Do (номер временного промежутка)

Begin

For J := 0 To N1 - 1 Do (номер участка стержня)

Begin

Flag := False;

For К := 0 To Shag Do

Begin

For y:= 0 To Shir Do

Begin

Color := 1 + Round((U[I, J] + (определение номера цвета)

(U[I, J + 1] - U[I, J]) \* К / Shag - U[0, 0]) / 3 \* 16);

If Random(64) > 32

Then If Random(64) > 32 Then Color := Color + 1 Else

Color := Color - 1;

If Not Flag Then (вывод текущей температуры)

Begin

Str((U[I,J]+(U(I,J+l]-U[I,J])\*K/Shag) : 5 : 3, Ff);

OutTextXY(K+X\_N+Shag\*J, Y\_N\*(1+1)-19, Ff) ;

Flag := True

End;

{рисование точки}

PutPixel(K+X\_N+Shag\*J, Y+Y\_N\*(1+I), Col[Color])

End

End

End

End;

SetColor(White); OutTextXY(150, 450, 'Нажмите любую клавишу ');

Repeat Until KeyPressed; CloseGraph

End;

Begin (ОСНОВНАЯ ПРОГРАММА)

L := 4; Т := 10; Hi := 1; Ht := 1;

N1 := Trunc(L / HI); Nt := Trunc(T / Ht); MaxF := 5;

Postanovka (U. Nt, N1, HI, L, MaxF)

End.