Министерство образования Российской Федерации

Северо-Западный государственный заочный

технический университет

В. В. Дембовский

Компьютерные технологии

в металлургии и литейном производстве

Часть 2

Утвердено редакционно - издательским советом университета
в качестве учебного пособия

Санкт – Петербург
2003

УДК 512/972

Дембовский В.В. Компьютерные технологии в металлургии и литейном производстве. Учеб. пособие, часть 2. – СПб.: СЗТУ, 2003. – 155 с.

В части 2 учебного пособия приведены сведения о специализированных пакетах прикладных программ для решения математических задач, описаны принципы программирования задач в среде Visual Basic, моделирования металлургических процессов, элементы САПР и пр.

Пособие предназначено для студентов специальности 110400 − «Литейное производство чёрных и цветных металлов» (специализация 110409 – «Литейное производство и экономика металлургии»). Может быть использовано студентами специальности 060800 – «Экономика и управление на предприятии» (специализация 060802 – «Экономика и управление на предприятиях металлургии»), а также - слушателями факультета повышения квалификации профессорско-преподавательского состава, аспирантами, инженерами и всеми, кто желает в короткое время освоить персональный компьютер и использовать его в своей повседневной деятельности.

Рецензенты:

1. Кафедра металлургии и литейного производства Северо – Западного государственного заочного технического университета (заведующий кафедрой А.А.Яценко, канд. техн. наук, доц.);
2. Н.А.Хлямков, канд. техн. наук, ст. научн. сотрудник ЦНИИ КМ «Прометей».

ПРЕДИСЛОВИЕ

Успехи микроэлектроники, начиная с 80-х гг. XX в., обусловили бурный рост производства и стремительное совершенствование схемно-конструктивных решений в области персональных компьютеров. Соответственно этому быстрыми темпами развивается системное и прикладное программное обеспечение современных средств вычислительной техники. Без преувеличения можно утверждать, что компьютерная проблематика, включая её аппаратное и всё более интеллектуально - насыщенное программное обеспечение, а также вопросы практического использования персональных компьютеров находятся на острие научно-технического прогресса во всех промышленно развитых странах мира.

Еще десяток лет тому назад специалисты насчитывали до 400 тысяч видов применения компьютеров и компьютерных технологий в различных сферах человеческой деятельности.

Обеспеченный соответствующими программами компьютер многократно усиливает научно-технические и производственные возможности персонала, существенно повышает эффективность производства и научных исследований. Именно эта эффективность является движущей силой неуклонно расширяющегося ежегодного выпуска многомиллионной массы персональных компьютеров, причем темпы качественного совершенствования и количественного роста производства этого вида продукции ведущих мировых фирм оставляют далеко позади аналогичные показатели для изделий других видов.

Металлургия, к которой относится и литейное производство, является наукоёмкой, сложной в производственном отношении и многосвязной отраслью промышленности. Металлурги и литейщики, кроме задач непосредственного управления технологическими процессами, в своей деятельности часто сталкиваются с необходимостью выполнения достаточно сложных научно-технических и инженерно-экономических расчетов, решать задачи математического моделирования и оптимизации металлургических (литейных) объектов, обоснованно принимать те или иные решения, причем последние должны быть оптимальными с точки зрения достигаемых при их выполнении результатов.

Обычно время для решения подобных задач жестко ограничено, а производственная ситуация может стремительно изменять-

ся. В подобных условиях неоценимую помощь оказывает персональный компьютер, но для реализации этой помощи требуются определенные знания и умение применить компьютер на практике.

Обретению таких знаний и практических навыков призвано содействовать настоящее учебное пособие. В нём приведены сведения теоретического характера и практические методики использования IBM - совместимых персональных компьютеров желательно класса Pentium с операционной системой Windows 95 /98 /2000), пакетами прикладных программ Microsoft Office 97 / 2000, Visual Basic 5.0 / 6.0, Mathcad 7.0 / 8.0 / 2000. Уделено внимание вопросам передачи информации по компьютерным сетям. Манипулятор типа «мышь» использован в режиме одного щелчка рабочей (левой) клавиши мыши при необходимости *выделения* того или иного объекта и двойного щелчка для *открытия* файлов, т.е. их загрузки в оперативную память для выполнения.

Даны подробные пояснения к методикам решения часто встречающихся задач и конкретным примерам металлургической и литейной тематики. Научившись с помощью данного пособия решать эти задачи, пользователь сможет использовать персональный компьютер для решения и других задач, технологической, научно-технической и инженерно-экономической направленности.

Длительный опыт преподавательской работы автора в системе высшей школы позволяет ему надеяться на то, что приводимые в данном учебном пособии примеры имеют достаточно «прозрачный» смысл, понятный широкому кругу читателей. При этом сообщаемые пояснения являются вполне достаточными не только для решения задач под руководством преподавателя, например, в компьютерном классе учебного заведения, но также и при самостоятельной работе. Последнее обстоятельство может оказать существенную помощь студентам - заочникам.

Для компактности изложения в тексте приняты следующие условные обозначения манипуляций с мышью:

[1Л] – одиночный щелчок левой клавишей;
[2Л] – двойной щелчок левой клавишей;
[1П] – одиночный щелчок правой клавишей;
[2П] – двойной щелчок правой клавишей;

Названия других клавиш и кнопок (псевдокнопок на экране монитора) приводятся в квадратных скобках. Команды, берущие

начало из главного меню, выделены полужирным шрифтом, последовательность перехода от команды к следующей подкоманде обозначена отрезком вертикальной линии.

Предполагается знакомство читателя с курсом «Информатика», знание общих принципов работы персонального компьютера и элементарных действий пользователя в среде упомянутых операционных систем, умение выполнять операции над файлами и файловыми структурами.

Учебное пособие непосредственно предназначено для студентов специальности 110400 − «Литейное производство чёрных и цветных металлов» (специализация 110409 – «Литейное производство и экономика металлургии») по дисциплине «Информационные технологии в металлургии»».. Оно может быть использовано студентами специальности 060800 – «Экономика и управление на предприятии» (специализация 060802 «Экономика и управление на предприятиях металлургии») при изучении дисциплины «Информационные технологии в экономике», а также - слушателями факультета повышения квалификации профессорско - преподавательского состава, аспирантами, инженерами и всеми специалистами, которые желают в короткое время усовершенствовать умение использовать персональный компьютер в своей повседневной работе.

Автор с благодарностью примет замечания читателей по материалам настоящего учебного пособия, которые следует направлять на имя автора – проф. Дембовского В.В. (кафедра металлургии и литейного производства СЗТУ) по адресу, указанному на с. 155.

1. Применение специализированных пакетов прикладных программ для выполнения математических расчётов

К настоящему времени разработан ряд программных продуктов, предназначенных для выполнения различных математических операций с помощью компьютера без необходимости использования численных методов и программирования. Так, известны пакеты Mathcad, MathLab, Maple, Mathematica, Scientific Workplace и др. Их сравнительный обзор приведен в литературе [1]. Пакеты различаются возможностями и сложностью в управлении.

Наиболее простым для использования является пакет Mathcad, разработанный фирмой MathSoft, Inc (США). Известны такие версии этого пакета, как Mathcad 6.0, Mathcad Plus 6.0, Mathcad 7 Pro, Mathcad 8,0 и, наконец, Mathcad 2000. Перечисленные версии построены на единой идеологической основе и различаются лишь некоторыми (непринципиальными) деталями сервиса. Мы будем придерживаться последней на сегодня версии, то есть Mathcad 2000.

Следует отметить, что в сравнении с такими пакетами, как MathLab и Maple, Mathcad обладает несколько меньшими возможностями в решении сложных математических задач. Однако для рассматриваемой области этих возможностей в большинстве случаев достаточно.

Конкретно к числу достоинств Mathcad относятся:

- Естественность записи математических выражений, что совершенно отличается от записи тех же выражений в известных языках программирования и упомянутых выше пакетах.
- Кнопочное управление процессом решения задач, в результате чего нужные функции из числа встроенных в пакет вызываются щелчками мыши [1] на кнопках высвечиваемых на экране монитора панелей инструментов

1.1. Начальное ознакомление с Mathcad

Запустим стандартными действиями Mathcad. На экране появляется операционное поле и над ним − главное меню. Последнее очень похоже на то, которое имеется у текстового процес-

сора Word, а также у Excel и других разработок фирмы MicroSoft.

Из главного меню вводим последовательность команд

Вид (View) | Панели инструментов (Tool Boxes) | Математика (Math Palette)

для вызова на экран *математической* панели. Последняя содержит ряд кнопок, названия которых поясняются всплывающими подсказками при подведении к ним курсора.

С помощью кнопок математической панели можно вызвать другие панели инструментов (табл.1.1, с.8). В связи с широким распространением англоязычных версий Mathcad, приведены названия панелей на русском языке и их английский перевод.

Из главного меню аналогично разработкам фирмы MicroSoft можно варьировать размеры символов и их начертание.

Для изменения знаков, букв и цифр в формулах необходимо выделить символ установкой синего ограничителя, а затем задать нужный размер шрифта в окнах главной панели инструментов Mathcad действиями, аналогичными принятым в текстовом процессоре Word. Точно так же можно вместо обычного начертания выбрать шрифты:

[**Ж**], или [**B**]– полужирный (bold),
[***К***], или [***I***]– курсив (*italic*),
[**Ч**], или [**U**]– подчеркнутый (underline).

Кроме выборочного изменения отдельных символов в формуле можно заранее оговорить нужный стиль написания переменных и констант командами

Формат | Уравнение
(Format) | (Equation)

Возможно по желанию изменять значность чисел командой

Формат | Результат
(Format) | (Number)

Таблица 1.1. **Содержание математической панели инструментов Mathcad**

Кнопки	Назначение
Арифметические инструменты (Arithme – tics)	Вызов арифметической панели для ввода часто используемых символов
Инструменты некоторых знаков (Signs)	Вызов средств ввода знаков, дополняющих арифметическую панель
Программирование (Programming)	Ввод ключевых слов при составлении программы вычислений
Инструменты графики (Graph)	Вызов средств построения дву - и трехмерных графиков, гистограмм и пр. по результатам вычислений
Операторы математического анализа (Calculus)	Вызов панели ввода символов сумм, пределов, произведений, производных и интегралов
Символы греческого алфавита (Greek)	Ввод заглавных и строчных греческих букв
Векторные и матричные операции (Vector & Matrix)	Ввод символов создания векторов, матриц и их отдельных элементов
Панель инструментов «булево» (Boolean)	Ввод символов отношения и логических функций
Символические операторы (Symbolic)	Ввод знаков и ключевых слов символьных преобразований

с последующей установкой требуемого количества значащих цифр в дробной части числа.

Имеется средство масштабировать изображения *на экране* ([Scale] – масштаб), но это – только для визуального восприятия. Для того чтобы изменить шрифт *в распечатке*, нужно воспользоваться главным меню.

Курсор на операционном поле в Mathcad имеет вид красного крестика. Для фиксации курсора в нужном месте поля нужно щёлкнуть [1Л] на нём мышью.

Решим для начала несколько простых примеров с помощью Mathcad, для чего нужно внимательно следить за текстом и строго следовать приводимым в нём пояснениям.

Пример 1.1. Пусть необходимо вычислить

$$ln(14{,}7) / \cos(0{,}25\,\pi)$$

Решение

Выбираем и фиксируем начальное положение курсора. Проверяем наличие на экране *математической* панели инструментов и, если она отсутствует, вызываем её из главного меню описанными выше действиями. Из математической панели вызовем панель *арифметическую* путем [1Л] на её кнопке.

В арифметической панели делаем [1Л] на кнопке с изображением символа ln. На месте курсора появляется изображение $ln(\blacksquare)$ в рамке, внутри которой расположен *ограничитель* в виде пересекающихся под прямым углом отрезков прямых линий, выделенных синим цветом. Зачерненный прямоугольник в скобках служит *держателем знакоместа* (place holder), или маркером. Здесь и далее на место таких элементов формульных изображений следует вводить значения аргументов или других чисел, запрашиваемых компьютером. Для ввода числа 14,7 переводим ограничитель в скобки с помощью мыши или клавиши управления курсором и средствами клавиатуры на месте держателя знакоместа пишем 14.7. Здесь десятичный разделитель в виде точки используется во всех версиях Mathcad. Переводим ограничитель вправо за пределы формулы и вводим знак деления. Это можно сделать непосредственно с помощью клавиатуры [/] или взять этот знак из арифметической панели, сделав [1Л] на его изображении. Видим, что на экране появился знак деления в виде *горизонтальной черты* ! Так принято в Mathcad в качестве стандарта.

Заметим, что в других случаях знаки арифметических действий (сложение, вычитание, умножение, возведение в степень) также можно вводить любым из способов: или средствами клавиатуры (знак умножения – «звездочка», другие знаки – обычные) или – брать их из арифметической панели инструментов с учётом содержания *всплывающих подсказок*.

Продолжаем набор выражения, значение которого нам требуется вычислить.

Переводим ограничитель на место держателя знакоместа в знаменателе. Сюда необходимо записать функцию косинуса и указать далее значение его аргумента. Символ «cos» находим на арифметической панели, щёлкаем на нем [1Л]. Этот символ появляется в знаменателе создаваемого нами выражения на экране. Остается ввести аргумент $0{,}25\cdot\pi$, что и выполняем уже знакомыми действиями. Кнопка с изображением «π» имеется на арифметической панели.

Для получения результата вычисления вводим знак равенства «=» с клавиатуры или из арифметической панели инструментов. На экране получаем в рамке

$$\frac{\ln(14.7)}{\cos(0.25\cdot\pi)}=3.801$$

Остается убрать рамку действием [1Л] за её пределами или – нажатием на [Enter].

Если при написании формулы были допущены ошибки, то для исправления (редактирования) формулы на ней следует сделать [1Л]. Формула окружится рамкой и, действуя ограничителем, можно внести требуемые исправления. При этом следует руководствоваться сообщениями на экране об ошибках, что осуществляется выделением цветом и соответствующим текстом.

Для вызова контекстной помощи, следует использовать [F1] с выбором нужного из меню.

В заключение этого примера отметим следующие обстоятельства, важные для дальнейшего:

а) символы функций по желанию пользователя не обязательно брать из арифметической панели. Их можно набирать и средствами клавиатуры. Однако, для начинающего это чревато риском ошибок и поэтому не рекомендуется.

б) если в процессе редактирования формулы изменить, по крайней мере, одно из чисел, то произойдет *автоматический пересчёт* значения искомого результата, что повышает оперативность Mathcad в работе.

в) Сформированные на экране формулы и результаты вычисления можно сохранить на диске обычными действиями и вывести на печать (см. меню Файл).

Пример 1.2. Ознакомимся с методикой определения аргумента x для последующего вычисления функции $y = f(x)$.

Дано: $x = 5$

Найти:

$$y = f(x) = \sin(x) / (3 + x^2)$$

Решение

Для определения аргумента применяем оператор присваивания ему определённого значения (:=). Такой оператор имеется в арифметической панели инструментов. Его же можно ввести с клавиатуры, нажав на клавишу с изображением двоеточия [:], верхний регистр. Пишем на экране

$$x := 5$$

Затем изображаем вычисляемую функцию и получаем

sin(*x*) / (3+*x*^2)= – 0.034.

По общему правилу аргумент должен быть определен до того, как компьютер приступит к вычислению функции. Определять аргумент следует левее или выше того места на экране, где написана формула.

Знак равенства здесь обычный. Его также можно ввести либо из арифметической панели, либо непосредственно с клавиатуры. Замечаем, что немедленно после написания знака равенства на экране появляется результат вычислений (в рассматриваемом примере число – 0,034). Для обозначения аргументов и функций в

Mathcad можно использовать любые буквы латинского и греческого алфавитов.

При наборе формул следует знать приём вывода ограничителя из так называемых «липучек», к которым относится знак квадратного корня, знаменатель дроби и некоторые другие. Клавишами управления курсором сделать это не всегда удается, и для продолжения написания формулы необходимо нажимать на клавишу пробела. Чтобы получить результат вычисления, ограничитель должен охватывать всю формулу, после чего можно вводить знак равенства.

После переопределения значения аргумента автоматически следует *пересчёт функции* (всех функций). В этом можно убедиться, придав выше аргументу значение, например

$$x := 3.$$

Следует отметить, что если строку определения аргумента после окончания решения задачи не удалить, то данное нами определение x будет продолжать действовать и во всех последующих решаемых нами задачах.

При использовании тригонометрических функций значения аргументов должны быть выражены в *радианах*.

Пример 1.3

Дано:

$$f(x) = \ln(x) / \sqrt{x+5}$$

Найти:

$f(x)$ при x =10

Решение

Покажем здесь, что вычислить значение функции можно и при другом способе определения её аргументов, а именно так. Пишем на экране:

$$f(x) := \frac{\ln(x)}{\sqrt{x+5}}$$

Выводим ограничитель из-под знака квадратного корня и завершаем, как обычно, изображение формулы нажатием на клавишу [Enter] или путем [1Л] на поле за пределами рамки с формулой, правее или ниже неё. Таким образом мы определили *функцию*. Для получения её значения необходимо и достаточно написать:

$$f(10) =$$

На экране – результат в виде числа 0.595**.**

В ряде расчетов используются переменные *диапазонного* типа, например $x_{min} \le x \le x_{max}$ с заданным шагом Δx варьирования.

Пример 1.4

Дано:

Функция $f(x) = sin(x)/\sqrt{x + 5}$ при диапазоне изменения аргумента x в пределах от $x_{min} = 0$ до $x_{max} = 10$ с шагом $\Delta x = 2$.

Средства Mathcad требуют представления такой переменной записью следующего содержания:

$$x := x_{min}, x_{min} + \Delta x, .., x_{max}$$

Рекомендуется обратить внимание на наличие «горизонтального» двоеточия между значениями $x_{min} + \Delta x$ *и* x_{max}. Здесь ввод последовательно двух точек не срабатывает, и нужно ввести данный символ с клавиатуры, нажав на клавишу [;] или выбрав соответствующий знак в арифметической панели.

Найти:

Значения функции, соответствующие каждому из значений аргумента в заданном диапазоне.

Решение

Определяем на экране функцию действиями, аналогичными использованным в примере 1.3. Переместившись по экрану вниз до достижения на нем свободного мести с помощью полосы вертикальной прокрутки, определяем далее аргумент диапазонного типа посредством записи

$$x := 0, 2 .. 10$$

Выбрав свободное место на поле правее или ниже этой записи, пишем там:

x и нажимаем на [=].

В ответ высвечивается столбец значений аргумента.

Аналогично, изобразив на экране

$f(x)$ и нажав на [=],

получаем столбец соответствующих значений функции.

Как видим, в особых случаях аргумент может быть определен и после определения функции. Этот особый случай характерен тем, что при определении как функции, так и аргумента мы использовали вместо обычного знака равенства символ присваивания.

1.2. Вычисление корней полиномов

Пример 1.5

Дано:

$$C_5x^5 + C_4x^4 + C_3x^3 + C_2x^2 + C_1x + C_0 = 0,$$

где коэффициенты $C_5 = 5$; $C_4 = 4$; $C_3 = 0$; $C_2 = 2$; $C_1 = 1$; $C_0 = -10$.

Найти:

Корни этого полинома r_i, где $i = \overline{1, 5}$.

Решение

Изобразим на экране вектор коэффициентов в виде матрицы, содержащей 1 столбец и $\boldsymbol{n}$ + 1 = 6 строк, где $\boldsymbol{n}$ = 5 – степень полинома. Вообще Mathcad справляется с задачами, где $1 \le \boldsymbol{n} \le 100$.

Записываем:

C:=

и вызываем из панели векторных и матричных операций шаблон матрицы. По запросу компьютера указываем требуемое количество столбцов и строк и делаем [1Л] на кнопке [Вставить].

В места, обозначенные символами «■» вводим значения коэффициентов в направлении сверху вниз, начиная с младшего (C_0) и включая нулевые (здесь C_3).

Результаты нашей работы отобразятся на экране в виде

$$C = \begin{bmatrix} -10 \\ 1 \\ 2 \\ 0 \\ 4 \\ 5 \end{bmatrix}$$

Далее следует написать:

$$\boldsymbol{r} := \boldsymbol{polyroots}(C)$$

где ***polyroots*** – имя функции вычисления корней полиномов.

Для вызова решения пишем:

$$\boldsymbol{r} =$$

На экране высветится вектор искомых корней, как вещественных, так и комплексных.

1.3. Решение трансцендентных уравнений

Пример 1.6.

Дано:

$$y = x^2 - \sin(x)$$

Найти:

корень этого уравнения.

Решение

Задача решается методом итераций. Примем в качестве начального приближения, например $x := 1$. Затем вызовем функцию вычисления корней

$$root\left(x^2 - \sin(x), x\right)$$

Здесь запятая отделяет записываемый справа от неё аргумент x от стоящего от неё слева выражения функции.

Для приведения системы Mathcad в действие справа от последней закрывающей скобки поместим знак равенства, после чего получим на экране значение корня в виде числа 0.877.

1.4. Решение систем алгебраических уравнений

Mathcad позволяет решать системы алгебраических уравнений, по крайней мере, одним из следующих возможных способов.

1.4.1. Общий случай

Рассматриваемый способ позволяет решать системы любых уравнений, в том числе нелинейных и трансцендентных. Поясним особенности использования этого способа на следующем примере.

Пример 1.7

Дано:

система из двух уравнений [2] с двумя неизвестными x и $\boldsymbol{y}$

$$\begin{cases} x^3 + \sin(\boldsymbol{y}) = 25 \\ \boldsymbol{y}^2 - \boldsymbol{cos}(x) = 27 \end{cases}$$

Найти:

значения корней этой системы.

Решение

Поскольку Mathcad решает подобные задачи методом итераций, необходимо задаться начальными приближениями $\boldsymbol{x}$ и $\boldsymbol{y}$. Зададим их в виде, например:

$$\boldsymbol{x := 1; \quad y := 1}$$

Затем вводим ключевое слово ***Given*** (Дано), после чего изображаем уравнения системы по схеме:

Given

$$x^3 + \sin(\boldsymbol{y}) \equiv 25$$
$$\boldsymbol{y}^2 - \cos(x) \equiv 27$$

Примечание: здесь «**=**» – особый («жирный») знак равенства, который вводится аккордом [Ctrl] и [=]. Обычный знак равенства или знак присваивания значений здесь не пригодны. Записывать показатель степени можно из арифметической панели или непосредственно с помощью клавиатурного символа [∧], соответствующего операции возведения в степень.

Далее вводим следующее ключевое слово ***Find*** (найти) записью

$$\textbf{\textit{Find (x, y) =}}$$

В ответ на ввод знака равенства на экране высвечивается вектор искомых корней

$$\begin{bmatrix} 2.96 \\ 5.101 \end{bmatrix}$$

Пример 1.8

Температуру расплавленной стали под слоем шлака (серое тело) измерили двумя пирометрами излучения (табл. 1.2).

Таблица 1.2. **Результаты измерения температуры**

Пирометр	Обозначение псевдотемператур, К	Показания приборов, ˚С
1. Яркостный	$\boldsymbol{T}_b$	1525
2. Радиационный	$\boldsymbol{T}_r$	1400

Требуется определить истинную температуру стали и степень черноты объекта измерения.

Решение

Известно, что яркостная $\boldsymbol{T}_b$ и радиационная $\boldsymbol{T}_r$ псевдотемпературы зависят как от истинной температуры $\boldsymbol{T}$, так и от степени черноты измеряемого объекта [11, 12, 13]. При этом имеют место зависимости:

$$\frac{1}{T_b} - \frac{1}{T} = \frac{\lambda}{C_2} \ln\left(\frac{1}{\varepsilon_\lambda}\right); \qquad (1.1)$$

$$T_r = T \sqrt[4]{\varepsilon}\ , \qquad (1.2)$$

где ε_λ – спектральная степень черноты при данной длине волны $\boldsymbol{\lambda}$, м;

$\boldsymbol{\varepsilon}$ – общая (интегральная) степень черноты;

$C_2 = 1{,}43 \cdot 10^{-2}$ м · К.

Для серых тел при $\boldsymbol{\varepsilon_\lambda = \varepsilon} < 1$ существует соотношение

$$T_r < T_b < T\ .$$

Обычно яркостные пирометры снабжаются красным светофильтром, пропускающим излучение с эффективной дли-ной волны $\boldsymbol{\lambda}$ = 650 нм = 650 · 10^{-9} м, а измеренные значения абсолютных псевдотемператур в примере составили $\boldsymbol{T_b}$ = 1525 ºС = 1525 + 273 = 1798 К ; $\boldsymbol{T_r}$ = 1400 ºС = 1400 + 273 ºС = 1673 К.

Для решения примера загружаем Mathcad и формулируем условия задачи на экране согласно данным рис.1.1, где принято обозначение $\varepsilon = \varepsilon_\lambda$.

Mathcad решает подобные задачи методом итераций. Поэтому в первой строке задаёмся произвольными начальными приближениями искомых переменных $\boldsymbol{T}$ и ε, а также присваиваем значения величинам $\boldsymbol{T_b}$ и $\boldsymbol{T_r}$.

Ведущей идеей решения поставленной задачи здесь является представление равенств (1.1) и (1.2) в виде системы двух уравнений с двумя неизвестными: $\boldsymbol{T}$ и ε. Эти уравнения преобразуем путём переноса членов с неизвестными в левую часть, а известные величины – в правую часть.

$T := 1000 \quad \varepsilon := 1 \qquad Tb := 1798 \qquad Tr := 1673$

Given

$$\frac{1}{T} + 650 \cdot \frac{10^{-9}}{1.43 \cdot 10^{-2}} \cdot \ln\left(\frac{1}{\varepsilon}\right) = \frac{1}{Tb}$$

$$T \cdot \varepsilon^{0.25} = Tr$$

$$\mathbf{Find}(T, \varepsilon) = \begin{pmatrix} 1.864 \times 10^{3} \\ 0.649 \end{pmatrix}$$

Рис.1.1. К решению примера 1.8

Используем ключевые слова ***Given*** и ***Find***, как это было показано в предыдущем примере. На экране получаем результат решения:

T = 1864 К = 1864 – 273 = 1591 ºС ; $\varepsilon = \varepsilon_{\lambda}$ = 0,649.

1.4.2. Матричный способ решения систем алгебраических уравнений

Пример 1.9

Дано:

$$\left.\begin{aligned} 3x + 2y &= 137 \\ x + 5y &= 180 \end{aligned}\right\}.$$

Найти:

корни данной системы.

Решение

Для того чтобы порядковый номер элементов массивов начинался с единицы, а не с нуля, любым шрифтом пишем на экране:

$$\boldsymbol{origin := 1}$$

Вводим матрицу коэффициентов левой части уравнений системы, указывая число строк (rows) 2 и столбцов (columns) 2:

$$\boldsymbol{A} := \begin{bmatrix} 3 & 2 \\ 1 & 5 \end{bmatrix}$$

Задаем матрицу правых частей:

$$\boldsymbol{B} := \begin{bmatrix} 137 \\ 180 \end{bmatrix}$$

Затем пишем известную из математики формулу решения:

$$\boldsymbol{X := A^{-1} \cdot B}$$

Вектор корней $\begin{pmatrix} 25 \\ 31 \end{pmatrix}$ высветится после ввода $\boldsymbol{X} =$

Примечание: Векторы и матрицы для наглядности следует изображать заглавными буквами, однако замена их строчными буквами на процесс решения не влияет.

1.5. Вычисление производных

Пример 1.10

Дано:

$$f(z) = 5z + 2z^2 + \sin^2(z).$$

Найти:

$f'(z)$ и $f''(z)$ в точке $z = 3$.

Решение

Определяем на экране дифференцируемую функцию

$$f(z) := 5 \cdot z + 2 \cdot z^2 + \sin(z)^2$$

и точку вычисления первой и второй производных

$$z := 3$$

Вызываем символ первой производной $\left[\frac{d}{dx}\right]$ из панели математического анализа.

На экране появляется этот символ

$$\frac{\mathrm{d}}{\mathrm{d}\blacksquare}\blacksquare$$

в сопровождении держателей знакомест (маркеров). На месте маркера напротив горизонтальной черты пишем $f(z)$, а маркер в знаменателе заполняем символом z. Переводим ограничитель в положение правее записи $f(z)$ и вводим [=]. Появляется значение первой производной (16.721).

Заметим, что для перехода от одного маркера к другому можно воспользоваться клавишей [Tab].

Для вычисления второй производной из той же панели инструментов вызываем символ производной старшего порядка

$$\frac{\mathrm{d}^{\blacksquare}}{\mathrm{d}\blacksquare^{\blacksquare}}\blacksquare$$

Заполняем места, указываемые маркерами, нашими данными, которые соответствуют записи второй (а вообще – произольногопорядка) производной в виде

$$\left[\frac{d^2}{dz^2} f(z)\right].$$

После ввода знака равенства правее $f(z)$ на экране получаем искомый результат (5.92).

Примечание: Символ порядка производной в числителе генерируется автоматически после указания его на месте маркера в знаменателе символа производной.

1.6. Вычисление конечных сумм и интегралов

Пример 1.11

Дано:

$$\sum_{n=n_{\min}}^{n=n_{\max}} \frac{1}{n!}$$

Найти:

значение этой суммы при $n_{\min} = 0$ и $n_{\max} = 25$.

Решение

Из математической панели инструментов вызываем панель математического анализа и из последней берем символ

$$\sum_{\blacksquare=\blacksquare}^{\blacksquare} \blacksquare.$$

На места маркеров вводим выражение функции − факториала $\frac{1}{n!}$, а также нижний $n_{min} = 0$ и верхний $n_{max} = 25$ пределы данной суммы.

Остается, выведя ограничитель за правую границу формулы, написать знак равенства [=], чтобы получить искомый результат (2.718).

Аналогично вычисляются и интегралы.

Пример 1.12

Дано:

$$\int_a^b \frac{1}{1+y^2}\,dy.$$

Найти:

значение этого интеграла при $a = 0$ и $b = 25$.

Решение

Из этой же панели математического анализа вызываем символ определенного интеграла

$$\int_{\blacksquare}^{\blacksquare} \blacksquare\, d\blacksquare$$

Заполняем данными примера места расположения маркеров. Перемещаем ограничитель за пределы формулы и вводим с клавиатуры или из арифметической панели знак равенства [=]. На экране – результат вычисления интеграла (1.531).

Сходным образом можно вычислить *кратные* интегралы.

Пример 1.13

Дано:

$$I = \int_a^b \int_c^d \int_e^f (x\,y\,z)\,dx\,dy\,dz.$$

Найти:

Значение интеграла при $a = 0$; $b = 2$; $c = 3$; $d = 6$; $e = 1$; $f = 5$.

Решение

Формулируем на экране задачу:

$$\boldsymbol{I} :=$$

после чего вызываем символ определенного интеграла действиями, аналогичными использованным в предыдущем примере. На место маркера в позицию подынтегрального выражения первого интеграла вводим второй интеграл, а вместо подынтегральной его функции вводим третий интеграл. На экране видим:

$$I := \int_{\blacksquare}^{\blacksquare} \int_{\blacksquare}^{\blacksquare} \int_{\blacksquare}^{\blacksquare} \blacksquare \, d\blacksquare \, d\blacksquare \, d\blacksquare$$

В позицию, непосредственно следующую за символом третьего интеграла, вводим выражение подынтегральной функции $(x \cdot \boldsymbol{y} \cdot z)$. На местах следующих трёх маркеров располагаем символы наших переменных ***x, y, z***. В итоге задача на экране представляется в виде

$$\boldsymbol{I} := \int_{0}^{2} \int_{3}^{6} \int_{1}^{5} \boldsymbol{(x \cdot y \cdot z)\, dx\, dy\, dz}$$

Искомый результат вызываем записью:

$$\boldsymbol{I} =$$

При отсутствии ошибок пользователя, этот результат составляет ***I*** = 324.

1.7. Решение дифференциальных уравнений и их систем

Рассмотрим примеры применения Mathcad для решения обыкновенных дифференциальных уравнений, последовательно усложняя их. При этом будем иметь в виду, что примечание, приведенное выше на с. 21, остаётся в силе и при действиях с дифференциальными уравнениями.

Пример 1.14

Дано:

дифференциальное уравнение $T\frac{d\boldsymbol{y}(t)}{dt}+\boldsymbol{y}(t)=Kx$

при $\boldsymbol{T}$ = 3, $\boldsymbol{K}$ = 1 и $\boldsymbol{x}$ = 1.

Найти:

решение уравнения в интервале значений аргумента $0 \le \boldsymbol{t} \le 10$.

Решение

Для поиска решения может быть использована встроенная во все версии Mathcad функция ***rkfixed*** по методу Рунге---Кутта с фиксированным шагом интегрирования (рис.1.2).

После переноса $\boldsymbol{y}$ в правую часть и сокращения на 3 условная запись первой производной этой функции по аргументу $\boldsymbol{t}$ приобретает вид:

$$f(t,\boldsymbol{y}) := (1-\boldsymbol{y})/3$$

Принимаем начальные условия, например нулевые:

$$\boldsymbol{t}0:=0;\quad \boldsymbol{y}0:=0$$

и параметры решения:

$\boldsymbol{t}1$:=10 – конечная точка интервала аргумента,

$\boldsymbol{n}$:=100 – число решений (расчетных точек) на этом интервале.

$$f(t, y) := \frac{(1 - y)}{3}$$

$t0 := 0 \qquad y0 := 0$

$t1 := 10 \qquad n := 100$

$ic := y0 \qquad D(t, Y) := f(t, Y)$

$S := rkfixed(ic, t0, t1, n, D)$

$t := S^{\langle 0 \rangle} \qquad y := S^{\langle 1 \rangle}$

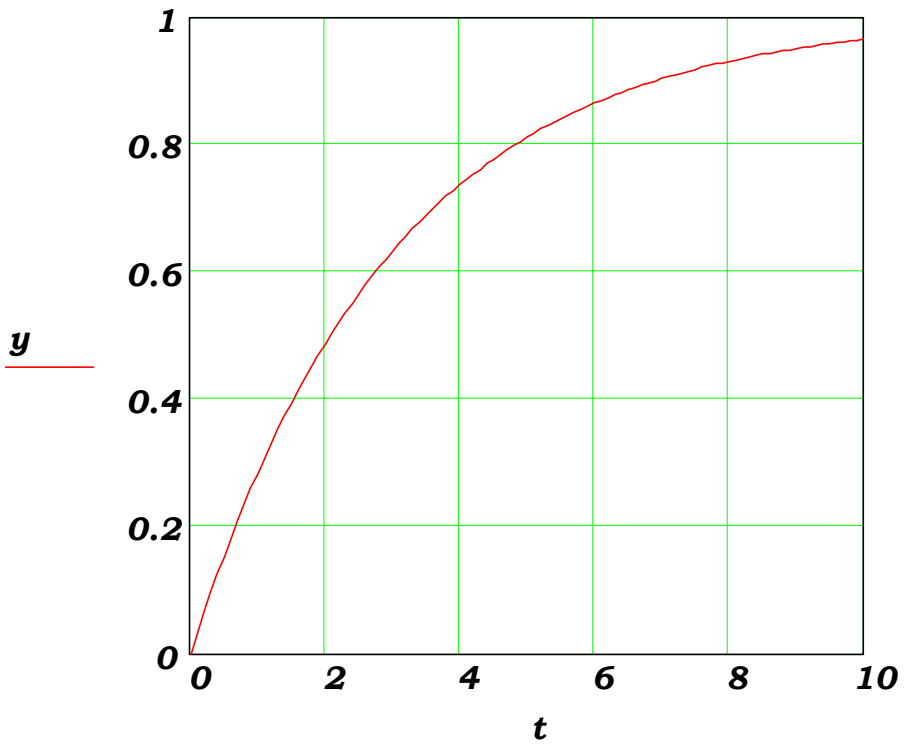

Рис.1.2. Решение дифференциального уравнения 1- го порядка с помощью функции ***rkfixed***

Следует также определить конечную точку интервала функции ***y***1.

Это можно сделать по оценке исходного дифференциального уравнения. Так, заметим, что в нашем примере после окончания переходного процесса изменения ***y*** от его начального ***y***0 до конечного ***y***1 = const состояний имеем ***dy/dt*** = 0. Отсюда ***f(t,y)*** = 0 и ***y***1 := 1.

Определяем *вектор* начальных условий:

$$ic := y0$$

и функцию производной:

$$D(t,Y) := f(t,Y)$$

Здесь второй аргумент ***Y*** является *вектором* вычисляемых значений функции ***y(t).***

Формируем матрицу решений

$$S:=rkfixed(ic,\ t0,\ t1,\ n,\ D).$$

Определяем её столбцы:

первый (фактически нулевой)

$$t:=S^{<0>}$$

и второй

$$y:=S^{<1>}$$

Если нас интересуют численные значения аргумента ***t*** и функции ***y*** в отдельных точках решения, то после последнего выражения можно ввести

$$S =$$

Это вызовет появление на экране начальной области матрицы, содержащей в первом столбце значения ***t***, а во втором – ***y***. Её можно расширить путем действия [1Л], после чего растянуть вниз по экрану с помощью мыши.

Для получения *графика* ***y = f(t)*** щёлкаем [1Л] на кнопке [Инструменты графики] Математической панели, а затем − на кнопке [Декартов график] Панели графики. После этого на экране появляется шаблон автоматически масштабируемого графика исследуемой функции с маркерами для обозначения величин ***t*** и ***y*** по центру соответственно абсциссы и ординаты, а также символов начального и конечного значений ***t0, t1*** аргумента и ***y***0, ***y***1 функции (рис.1.2). Остаётся щёлкнуть [1Л] вне рамки шаблона графика, и на экране высветится законченный график с автоматической заменой введенных нами символов числами, соответствующими условиям решаемой задачи.

Для успеха в решении задачи верхняя грань рамки графика должна располагаться правее или ниже выражения, определяющего значение ***y***. Если кривая графика не обозначена на экране, рамку следует вызвать действием [1Л] на поле графика и переместить рамку в нужное положение при удерживаемой в нажатом состоянии левой клавише мыши. При перетаскивании рамки курсор, остановленный на ней, принимает вид кисти руки человека.

Аналогичным способом можно перетаскивать рамки с формулами и надписями на рабочем листе.

Достоинством Mathcad является автоматизация пересчёта данных решаемых задач. Так, после изменения любого численно определенного параметра (коэффициента уравнения или значения аргумента) следует пересчёт значений функций и автоматическое перестроение графика. В этом читатель может убедиться сам в процессе выполнения своих упражнений по теме настоящего примера.

Для редактирования графика следует сделать на его поле [2Л] или [1П]. В ответ на это высветится диалоговое окно с предлагаемыми пользователю командами. Можно, например, ввести вспомогательные линии по осям координат и выполнить другие операции.

Решение дифференциального уравнения порядка ***n,*** более высокого, чем первый, может быть выполнено одним из, по крайней мере, двух возможных способов. Применение первого способа требует представления дифференциального уравнения ***n*** − го порядка в виде системы ***n*** дифференциальных уравне-

ний, каждое из которых имеет первый порядок.

Пример 1.15

Дано:

$$a_2 \frac{d^2 y}{dt^2} + a_1 \frac{dy}{dt} + a_0 y = b_0 x \qquad (1.3)$$

Найти:

решение этого уравнения.

Решение

В первом варианте методика поиска решения дифференциального уравнения сводится к следующему.

Обозначим

$$y_0 = y;$$
$$y_0' = \frac{dy}{dt} = \frac{dy_0}{dt} = y_1;$$
$$y_1' = \frac{d^2 y}{dt^2}.$$

Тогда вместо уравнения (1.3) получаем систему уравнений

$$y_0' = y_1$$
$$y_1' = \frac{b_0}{a_2} x - \frac{a_0}{a_2} y_0 - \frac{a_1}{a_2} y_1.$$

Пусть $a_2 = 2$; $a_1 = 5$; $a_0 = 1$; $b_0 = 1$; $x = 1$ (в общем случае правая часть может быть функцией t).

Определим функцию, характеризующую *вектор* значений производных в каждой точке решения (***t, Y***) (рис.1.2):

$$D(t,Y):=\begin{bmatrix} Y_1 \\ \frac{1}{2}-\frac{1}{2}Y_0-\frac{5}{2}Y_1 \end{bmatrix}$$

Зададим дополнительно:

t0:=0 – начальное значение аргумента;
t1:=10 – его конечное значение;
Y0:=$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ – вектор начальных (например нулевых) значений функций;
N = 100 – число решений на интервале [***t0, t1***].

Для решения можно использовать уже знакомую нам из предыдущего примера функцию ***rkfixed***, но она не единственная в арсенале встроенных функций Mathcad. Испытаем из них фунцию ***Rkadapt*** – также по методу Рунге - Кутта, но с адаптируемым шагом интегрирования, для чего напишем:

S:= Rkadapt(***Y***0, ***t***0, ***t***1, ***N, D***)
t:=S$^{<0>}$
y0:=***S***$^{<1>}$
y1:=***S***$^{<2>}$

После этого можно ввести:

S =

и на экране высветится матрица значений ***t***, ***y***0, ***y***1 (всего по умолчанию 15 строк). Затем строим график (рис. 1.3) действиями, знакомыми из предыдущего примера.

Средства редактирования полученного графика аналогичны упомянутым в примере 1.14.

$$D(t, Y) := \begin{pmatrix} Y_1 \\ \frac{1}{2} - \frac{1}{2} Y_0 - \frac{5}{2} Y_1 \end{pmatrix}$$

$t0 := 0 \qquad t1 := 10$

$$Y0 := \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad N := 100$$

$S := Rkadapt(Y0, t0, t1, N, D)$

$t := S^{\langle 0 \rangle} \qquad y0 := S^{\langle 1 \rangle} \qquad y1 := S^{\langle 2 \rangle}$

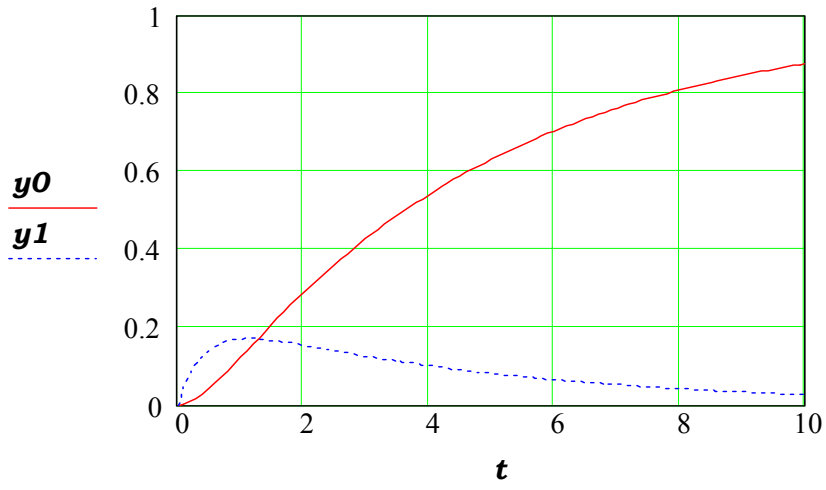

Рис.1.3. Решение системы дифференциальных уравнений с помощью функции *Rkadapt*

Рассмотренная методика позволяет решать и более сложные системы из n уравнений первого порядка.

В составе Mathcad имеется также функция Odesolve, позволяющая решать дифференциальные уравнения n - го порядка без их преобразования в систему n уравнений первого порядка.

В этом заключается второй вариант решения подобных уравнений.

Пример 1.16

Дано:

$$25\frac{d^2x(t)}{dt^2}+2\frac{dx(t)}{dt}+x(t)=1.$$

Найти:

решение этого уравнения при нулевых начальных условиях в интервале $0 \le t \le 150$.

Решение

Записываем исходное уравнение по правилам Mathcad, как показано на рис.1.4, и начальные условия $x(0) = 0$ $x'(0) = 0$.

При наборе условий задачи на экране используем «жирные» знаки равенства, вводимые клавишным аккордом [Ctrl][=], а штрихи при обозначении производных ‘, “ и т. д. − клавишным аккордом [Ctrl][F7].

Вызываем функцию решения с её параметрами

x:=Odesolve (t, 150).

Теми же средствами, что и в предыдущем примере, получаем график решения ***x(t)*** − см. рис. 1.4.

В заключение настоящего раздела отметим, что многие другие функции решения дифференциальных уравнений могут быть рассмотрены в QuckSheets (Быстрых страницах), вызываемых кнопкой Ресурс Центр (Resource Center) из Главного меню Mathcad. Этот же инструментарий пригоден для ознакомления и с другими возможностями Mathcad.

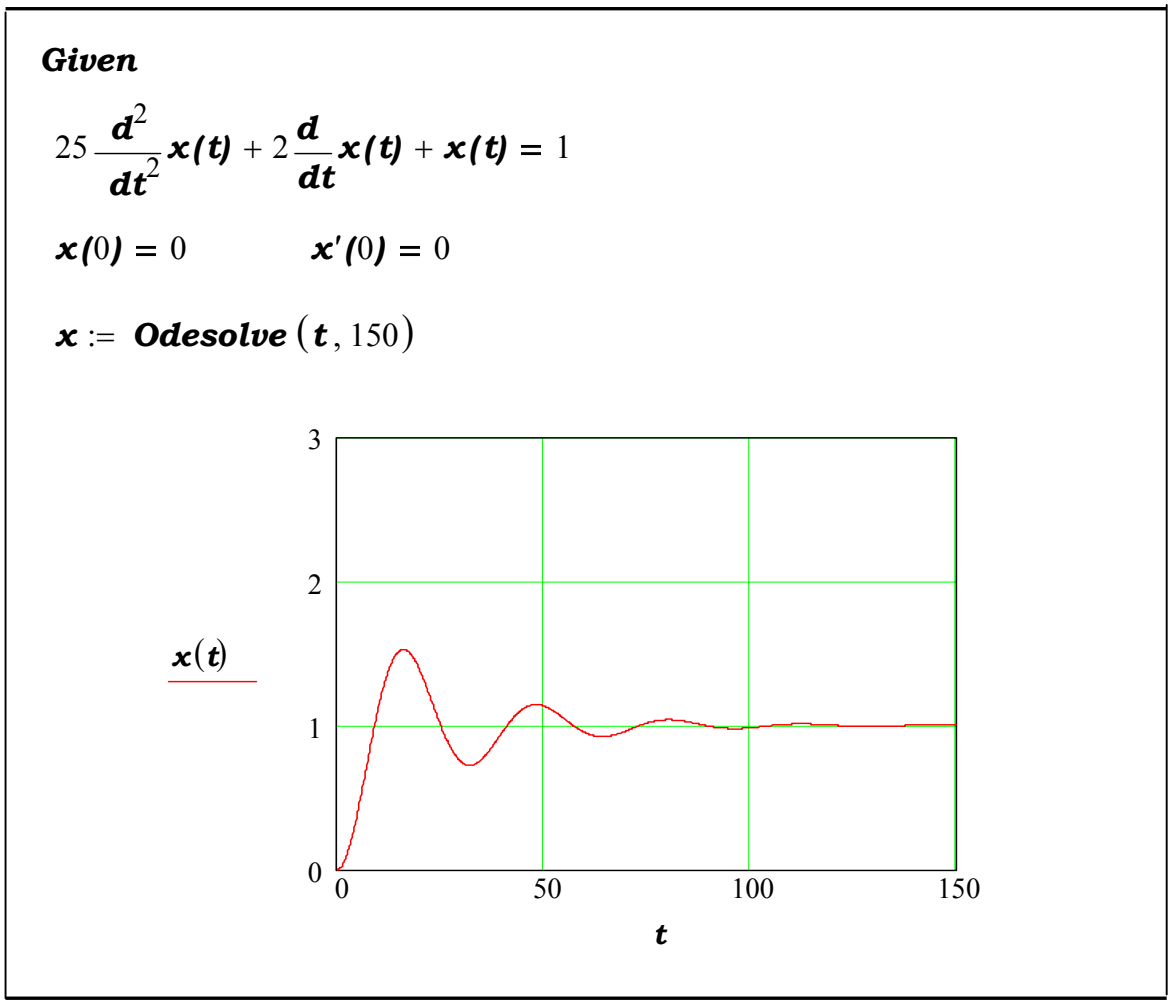

Рис.1.4. Решение дифференциального уравнения с помощью функции *Odesolve*

1.8. Элементы искусственного интеллекта в составе Mathcad

Начнем с того, что название этого раздела носит несколько утрированный характер (по-английски Smart Math – «ловкая» математика). Тем не менее разработчики Mathcad включили в его состав ряд скрытых от пользователя программ, которые автоматически включаются в ответ на введенную пользователем команду и осуществляют вычислительный процесс некоторым *оптимальным* способом в обход наиболее очевидного, тем самым в одних случаях повышая точность вычислений, а

в других случаях – сокращая затраты времени на переработку информации.

Пример 1.17

Дано:

формула для вычисления произвольного (n - го) члена геометрической прогрессии

$$a_n = a_1 q^{n-1},$$

где: a_1 – первый член;

q – знаменатель прогрессии.

Найти:

сумму n членов этой прогрессии при $n = 10$; $a_1 = 5$; $q = 2$.

Решение

Действуя на примитивном уровне, неискушенный пользователь мог бы вычислить отдельно каждый член прогрессии, т.е. a_1, a_2, $a_3,\ldots,a_{10}$, а затем их суммировать. Этот путь долог, так как существует формула для вычисления суммы n членов геометрической прогрессии

$$S_n = \frac{a_n q - a_1}{q - 1}.$$

Данную формулу Mathcad автоматически отыскивает в своем «арсенале», подключает её и по ней определяет искомое решение.

Наши действия на рабочем листе Mathcad конкретно сводятся к следующему (рис.1.5, с. 36).

Определяем данные:

a_1:=5 q:=2.

Вводим формулу

$$S := \sum_{n=1}^{10} (a_1 q^{n-1})$$

и не убираем рамку вокруг неё.

Далее из главного меню командуем:

Математика (Math) | Оптимизация (Optimization)

Замечаем, что в рамке правее формулы появляется символ оптимизации в виде звездочки * красного цвета.

Наконец, записываем

$$S =$$

и получаем результат (число 5115) − рис.1.5, на котором наши действия записаны по правилам Mathcad.

$$a_1 := 5 \qquad q := 2$$

$$S := \sum_{n=1}^{10} \left(a_1 \cdot q^{n-1}\right) \ *$$

$$S = 5.115 \times 10^3$$

Рис.1.5. К оптимизации вычислений

1.9. Графика Mathcad

Методику построения декартовых, т.е. двумерных, графиков мы уже рассмотрели на примерах решения дифференциальных уравнений. Для закрепления этих понятий выполним ещё несколько упражнений.

Пример 1.18

Дано:

$$Y(t) = e^{-0.3t} \cdot \cos(\omega t).$$

Найти:

результаты вычисления этого выражения при $0 \le t \le 5$ с шагом 0,1 и для случая $\omega = 5$ − график функции ***Y(t).***

Решение

Подготовительная процедура кроме задания вида функции ***Y(t)*** в обозначениях Mathcad требует также формирования вектора значений аргумента ***t*** диапазонного типа и вектора значений функции ***Y(t)*** в том же интервале ***t.***

Последовательно определяем компоненты решения (рис. 1.6).

t:= 0, 0.1 .. 5 ω:= 5

$$Y(t) := \exp(-0.3 \cdot t) \cdot \cos(\omega \cdot t).$$

Вызываем векторы ***t*** и ***Y(t)*** действиями

t= и ***Y(t)=***

В ответ на экране высвечиваются столбцы значений ***t*** и ***Y(t).***

Остается установить курсор правее или ниже места расположения этих столбцов и, пользуясь кнопками Математической панели, а затем – панели Инструментов графики (см. всплывающие подсказки), вызвать «заготовку» графика, разметить его оси в соответствии с условиями задачи и после ввода последнего числа получить искомый график.

В автоматизации пересчёта данных и перестроения графика читатель может убедиться, самостоятельно варьируя, например значения ω.

$t := 0, 0.1 .. 5 \quad \omega := 5$

$Y(t) := \exp(-0.3 \cdot t) \cdot \cos(\omega \cdot t)$

t =	$Y(t)$ =
0	1
0.1	0.852
0.2	0.509
0.3	0.065
0.4	-0.369
0.5	-0.69
0.6	-0.827
0.7	-0.759
0.8	-0.514
0.9	-0.161
1	0.21
1.1	0.509
1.2	0.67
1.3	0.661
1.4	0.495
1.5	0.221

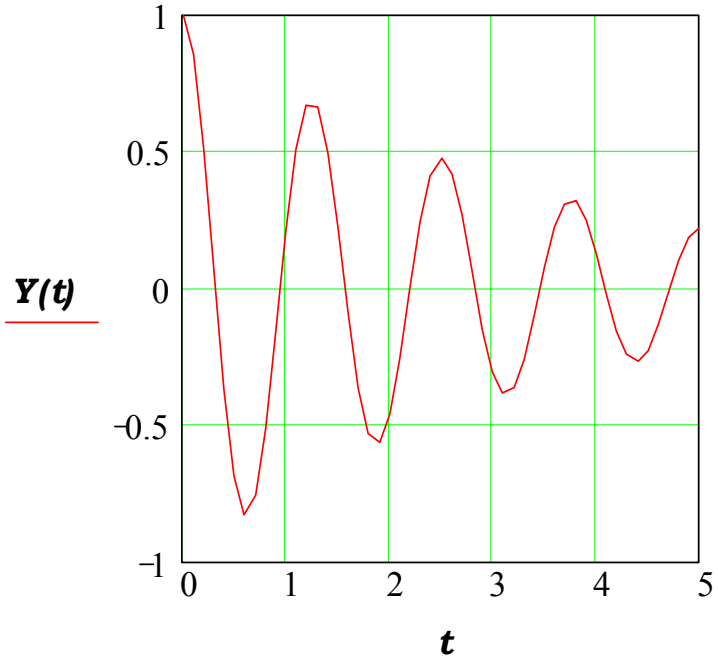

Рис.1.6. К примеру построения графика функции.

Mathcad располагает также возможностями построения графиков в полярных координатах, карт линий уровня, а также – *трёхмерных* графиков и столбиковых гистограмм (см. Quick Sheets).

Пример 1.19

Наиболее точным из числа бесконтактных методов измерения высоких температур является *цветовой метод*. Его сущность заключается в том, что темература, называемая цветовой, или температурой спектрального отношения, T_c, К определяется как функция отношения интенсивностей излучения $I_{\lambda 1}$ и $I_{\lambda 2}$ контролируемого объекта при двух различных длинах волн λ_1 и λ_2

$$T_c = f_0 (I_{\lambda 1} / I_{\lambda 2}).$$

И, обратно, отношение интенсивностей излучения является функцией температуры

$$I_{\lambda 1} / I_{\lambda 2} = f(T).$$

При этом каждая из интенсивностей излучения на основании закона М. Планка составляет

$$I_{\lambda i} = \frac{\varepsilon_{\lambda i} C_1 \lambda^{-5}}{e^{\frac{C_2}{\lambda_i T}} - 1}, \qquad (1.4)$$

где $i = \overline{1, 2}$;
$C_1 = 0{,}374 \cdot 10^{-15}$ Вт · м^2 ;
$C_2 = 1{,}43 \cdot 10^{-2}$ м · К ;
$\varepsilon_{\lambda i}$ − спектральная степень черноты (у абсолютно чёрного тела $\varepsilon_{\lambda i} = 1$; у серых тел $\varepsilon_{\lambda i} < 1$ и $\varepsilon_{\lambda i} \neq f(\lambda)$);
T − абсолютная температура тела, К.

В соответствии с изложенным рассматриваемое отношение спектральных интенсивностей излучения как мера цветовой температуры выражается формулой

$$f(T) = \frac{I_{\lambda 1}}{I_{\lambda 2}} = \frac{\lambda_1^{-5} (e^{\frac{C_2}{\lambda_2 T}} - 1)}{\lambda_2^{-5} (e^{\frac{C_2}{\lambda_1 T}} - 1)} \quad . \qquad (1.5)$$

Полагаем, что цветовой пирометр снабжен светофильтрами, пропускающими излучения с эффективными длинами волн $\lambda_1 = 550$ нм и $\lambda_2 = 650$ нм.

Требуется рассчитать и построить график зависимости значения $f(T)$ от температуры поверхности ванны сталеплавильной печи, покрытой шлаком (серое тело), в диапазоне 1400 … 1650 °С = 1673 … 1923 К.

Решение

На экране Mathcad формулируем условия задачи (рис.1.7).

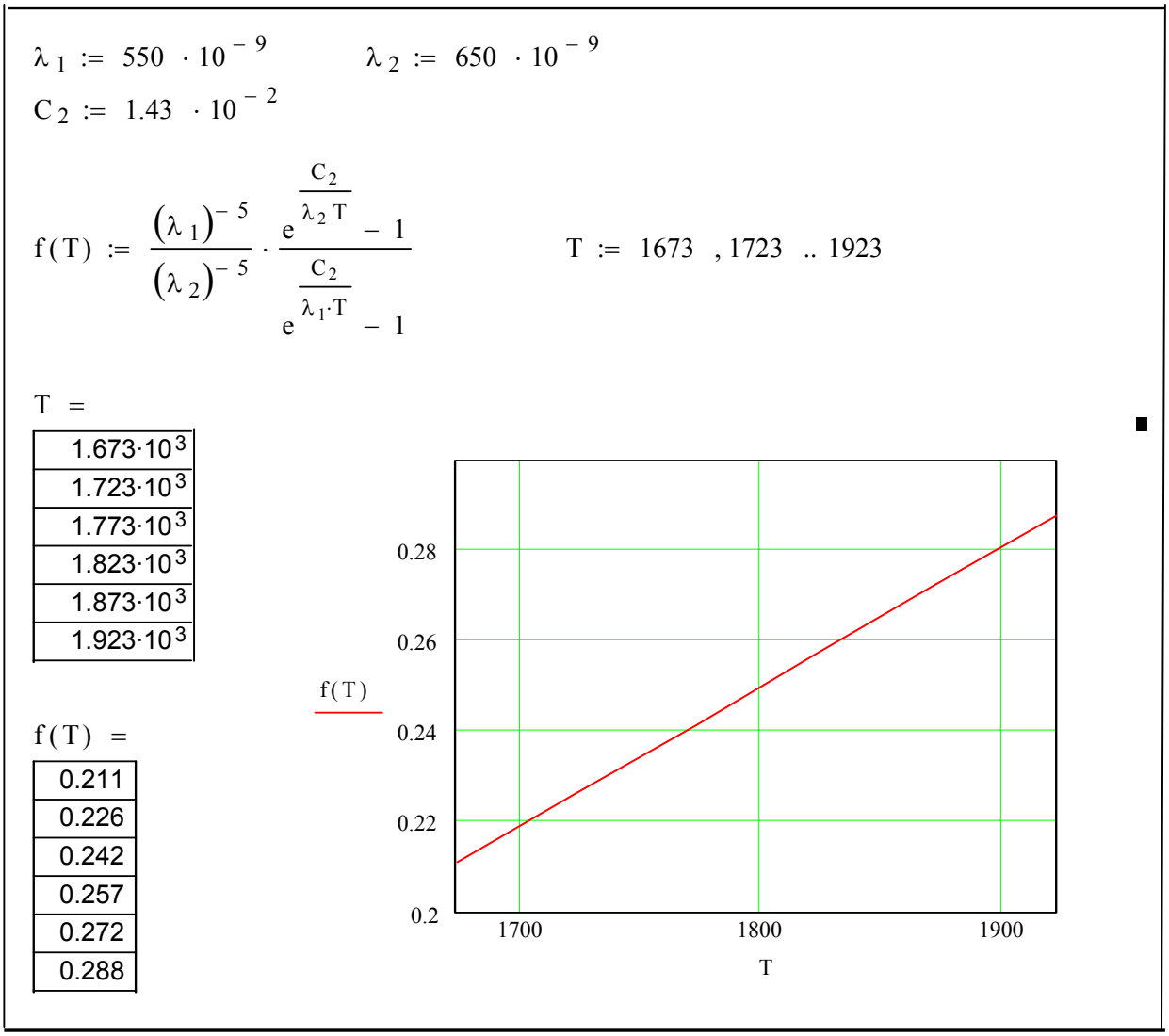

Рис.1.7. К примеру вычисления и построения графика спектральной характеристики цветового пирометра при измерении температуры *T*, К жидкой стали

Определяем исходные данные в составе λ_1, λ_2, C_2. Присваиваем конкретное выражение исследуемой функции ***f(T),*** исходя из уравнения (1.5). После этого определяем аргумент этой функции *T,* относящийся к переменным *диапазонного типа* и изменяющийся в пределах от 1673 до 1923 К с шагом 50 К.

Введя ***T*** **=** , получаем столбец значений температур, а после ввода ***f (T*****)** **=** видим на экране столбец значений исследуемого спектрального отношения.

Остаётся организовать вывод графического отображения результатов нашего вычислительного эксперимента. Для этого делаем [1Л] на кнопке [Инструменты графики], а затем − [1Л] на кнопке [Декартов график]. На экране высвечивается шаблон для построения графика в виде заключённых в рамку координатных осей и расположенных по центру каждой из них маркеров. В места, занимаемые этими маркерами, вводим символы аргумента *T* (абсцисса) и функции ***f(T)*** (ордината). После такого ввода появляются маркеры для записи пределов изменения наших переменных. Вслед за этим, нажав на [Enter] или сделав [1Л] за пределами поля графика, вызываем его кривую. Действие [2Л] или [1П] на поле графика сопровождается появлением на экране диалогового окна с перечнем команд редактирования полученного графика. Из них пользователю предоставляется возможность выбрать те, с помощью которых можно изменить масштаб изображения, ввести вспомогательные линии сетки, обозначить соответствующие метки и пр. Проделанную работу рекомендуется сохранить, выбрав для неё то или иное собственное имя. Расширение имени в виде mcd присваивается компьютером автоматически.

Пример 1.20

Дано:

функция двух аргументов [2], с.45:

$$Z(b,\alpha)=\frac{G(b+c)\sin(\alpha)\cos(\alpha)}{c}.$$

Найти:

график этой функции.

Решение

На этом примере ознакомимся с возможностями использования размерностей, единицы которых в СИ обозначаются так:

newton – Ньютон,
m – метр,
deg – угловые градусы.

Эти единицы после записи каждой из размерных величин указывают через знак умножения.

О других единицах см. в Главном меню по команде

Вставка | Единицы измерения

Читатель может испытать и безразмерные величины, но тогда аргументы всех тригонометрических функций должны быть выражены не в градусах (degrees), а в радианах.

Шаг 1-й: определяем аргументы c:=1·m; G:=20·newton. Записываем функцию в обозначениях Mathcad:

$$Z(b,\alpha) := \frac{G\cdot(b+c)\cdot\sin(\alpha)\cdot\cos(\alpha)}{c}$$

Шаг 2-й: нумеруем узлы сетки плоскости xOy с шагом (по умолчанию), равным единице:

$$i := 0 .. 40$$

Шаг 3-й: формируем вектор первого аргумента:

$$b_i = -5\cdot m + 0.25\cdot m\cdot i$$

Шаг 4-й: нумеруем узлы сетки той же плоскости:

$$j := 1 .. 40$$

Шаг 5-й: формируем вектор значений второго аргумента:

$$\alpha_j := 9 \cdot \mathrm{deg} \cdot j$$

Шаг 6-й: заполняем матрицу *M* значениями ***Z(x,y)***. Для этого записываем:

$$M_{i,j} = Z(b_i, \alpha_j).$$

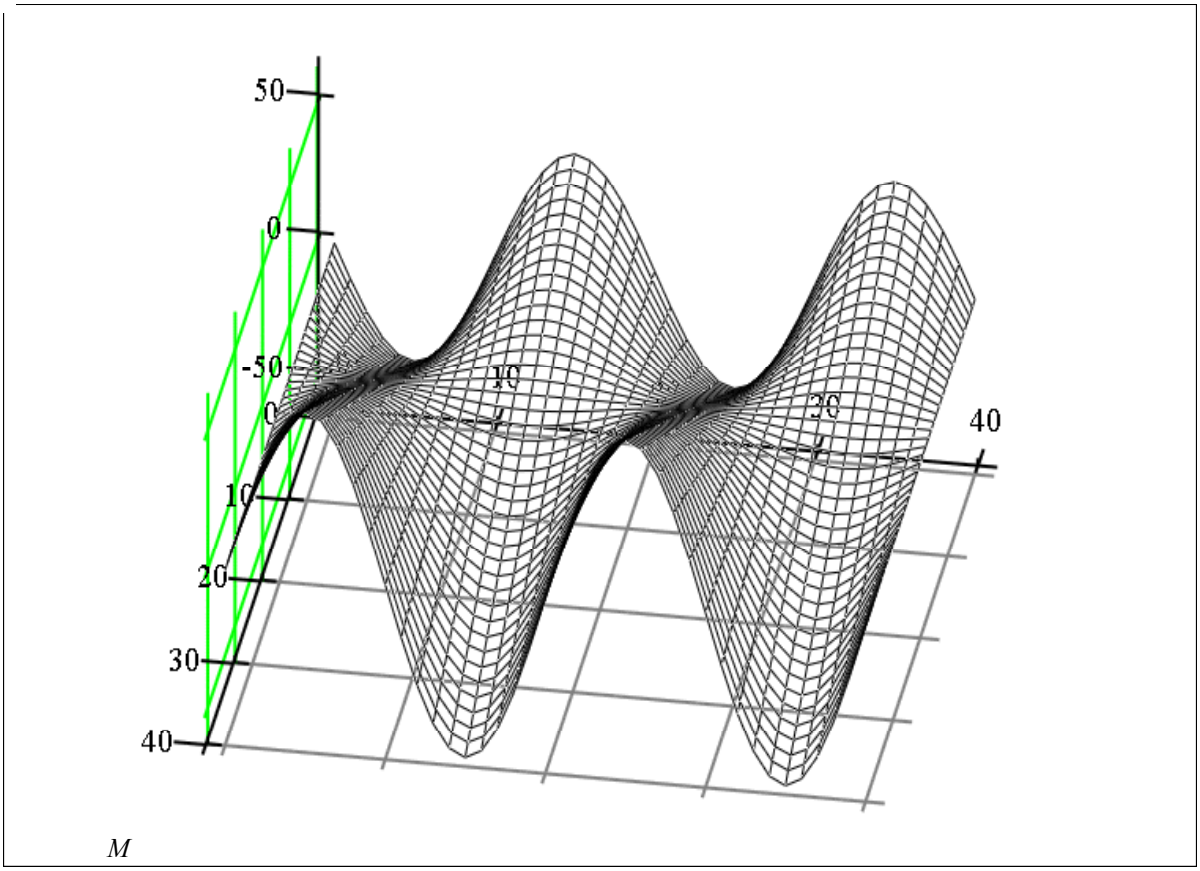

Рис.1.8. Пример построения трёхмерного графика

Шаг 7-й: строим и форматируем график *поверхности* (рис.1.8), последовательно нажимая на кнопки [Инструменты графики] математической панели инструментов и [График поверхности] панели графики. График масштабируется согласно введённым данным.

Для достаточно качественного отображения подобных графиков требуется монитор с разрешающей способностью не менее 800 x 600 точек на дюйм (dpi).

1.10. Символьные преобразования

Mathcad способен осуществлять в автоматическом режиме ряд символьных преобразований формул, отыскивая соответствующее решение или упрощая их написание в общем виде, то есть без получения числового результата.

Пример 1.21

Дано:

$$\int \sin x \, dx.$$

Найти:

неопределенный интеграл данного выражения как его общее выражение.

Решение

На рабочем листе Mathcad пишем:

$$\int \sin(x) \, dx$$

и не убираем рамку вокруг этого выражения. Проверяем, чтобы ограничитель в рамке охватывал всю формулу, при необходимости перемещая его с помощью мыши, клавиш управления курсором или нажимая на клавишу [End] или «Пробел».

Из главного меню вводим

Символы (Symbolics) | Расчеты (Evaluate) | Символические (Symbolically)

На экране получаем результат:

$$-\cos(x)$$

К нему следует добавить постоянную интегрирования.

Место вывода результата, например по горизонтали справа от исходного выражения, можно предварительно задать командами

Символы (Symbolics) | Стиль вычислений … (Style…) | Горизонтально (Horizontally)

[OK]

Если при этом в составе меню Стиль вычислений установить флаг ☑ на позиции Показать комментарии, то исходная формула будет связана с результатом её преобразования словом yields (даёт).

Читателю рекомендуется испытать возможности Mathcad на других примерах, например на выражении суммы членов геометрической прогрессии, используя команды

Символы (Symbolics) | Упростить (Simplify)

В ответ между исходной формулой и результатом её упрощения на экране видим слово Simplifies to (т.е. «упрощается к … »).

В том же меню Символы приведен ряд других полезных команд.

1.11. О программировании в среде Mathcad

Если математических средств, встроенных в Mathcad, недостаточно, имеется возможность использовать его собственный внутренний язык программирования высокого уровня. Мы этот язык в рамках настоящего учебного пособия рассматривать не будем, поскольку ниже дана специальная глава, посвященная основам программирования в среде Visual Basic. Интересующимся рекомендуем обратиться к первоисточникам [2, 3].

2. Основы программирования в среде Visual Basic

К числу достоинств современных «визуальных» языков программирования относят:
1) высокую степень наглядности отображения на экране монитора процесса переработки информации;
2) оперативность как ввода исходных данных, так и получения результатов счета;
3) структурированность программ, которые представляются состоящими из ряда отдельных процедур (подпрограмм или функций);
4) возможность реализации развитых средств графики.

Для широкого круга пользователей Visual Basic может быть использован в любом из следующих двух вариантов:
а) усечённом варианте в виде Visual Basic for Applications (в дальнейшем VBA), встроенном в Excel;
б) полном варианте – Visual Basic (сокращенно VB) в виде независимого программного комплекса версий 5.0 или 6.0, что будет далее обозначаться обобщённо как VB.

Полный вариант отличается от усеченного значительно большими возможностями создания программных проектов, однако для начинающих пользователей эти различия сводятся в основном к характеру использования графических средств представления результатов вычислений. VBA использует графику Excel, знакомство с которой состоялось в части 1 настоящей работы, а VB5 (VB6) располагает собственными средствами управления процессом построения различных графических объектов без необходимости обращения к Excel.

2.1. Программирование вычислительного процесса в VBA и VB

Как известно из курсов программирования, любая программа состоит из последовательности инструкций (операторов), каждая из которых описывает одно из выполняемых компьютером действий. Эти действия выполняются над переменными и постоянными величинами (операндами).

Соответственно решаемой задаче, эти величины в общем случае следует отнести к определенному типу и описать их в начале программы.

Пусть, например, для переменной ***x*** достаточна одинарная точность (7 верных значащих цифр), переменная ***y*** требует удвоенной точности (15 верных значащих цифр), а ***z*** является целым числом. Тогда программа должна начинаться с описания этих (а вообще и всех, вовлекаемых в процесс счёта) переменных следующим образом:

```
DIM  x  As Single, y  As Double
DIM  z  As Integer
```

Здесь аббревиатура DIM (одно из ключевых, т.е. распознаваемых компьютером слов) – от англ. Dimension (размер), Single – одинарная (единичная), Double – удвоенная (двойная) точность; As – как (в качестве), Integer – целое.

Та же символика служит для описания *массивов* переменных, например, DIM x(1000), y(500,100), соответственно для одномерных и двумерных массивов. Порядковый номер первого элемента массива описывается оператором Option BASE (), где в скобках указывают 0 или 1.

Допускается также традиционное описание типов переменных оператором DefType, принимающего формы, например:

DefSng a – d – для переменных одинарной точности от ***a*** до ***d*** включительно;

DefDbl x, y – для переменных ***x*** и ***y*** удвоенной точности;

DefInt z – для целой z.

Заметим, что большинство современных языков программирования использует в качестве стандарта английский язык.

Вообще переменные могут быть и не описаны явно. Тогда по умолчанию тип переменной – Variant присваивается автоматически компьютером. Однако это занимает больший объём оперативной памяти, снижает производительность вычислительного процесса, усложняет процедуру поиска ошибок.

Ввод исходных значений переменных в языках VBA и VB отличается от их предшественников – языков Turbo Basic, QBasic, Quick Basic [4, 5] отсутствием оператора INPUT (априорно предполагается знакомство читателя, по крайней мере, с одним

из таких языков программирования). Вместо этого ввод часто осуществляют через текстовое окно (TextBox) на экране монитора. При этом вообще такие окна непосредственно предназначены для ввода текста (строковых переменных). Для ввода числовых исходных данных оператор распознавания компьютером введенного, например значения переменной x имеет вид

x = Val (TextBox1.Text),

где Val – от англ. Value – величина, значение;
TextBox1 – имя текстового окна, предлагаемое пользователю компьютером в среде VBA (в VB вместо этого предлагается имя Text1.

Последнее может быть принято пользователем или видоизменено им по собственному усмотрению для большей наглядности и удобства для запоминания. Например, здесь может быть принято имя txtX.

Символы математических действий в VBA и VB почти целиком аналогичны таковым в других языках программирования:
(+) сложение; (–) вычитание; (∗) умножение; (/) деление; (∧) возведение в степень; SQR (x) – извлечение квадратного корня из значения x.

Также практически аналогичны используемым в упомянутых предшественниках операторы цикла; условного оператора и оператора безусловного перехода, например,

а) FOR I = 1 TO 100 Step 2
<тело цикла>
Next I

Здесь могут быть также использованы знакомые операторы цикла вида

Do Until – Loop;
Do While – Loop;
While – Wend.

б) If x > 0 Then GoTo 3 Else GoTo 5,
где 3 и 5 – метки соответствующих операторов условного перехода в зависимости от выполнения условия (x>0).

If x > 0 Then x = y + 65 Else y = – 3.

Допускается также блочная конструкция:

If ⟨условие 1⟩ Then

⟨операторы, реализуемые при выполнении условия 1⟩

ElseIf ⟨условие 2⟩ Then

⟨операторы, реализуемые при выполнении условия 2⟩

Else

⟨операторы, реализуемые при прочих условиях ⟩

End If

в) GoTo 10

Оператор вывода значения ***Y*** в текстовое окно имеет вид

textY.Text = Str (Y)

где Str – от англ. String – строка, строковая переменная. Вместо числа может быть также выведен текст в кавычках, например

txtT.Text = Str (“…”)

Вывод контрольных или промежуточных значений переменных на экран может быть также осуществлен через *информационное окно* (окно сообщений). Например

MsgBox “A= ” & A

где Msg Box – Message Box – имя окна сообщений;
A – имя переменной;
& – оператор конкатенации, то есть соединения в строке записи «A =» и значения A.

Как упоминалось выше, программа, написанная на языках VBA и VB, состоит из отдельных процедур. Эти процедуры могут быть одного из следующих видов:

процедура – *функция*, обозначаемая словом Function и предназначаемая для вычисления одного её значения,

процедура – *подпрограмма*, именуемая Sub (от англ. Subroutine – подпрограмма), с помощью которой вычисляется ряд зна -

чений переменных.

Имена соответствующих процедур записываются с указанием области их «видимости» со стороны других процедур и возможности информационного обмена, а также с учётом способа приведения их в действие и обязательно указываются в заголовке процедуры, что будет показано в следующем параграфе.

Каждая подпрограмма заканчивается записью End Sub, а каждая функция – End Function.

2.2. Практика реализации языков программирования VBA и VB

Если на компьютере уже установлен пакет Microsoft Office с имеющейся в нём системой электронных таблиц Excel, то для начинающего пользователя имеет смысл обратиться к встроенному в Excel варианту Visual Basic (VBA). К тому же это не требует от компьютера дополнительных ресурсов. Тогда после включения компьютера и загрузки операционной системы Windows следует загрузить Excel, а затем из главного меню вызвать VBA

Вид | Панели Инструментов | Visual Basic

На экране появится дополнительная шестикнопочная панель инструментов. Сделав [1Л] на четвёртой слева кнопке панели, вызываем РЕДАКТОР Visual Basic. Замечаем, что назначение кнопок панелей инструментов поясняется всплывающими подсказками.

Для продолжения работы следует произвести [1Л] на второй слева кнопке главной панели инструментов – Вставить User Form.

User Form – это пользовательская *форма* (в VB обозначается просто как Form), представляющая собой прямоугольную область на экране, на которой располагаются органы управления формой, поясняющие надписи и окна для ввода и вывода информации в виде чисел и фрагментов текста. В заголовке отображается имя формы и её порядковый номер. За каждой формой скрывается её *код,* являющийся программой её действий. Вместе с формой на экране появляются *панель инструментов* управления формой (Control Panel) с рядом кнопок (см. всплываю-

щие подсказки), а также *окно обозревателя* проекта (Project Explorer) и *окно свойств* (Properties). Если их нет, они могут быть вызваны действиями [1Л] на кнопках панели управления Visual Basic или с использованием главного меню (меню ВИД).

Сочетание форм, описывающих их кодов и модуля образует *проект*.

Большие проекты могут содержать множество форм. Каждую новую форму можно открыть, сделав [1П] на значке проекта в его окне. Затем ввести команды

Вставить | Форму

Доступ к формам осуществляется через окно проекта. В нём графически отображается структура проекта в целом, содержащего ряд форм и *модуль*.

Модуль необходим для передачи данных из одной формы или процедуры в другую. В модуль следует вносить описания массивов, например упоминавшихся выше, путем записи

```
Public x(1000), y(500, 100)
```

причем запись Public (общий) объявляет данные, содержащиеся в этих массивах, доступными для процессов переработки информации в любой из форм и процедур проекта. В отличие от этого процедура, описывающая такой процесс только в пределах данной формы, относится к категории Private (частная).

Вставить модуль в состав проекта можно так же, как и форму, сделав [1Л] на слове Модуль в функции Вставить.

Переход к модулю, а также от одной формы к другой при создании или редактировании проекта в целом осуществляется щелчком (click) левой клавиши мыши на соответствующем объекте *в окне проекта*. Аналогично можно перейти от формы к её коду и обратно путём [1Л] на соответствующих кнопках в верхней части окна проекта (см. всплывающие подсказки).

Если проект содержит только одну форму, то наличие модуля в проекте не является обязательным.

Для заполнения формы необходимым для решения конкретной задачи содержимым с помощью мыши выбирают на панели управления нужный символ. Делают на нём двойной щел-

чок [2Л]. Соответствующий элемент управления при этом автоматически переносится на поле формы, после чего следует, удерживая в нажатом положении левую клавишу мыши, перетащить данный элемент к нужному, удобному для обозрения и управления месту формы.

Можно ограничиться и одним щелчком [1Л] на выбранном элементе панели управления, а затем с помощью мыши очертить область расположения этого элемента на поле формы. Также с помощью мыши возможно изменение размеров элементов управления при их размещении в пределах формы путём перетаскивания границ элемента за находящиеся на них маркеры.

2.3. Пример создания проекта VBA или VB

Рассмотрим в качестве примера процесс разработки простейшего проекта вычисления корней квадратного уравнения в среде VBA (VB).

Пусть дано квадратное уравнение

$$Ax^2 + Bx + C = 0 .$$

Известно, что корни этого уравнения R_1 и R_2 вычисляются по формуле

$$R_{12} = \left(-B \pm \sqrt{B^2 - 4AC}\right) \Big/ (2A).$$

Создаем форму, содержащую три окна ввода (значений A, B и C) и два окна для вывода результатов решения (R_1 и R_2). На форме необходимо также разместить кнопки запуска на счёт [ПУСК] и отключения [СБРОС], а также ввести поясняющие надписи (рис.2.1, с. 54).

Конкретно действия пользователя в рамках поставленной задачи сводятся к следующему.

1) Убеждаемся в том, что на экране высвечена форма (пока пустая) и имеются панель управления, а также окно проекта и окно свойств. Если их нет, вызываем эти компоненты и средст-

ва создания нашего проекта с помощью панели инструментов Visual Basic или – из главного меню.

2) Вызываем из панели управления текстовое окно (TextBox) для ввода значения A посредством [1Л] или [2Л] на кнопке с символом ab|. Здесь и далее не оставляем без внимания всплывающие подсказки. Определяем положение этого окна на форме и устанавливаем его размеры, достаточные для отображения числа A.

3) Удаляем запись Text1 в окне. Для этого обращаемся к окну свойств, находим в нем строку с заголовком Text и в столбце, расположенном справа от заголовка в той же строке, удаляем ненужное нам слово Text1 средствами клавиатуры (клавишами [Delete] или [Backspace]).

4) Описываем свойства окна, для чего в окне свойств выбираем первую сверху строку с заголовком (Name) и в расположенную правее этого заголовка ячейку, удалив высвеченное там то же слово Text1, вводим запись txtA.

5) Рядом с окном или над ним устанавливаем поясняющую надпись, используя кнопку с символом **A** на панели управления, и пишем с помощью клавиатуры аналогичный, в данном случае, символ на форме. Можно выбрать шрифт надписи и его размеры, используя свойство Font (шрифт) в окне свойств, причем [1Л] на кнопке с изображением знака «минус» позволяет выбрать начертание и размер шрифта действиями, аналогичными используемым при работе в текстовом процессоре WORD. Заметим, что это же свойство Font применимо и к самому окну ввода, позволяя предопределить размеры вводимых впоследствии цифр, а также установить их расцветку (свойство ForeColor – цвет переднего плана) и цвет фона (свойство BackColor). Нужные цвета пользователь имеет возможность выбрать из включенных в состав этих свойств палитр.

6) Создаём два окна вывода корней со свойствами (Name): txtR1 и txtR2 и поясняющими надписями «Корни», «R1=» и «R2=».

7) Для лучшего эстетического восприятия уточняем размеры и размещение наших объектов (окон, кнопок и надписей) на форме. Для этого на объекте делаем [1Л]. Объект окружается маркерами, за которые с помощью мыши, установив указатель мыши на нужном маркере и удерживая в нажатом состоянии её левую клавишу, можно перетаскивать любую из границ объекта по полю

формы. Для перетаскивания объекта целиком указатель мыши устанавливают на поле объекта между маркерами и также при нажатой левой клавише мыши изменяют расположение объекта. Закончив перетаскивание, нужно отпустить левую клавишу

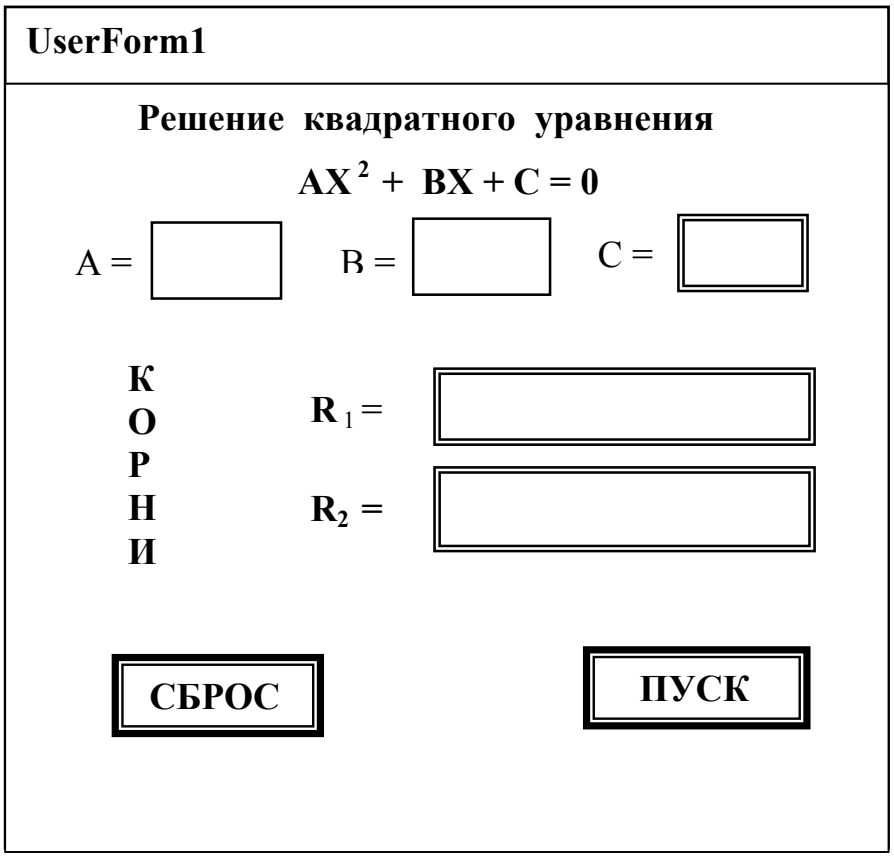

Рис.2.1. Форма к решению квадратного уравнения

мыши и произвести [1Л] вне области, окруженной маркерами, для того чтобы их убрать с экрана.

8) Окна вывода текста и данных рекомендуется сделать недоступными для пользователя установкой свойства Enabled (сделать доступным) в состоянии False (ложь).

9) В верхней части поля формы можно сделать общий заголовок «Решение квадратного уравнения».

10) Для того чтобы закончить проектирование формы, остается установить на ней (обычно – в нижней части поля формы справа и слева) кнопки запуска на решение и останова программы. Эти кнопки вызываем из панели управления теми же действиями,

которые выше были использованы для вызова окон и надписей. Кнопке запуска для наглядности следует придать свойство (Name), первоначально обозначаемое компьютером как CommandButton1 (командная кнопка 1) и подлежащее удалению, в виде, например, cmd_Start, где аббревиатура cmd_ от слова command – команда. Кнопке останова, действие которой называют также сбросом или выходом, целесообразно присвоить имя (Name) как cmd_Quit от английского quit – прекратить действие.

11) На кнопках пуска и останова нужно также сделать поясняющие надписи, причем лучше по-русски. Для ввода этих надписей делаем [1Л] на кнопке запуска. Кнопка выделяется маркерами. Снова обращаемся к окну свойств. Находим там свойство Caption (заголовок). Удаляем продублированную там запись CommandButton1. Вместо неё русским шрифтом пишем ПУСК. Форматирование шрифта можно при этом осуществлять действиями, описанными в п. 5). Точно так же вместо заголовка CommandButton2 на кнопке останова размещаем надпись, например, СБРОС.

12) Во избежание потери выполненного этапа работы по созданию проекта, следует форму сохранить. Это делаем, обращаясь к меню Файл (File) главного меню, входим в него и выбираем из числа предлагаемых нам функций

Save Project1 As…

(Сохранить проект в качестве …). Такой выбор продиктован тем, что нашему проекту необходимо дать уникальное имя. Это имя и адрес сохранения файла (файлов) проекта вводим в открывшемся на экране диалоговом окне. Примем во внимание, что появляющееся по умолчанию имя Project1 – это текущее имя (и его порядковый номер) в данном сеансе работы на компьютере. Уникальное, т.е. не повторяющееся в устройствах накопления информации в виде дискет и жестких дисков имя, исключит в дальнейшем возможность наложения проектов и уничтожения обозначенного под тем же именем предыдущего проекта в следующем сеансе работы.

Завершаем процесс сохранения проекта указанием имени диска (например диск A:), папки (например, VBAProjects) и собст-

венного имени проекта (например, VBAProject1.xls[1]), после чего нужно щёлкнуть левой клавишей мыши на кнопке «Сохранить».

В процессе дальнейшей работы при внесении каких-либо исправлений или дополнений проект можно будет сохранить более простой командой

Save Project

(Сохранить проект), поскольку имя его нами дано и оно остается в памяти компьютера. Заметим, что в окне проекта после текущего имени Project1 в скобках автоматически отображается данное нами имя проекта.

Расширение имени (файловый тип) присваивается компьютером автоматически. При использовании VBA это расширение приобретает вид xls, а при работе с VB5 (или VB6) оно обозначается как vbp (Visual Basic Project).

13) Следующим этапом создания проекта является написание программы управления формой. Эта программа называется кодом (code). Вызываем окно кода щелчком [1Л] на кнопке с поясняющей (всплывающей) надписью code в верхней части окна проекта.

Применительно к рассматриваемому примеру решения квадратного уравнения заполняем окно кода в следующей последовательности (Приложение 1):

А) В разделе General проверяем наличие формируемой по умолчанию записи Option Explicit (явное описание переменных). Если этой записи нет, то вводим её.

Б) Пишем программу по правилам Visual Basic, причем здесь после знака ´ (апостроф) приведены поясняющие для начинающего комментарии.

В) Дописываем код процедурой (подпрограммой) останова, то есть прекращения действия проекта в целом. Для этого пользователь должен нажать (click) один раз [1Л] левой клавишей мыши на кнопку [СБРОС]. И вот что нам требуется дописать:

```
Private Sub cmd_Quit_Click ()
End
End Sub
```

[1] При работе в среде VBA.

Осуществим теперь пробный запуск созданного нами проекта, содержащего пока одну форму и код к ней.

Для запуска необходимо щёлкнуть [1Л] на кнопке с изображением [▶] (запуск), находящейся на главной панели инструментов.

Если при написании кода не были допущены ошибки, то в ответ на экране появится форма в своем рабочем обличии, т.е. без сетки точек и без маркеров. В первом по порядку созданном нами окне, предназначенном для ввода значения ***A,*** мигает курсор, приглашая нас ввести соответствующее число. Вводим его, например – 12

Для перевода курсора в следующее окно с целью ввода значения ***B*** можно воспользоваться либо нажатием на клавишу табуляции [Tab], находящуюся на клавиатуре слева, либо непосредственно указать это окно с помощью мыши. Введём здесь, например число 5,5. Аналогичными действиями введём число 3 в окно ***C***.

Важно отметить, что при использовании как VBA, так и VB5 (VB6) десятичным разделителем остается «англоязычная» точка, несмотря на возможно русскую версию Excel, при действии в которой десятичным разделителем становится запятая.

Д) Нажимаем еще раз на клавишу [Tab] и замечаем, что после ввода значений ***A, B*** и ***C*** оказалась выделенной двойным контуром и затенением кнопка [ПУСК]. Таким образом система сообщает нам о готовности проекта к запуску.

При использовании клавиши [Tab] текстовое окно и окна вывода результатов счёта обходятся курсором. Точно так же не увенчается успехом попытка ввести какую-либо информацию в эти окна с помощью мыши. Это служит наглядной иллюстрацией действия установленного нами свойства Enabled в состояние False. Если же изменить это свойство, устанавливаемое по умолчанию во всех окнах в состояние True (Истина), то окна доступны для ввода или корректировки информации.

В случае отсутствия ошибок запуск проекта сопровождается передачей в окна вывода результатов счета согласно написанному нами коду.

Если в коде оказались ошибки, то на экране появится информационное окно с указанием на природу этих ошибок. На -

жав [1Л] на кнопку [OK] или кнопку [Debug] (Отладка) такого окна, переходим в окно кода, где ошибочные записи оказываются выделенными желтым (или иным) цветом. Ошибки следует исправить, снова сохранить проект и повторно запускать на выполнение до тех пор, пока не будут выявлены и устранены все ошибки кода. О многих ошибках вычислительная система сообщает пользователю непосредственно в процессе набора кода соответствующим выделением ошибочных строк цветом или – выводом замечаний текстом.

В ходе тестирования оцениваем достаточность установленных нами свойств элементов формы, в том числе – цвета, формы и размера шрифта, в окнах ввода и вывода информации. При необходимости корректируем их, осуществив останов программы нажатием [1Л] на кнопку [СБРОС].

Примем во внимание, что при ошибках в коде работа проекта автоматически приостанавливается. Он не будет отвечать ни на какие наши действия до тех пор, пока не нажмём [1Л] на находящуюся на главной панели инструментов кнопку с изображением [■] (СТОП).

Е) Проверим работу проекта вводом контрольных цифр:

а) ***A*** = – 12*;* ***B*** = 5.5*;* ***C*** = 3. Ответ – корни действительные: *R1* = – 0.321*; R2* = 0.779.

б*)* ***A*** = 5*;* ***B*** = 1*;* ***C*** = 50. Ответ – корни комплексные :

R1 = – 0.1 + i · 3.161*; R2* = – 0.1 – i · 3.161 (с точностью до трёх десятичных знаков)*.*

Несколько усложним теперь код рассматриваемого примера, введя ограничения на переменные ***A*** и ***B***. Пусть в условиях некоторой задачи

$$-20 \le \boldsymbol{A} \le -5; \quad 1 \le \boldsymbol{B} \le 10.$$

Эти данные приводят к получению действительных корней, в чем читатель может убедиться.

После оператора B = Val(TextBoxB.Text) введём две строки следующего содержания:

```
If  A < – 20  Or A > – 5 Then  End
```

If B < 1 Or B > 10 Then End

Вместо Then End можно использовать оператор Then Exit Sub (выход из подпрограммы).

Как и принято в WORD, для раздвижки строк текста следует установить курсор в первую позицию (Home) строки, следующей за указанным оператором ввода значения ***B***, и дважды нажать на клавишу [Enter]. В освободившиеся две строки кода вводим принятые ограничения.

При запуске проекта с такими ограничениями в случае ошибочного ввода данных, не укладывающихся в заданные пределы, автоматически произойдет останов программы, аналогичный нажатию на кнопку [СБРОС].

Можно усовершенствовать условия останова, заставив компьютер сообщать о том, что ему «не понравилось» в составе исходных данных. Такое сообщение может быть выведено в *информационное окно* (Message Box) следующим образом. Вместо указанных двух дополнительных строк кода введём

```
If A < – 20 Or A > – 5 Then
MsgBox(«Недопустимое значение A»)
End
End If
If B < 1 Or B > 10 Then
MsgBox(«Недопустимое значение B»)
End
End If
```

После появления информационного окна и прочтения пользователем содержащегося в нём сообщения необходимо сделать [1Л] на кнопке [OK], после чего информационное окно исчезает, и программа выходит на завершающий оператор End (или Exit Sub), после чего проект следует перезапустить с уточнёнными значениями исходных данных.

Можно расширить содержание информационного окна, вызываемого ключевым словом MsgBox, заставив, таким образом, компьютер выводить не только текст, но и численные значения тех или иных промежуточных переменных, что удобно, в частности при отладке программ. Например, в рассмотренной задаче

вычисления корней квадратного уравнения после определения дискриминанта

$$D = B^2 - 4AC$$

в состав программы пользователь в виде эксперимента может включить оператор

```
MsgBox("Дискриминант равен" & D)
```

Тогда в информационном окне на экране монитора появится не только выделенный кавычками текст, но и вычисленное значение ***D***, причём знакомый символ & (знак конкатенации) означает объединение текстовой и числовой информации в одном окне.

Инструментарий Visual Basic в любой его форме: VBA или VB − располагает также возможностью организации ввода данных не только через текстовое окно, но также с помощью особого вида информационного окна − окна ввода (Input Box). Оператор вызова этого окна на экран, например при необходимости ввода значения некоторой величины ***A*** имеет вид

```
A = Val (InputBox("A="))
```

При прочтении компьютером данного оператора на экране появляется окно, в заголовке которого отображается запись "A=", и значение ***A*** пользователь вводит на место мигающего курсора в этом окне. После нажатия [1Л] на кнопку [OK] окна ввода оно исчезает с экрана, а введённое значение вовлекается в вычислительный процесс.

Подобное средство общения человека с компьютером является одной из форм *диалогового (интерактивного) режима*.

2.4. Пример построения графика средствами VBA

Пусть даны функции

$$Z_1 = \left(X^2 - 2Y\right) / \left(2Y^2\right);\ Z_2 = e^{Z_1},$$

и требуется построить графики

$$Z_1 = f_1(Y);\ Z_2 = f_2(Z_1).$$

Примем ***X*** = 0,5 = const, а диапазон варьирования другого аргумента в пределах $1 \le Y \le 20$ с шагом 1.

Построение таких графиков средствами одного лишь языка VBA невозможно, и задача решается во взаимодействии средств этого языка и электронных таблиц Excel.

Первым этапом решения является организация обмена данными между VBA и Excel, причём необходимо разработать на языке VBA программу автоматического заполнения данными Рабочего Листа Excel с последующим привлечением графических средств, принципы управления которыми рассмотрены в части 1 настоящей работы.

После загрузки Excel и Visual Basic for Application (VBA) делаем [1П] на второй слева кнопке главного меню (Вставить UserForm) и подтверждаем [1Л] на UserForm. Появляется новая форма, готовая для размещения на ней окон ввода данных, элементов управления и надлежащих надписей. Щелчками [1Л] на кнопках главной панели управления вызываем окно проекта – VBAProject и окно свойств UserForm1. Если почему бы то ни было отсутствует панель элементов управления, делаем [1Л] на кнопке [Панель элементов].

Теперь конструируем форму UserForm1 (рис.2.2, с. 62). Она должна содержать шесть окон ввода, из которых первое – для ввода постоянного в данном примере значения аргумента ***X***, второе – для начального значения аргумента ***Y***, третье – для его конечного значения, четвертое – для шага варьирования в пределах от начального до конечного значений включительно. Еще два окна – пятое и шестое, предназначены для ввода формул.

Вводим поясняющие надписи на форме и три командных кнопки управления с надписями ВЫЧИСЛЕНИЕ, СБРОС, ГРАФИК.

Ознакомимся в этом примере с возможностью упрощенного использования имен окон и командных кнопок − без назначения им пользовательских обозначений, а принимая предлагаемые

компьютером по умолчанию. При этом достаточную наглядность обеспечиваем приведенными в предыдущем абзаце поясняющими надписями. По умолчанию вычислительная система назначает нашим объектам следующие имена:

Для окон	Для командных кнопок
TextBox1	CommandButton1
TextBox2	CommandButton2
TextBox3	CommandButton3

и т.д.

Рис.2.2. К примеру вычислений и построения графика в среде VBA (VBAProject2.xls)

Так и будем называть их в коде, т.е. в программе действий создаваемого нами проекта.

Условимся Рабочий Лист1 Excel, куда из формы будут выводиться исходные данные и результаты счёта, заполнять следующим образом:

Столбец A – первый аргумент $\boldsymbol{X}$ = const;
Столбец B – второй (варьируемый) аргумент $\boldsymbol{Y}$ = var;
Столбец C – первая формула (для вычисления Z_1);
Столбец D – вторая формула (для вычисления Z_2).

Первая строка этого диапазона столбцов в дальнейшем будет отведена под заголовки данных каждого столбца. Непосредственно информация будет заноситься начиная со второй_строки. Тогда надпись “Формула 1” обозначает окно ввода формулы для вычисления значения функции Z_1, а надпись “Формула 2“ предназначена для обозначения окна написания формулы для вычисления Z_2 .

В соответствии с принятыми допущениями составим текст кода формы (Приложение 2), руководствуясь приведенными в нём комментариями.

В составе кода процедура под условным наименованием Private Sub CommandButton1_ Click() используется для автоматизации пересчета исходных данных и передачи их в таблицу (Рабочий Лист 1). Полученные результаты используются собственно для построения графика процедурой Private Sub CommandButton4_Click(). Последняя автоматически вызывает известную из части 1 программу Мастер Диаграмм и сообщает этой программе необходимые параметры построения графика.

Процедура Private Sub CommandButton3_Click() предназначена для закрытия формы.

Процедура Private Sub UserForm_Initialize () служит для активизации окна формы и назначает клавише [Enter] функцию кнопки ВЫЧИСЛЕНИЕ, а клавише [Esc] – функцию кнопки СБРОС (для сброса команду нужно вводить дважды).

Переменная $\boldsymbol{n}$ является числом строк, занятых варьируемыми значениями (здесь – аргумента $\boldsymbol{Y}$). Поскольку эта переменная используется в двух процедурах, в состав проекта включаем модуль. В код последнего вносим запись

```
Public n As Integer
```

A	B	C	D
Аргумент 1	Аргумент 2	Функция 1	Функция 2
0,5	1	-0,875	0,4168
0,5	2	-0,4687	0,6257
0,5	3	-0,3194	0,7265
0,5	4	-0,2421	0,7849
0,5	5	-0,195	0,8228
0,5	6	-0,1631	0,8494
0,5	7	-0,1403	0,8690
0,5	8	-0,1230	0,8842
0,5	9	-0,1095	0,8962
0,5	10	-0,0987	0,9059

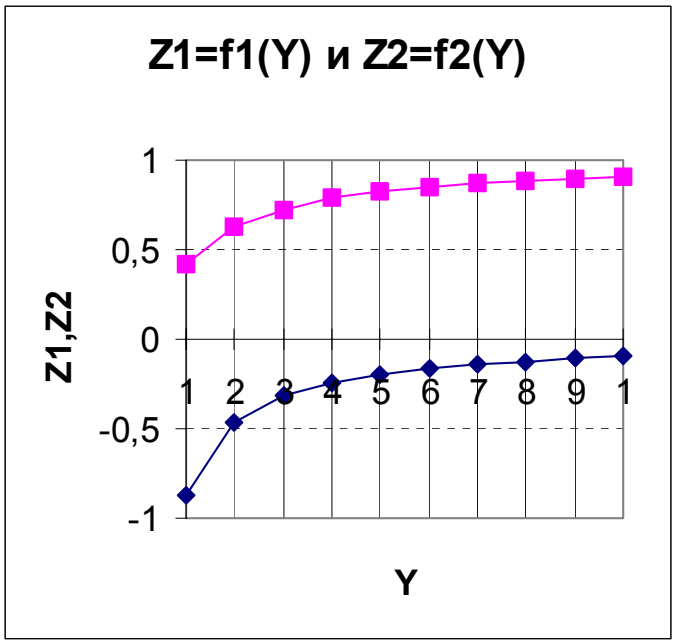

Рис. 2.3. Результаты вычислений
и построение графика средствами VBA

После ввода исходных данных нажимаем на кнопку [ВЫЧИСЛЕНИЕ], готовя данные для построения графика. Непосредственно график создается после нажатия на кнопку [ГРАФИК]. При этом работающая форма остается на экране монитора. Деактивация её производится кнопкой [СБРОС] (дважды).

Для просмотра результатов делаем [1Л] на кнопке (первой слева на главной панели инструментов), всплывающая подсказка к которой сообщает её имя – ВИД Microsoft Excel. После этого открывается Рабочий Лист 1 с результатами счёта и графиком (рис.2.3.). Последний можно редактировать действиями, рассмотренными в части 1, начиная с действия [1П] на поле графика и выбора команды Параметры Диаграммы для корректировки таких элементов изображения, как оси и линии сетки графика.

Очевидно, что аналогичным путём можно построить и трехмерную диаграмму [6, 7].

2.5. Пример вычислений и построения графика средствами VB5 (VB6)

Пусть дано уравнение

$$Z^2 + N\,Z + M = 0,$$

где $N = N_i$ при i, варьируемом в пределах от 0 до 10 включительно с шагом 0,5, и $M = -10 = \text{const}$. Требуется вычислить вещественные корни этого уравнения $R1_i$ и $R2_i$, а также построить графики функций

$$Y1_i = 3e^{7R1_i};\; Y2_i = e^{R2_i},$$

Решение примера здесь принципиально отличается от рассмотренного в разделе 2.4 тем, что Excel здесь совершенно не используется. Вместо этого нам необходимо включить в состав проекта ещё одну форму с графическим окном для отображения кривых графика, а для передачи данных из одной формы (вычисления, рис.2.4) в другую (график, рис.2.5) потребуется ещё и модуль.

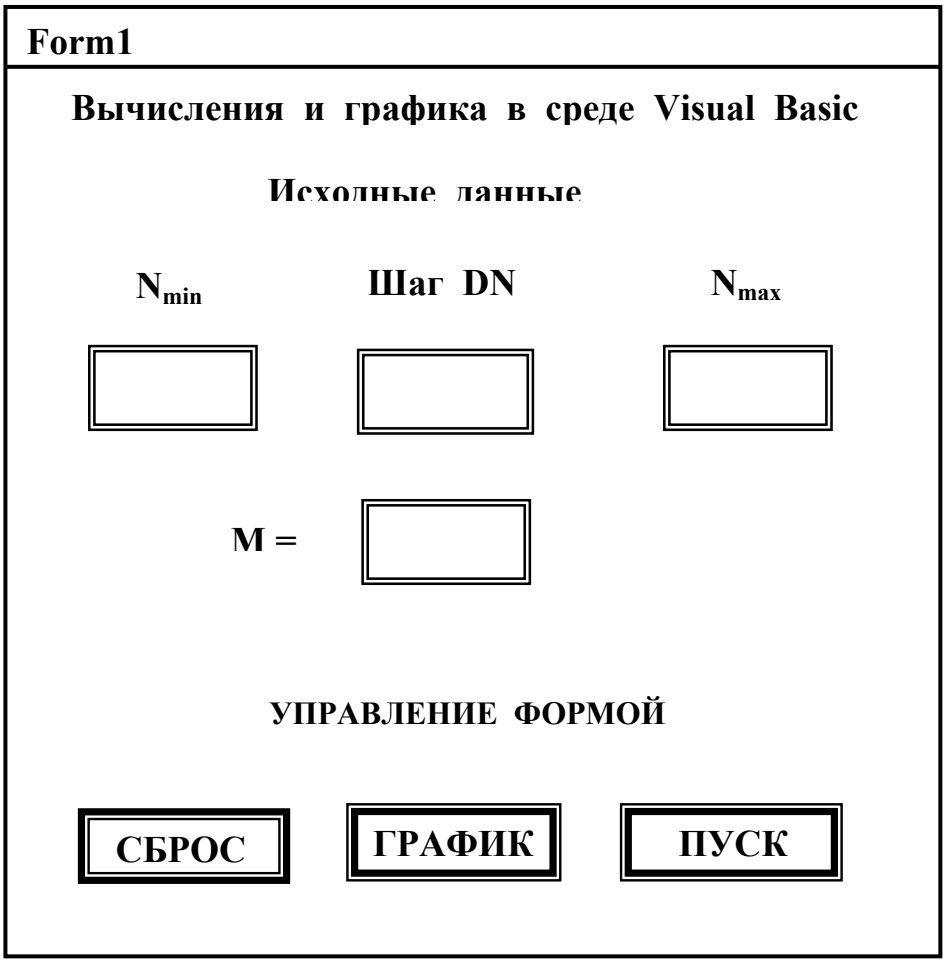

Рис.2.4. Первая форма учебного проекта Project1VB

После запуска Visual Basic, выполнив щелчок [1П] на значке проекта (окно Проект), вызываем контекстное меню. Выбираем команду

Add (Добавить) | **Form** (Форму)

Это будет вторая форма (Form2), так как первая (Form1) загружается автоматически. Аналогичным путём вставляем в состав проекта МОДУЛЬ

Add | Module

На второй форме нужно разместить графическое окно (PictureBox), взяв его с помощью мыши из панели управления и увеличив его размеры по ширине до предела, позволяемого

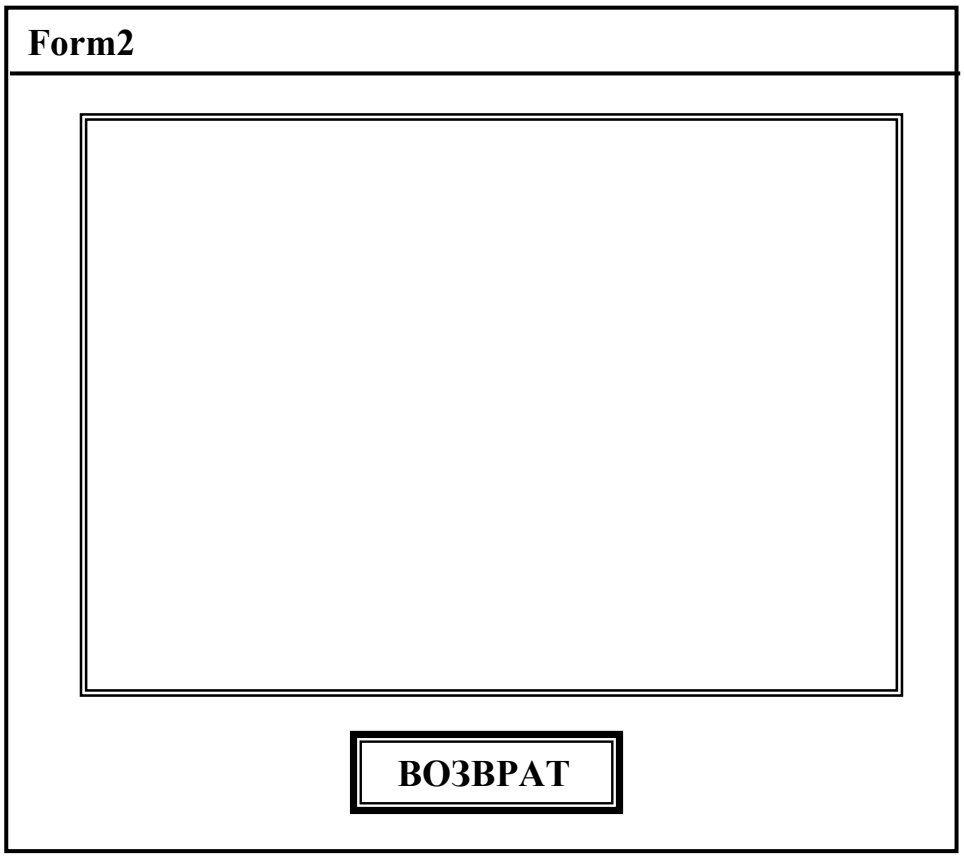

Рис.2.5. Вторая форма учебного проекта Project1VB

формой. По высоте следует установить такой размер, чтобы оставалось место для помещения на форму кнопки возврата к первой форме с целью изменения исходных данных, а сверху – для заголовка графика (рис.2.5).

Замечаем, что по умолчанию свойство (Name) графического окна устанавливается как Picture1. Цвет фона (свойство BackColor) следует установить белым.

Кнопку возврата также берем мышью из панели управления. Придаем ей свойство (Name) в окне свойств в виде предложенного компьютером имени Command1, а надпись на ней – свойство Caption – может быть, например, как ВОЗВРАТ.

Первую форму, перейти к которой можно путем [1Л] на Form1 в окне Проекта, необходимо дополнить ещё одной кнопкой уп-

равления, для которой в окне свойств задать (Name), например, именем Command3, а надпись (Caption) обозначить как ГРАФИК.

Действия по установке кнопок аналогичны описанным в предыдущем разделе.

Для вычисления корней и заданных функций нужно написать код согласно подпрограмме Private Sub Command2_Click() по образцу данного в Приложении 3 с учётом приведенных там же комментариев, полезных для начинающего. После этого следует организовать вывод графика.

Для построения элементов графики в среде VB5 (VB6) используются те же операторы, которые приняты в предшествующих версиях языка Basic, например:

- Для построения круга

Circle (x, y), R, C,

где x, y – координаты центра;
R – значение радиуса;
C – код цвета.

- Для построения точечного графика

PSet (x, y), C,

где x, y – координаты текущей точки.

- Для построения отрезка прямой линии

Line (x1, y1) - (x2, y2), C,

где x1, y1 и x2, y2 – координаты точек начала и конца линии.

Более подробно остановимся на некоторых деталях графических средств языка Visual Basic (полного в отличие от VBA). Здесь начинать построение графических образов следует с написания оператора масштаба (Scale) графического окна:

Scale (m, n) – (p, q),

где m, n – соответственно, абсцисса и ордината левого верхнего угла графического окна;
p, q – абсцисса и ордината правого нижнего угла того же окна.

Существует несколько способов отсчёта этих координат [4, 5, 8 … 10], из которых мы примем прямое их задание в единицах измерения величин x (абсцисса) и $y = f(x)$ (ордината) графика.

Примем область отображения переменных на графике, ограниченную при ***M*** = –10 = const значениями абсциссы

$$0 \le N_i \le 10$$

с шагом варьирования 0,5 и ординаты – по результатам пробных оценок функций ***Y***2 как большей, чем ***Y***1

$$0 \le Y2 \le 4.$$

Вообще координаты левого нижнего угла графического окна желательно было бы иметь ***x = 0; y = 0.*** Именно в таком виде графики представляются в литературе. Однако с учетом необходимости оцифровки координатных осей и внесения элементов автомасштабирования графика по оси абсцисс примем оператор Scale с такими координатами:

Scale(–1.25, 5) – (Nmax+1, –1.25)

В результате оставляем полосы шириной в 1.25 единицы измерения вдоль каждой из координатных осей и приступаем к написанию подпрограммы построения графика Private Sub Command3_Click() по образцу представленной в Приложении 3 с необходимыми комментариями.

Остановимся на некоторых пояснениях к тому, что было использовано нами выше.

1) Нужна пунктуальность в наборе текста кода. Недопустимыми оказываются лишние пробелы, и там, где запись изображена без пробелов, их и не должно быть в коде.

2) Если запись не помещается в строке, часть её текста может быть перенесена на следующую строку. Символами переноса в конце строки служат пробел и знак подчеркивания.

3) В качестве вспомогательной переменной использованы величины ***x*** и *V* для компактности обозначений.

4) Операторы

Form1. Hide

```
Form2. Show
```

означают команду закрыть («спрятать») Форму 1 и открыть («показать») Форму 2.
5) То же, но в другой последовательности, а именно

```
Form2.Hide
Form1.Show
```

используется в подпрограмме для возврата от второй формы к первой (кнопка [ВОЗВРАТ]). Требуется описать эту процедуру (в окне кода второй формы) следующим образом:

```
Private Sub cmd_Return_Click ()
    Form2. Hide
    Form1. Show
End Sub
```

6) Символикой QBColor() с соответствующей цифрой в скобках может быть задан цвет линии таким же образом, как это принято в языке QBasic.
7) Сочетание

```
CurrentX=
CurrentY=
```

используется для задания координат специальных обозначений и поясняющих надписей.
8) Оператор Print действует точно так же, как и в предшествующих вариантах языка Basic и позволяет, в частности, осуществлять печать непосредственно на поле формы.
9) В операторах графики языка Visual Basic можно использовать не только константы (как это было показано в For − Next и Line), но и переменные, значения которых определены.
10) После вставки модуля следует вызвать его окно на экран и ввести туда следующую информацию, доступную из любых форм и процедур проекта:

```
Public Nmin As Single, Nmax As Single
```

```
Public DN As Single, M As Single
Public Y1(200) As Single, Y2(200) As Single
Public R1(200) As Single, R2(200) As Single
```

Это – резервирование ячеек оперативной памяти компьютера для хранения (с запасом) элементов соответствующих массивов индексированных переменных. Как видим, и это описание не отличается от используемого в других версиях языка Basic. Слово Public (общие) служит антонимом, т.е. словом противоположного значения, по отношению к слову Private (частный).

Переменные в составе процедуры типа Private доступны только в пределах этой процедуры или формы.

Итогом нашей работы служат тексты кодов, представленные в Приложении 3.

В завершение выполненных действий можно приступить к тестированию и отладке проекта в целом. При этом пользователю настоятельно рекомендуется самостоятельно поработать в направлении дальнейшего улучшения вида графика, в том числе, и методом проб и ошибок. Смелее экспериментируйте ! Последовательно изменяя тот или иной параметр, Вы сразу оцените на экране особенности его влияния на результаты Вашей работы. Не забудьте сохранить отлаженный проект.

Разумеется, приведенные примеры не исчерпывают всего многообразия средств вычислений и графики в среде Visual Basic. Эти примеры особенно ценны для *первых шагов* в освоении новых средств программирования, чему в существующей литературе должного внимания совершенно не уделяется. А для дальнейшего совершенствования следует обратиться к фундаментальным руководствам, например, [8, 9, 10].

3. Принципы компьютерного моделирования металлургических процессов

Моделирование представляет собой такую разновидность научного исследования различных явлений в природе и технике, при которой исследуемый реальный *объект* (технологический процесс, оборудование, где он осуществляется, или вещество) заменяется его *моделью.* При этом модель должна обладать определённой совокупностью важнейших для данного

исследования свойств, аналогичных свойствам исследуемого объекта.

Результаты моделирования используются при дальнейшем исследовании объекта, проектировании новых или модернизации существующих технологий, оптимизации схемно - конструктивных решений в области совершенствования структуры и режимов работы оборудования, оптимальном планировании производства и автоматизированном управлении технологическими процессами.

Из многих видов моделей, используемых в металлургии и литейном производстве, наиболее характерными оказались модели физические и математические.

Физические модели позволяют исследовать гидравлические и газомеханические процессы, протекающие в металлургических печах, а также − некоторые явления теплопереноса. В отдельных случаях физические модели дают возможность исследовать процессы затвердевания расплавов при различных способах литья с использованием легкоплавких веществ, например гипосульфита или парафина, вместо высокотемпературных металлических расплавов.

Значительно большими возможностями располагает *компьютерное моделирование*, основанное на использовании математических моделей.

Математическая модель (рис. 3.1) представляет собой выражение математическим языком причинно − следственных связей между входными воздействиями x_i на объект (*входами*, или технологическими факторами) и ответными реакциями y_j объекта (его *выходами*, или откликами)

$$y_j = f(x_1, x_2, x_3, \ldots, x_k), \quad (3.1)$$

где $j = \overline{1, n}$ при общем числе входов, равном k.

Математическая модель служит основой алгоритмизации процесса моделирования, и на этой основе строится процедура компьютерного моделирования с использованием соответствующих средств программного обеспечения.

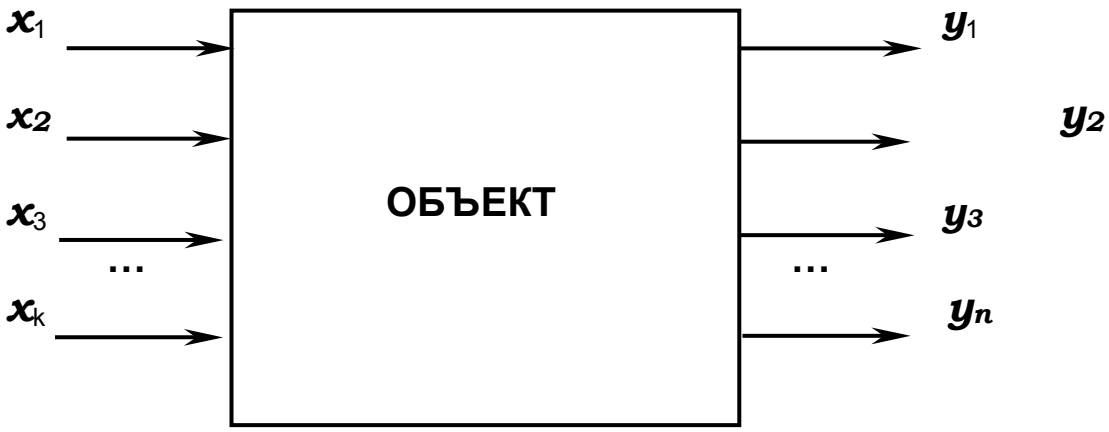

Рис. 3.1. Моделируемый объект с его входами и выходами

3.1. Общая классификация математических моделей

Основные разновидности математических моделей включены в состав схемы, представленной на рис. 3.2, с.74.

Стохастические модели позволяют прогнозировать состояние объекта в определённых пределах значений его параметров, характеризуемых доверительной вероятностью. Принципы построения таких моделей должны быть знакомы читателю по материалам части 1 настоящего учебного пособия (главы 8, 9), посвящённым организации пассивного и активного эксперимента с примерами статистической обработки полученных результатов (например, с помощью Excel). В отличие от этого *детерминированные* модели дают возможность прогнозировать состояние объекта *однозначно*. Такие модели строятся *аналитическими методами*, характерным примерам применения которых в рассматриваемой отрасли посвящён последующий материал данной главы.

В состав математической модели могут входить как линейные, так и нелинейные математические выражения. По виду этих выражений и соответствующих методов решения уравнений моделей последние подразделяются на *линейные* и *нелинейные*.

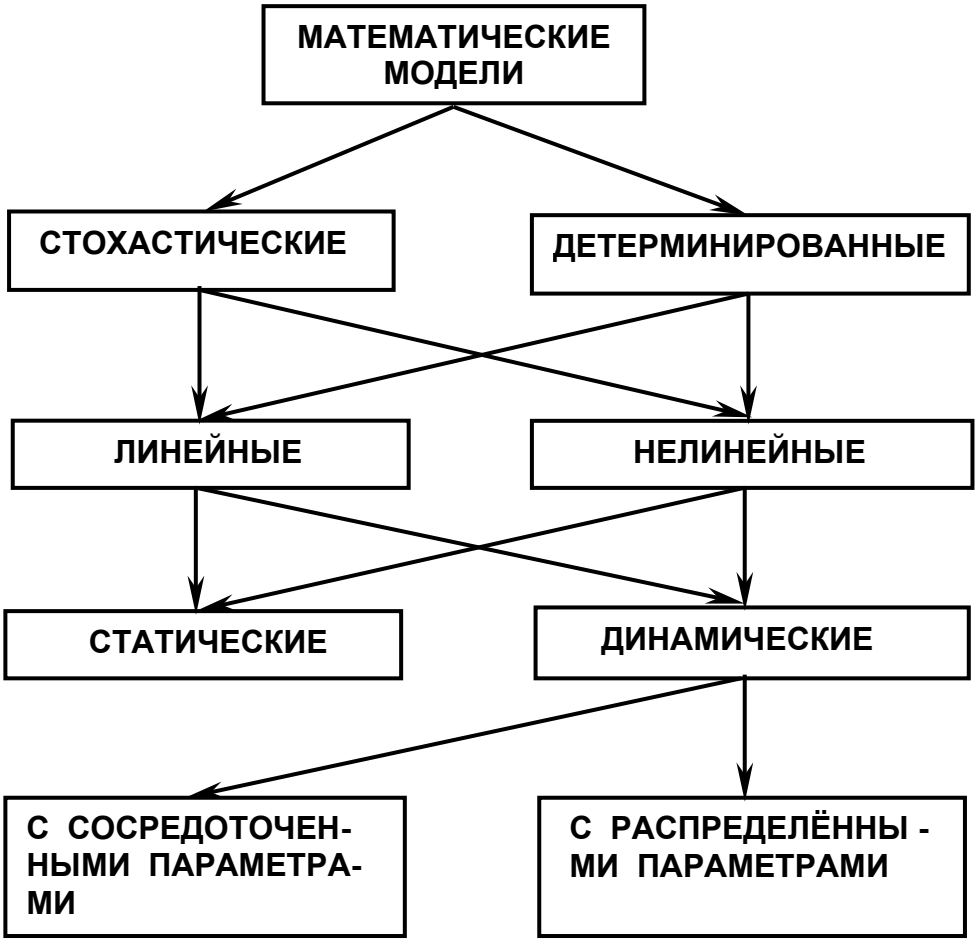

Рис. 3.2. Основные виды математических моделей

Статические модели характеризуют объект в состоянии равновесия, признаком которого является постоянство входов и выходов во времени. В противоположность этому *динамические* модели призваны прогнозировать развитие переходного процесса в объекте во времени − от заданного начального состояния до достижения его конечного состояния.

Отличительным признаком моделей с *сосредоточенными параметрами* (как и самих моделируемых объектов) является одинаковое значение переменных состояния объекта − входов, выходов и промежуточных переменных − во всём его пространстве. Примером может служить модель процесса формирования тонкостенной отливки, у которой температура, по крайней мере, в пределах доступной точности измерения, равномерно распределена по толщине стенки. При формировании же

толстостенной отливки температура как важнейший технологический параметр распределяется по толщине стенки неравномерно, и характер этого распределения изменяется во времени. Соответствующие математические модели относятся к классу моделей *с распределёнными* параметрами.

Особый класс математических моделей составляют *оптимизирующие* модели. В части 1 настоящей работы были рассмотрены модели статической оптимизации, основанные на применении *методов математического программирования* (главы 12, 13,14).

3.2. Общая методика построения математических моделей

Несмотря на огромное многообразие видов математических моделей, общая методика их построения укладывается в некоторую общую последовательность шагов (рис.3.3, с.76).

Соответственно *цели* моделирования выбирают математический *метод* построения модели. Например, большинство статических моделей базируется на использовании принципов *алгебры*. В то же время, основой динамических моделей являются *дифференциальные уравнения*, описывающие развитие переходных процессов *во времени*, а при моделировании объектов с распределёнными параметрами требуются дифференциальные уравнения *с частными производными,* позволяющими описать переходные процессы *в пространстве*.

Структурный синтез математической модели представляет собой составление уравнений и других математических выражений (например целевых функций, ограничений) в общем виде на основании привлечения знаний из конкретных наук таких как физика, химия, физическая химия, металлургическая теплотехника, теория металлургических и литейных процессов и пр., сообразно природе явлений в объекте. При этом, используя принципы системного подхода и системного анализа, исследователь стремится в структуре модели отобразить свойства объекта, существенные с точки зрения достижения цели моделирования.

Параметрическая идентификация заключается в опреде-

лении конкретных числовых значений коэффициентов, входящих в состав модели.

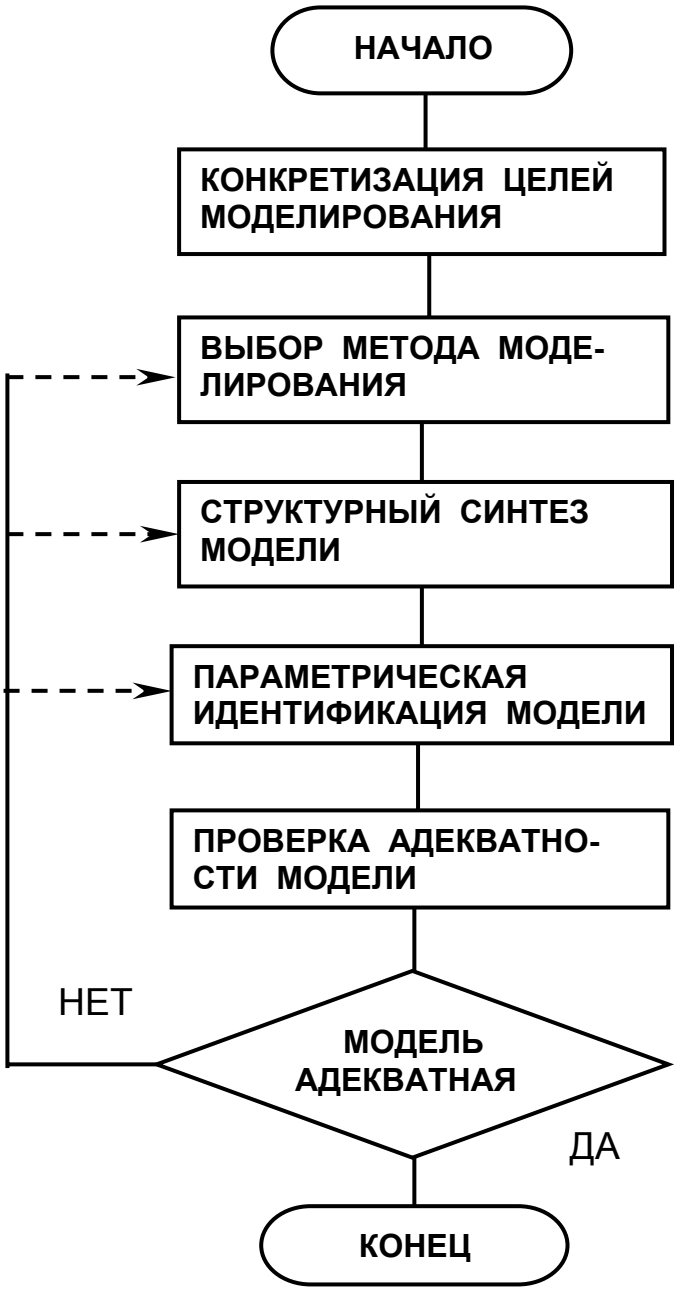

Рис. 3.3. Блок − схема алгоритма процесса построения математической модели

Заключительным этапом построения математической модели является проверка её *адекватности*, т. е. соответствия моделируемому объекту. Суждение об адекватности либо неадекватности модели (в пределах принимаемых в каждой конк-

ретной задаче критериев), вообще, основывается на опыте. В том числе, могут быть использованы результаты ранее проведенных исследований, приводимые в литературе, или данные контрольных измерений. Если реакции модели и моделируемой системы на одинаковые внешние воздействия существенно различаются, переходят согласно рис. 3.3 к другому методу моделирования: используют иной математический аппарат, уточняют структуру модели и её коэффициенты. После этого вновь проверяют модель на адекватность.

Ряд примеров построения математических моделей приведен в работе [14].

Адекватная модель правильно отображает свойства объекта как в качественном, так и в количественном отношении.

3.3. Принципы моделирования металлургических объектов с сосредоточенными параметрами

Для осуществления компьютерного моделирования, иногда называемого вычислительным экспериментом, исследователь должен располагать адекватной математической моделью, компьютером и подходящим его программным обеспечением. В одних случаях модель может иметь весьма специфический вид, характерный для достаточно узкого класса исследуемых объектов, причём с переходом от одного технологического процесса к другому вид модели принципиально изменяется. Здесь рационально использовать Visual Basic. Если же рассматривается некоторое множество объектов, характеризуемых моделями типовой структуры, различающимися только коэффициентами, то во многих случаях весьма полезным может оказаться Mathcad, в частности при решении обыкновенных дифференциальных уравнений.

3.3.1. Моделирование процесса аргонно – кислородного рафинирования стали

Известно, что в коррозионно-устойчивых (нержавеющих) сталях углерод является одной из наиболее вредных примесей,

снижающих стойкость металла против межкристаллитной коррозии. Для глубокого обезуглероживания одна только кислородная продувка недостаточна. Она вызывает значительный угар хрома из состава переплавляемого легированного стального лома и требуемого окисления углерода не обеспечивает.

Согласно методике Московского института стали и сплавов (технический университет) [15, 16], считают, что роль вводимого кислорода сводится к поддержанию окисленности металла на уровне равновесия с Cr_3O_4. При содержании последнего в шлаке более 9% процесс обезуглероживания описывается следующими уравнениями:

$$\frac{1}{4}(\boldsymbol{Cr_3O_4}) + [\boldsymbol{C}] = \frac{3}{4}[\boldsymbol{Cr}] + \{\boldsymbol{CO}\}; \qquad (3.2)$$

$$\lg \boldsymbol{K} = \lg \frac{[\boldsymbol{Cr}]^{3/4}\, \boldsymbol{P_{CO}}}{\boldsymbol{f_C}[\boldsymbol{C}]} = -\frac{11520}{T} + 7{,}64\ , \qquad (3.3)$$

где в квадратных, круглых и фигурных скобках – концентрации вещества в металле, шлаке и газовой фазе соответственно;
$\boldsymbol{f_C}$ – коэффициент активности углерода;
P_{C0} - парциальное давление CO;
$\boldsymbol{K}$ – константа химического равновесия для реакции (3.2);
T – абсолютная температура, К.
Коэффициент активности углерода, например для условия 18% Cr, 10% Ni, C< 0,1 % определяется выражением

$$\lg \boldsymbol{f}_C = [\mathrm{Cr}]e_C^{Cr} + [\mathrm{Ni}]e_C^{Ni} + [\mathrm{C}]e_C^{C}\ ,$$

в котором параметры взаимодействия первого порядка:

$$e_c^{Cr} = -0{,}0024; \quad e_c^{Ni} = 0{,}012; \quad e_c^{C} = 0{,}14,$$

а концентрации Cr, Ni, C в металле выражены в процентах по массе.

Если в расплав ввести некоторое количество аргона dV_{Ar}, $м^3/Т$, то концентрация углерода снизится за счёт перевода CO в аргонный пузырек на величину $-d[C]$, % по массе, а количество образовавшегося при этом CO составит, $м^3/Т$:

$$dV_{CO} = \frac{10 \cdot 22{,}414}{12} \cdot \left(-d[C]\right) = -18{,}678 \cdot d[C]. \qquad (3.4)$$

Здесь $d[C] < 0$, так как имеет место обезуглероживание стали.

Относительное парциальное давление CO в пузырьках газовой фазы, выходящих из ванны, будет

$$P_{CO} = \frac{dV_{CO}}{dV_{CO} + dV_{Ar}} = \frac{-18{,}678 d[C]}{dV_{Ar} - 18{,}678 d[C]} . \qquad (3.5)$$

Отношение фактического P_{CO} к равновесному из уравнения (3.2) примем равным $\alpha \le 1$. Тогда

$$P_{CO} = \frac{\alpha K f_C [C]}{[Cr]^{3/4}} \qquad (3.6)$$

Последнее выражение можно преобразовать к виду

$$\frac{d[C]}{dV_{Ar}} = \frac{[C]}{18{,}678\left([C] - \frac{[Cr]^{3/4}}{a K f_C}\right)} \qquad (3.7)$$

Для численного решения уравнения (3.7) примем достаточно малые шаги изменения функции [C], % и её аргумента V_{Ar} , м^3/Т

$$d[C] \rightarrow \Delta[C] = [C]^* - [C];$$

$$dV_{Ar} \rightarrow \Delta V_{Ar},$$

где [C] – концентрация углерода на момент начала ввода порции аргона в количестве ΔV_{Ar} , м^3/Т,

[C]* - концентрация углерода на момент завершения ввода данной порции аргона.

Примем расход аргона равным Q_{Ar} , м^3/(Т· мин). Тогда, м^3/Т

$$\Delta \boldsymbol{V}_{\text{Ar}} = Q_{\text{Ar}} \Delta \boldsymbol{t} ,$$

где $\Delta \boldsymbol{t}$ – шаг по времени, например равный 1 мин.

В результате получаем

$$[\boldsymbol{C}]^{*} = [\boldsymbol{C}] - \frac{Q_{\boldsymbol{Ar}} [\boldsymbol{C}] \Delta t}{18{,}678 \left(\boldsymbol{N} - [\boldsymbol{C}]\right)} , \qquad (3.8)$$

где

$$N = \frac{[\boldsymbol{Cr}]^{3/4}}{\alpha K f_{\text{C}}}.$$

Для поддержания постоянной концентрации Cr_3O_4 каждая тонна металла за всё время продувки теряет углерода от начального $[C]_{\text{Н}}$ до конечного $[C]_{\text{К}}$ содержаний, %, в количестве $1000([C]_{\text{Н}} - [C]_{\text{К}})/100$, кг. Это требует кислорода по реакции:

$$[C] + [O] = \{CO\} \qquad (3.9)$$

в количестве, кг/Т

$$1000 \cdot 16 \left([C]_{\text{Н}} - [C]_{\text{К}}\right) / (100 \cdot 12),$$

где 16 и 12 соответственно – атомные массы С и О, , кг/кмоль.

Отсюда потребный общий объём кислорода составляет

$$\boldsymbol{V}_{\text{O2}} = 1000 \cdot 22{,}414 \cdot 16 ([C]_{\text{Н}} - [C]_{\text{К}}) / (100 \cdot 12 \cdot 32), \qquad (3.10)$$

где 32 – молекулярная масса кислорода, кг/кмоль; 22,414 – объём одного киломоля газа при нормальных условиях, $м^3$. При общей длительности продувки $\boldsymbol{t}$, мин, требуемый расход кислорода будет $\boldsymbol{Q}_{\text{O2}} = \boldsymbol{V}_{\text{O2}} / \boldsymbol{t}$, $м^3$/ (Т· мин).

Соотношение расходов и общих объёмов кислорода и аргона:

$$R = \frac{Q_{\boldsymbol{Ar}}}{Q_{\boldsymbol{O_2}}} = \frac{V_{\text{Ar}}}{V_{\text{O2}}} . \qquad (3.11)$$

Кроме режима окисления углерода, необходимо определить параметры температурного режима плавки.

Обозначим T и T^*, К абсолютные температуры металла на моменты начала и окончания шага по времени Δt, соответственно. Тогда для выявления скорости изменения температуры ванны необходимо составить уравнение теплового баланса в расчёте на 1 Т металла за шаг по времени. Для этого имеем, кДж / Т:

− приход теплоты

$$\frac{1000}{100} \cdot \overline{Q}_C \cdot \Delta C, \qquad (3.12)$$

где $\overline{Q}_C$ − тепловой эффект реакции (3.1), в расчёте на 1 кг С, кДж / кг;
ΔC − изменение концентрации углерода, %;
1000 − коэффициент приведения данных к 1 Т металла, кг / Т;

− расход теплоты

а) на изменение теплосодержания металла

$$Q_1 = 1000\, c_{ME}\, (T^* - T), \qquad (3.13)$$

где c_{Me} − удельная теплоёмкость металла, кДж / (кг· К);

б) на нагрев аргона

$$Q_2 = c_{Ar}\, Q_{Ar}\, (T - T_H)\, \Delta t, \qquad (3.14)$$

где c_{Ar} − удельная теплоёмкость аргона, кДж / ($м^3$ · К);
Q_{Ar} − удельный расход аргона, м3 / (Т · мин);
T_H = 293 К − нормальная температура;
Δt − шаг по времени, мин;

в) на нагрев кислорода

$$Q_3 = c_{O2}\, Q_{O2}\, (T - T_H)\, \Delta t, \qquad (3.15)$$

где c_{O2} – удельная теплоёмкость кислорода, кДж / (м^3 · К);
Q_{O2} – удельный расход кислорода, м^3 / (Т · мин).

Приравнивая приход теплоты её расходу, получаем

$$10\, \overline{Q}_C\, \Delta C = 1000\, c_{ME}\, (T^* - T) + c_{Ar}\, Q_{Ar}\, (T - T_H)\, \Delta t + c_{O2}\, Q_{O2}\, (T - T_H)\, \Delta t,$$

или

$$\frac{\Delta T}{\Delta t} = \frac{T^* - T}{\Delta t} = 0{,}01\, \overline{Q}_C \frac{\Delta C}{\Delta t} - (T - T_H) \cdot (c_{Ar} Q_{Ar} + c_{O2} Q_{O2}) \cdot 10^{-3} / c_{Me}, \qquad (3.16)$$

Последнее выражение является численным аналогом дифференциального уравнения

$$\frac{dT}{dt} = 0.01\, \overline{Q}_C \frac{dC}{dt} - (T - T_н) \cdot (c_{Ar} Q_{Ar} + c_{O2} Q_{O2}) \cdot 10^{-3} / c_{Me}.$$

Для жидкой стали можно принять c_{Me} = 0,75 кДж / (кг· К), а тепловой эффект $\overline{Q}_C$ = 11200 кДж / кг углерода.

Уравнения (3.7), (3.8), (3.16) вместе с входящими в них величинами являются основой компьютерного моделирования процесса аргонно – кислородного рафинирования стали.

На основании этих уравнений с учётом промежуточных переменных автором разработан алгоритм численного моделирования процесса кислородно - аргонного обезуглероживания стали (рис.3.4, с. 83).

Этот алгоритм реализован в среде Visual Basic. В составе фонда алгоритмов и программ кафедры металлургии и литейного производства СЗТУ соответствующему VB проекту присвоено имя P9VB.

Проект P9VB содержит три формы. Первая из них предназначена для ввода исходных данных: начального C0 и конеч-

ного CF содержаний углерода в металле, а также хрома и никеля, начальной температуры и принятого расхода аргона. Вторая форма служит для вывода результатов счёта: продолжительности продувки, количеств израсходованных аргона и кислорода, конечной температуры металла. С помощью третьей формы отображается график выгорания углерода во времени.

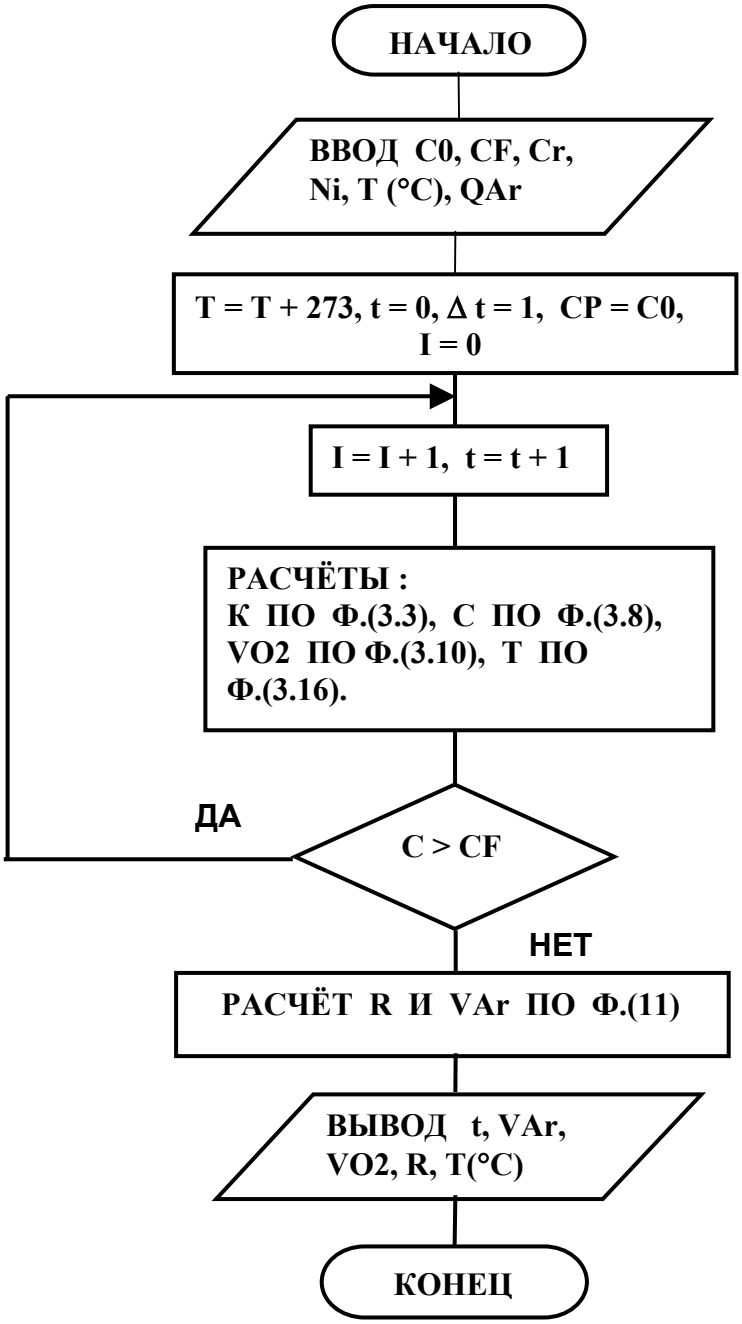

Рис. 3.4. Блок − схема моделирования процесса окисления углерода

3.3.2. Моделирование и оптимизация процесса оборота литейных материалов

Литейное производство как мощная заготовительная база современного машиностроения характеризуется широким использованием оборотных материалов. Так, в процесс плавки вовлекаются литники, прибыли, скрап, металл, остающийся в каналах индукционных печей канального типа и «болоте», если с таковым ведётся плавильный процесс в печах любого типа. В массовых масштабах используются оборотные формовочные смеси (рис. 3.5, с.85).

Поскольку на различных участках производства происходят потери материалов, к оборотным продуктам добавляют свежие. При этом количество освежающих добавок не должно быть меньше потерь.

При равенстве добавки сумме всех видов потерь имеет место процесс кругооборота, установившийся *по балансу материалов*. Однако, в то же время, весьма вероятен процесс, неустановившийся *по содержанию вредных примесей* в оборотных продуктах. К таким примесям относятся, например, железо в силуминах, шамотизированная глина в составе песчано - глинистых или Na_2O в песчано - жидкостекольных формовочных смесях.

Рассмотрим в качестве примера оборотную песчано-жидкостекольную смесь. Вредной примесью в ней является остаточное после регенерации содержание Na_2O из состава жидкого стекла, используемого в качестве связующего материала. Эта примесь, обволакивая зёрна кварцевого песка, снижает прочность связи между ними, а химическое взаимодействие Na_2O с кремнезёмом отрицательно влияет на огнеупорность и термическую устойчивоть формовочной смеси.

Предельно допустимым содержанием Na_2O в оборотной формовочной смеси считают ~ 2,5% по массе, тогда как в исходном кварцевом песке содержится 0,5…0,7 % Na_2O, а в стандартном жидком стекле − от 11, 8 до 14,6 %.

Количество жидкого стекла обычно составляет 9...13 процентов по массе.

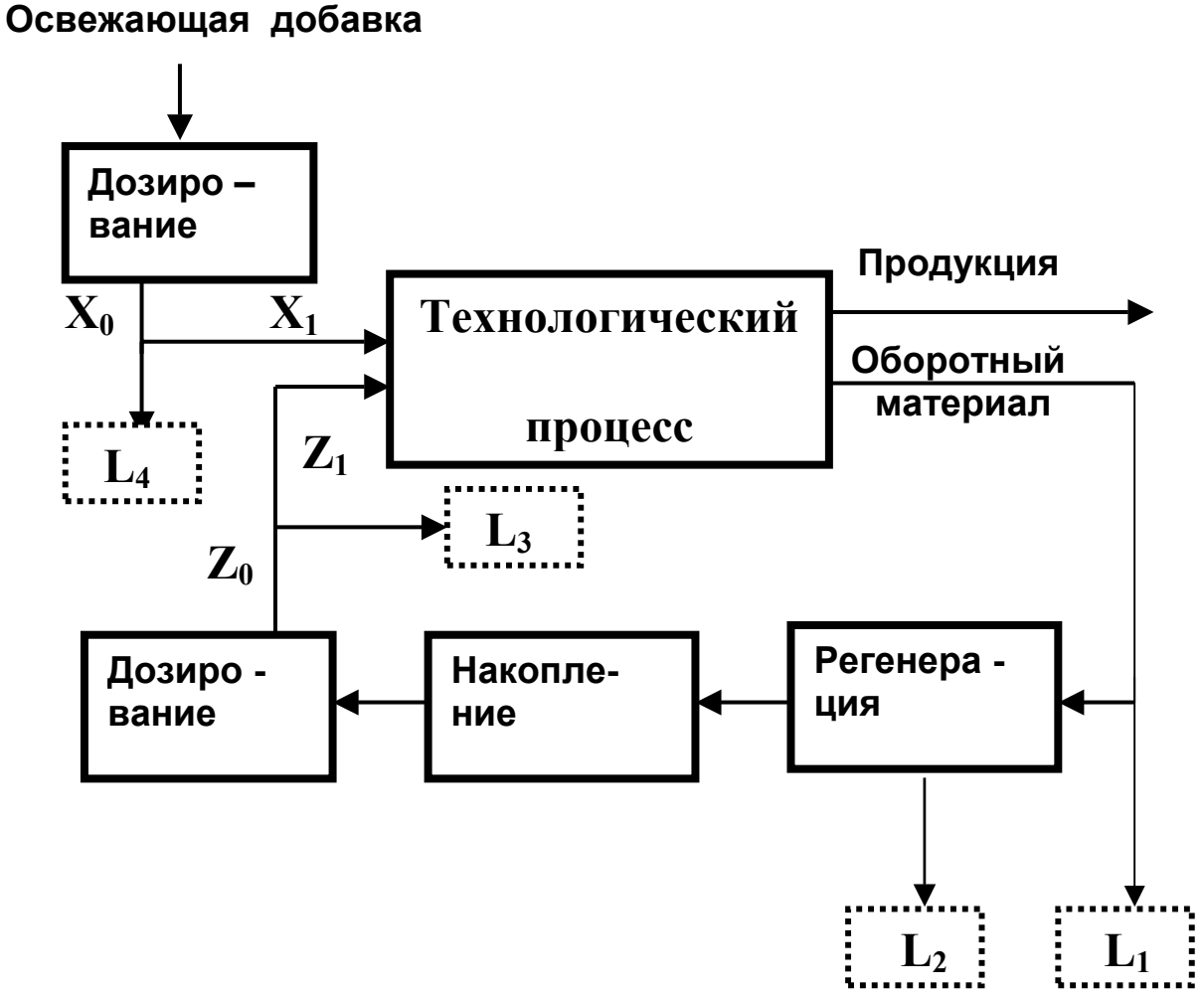

Рис. 3.5. Схема процесса оборота литейных материалов. Обозначения : **L_1, L_2, L_3, L_4** – потери

Установлено [17 – 20], что текущее содержание Na_2O на N − м цикле оборота выражается формулой:

$$Y_N = \frac{X \cdot p + (100 - X) \cdot Y_{N-1}}{100 + G} + \frac{q \cdot G}{100 + G}, \qquad (3.17)$$

где $\boldsymbol{X}$ – концентрация добавляемого свежего кварцевого песка в составе сухой формовочной смеси, возвращаемой в рецикл;

G – концентрация вводимого жидкого стекла в готовой формовочной смеси;

q – концентрация Na_2O в кварцевом песке;

p – концентрация Na_2O в жидком стекле.

Установлено также, что с возрастанием ***N*** значение Y_N стремится к определенному пределу, который в процентах по массе составляет:

$$Y_N = \frac{X \cdot p + q \cdot G}{X + G}. \qquad (3.18)$$

При этом после *N* циклов оборота материалов остаток *R* первой порции кварцевого песка в смеси (в процентах по массе) не превышает

$$R = \left(\frac{100 - X}{100}\right)^N \cdot (100 - X). \qquad (3.19)$$

Можно задаться некоторым малым значением остатка, например $R \le 1\%$. После прохождения соответствующего этому числа циклов $N = f(R)$ остаток первой порции освежающей добавки становится столь малым, что даже возможная последующая деградация её свойств уже не способна повлиять на свойства оборотного продукта в целом. Скорость уменьшения рассматриваемого остатка в рецикле возрастает с ростом степени освежения ***X***. Однако оптимальное значение добавки свежего кварцевого песка призвано обеспечить наиболее экономичное его расходование при сохранении качества отливок и соответственно – снижении себестоимости их изготовления.

Блок – схемой итерационной процедуры моделирования процесса изменения содержания вредной примеси в оборотном продукте (рис. 3.6, с. 89) предусмотрено использование двух видов ограничений: предельно допустимого числа циклов и максимально допустимого содержания вредной примеси. Эти ограничения определяют условие останова действия программы с выводом полученных результатов. Основные черты этого алгоритма положены в основу проекта P10VB, разработанного автором для использования в среде Visual Basic.

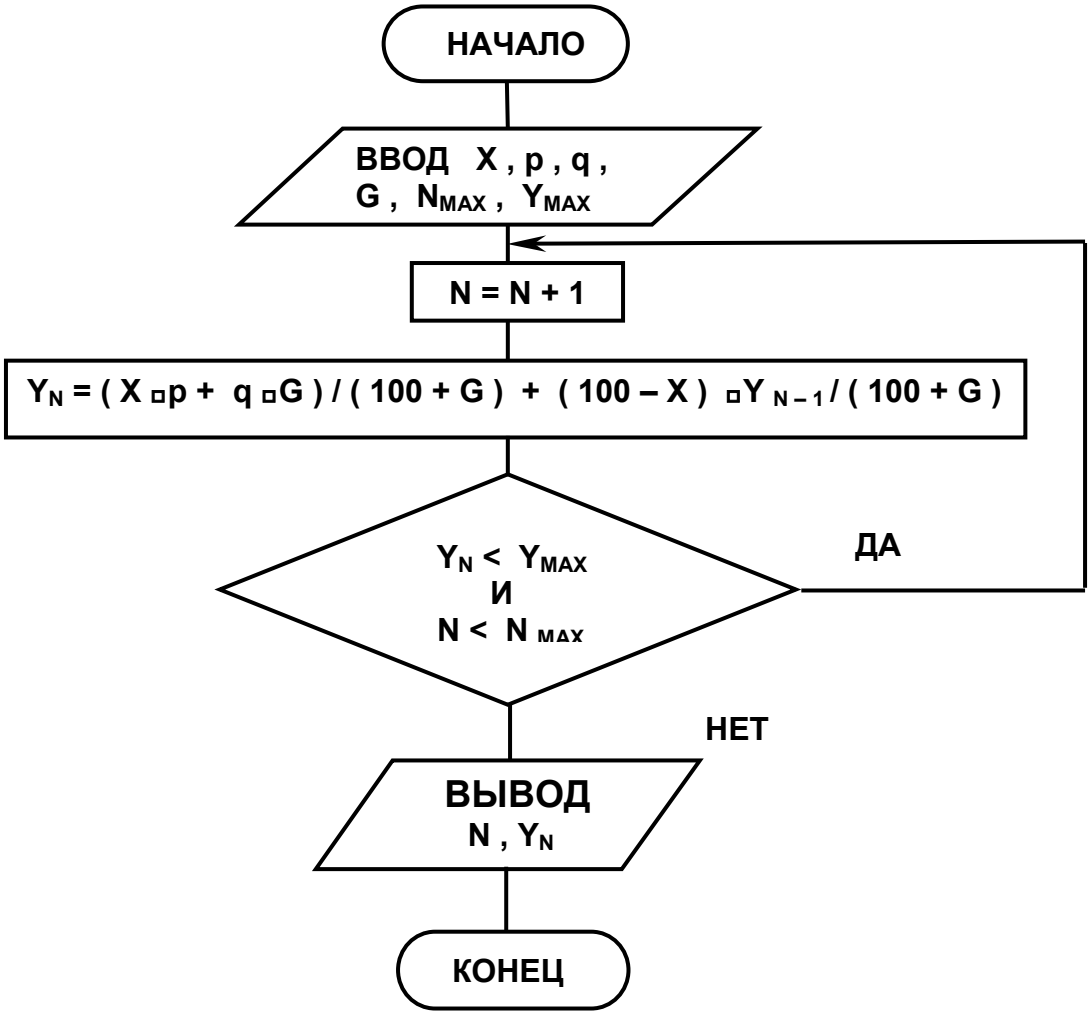

Рис. 3.6. Блок – схема алгоритма моделирования процесса накопления вредной примеси по ходу рецикла

При дозировании материалов имеют место случайные *погрешности* и *потери.*

Случайные погрешности дозирования материалов складываются из *статических* и *динамических*. В свою очередь, статические погрешности представляют собой *погрешности измерения* доз оборотных материалов и освежающих добавок. Наиболее совершенные – весовые дозаторы базируются на использовании тензометрических и магнитоанизотропных измерительных датчиков [12, с. 34 ... 37], *предельная* погрешность которых $\Delta_{1пред} \le \pm 1{,}5\%$. Такова же предельная погрешность (согласно классу точности) вторичных приборов $\Delta_{2пред}$, работающих в комплекте с этими датчиками.

Как известно, случайные погрешности измерения часто бывают распределены по нормальному закону (Гаусса):

$$P(\Delta_{1,2}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\Delta_{1,2}^2/(2\sigma^2)\right], \qquad (3.20)$$

где σ – средняя квадратическая погрешность, % ; $P(\Delta_{1,2})$ – плотность вероятности распределения *текущих* пог -решностей Δ_1, Δ_2.

При этом за *предельные* погрешности измерения принимают значения $\Delta_{1\,пред}$, $\Delta_{2\,пред}$, равные 3σ (правило “трёх сигм“).

Динамическая погрешность Δ_3 дозы материала («недовес» или «перевес») обусловлена несвоевременностью ввода и исполнения команды на отсечку его подачи и зависит от применяемого весодозирующего оборудования. С необходимым запасом точности можно принять $\Delta_{3пред} \le 2\Delta_{1,2\,пред}$.

На основании закона сложения погрешностей [21, с. 29] суммарная предельная погрешность дозирования каждого из рассматриваемых материалов составляет

$$\Delta_{д\,пред} = \pm\sqrt{\Delta_{1пред}^2 + \Delta_{2пред}^2 + \Delta_{3пред}^2} = \pm\sqrt{1{,}5^2 + 1{,}5^2 + 3^2} = \pm 3{,}7 \ \%\ .$$

Что касается учета случайных потерь материалов $\Delta_n, \%$ на пути от пункта дозирования до приемника (например смесителя на участке смесеприготовления литейного цеха), то они целиком зависят от местных условий производства. При моделировании можно принять эти потери равномерно распределенными в интервале от 0 до их предельного значения $\Delta_{n\,пред}$.

Таким образом, в уточнённый расчет вместо заданных на дозаторах значений X_0 и $Z_0 = 100 - X_0$ с учетом погрешностей и потерь фактически вводятся величины соответственно

$$X_1 = X_0\left(1 \pm \frac{\Delta_{д1}}{100} - \frac{\Delta_{n1}}{100}\right); \qquad (3.21)$$

$$Z_1 = (100 - X_0)(1 \pm \frac{\Delta_{\partial 2}}{100} - \frac{\Delta_{n2}}{100}), \qquad (3.22)$$

где - $\Delta_{\partial 1}$ – погрешность дозирования освежающей добавки, %;

$\Delta_{\partial 2}$ – погрешность дозирования оборотного материала, %;

Δ_{n1} – потери освежающей добавки, %;

Δ_{n2} – потери оборотного материала, %.

Величины $\Delta_{\partial 1}$, $\Delta_{\partial 2}$, Δ_{n1}, Δ_{n2} в процессе имитационного моделирования определяются с использованием аппарата случайных чисел и независимо друг от друга, а фактическое значение степени освежения составляет, % :

$$X_2 = \frac{X_1}{Z_1} 100 . \qquad (3.23)$$

Учёт погрешностей дозирования компонентов и потерь предусмотрен следующим вариантом алгоритма моделирования процесса нарастания вредной примеси в оборотном продукте, отличающимся от рассмотренного выше ещё и тем, что ограничение по предельно допустимому числу циклов N_{max} здесь снято, так как оно не всегда априорно известно. Вместо этого пользователь в информационном окне на экране монитора получает сообщение о фактически достигнутом числе циклов N, после которого остаток R начальной порции освежающей добавки становится пренебрежимо малым. Если это число вызывает сомнение технолога на основании его предшествующего опыта или интуиции, то следует увеличить значение X. Разумеется, адекватность результата любого моделирования тре-бует проверки соответствующим экспериментом в конкретных производственных условиях. Однако сам процесс моделирова-ния, как правило, позволяет сократить число потребных опытов и, тем самым, снизить затраты на их постановку.

Поэтому для среды Visual Basic создан проект P10MVB, в котором случайные погрешности и потери материалов обрабатываются с использованием встроенного аппарата случайных

чисел. В результате получаем возможность поста-новки *имитационного моделирования*.

Результаты имитационного моделирования приближаются к реальным производственным данным и могут быть применены на практике. Его использование обеспечивает рациональное расходование освежающих добавок. Возникает возможность достижения требуемого состава оборотных материалов на каждом цикле. Стимулируются мероприятия по стабилизации и повышению качества отливок, а также – снижению себестоимости их изготовления.

Разработан также проект P11VB непосредственной оптимизации меры освежающей добавки к оборотной смеси материалов (рис. 3.7 ***a*** и ***b***, с. 91, 92). Здесь итерационная процедура осуществляется от начального приближения $\boldsymbol{X} = \boldsymbol{X}_{НАЧ}$ с последующим уменьшением шага $\Delta\boldsymbol{X}$ изменения величины $\boldsymbol{X}$ в процессе поиска оптимума $\boldsymbol{X} = \boldsymbol{X}_{опт}$ до достижения заданной точности ε. Итогом работы программы служит вывод значения оптимальной освежающей добавки $\boldsymbol{X}_{опт}$, соответствующей заданным ограничениям $\boldsymbol{N}_{max}$ и $\boldsymbol{Y}_{max}$.

Таким образом, работы по моделированию и оптимизации процесса оборота литейных материалов могут быть выполнены в любом из возможных вариантов (табл.3.1, с. 93). Необходимые для этого компьютерные программы (VB – проекты) имеются в фонде алгоритмов и программ кафедры металлургии и литейного производства СЗТУ. После загрузки каждого из этих проектов запуск его в работу осуществляется нажатием на кнопку с изображением [►] в составе главного меню. От пользователя требуется ввод исходных данных в окна форм, а затем – нажатие на кнопку [ПУСК]. Численные результаты счёта отображаются в окнах вывода на тех же формах, а в вариантах 1, 2 при нажатии на кнопку [ГРАФИК] высвечивается форма с графиком нарастания содержания Na_2O с каждым циклом оборота формовочной смеси.

При необходимости повторения счёта для другого набора исходных данных нажимают на кпопку [ВОЗВРАТ], изменяют данные в окнах ввода и снова запускают задачу на счёт нажатием на кнопку [ПУСК].

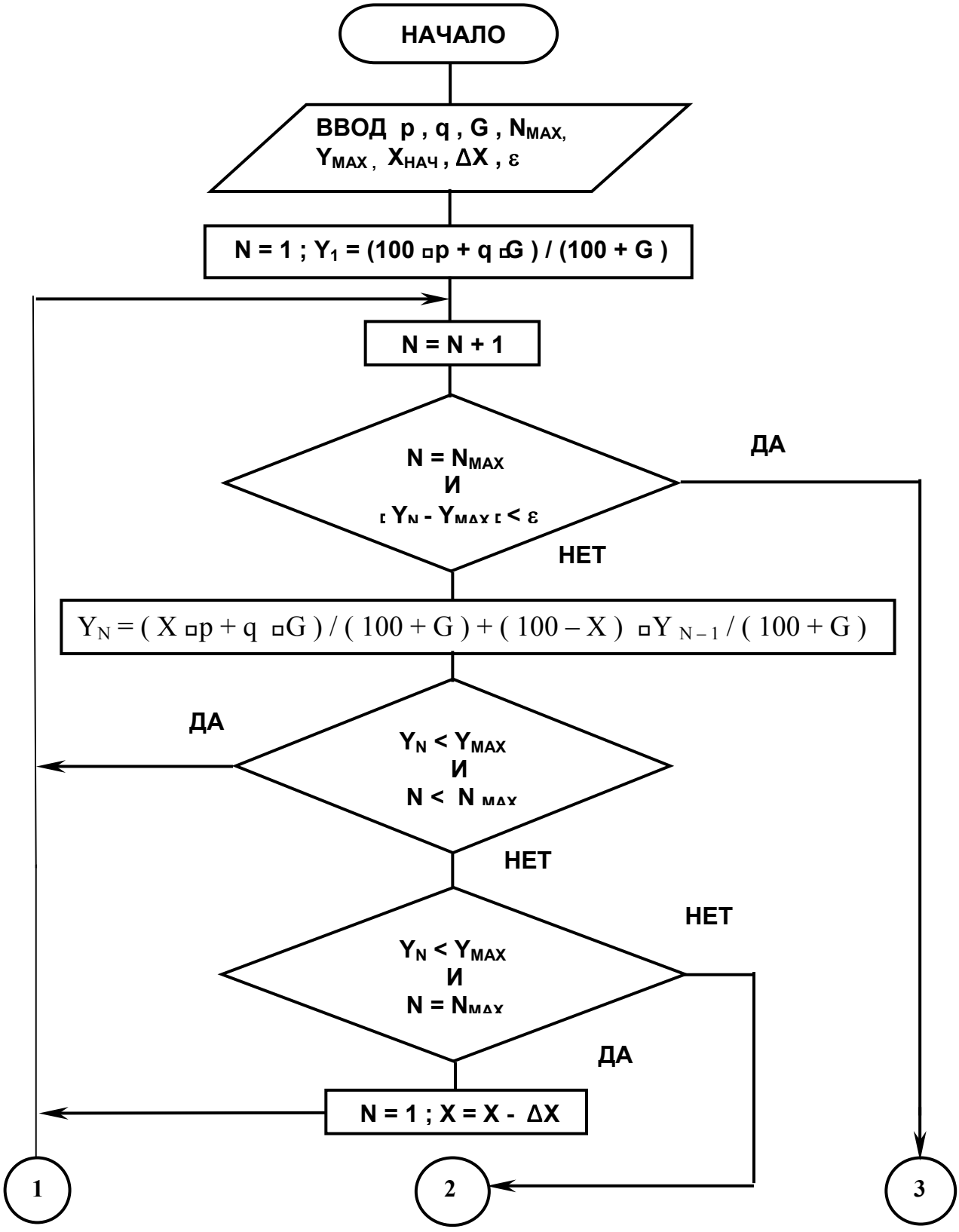

Рис. 3.7*а*. Блок – схема ядра алгоритма оптимизации освежающей добавки к оборотному продукту

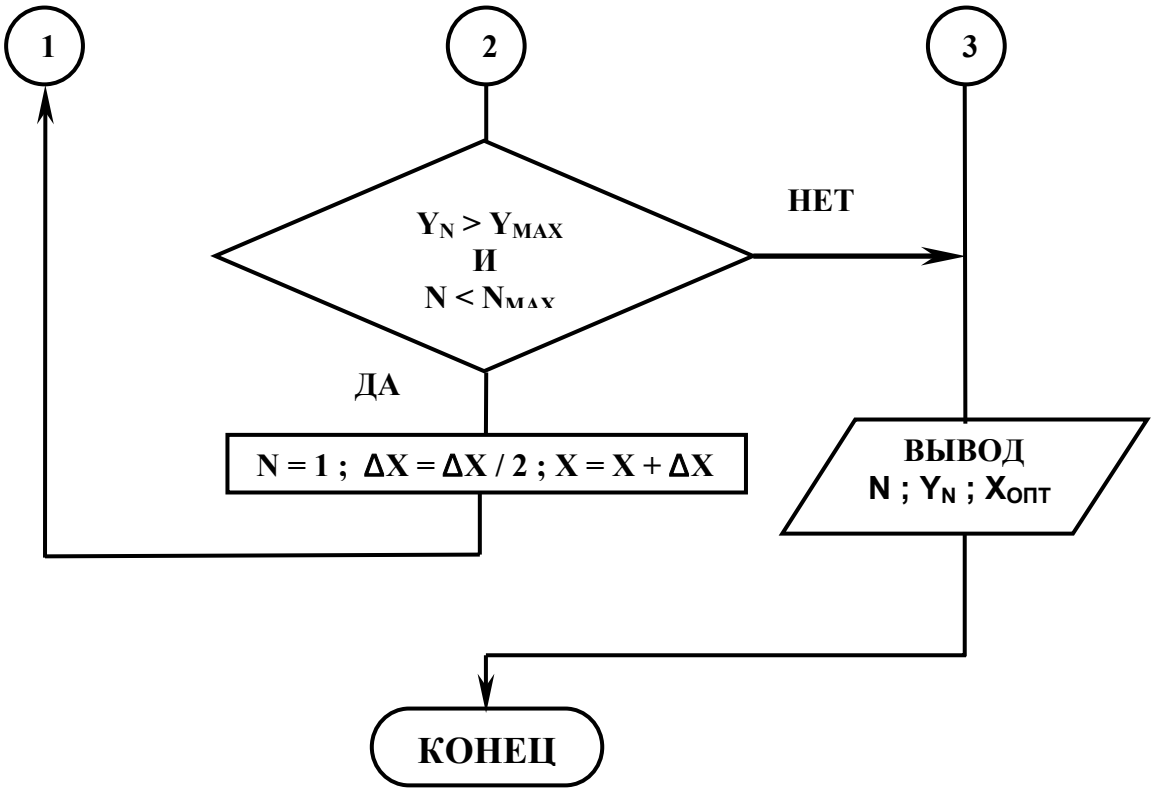

Рис. 3.7***b***. Условие окончания итерационной процедуры поиска оптимума к блок – схеме алгоритма, показанного на рис. 3.7***a*** (обозначения согласно ГОСТ 19.701 – 90)

3.3.3. Моделирование переходных процессов в системах автоматического регулирования металлургических процессов

Металлургические процессы, как правило, характеризуются высокой сложностью своего математического описания. Тем не менее с достаточной для цели автоматического регулирования точностью математические модели технологических объектов управления (ТОУ) в металлургии во многих случаях удаётся аппроксимировать обыкновенными дифференциальными уравнениями. Порядок этих уравнений и значения их коэффициентов обычно устанавливаются экспериментально [14], [22] либо путём нанесения пробных входных воздействий с последующей математической обработкой отклика объекта, либо методами статистической динамики с использованием, например, уравнения Винера – Хопфа. Так или иначе, дифференциальные уравнения, ап-

Таблица 3.1. **Варианты работы по моделированию и оптимизации оборота регенерируемых формовочных смесей**

Варианты	Тема	Программа	Данные ввода	Результаты счёта
1	Исследование процесса накопления Na_2O в оборотной песчано−жидкостекольной смеси при ограничениях по предельному числу циклов и максимально допустимому содержанию Na_2O в формовочной смеси.	P10VB	Количество свежего кварцевого песка, содержание в нём Na_2O, количество жидкого стекла, содержание в нём Na_2O, предельно допустимые значения числа циклов и содержания Na_2O в формовочной смеси.	Фактически достижимое число циклов при предельном содержании Na_2O в смеси, график роста Na_2O в оборотной смеси по циклам оборота.
2	То же − при ограничении по предельному содержанию Na_2O и с учётом погрешностей дозирования и потерь материалов.	P10MVB	То же с дополнительным заданием предельных погрешностей дозирования и потерь материалов.	То же
3	Оптимизация освежающей добавки кварцевого песка с ограничениями по числу циклов и максимально допустимому содержанию Na_2O в формовочной смеси.	P11VB	Содержание Na_2O в кварцевом песке и жид ком стекле, количество жидкого стекла, предельно допустимые значения числа циклов и содержания Na_2O в смеси.	Оптимальное значение добавки свежего кварцевого песка к оборотной формовочной смеси.

проксимирующие реальные процессы плавки, перемешивания материалов, ряда процессов литья, различаются лишь своим порядком и численными значениями коэффициентов. Это даёт возможность во многих случаях осуществлять моделирование переходных процессов в системах автоматического регулирования таких параметров, как температура в нагревательных и плавильных печах, давление жидких и газообразных сред, уровень расплава в кристаллизаторе машины непрерывного литья и пр., с помощью Mathcad, причём основное средство моделирования представляется в виде возможностей оперативного решения названных дифференциальных уравнений при различных начальных условиях.

Предполагается знакомство читателя с основами автоматики [12], [14], [23], [24].

В составе *системы автоматического регулирования* (САР) – рис. 3.8, с. 95 – различают такие звенья, как объект (ТОУ) и регулятор со следующими их ***входами*** (входными воздействиями) и ***выходами*** (откликами на входные воздействия):

$\boldsymbol{y}_З$ – заданное значение выхода объекта;

$\boldsymbol{y}$ – фактическое значение выхода объекта;

$\boldsymbol{x}_Р$ – вход регулятора:

$$\boldsymbol{x}_Р = \boldsymbol{y}_З - \boldsymbol{y}; \qquad (3.24)$$

$\boldsymbol{x}$ – управляемый вход объекта:

$$\boldsymbol{x} = \boldsymbol{y}_Р + \boldsymbol{z}; \qquad (3.25)$$

где $\boldsymbol{y}_Р$ – выход регулятора;

$\boldsymbol{z}$ – возмущение.

Заметим, что независимо от конструктивных особенностей любой регулятор реагирует на *отклонение* выхода объекта от своего заданного по условиям технологии значения в соответствии с уравнением (3.24). При этом само отклонение является движущей силой процесса автоматического регулирования. В ходе такого процесса регулятор своим выходом воздействует на вход объекта так, чтобы компенсировать действие возмуще-

ния, то есть различного рода помех, вызывающих нежелательные отклонения выхода объекта от нормального по условиям ведения технологии режима. К числу возмущений относятся, например, колебания свойств сырых материалов, напряжения в сетях электропитания, давления сжатого воздуха. Если выходом (главным технологическим параметром) плавильной печи является температура рабочего пространства, то её отклонения регулятор температуры стремится компенсировать соответствующим изменением подводимой электрической мощности или – расхода топлива.

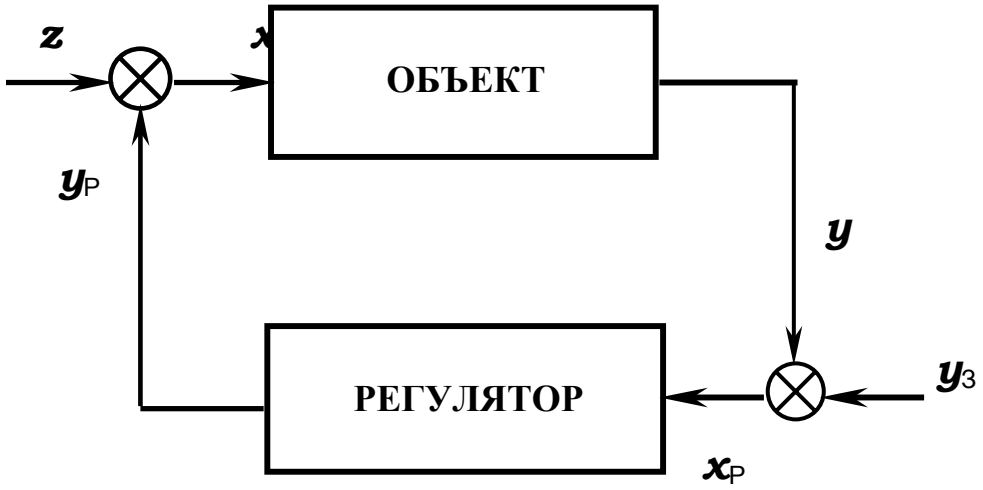

Рис. 3.8. Блок – схема САР

Примем, например, математические модели объектов второго порядка в двух вариантах (табл. 3.2).

Таблица 3.2. **ТОУ и их математические модели**

Объект	Модель
статический	$A_2 \frac{d^2 \boldsymbol{y}}{d t^2} + A_1 \frac{d \boldsymbol{y}}{d t} + \boldsymbol{y} = K_0 \boldsymbol{x}$
астатический	$A \frac{d^2 \boldsymbol{y}}{d t^2} + \frac{d \boldsymbol{y}}{d t} = \overline{K}_0 \boldsymbol{x}$

Обозначения: K_0 – передаточный коэффициент объекта; $\bar{K}_0$ – условный передаточный коэффициент объекта; A, A_1, A_2 – коэффициенты.

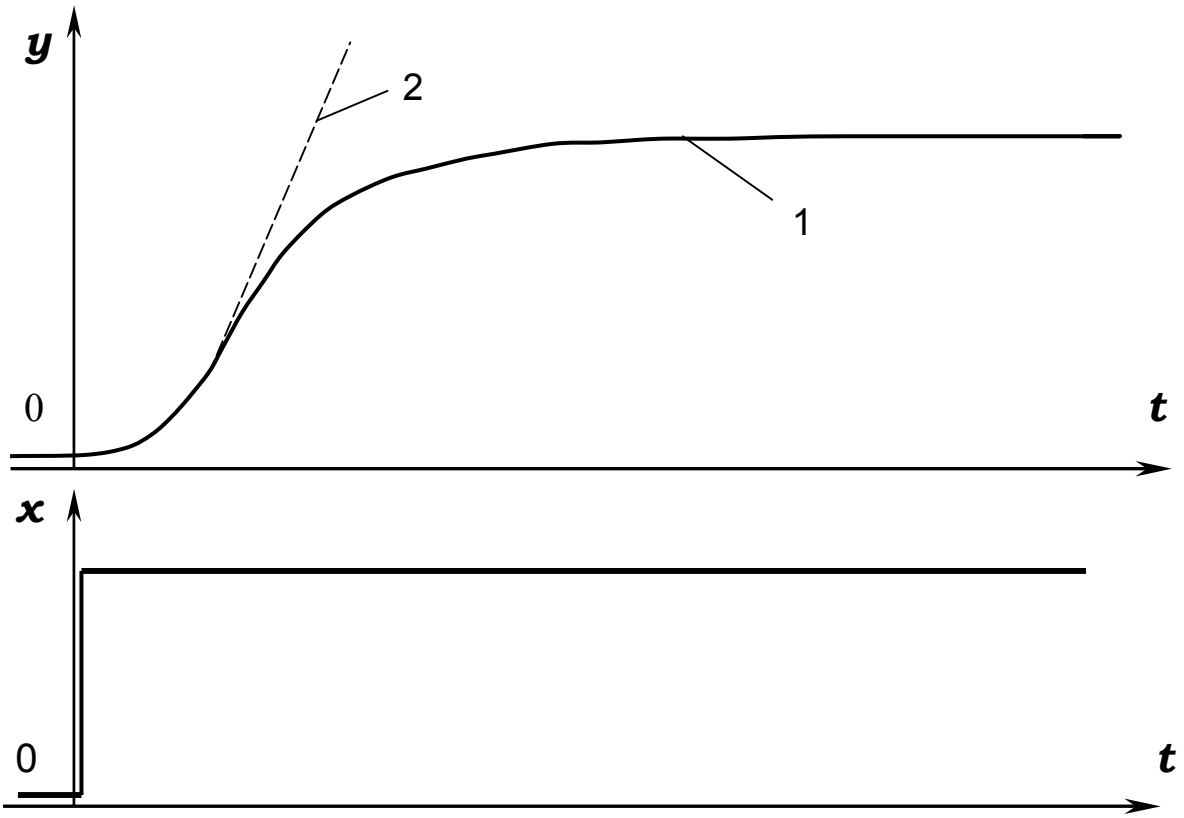

Рис. 3.9. Кривые разгона объектов второго порядка

Соответствующие кривые разгона объектов (1 – статического и 2 – астатического) как реакции на ступенчатое изменение входа представлены на рис.3.9.

В качестве алгоритмов (законов) регулирования, то есть выражений, связывающих входы и выходы регулятора, примем линейные, ставшие стандартными, варианты (табл. 3.3). Располагая приведенными данными, нетрудно составить дифференциальное уравнение САР, принять значения отдельных параметров, решить это уравнение и построить график переходного процесса. Варьируя тот или иной параметр настройки регулятора ($K_Р$, $\bar{T}_И$, $\bar{T}_Д$), можно проанализировать характер влияния этого параметра на форму кривой переходного процесса и соответственно оценить устойчивость САР и качество регулирования

(точность выполнения задания и степень колебательности величины y во времени t).

Условными здесь называют величины, имеющие *смешанную размерность.* Так, в размерности условного передаточного коэффициента объекта, кроме обычно требующихся единиц измерения его входа и выхода, содержатся также единицы времени.

Таблица 3.3. **Линейные алгоритмы регулирования**

Алгоритм	Формула
пропорциональный (П)	$y_P = K_P\, x_P$
интегральный (И)	$y_P = \frac{1}{\overline{T}_И} \int_0^t x_P\, d\,t$
пропорционально - интегральный (ПИ)	$y_P = K_P\, x_P + \frac{1}{\overline{T}_И} \int_0^t x_P\, d\,t$
пропорционально - дифференциальный (ПД)	$y_P = K_P\, x_P + \overline{T}_Д \frac{d\,x_P}{d\,t}$
пропорционально - интегрально - дифференциальный (ПИД)	$y_P = K_P\, x_P + \frac{1}{\overline{T}_И} \int_0^t x_P\, d\,t + \overline{T}_Д \frac{d\,x_P}{d\,t}$

В табл. 3.2 приняты следующие условные обозначения: $K_Р$ – передаточный коэффициент регулятора, $\overline{T}_И$ – условная постоянная времени интегрирования, $\overline{T}_Д$ – условная постоянная времени дифференцирования.

Что касается параметров объектов вообще, то с точки зрения автоматизации они представляются некоторой данностью, и для каждого вида объекта они определяются конструкцией объекта, режимом его работы и особенностями протекающего в нём технологического процесса.

Пример 3.1

Дано:

САР в составе статического объекта (табл. 3.2) и П – регулятора (табл. 3.3) при следующих значениях параметров:

$A_1 = 10$; $A_2 = 75$; $K_0 = 1$; $K_P = 10$.

Целесообразно отметить, что соотношение $A_1^2 \geq 4\,A_2$ придаёт объекту апериодические (неколебательные) свойства, что характерно для объектов тепловой природы, связанных с процессами нагрева или охлаждения тел.

Найти:

Переходный процесс в системе в двух вариантах:
А) при выводе объекта на задание ($\boldsymbol{y}_З = 50$, $\boldsymbol{z} = 0$);
В) при действии возмущения ($\boldsymbol{y}_З = 50$, $\boldsymbol{z} = -100$).

Решение

Составляем уравнение САР с учётом выражений (3.24) и (3.25):

$$A_2\,\boldsymbol{y}'' + A_1\,\boldsymbol{y}' + \boldsymbol{y} = \boldsymbol{K_0}\,(\boldsymbol{y_p} + \boldsymbol{z}) = K_0\,(K_P\,(\boldsymbol{y_З} - \boldsymbol{y}) + \boldsymbol{z}),$$

или

$$\frac{A_2}{1 + K_0\,K_P}\,\boldsymbol{y}'' + \frac{A_1}{1 + K_0\,K_P}\,\boldsymbol{y}' + \boldsymbol{y} = \frac{K_0\,K_P}{1 + K_0\,K_P}\,\boldsymbol{y_З} + \frac{K_0}{1 + K_0\,K_P}\,\boldsymbol{z}\ . \qquad (3.26)$$

Вызываем Mathcad и на его рабочем поле записываем условия задачи согласно варианту А и в соответствии с материалами Главы 1 (см. пример 1.16, с. 33), не забывая о “жирных” знаках равенства. Принимаем нулевые начальные условия для величины $\boldsymbol{y}$ и её первой производной $\boldsymbol{y}'$ по времени, изменяющемуся в диапазоне $0 \leq \boldsymbol{t} \leq 100$.

После запуска на решение получаем результат (рис. 3.10).

Замечаем, что при заданных настройках имеет место колебательный затухающий переходный процесс. Задание выполнено не точно, что для П – регулятора согласуется с теорией автоматического регулирования.

Увеличивая значение передаточного коэффициента регулятора K_P, читатель может наблюдать повышение точности выполения задания $\boldsymbol{y} \to \boldsymbol{y}_З$, однако при этом увеличится степень колебательности переходного процесса, а в системах более высокого порядка, чем второй, это может привести к потере устойчивости системы.

Для реализации варианта В той же задачи примем лишь следующие изменения: $\boldsymbol{y}(0) = 50$; $\boldsymbol{z} = -100$. Остальные данные те же, что в варианте А.

Полученный результат (рис. 3.11) свидетельствует о том, что задание и в этом случае в точности не выполняется ($\boldsymbol{y} < \boldsymbol{y}_З$) и в конце переходного процесса остаётся *статическая ошибка.* Численное значение этой ошибки можно определить из выражения (3.26), приравняв к нулю все производные в левой части, что будет соответствовать состоянию равновесия системы при $\boldsymbol{y}$ = const:

$$\boldsymbol{y} = \frac{K_O K_P}{1 + K_O K_P} \boldsymbol{y}_З + \frac{K_O}{1 + K_O K_P} \boldsymbol{z} \neq \boldsymbol{y}_З .$$

Пример 3.2

Дано:

САР в составе того же объекта, который был рассмотрен в предыдущем примере, и ПИ – реулятора (табл. 3.3).

Найти:

переходный процесс по выводу объекта на задание и под действием возмущения.

$Ko := 1 \qquad Kp := 10 \qquad A1 := 10 \qquad A2 := 75 \qquad y3 := 50$

Given

$$\frac{A2}{1 + Ko \cdot Kp} \frac{d^2}{dt^2} y(t) + \frac{A1}{1 + Ko \cdot Kp} \frac{d}{dt} y(t) + y(t) = Ko \cdot \frac{Kp}{1 + Ko \cdot Kp} \cdot y3$$

$y(0) = 0 \qquad y'(0) = 0$

$y := Odesolve(t, 100)$

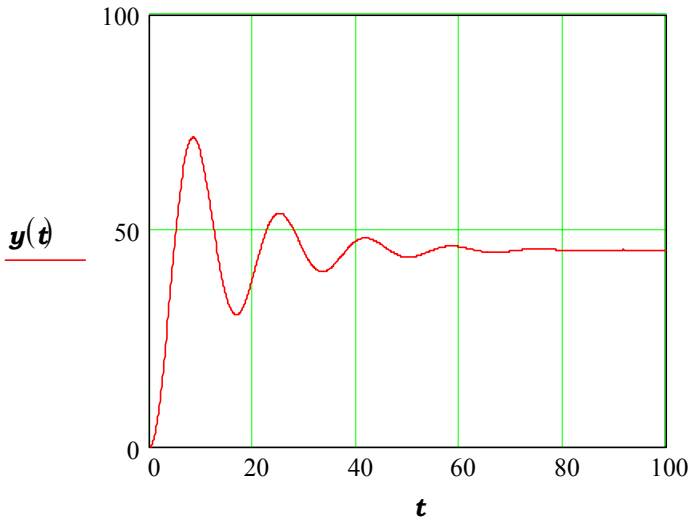

Рис. 3.10. Моделирование переходного процесса вывода объекта на задание

Решение

Теми же приёмами, что и в предыдущем примере, составим дифференциальное уравнение САР для последующего его решения с помощью Mathcad:

$$A_2\,\boldsymbol{y}'' + A_1\,\boldsymbol{y}' + \boldsymbol{y} = \boldsymbol{K_O}\left((K_P\,x_P + \frac{1}{\overline{T}_И}\int_0^t x_P\,d\,t) + \boldsymbol{z}\right),$$

или

$$A_2\,\boldsymbol{y}'' + A_1\,\boldsymbol{y}' + \boldsymbol{y} = \boldsymbol{K_O}\,\boldsymbol{K_P}\,(\boldsymbol{y_З} - \boldsymbol{y}) + \frac{K_O}{\overline{T}_И}\int_0^t (\boldsymbol{y_З} - \boldsymbol{y})\,d\,t + \boldsymbol{K_O z}. \qquad (3.27)$$

Ko := 1 **Kp** := 10 **A1** := 10 **A2** := 75 **y3** := 50 **z** := −100

Given

$$\frac{\boldsymbol{A2}}{1 + \boldsymbol{Ko}\cdot\boldsymbol{Kp}}\frac{\boldsymbol{d}^2}{\boldsymbol{dt}^2}\boldsymbol{y}(\boldsymbol{t}) + \frac{\boldsymbol{A1}}{1 + \boldsymbol{Ko}\cdot\boldsymbol{Kp}}\frac{\boldsymbol{d}}{\boldsymbol{dt}}\boldsymbol{y}(\boldsymbol{t}) + \boldsymbol{y}(\boldsymbol{t}) = \frac{\boldsymbol{Ko}\cdot\boldsymbol{Kp}}{1 + \boldsymbol{Ko}\cdot\boldsymbol{Kp}}\cdot\boldsymbol{y3} + \frac{\boldsymbol{Ko}}{1 + \boldsymbol{Ko}\cdot\boldsymbol{Kp}}\cdot\boldsymbol{z}$$

$\boldsymbol{y}(0) = 50$ $\boldsymbol{y}'(0) = 0$

$\boldsymbol{y} := \boldsymbol{Odesolve}(\boldsymbol{t}, 100)$

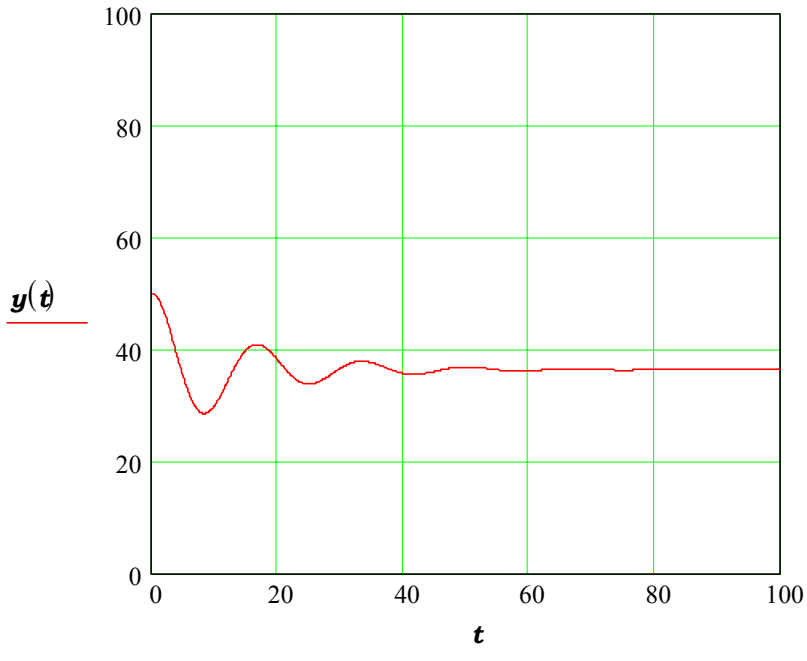

Рис.3.11. Моделирование переходного процесса под действием возмущения

Для того чтобы освободиться от интеграла в правой части последнего уравнения, почленно продифференцируем его по верхнему пределу, то ессть по t. Тогда после алгебраических преобразований получим при $y_з$ = const :

$$A_2\, y''' + A_1\, y'' + (1 + K_O\, K_P)\, y' + \frac{K_O}{T_И}\, y = \frac{K_O}{T_И}\, y_З .$$

Загрузим Mathcad и в соответствии с его правилами запишем условия задачи (рис.3.12), включая начальные условия (примем нулевые), после чего запустим её на решение.

Из полученных результатов следует, что задание выполнено точно. По этому поводу специалисты по автоматике шутят, говоря, что тот алгоритм регулирования точен, в формуле которого имеется интеграл. Однако здесь уместно ввести некоторое уточнение. Точное выполнение задания имеет место при постоянном возмущении. Если же оно оказывается функцией времени, то в согласии с уравнением (3.27) после его почленного дифференцирования величина z не исчезает и способна вызвать динамическую, то есть изменяющуюся во времени, ошибку выполнения задания регулятором.

Читателю представляется возможность аналогичным образом испытать другие алгоритмы регулирования из содержащихся в табл. 3.2 в применении не только к статическим, но также и к астатическим объектам и ознакомиться с их свойствами.

$A1 := 12 \quad A2 := 35 \quad Ko := 1 \quad Kp := 0.75 \quad Tu := 5 \quad y3 := 50$

Given

$$A2 \cdot \frac{d^3}{dt^3} y(t) + A1 \cdot \frac{d^2}{dt^2} y(t) + (1 + Ko \cdot Kp) \cdot \frac{d}{dt} y(t) + \frac{Ko}{Tu} \cdot y(t) = \frac{Ko}{Tu} \cdot y3$$

$y''(0) = 0 \quad y'(0) = 0 \quad y(0) = 0$

$y := Odesolve(t, 100)$

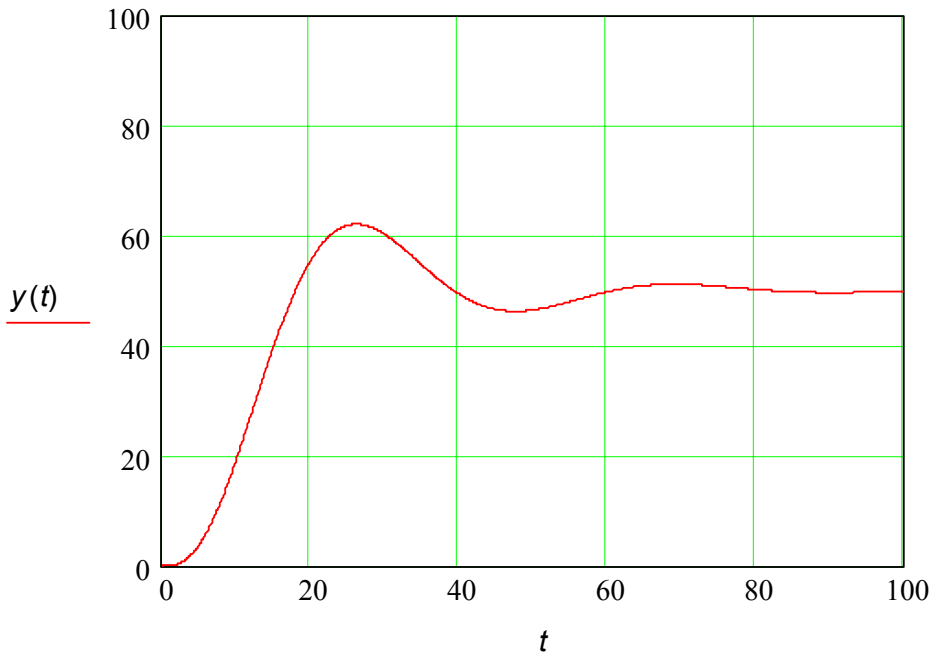

Рис.3.12. Переходный процесс в САР со статическим объектом 2-го порядка и ПИ – регулятором

3.3.4. Особенности моделирования динамических систем с запаздыванием

В ряде случаев металлургические объекты обладают чистым, называемым также транспортным, *запаздыванием*. Послед-

нее обусловлено временем транспортирования вещества (например компонентов формовочных или стержневых смесей) по протяжённым коммуникациям в процессе регулирования, в частности – состава этих смесей. Значение чистого запаздывания в секундах составляет

$$\tau_0 = \boldsymbol{V / L}, \qquad (3.28)$$

где $\boldsymbol{V}$ – скорость потока вещества, переносимого на ленте транспортёра, м / с;

$\boldsymbol{L}$ – протяжённость коммуникации, длина транспортёра, м.

Статический объект первого порядка с чистым запаздыванием характеризуется дифференциальным уравнением

$$T\frac{d\,\boldsymbol{y}(t)}{dt} + \boldsymbol{y}(t) = K_O\,\boldsymbol{x}(t-\boldsymbol{\tau}_0), \qquad (3.29)$$

где $\boldsymbol{T}$ – постоянная времени, с;

$\boldsymbol{K}$ – передаточный коэффициент, ед.выхода / ед.входа.

Для астатического объекта первого порядка с чистым запаздыванием справедливо дифференциальное уравнение другого вида:

$$\bar{T}\frac{d\,\boldsymbol{y}(t)}{d\,t} = \boldsymbol{x}(t-\boldsymbol{\tau}_0). \qquad (3.30)$$

Здесь $\bar{T}$ – условная постоянная времени объекта со смешанной размерностью (ед.времени · ед.входа) / (ед. выхода).

Иногда пользуются другой формой записи уравнения (3.29), а именно

$$\frac{d\,\boldsymbol{y}(t)}{d\,t} = \bar{K}_O\,\boldsymbol{x}(t-\boldsymbol{\tau}_0), \qquad (3.31)$$

где $\bar{K}_O$ – условный передаточный коэффициент объекта.

При этом

$$\bar{K}_O = 1 \;/\; \bar{T}_O.$$

На рис. 3.13 представлены кривые разгона объектов первого порядка с чистым запаздыванием: статического и астатического.

Для моделирования систем с чистым запаздывнием Mathcad непригоден. Поэтому мы должны прибегнуть к использованию принципов программирования задачи в среде, например, уже знакомого из предыдущего материала языка Visual Basic. Однако предварительно требуется применить один из численных методов преобразования дифференциальных уравнений в их алгебраический эквивалент, который «умеет» решать компью-

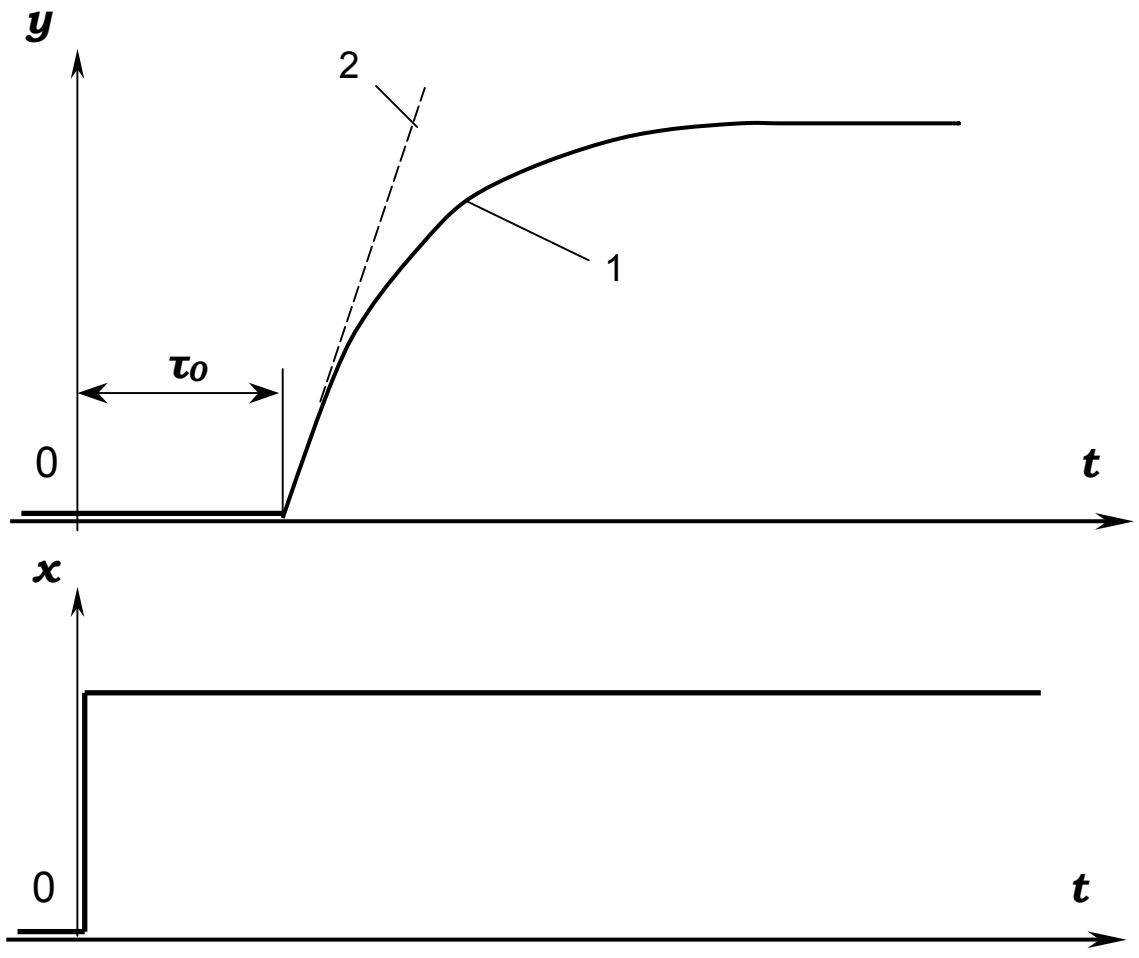

Рис. 3.13. Кривые разгона объектов первого поядка с чистым запаздыванием: статического (1) и астатического (2)

тер. Для этого используем *метод конечных разностей*. Сущность такого метода заключается в замене бесконечно малых величин: $d\boldsymbol{y}$, $d\boldsymbol{t}$ и пр. их достаточно малыми, но конечными изменениями, или разностями [14], [25]. В рассматриваемом случае разности *первого порядка* имеют вид:

$$d\,\boldsymbol{y} \approx \Delta \boldsymbol{y} = \boldsymbol{y}_i - \boldsymbol{y}_{i-1};$$

$$d\,t \approx \Delta t = t_i - t_{i-1},$$

где $\boldsymbol{i}$ – текущий номер данного шага $\Delta\boldsymbol{t}$ по времени.

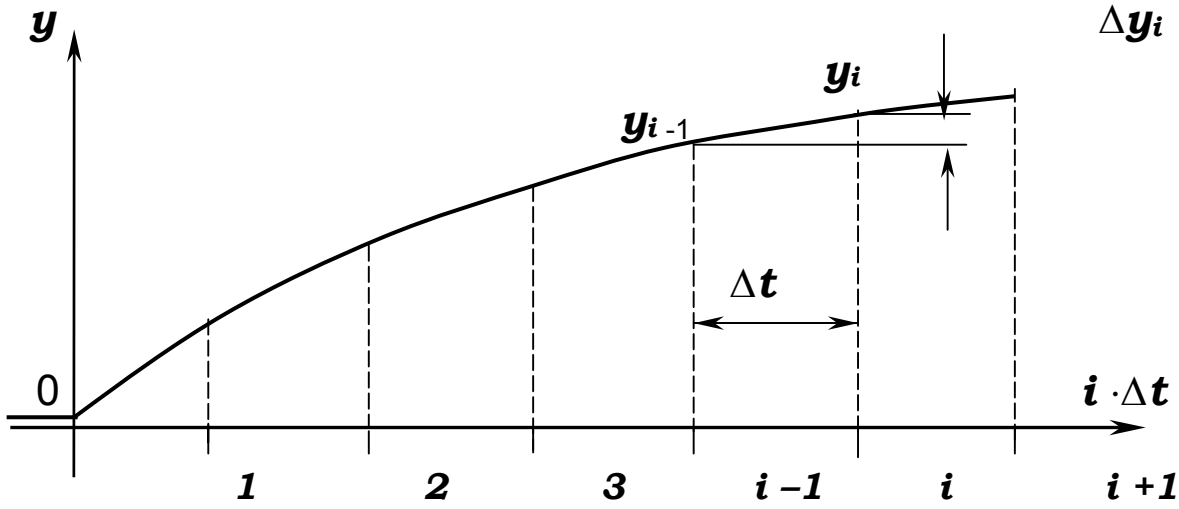

Рис. 3.14. Численное представление переходного процесса

Таким образом, вместо непрерывного течения времени $\boldsymbol{t}$ и соответственно плавного изменения величины $\boldsymbol{y}$ применяют приём *дискретизации* исследуемого процесса, то есть рассматривают его состояние лишь в последовательные ($\boldsymbol{i}$ – е) моменты времени.

Если $\tau_0 = 0$, то вместо уравнений (3.30), (3.31) можно перейти к конечно - разностным уравнениям соответственно

$$T\,\frac{\boldsymbol{y}_i - \boldsymbol{y}_{i-1}}{\Delta t} + \boldsymbol{y}_{i-1} = K_O\,\boldsymbol{x}_{i-1}; \qquad (3.32)$$

$$\overline{T}\,\frac{\boldsymbol{y}_i - \boldsymbol{y}_{i-1}}{\Delta t} = \boldsymbol{x}_{i-1}. \qquad (3.33)$$

Отсюда для компьютерного моделирования получаем численные модели статического (3.34) и астатического (3.35) объектов:

$$\boldsymbol{y}_i = \boldsymbol{y}_{i-1} + \frac{K_O\,\boldsymbol{x}_{i-1} - \boldsymbol{y}_{i-1}}{T} \cdot \Delta t; \qquad (3.34)$$

$$\boldsymbol{y}_i = \boldsymbol{y}_{i-1} + \frac{\Delta t}{T}\,\boldsymbol{x}_{i-1}. \qquad (3.35)$$

Таким образом, очередное ($\boldsymbol{i}$ – е) значение выхода $\boldsymbol{y}$ объекта определяется предшествующими значениями $\boldsymbol{x}$, $\boldsymbol{y}$, а также его динамическими параметрами (передаточным коэффициентом и постоянной времени).

Если теперь ввести чистое запаздывание $\tau_0 \neq 0$, то всегда можно подобрать шаг по времени, удовлетворяющий условию

$$\Delta \boldsymbol{t} = \frac{\boldsymbol{\tau}_0}{n}, \qquad (3.36)$$

где $\boldsymbol{n}$ – целое число шагов по времени в составе чистого запаздывания. Тогда наши численные модели рассматриваемых объектов приобретают формы:

– статический объект

$$\boldsymbol{y}_i = \boldsymbol{y}_{i-1} + \frac{K_O \boldsymbol{x}_{i-1-n}}{T} \cdot \Delta \boldsymbol{t}, \qquad (3.37)$$

– астатический объект

$$\boldsymbol{y}_i = \boldsymbol{y}_{i-1} + \frac{\Delta \boldsymbol{t}}{T}\,\boldsymbol{x}_{i-1-n}. \qquad (3.38)$$

Последние два уравнения использованы при построении блок – схемы алгоритма компьютерного моделирования САР с чистым запаздыванием (рис. 3.15, с. 109), где все обозначения, кроме особо оговорённых, соответствуют принятым выше. Вид

функций f_1 и f_2 определяется свойствами объекта и используемым алгоритмом регулирования (табл.3.3). Максимальное значение счётчика циклов i_{max} пользователь задаёт как условие останова программы.

В развитие изложенных основ компьютерного моделирования автором разработаны VB – проекты (программы) анализа САР (табл. 3.4) [24]. Программы отличаются удобством ввода и варьирования данных, возможностью задавать вид объекта, в том числе с чистым запаздыванием, алгоритм регулирования (при использовании P2VB и P2optVB как непрерывного, так и импульсного действия) и получать автоматически масштабируемый график переходного процесса во времени. При этом проектом P2VB предусмотрена автоматическая оценка качества регулирования по интегральному квадратичному критерию, что даёт возможность выбора оптимальных настроечных параметров регулятора. Названное программное обеспечение учебного процесса и НИР содержится в фонде алгоритмов и программ кафедры металлургии и литейного производства СЗТУ.

Таблица 3.4. **VB – проекты моделирования переходных процессов в САР**

Тема	Программа
1. Исследование устойчивости САР по критерию Найквиста - Михайлова	P02VB
2. Исследование переходных процессов в системах двухпозиционного регулирования	P2posVB
3. Исследование переходных процессов в линейных САР	P2VB
4. Моделирование процесса работы автоматического оптимизатора	P2optVB

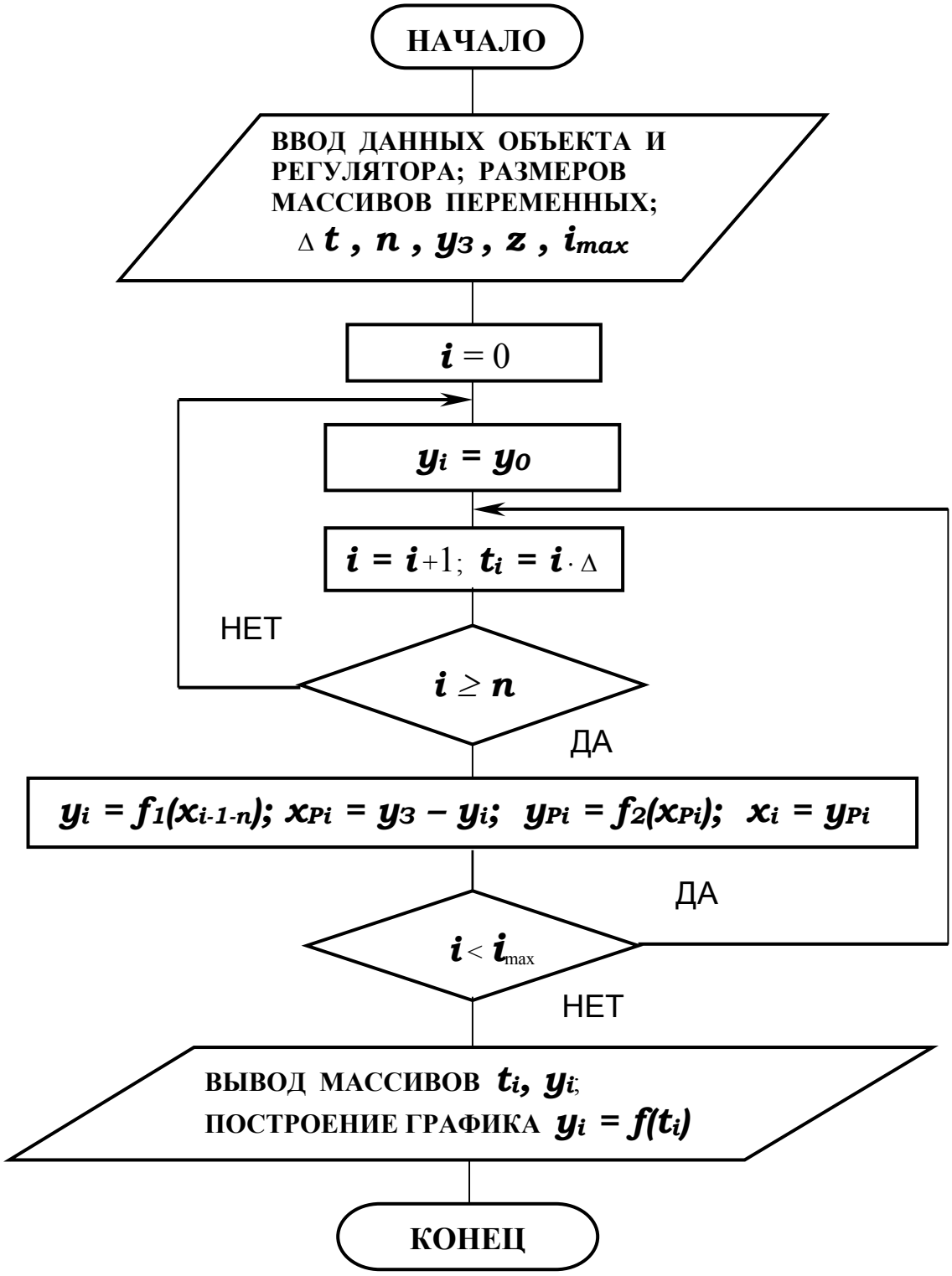

Рис. 3.15. Блок – схема моделирования САР

3.4. Основы моделирования объектов с распределёнными параметрами

В металлургических и литейных процессах параметром, распределённым в пространстве объекта, обычно оказывается *температура* (нагревание и охлаждение тел, их плавление и затвердевание). Реже таким параметром является *концентрация* химических элементов, например в процессах диффузионного легирования (химико – термическая обработка металлов).

В качестве математической основы для построения моделей объектов с распределёнными параметрами используется дифференциальное уравнение Фурье – Фика с *частными производными* общего вида

$$\frac{\partial \boldsymbol{\Pi}}{\partial \boldsymbol{t}} = W\left(\frac{\partial^2 \boldsymbol{\Pi}}{\partial \boldsymbol{x}^2} + \frac{\partial^2 \boldsymbol{\Pi}}{\partial \boldsymbol{y}^2} + \frac{\partial^2 \boldsymbol{\Pi}}{\partial \boldsymbol{z}^2}\right) \qquad (3.39)$$

в котором $\boldsymbol{\Pi}$ – параметр (температура θ, ºС или концентрация вещества C, например, в %);
$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ – координаты пространства, м;
$\boldsymbol{t}$ – время, с;
$\boldsymbol{W}$ – коэффициент, характеризующий скорость распространения теплоты или потока вещества в том же объекте.

Для процессов теплопереноса $\boldsymbol{W} = \boldsymbol{a}$ – коэффициент температуропроводности, м^2 / с

$$a = \frac{\lambda}{c\gamma}, \qquad (3.40)$$

где λ – коэффициент теплопроводности, кВт / (м · К);
c – удельная теплоёмкость, кДж / (кг · К);
γ – плотность, кг / м^3.

В процессах диффузионного массопереноса $\boldsymbol{W} = \boldsymbol{D}$ – коэффициент диффузии, м^2 / с.

Для решения уравнения (3.39), в соответствии с особенностями решаемой задачи, принимают определённые *краевые условия*: начальные и граничные.

Начальные условия характеризуют состояние объекта в начальный момент времени (например начальное распределение температур).

Граничные условия описывают особенности протекания процесса, например теплопереноса, на границах объекта.

Решение дифференциального уравнения (3.39) с учётом краевых условий требует применения одного из численных методов. К настоящему времени таких методов с достаточной полнотой разработано два:

А) Метод конечных разностей;

В) Метод конечных элементов.

Метод *конечных разностей* появился ещё в докомпьютерную эпоху [26] и применялся при графо - аналитических расчётах нестационарной теплопроводности. В наше время этот метод реализуется в многочисленных программных разработках моделирования тепловых задач как учебного характера, так и профессиональных. От данного метода в своих работах пока ещё не отказалась одна из всемирно известных коммерческих программистских фирм MagmaSoft. Отмечается [27], [28], что конечно - разностный метод является частным случаем более общего метода конечных элементов, обеспечивая необходимость применения менее сложного математического аппарата и более экономичные запросы к потребным машинным ресурсам в отношении объёма оперативной памяти и времени счёта. Вместе с тем, метод *конечных элементов* оказывается вне конкуренции при моделировании процессов формирования отливок сложных геометрических форм, ограниченных криволинейными поверхностями. Материалы сравнения методов конечных разностей и конечных элементов приводятся в литературе, в частности в [28], [31].

В рамках настоящей работы мы ограничимся рассмотрением конечно – разностного метода и его применения при моделировании литейных процессов.

3.4.1. Метод конечных разностей

С понятием конечных разностей мы уже встречались в разделах 3.3.3 и 3.4. Однако, здесь для преобразования урав-

нения (3.39) в его конечно – разностный эквивалент нам дополнительно потребуются некоторые новые понятия.

Рассмотрим *одномерный тепловой поток,* распространяющийся только в направлении оси ***x.*** Плоское тело, подвергающееся нагреву или охлаждению, разобьём по его толщине на элементарные слои толщиной Δx ***(***рис**.** 3.16***).***

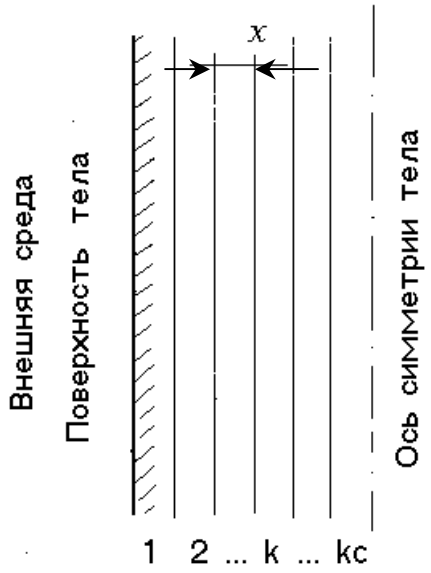

Рис. 3.16. Дискретизация пространства в методе конечных разностей (одномерная плоская задача с порядковыми номерами элементарных слоёв, например от поверхности к центру)

Тогда вместо исходного уравнения для одномерной задачи

$$\frac{\partial \theta}{\partial t} = a \frac{\partial^2 \theta}{\partial x^2}, \qquad (3.41)$$

для ***k*** – го элементарного слоя мы можем записать в конечно - разностной форме:

$$\frac{\Delta \theta_k}{\Delta t} = a \frac{\Delta^2 \theta_k}{\Delta x^2}, \qquad (3.42)$$

где

$\Delta\theta_k = \theta_k^* - \theta_k$ – приращение температуры от начального θ_k (на момент начала данного шага по времени) до конечного θ_k^* (на момент окончания данного шага по времени) значений, °С;

$\boldsymbol{\Delta t}$ – шаг по времени, с;

$\boldsymbol{a}$ – коэффициент температуропроводности, м^2 / с;

$\boldsymbol{\Delta x}$ – шаг по координате пространства (толщине тела), м;

$\Delta^2\theta_k$ – разность температур второго порядка, ºС :

$$\Delta^2 \boldsymbol{\theta_k} = (\boldsymbol{\theta}_{k-1} - \boldsymbol{\theta}_k) - (\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k+1}) = \boldsymbol{\theta}_{k-1} - 2\boldsymbol{\theta}_k + \boldsymbol{\theta}_{k+1}. \qquad (3.43)$$

Таким образом, и в этом случае переход от дифференциального уравнения к конечно - разностному может быть осуществлён путем замены бесконечно малых величин их достаточно малыми, но конечными разностями.

Уравнению (3.42) может быть придан определённый *физический смысл.* Для этого составим тепловой баланс $\boldsymbol{k}$ - го элементарного слоя, например в процессе охлаждения рассматриваемого тела, с учётом подвода теплоты теплопроводностью от $\boldsymbol{k}$ +1 слоя, частичной её аккумуляции $\boldsymbol{k}$ - м слоем и отвода теплопроводностью к $\boldsymbol{k}$ –1 слою за время $\Delta\boldsymbol{t}$ **:**

$$\frac{\lambda}{\Delta x}(\theta_{k+1} - \theta_k)\Delta t - \frac{\lambda}{\Delta x}(\theta_k - \theta_{k-1})\Delta t = c\gamma\Delta x(\theta_k^* - \theta_k), \qquad (3.44)$$

откуда нетрудно перейти непосредственно к выражению (3.42) или представить результат в общей развёрнутой форме:

$$\frac{\theta_k^* - \theta_k}{\Delta t} = \frac{a}{\Delta x^2}(\theta_{k-1} - 2\theta_k + \theta_{k+1}); \qquad (3.45)$$

$$\theta_k^* = \theta_k + Fo(\theta_{k-1} - 2\theta_k + \theta_{k+1}), \qquad (3.46)$$

где $Fo = \dfrac{a \cdot \Delta t}{\Delta x^2}$ – значение критерия подобия Фурье, вычисляе-

мое для шага по времени и шага по толщине тела.

Отсюда следует, что новое значение температуры k – го элементарного слоя тела определяется начальными температурами k -1, k и k+1 слоёв, а также теплофизическими свойствами и принятыми при расчёте (моделировании) значениями Δt и
Δx.

Для выявления динамики температурного поля в поперечном сечении тела моделирующим алгоритмом на каждом шаге по времени должно быть предусмотрено последовательное сканирование совокупности всех элементарных слоёв, причём новые значения температур на данном шаге принимаются за начальные для следующего шага по времени.

Выражение (3.46) называют *явной разностной схемой*. Она достаточно наглядна и даёт возможность построения на её основе простых и предельно компактных программ компьютерного моделирования. Однако следует иметь в виду, что для обеспечения *устойчивости расчёта* необходимо соблюдать условие:

$$Fo \leq 0,5 . \qquad (3.47)$$

Потеря устойчивости выражается в беспорядочности выдаваемых компьютером значений температур отдельных слоёв вместо плавного их изменения, соответствующего физическому смыслу моделируемых процессов. Причиной потери устойчивости является накопление *ошибок аппроксимации* и *ошибок счёта*.

Ошибки аппроксимации связаны с неизбежной дискретизацией пространства и времени, а ошибки счёта возникают из - за ограниченной значности чисел в компьютере.

Практически несложно принять соотношение шагов Δt и Δx, удовлетворяющее условию (3.47).

Можно использовать граничное значение критерия Фурье Fo = 0,5. Тогда получаем формулу Шмидта

$$\theta_k^* = \frac{\theta_{k-1} + \theta_{k+1}}{2}. \qquad (3.48)$$

Выполненными на кафедре МиЛП СЗТУ многочисленными исследованиями подтверждена приемлемая степень адекватности моделей, базирующихся на явной разностной схеме.

Вместе с тем известна и *неявная разностная схема* в виде уравнения

$$\theta_k^* = \theta_k + Fo(\theta_{k-1}^* - 2\theta_k^* + \theta_{k+1}^*). \qquad (3.49)$$

Сравнивая уравнения (3.46) и (3.49), замечаем, что первое из них является уравнением с одной неизвестной величиной – новым значением температуры, тогда как последнее содержит три неизвестных величины, помеченные звёздочками. Поэтому для решения неявной разностной схемы на каждом шаге вычислений необходимо решать систему уравнений вида (3.49), сдвинутых друг относительно друга по номеру элементарного слоя *k*.

Достоинством неявной разностной схемы является её абсолютная устойчивость при любых соотношениях Δt и Δx.

Метод конечных разностей в его физической трактовке позволяет учесть также и граничные условия. Для этого необходимо и достаточно рассмотреть тепловой баланс наружного элементарного слоя, которому мы присвоили порядковый номер 1. Тогда по аналогии с уравнением (3.44) и, принимая во внимание отдачу теплоты наружным слоем в окружающую среду с температурой θ_a, °С, получаем

$$\frac{\lambda}{\Delta x}(\theta_2 - \theta_1) - \alpha(\theta_1 - \theta_a) = c\gamma\Delta x\frac{\Delta\theta_1}{\Delta t}$$

или после последовательных преобразований

$$\frac{\Delta\theta_1}{\Delta t} = \frac{1}{c\gamma\Delta x}\left[\frac{\lambda}{\Delta x}(\theta_2 - \theta_1) - \alpha(\theta_1 - \theta_a)\right]; \qquad (3.50)$$

$$\theta_1^* = \theta_1 + \frac{\Delta t}{c\gamma\Delta x}\left[\frac{\lambda}{\Delta x}(\theta_2 - \theta_1) - \alpha(\theta_1 - \theta_a)\right]. \qquad (3.51)$$

Более “утончённый” способ учёта граничных условий путём добавления к поверхности тела фиктивного полуслоя с нулевым значением теплопроводности приведен в литературе [26], с. 74.

В ряде задач моделирования рассматриваются вопросы изменения теплового состояния не только плоских, но и цилиндрических тел. Для таких тел в цилиндрической системе координат (рис. 3.17, с. 7) уравнение Фурье - Фика записывается в форме

$$\frac{\partial \theta}{\partial t} = a\left(\frac{\partial^2 \theta}{\partial r^2} + \frac{1}{r}\frac{\partial \theta}{\partial r}\right), \qquad (3.52)$$

где r – текущий радиус сечения тела, м;
$\theta = \theta(\boldsymbol{r}, \boldsymbol{t})$ – температура, изменяющаяся по радиусу r и во времени $\boldsymbol{t}$, °С.

Из очевидных соотношений $r = \boldsymbol{k} \cdot \Delta r$; $\boldsymbol{k} = \overline{1, \boldsymbol{n}}$,
где $\boldsymbol{k}$ – порядковый номер элементарного слоя в направлении от центра цилиндра к его поверхности;
$\boldsymbol{n}$ – общее число слоёв,

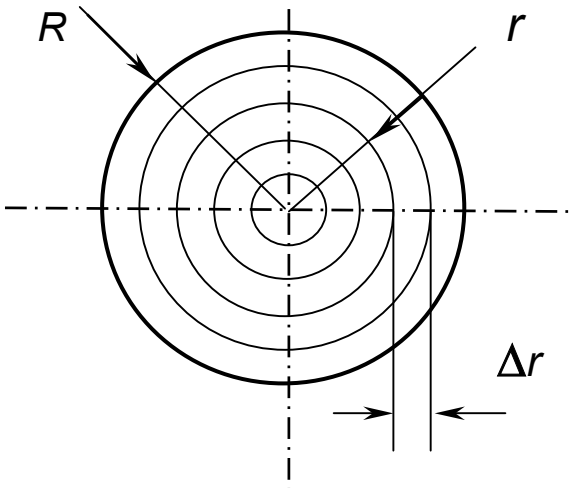

Рис.3.17. К динамике температурного поля цилиндрического тела

Уравнение (3.52) может быть представлено в конечно - разностной форме следующим образом:

$$\frac{\Delta\theta}{\Delta t}=a\left(\frac{\theta_{k-1}-2\theta_k+\theta_{k+1}}{\Delta r^2}+\frac{1}{k\,\Delta r}\cdot\frac{\Delta\theta}{\Delta r}\right)=$$

$$=a\left(\frac{\theta_{k-1}-2\theta_k+\theta_{k+1}}{\Delta r^2}+\frac{\theta_{k+1}+\theta_{k-1}}{2k\,\Delta r^2}\right)=$$

$$=\frac{a}{\Delta r^2}\left(\theta_{k-1}-2\theta_k+\theta_{k+1}+\frac{\theta_{k+1}-\theta_{k-1}}{2k}\right),$$

или в окончательной форме

$$\theta_k^*=\theta_k+\frac{a\,\Delta t}{\Delta r^2}\left(\theta_{k-1}-2\,\theta_k+\theta_{k+1}+\frac{\theta_{k+1}-\theta_{k-1}}{2k}\right). \qquad (3.53)$$

3.4.2. Моделирование процесса формирования отливки в малотеплопроводной форме

Условимся под малотеплопроводной формой подразумевать форму из материала, теплопроводность которого на много порядков ниже теплопроводности металла. Таким условиям отвечает форма, например на основе песчано - глинистой смеси. Согласно физическим представлениям [32] примем, что начальная температура металла равна температуре его кристаллизации $\theta_{кр}$, а температура рабочей поверхности формы, вступающей в контакт с жидким металлом, мгновенно повышается от начального значения $\theta_{0ф}$ до значения, равного $\theta_{кр}$. Выделившаяся при затвердевании *скрытая теплота* ρ передаётся материалу формы и расходуется на нагревание её элементарных слоёв (рис. 3.18, с. 118).

Ввиду малой теплопроводности материала формы, температура корочки затвердевшего металла толщиной δ и температура рабочей поверхности формы во времени $\boldsymbol{t}$ остаются близкими к температуре кристаллизации.

Условимся все температуры отсчитывать относительно начальной температуры формы.

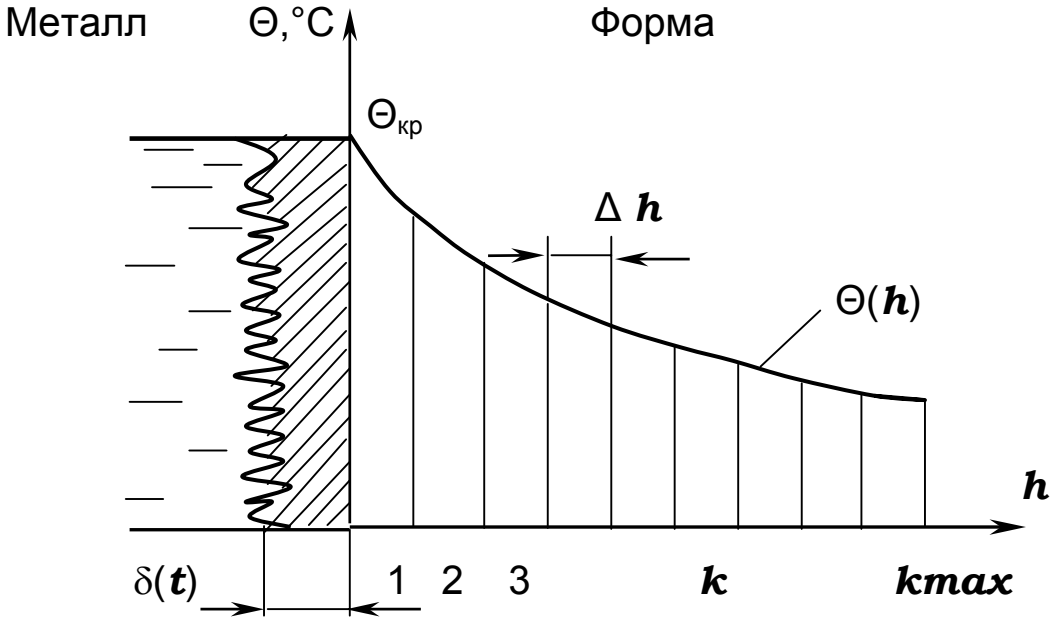

Рис. 3.18. Схема процесса формирования плоского фронта затвердевания металла

При отсутствии потерь вся теплота, выделяющаяся при затвердевании, воспринимается формой:

$$\rho \gamma_{м} S \delta = S \int_0^{\infty} c \gamma_{ф} \theta(h)\, dh, \qquad (3.54)$$

где ρ – скрытая теплота, кДж / кг;

$\gamma_м$ и $\gamma_ф$ – плотности металла и формы соответственно, кг / м^3;

S – поверхность контакта металла и формы, м^2;

c – удельная теплоёмкость материала формы, кДж / (кг · К);

$\theta(h)$ – температура бесконечно тонкого слоя формы толщиной dh, °С;

h – координата (толщина формы), м.

Интеграл в правой части уравнения (3.54) может быть определён численно. Тогда получим

$$\int_0^{\infty} c \gamma_{ф}\, \theta(h)\, d\, h = \sum_{k=1}^{n} c \gamma_{ф} \theta(k \cdot \Delta h)\, \Delta h,$$

где k – порядковый номер элементарного слоя;
n – общее число слоёв;
Δh – толщина слоя, м.

Отсюда можно рассчитать текущую (на момент времени t) толщину корочки затвердевшего металла:

$$\delta(t) = \frac{c\,\gamma_{\Phi}\,\Delta h \sum_{k=1}^{n} \theta(k\,\Delta h)}{\rho\,\gamma_{M}} \qquad (3.55)$$

при постоянных (например, средних в интервале рабочих температур) значениях свойств металла и формы.

Алгоритм численного моделирования процесса формирования плоского фронта затвердевания в авторском варианте (рис. 3.19) предусматривает своим внутренним циклом вычисление температур отдельных слоёв формы θ_k^* от второго до слоя с присвоенным ему порядковым номером k_{max}. В соответствии с принятыми положениями температура первого слоя формы, граничащего с металлом, в течение всего процесса остаётся постоянной. Затем вычисляется толщина корочки. Перед повторением внутреннего цикла осуществляется переопределение температур $\theta_k^* \rightarrow \theta_k$. Внешний цикл предназначен для счёта текущего времени по счётчику i и принятому шагу по времени Δt.

Условием останова служит начало прогрева слоя формы, которому пользователь присваивает порядковый номер z.

На основании рассмотренного алгоритма автором разработана программа Р12 (фонд алгоритмов и программ кафедры МиЛП СЗТУ) для осуществления моделирования процесса формирования плоского фронта затвердевания металла или эвтектического сплава. При работе со сплавом, затвердевающим в интервале температур, приближенные результаты можно получить, принимая за температуру кристаллизации среднее значение температур между точками ликвидуса и солидуса.

Если исследуется процесс формирования достаточно длинного слитка круглого сечения, уравнение (3.53) позволяет циклически рассчитывать температуры слоёв формы в цилиндрических координатах при том же граничном условии постоянства температуры поверхностного слоя $\theta_{пов} = \theta_{кр} = \text{const}$. При вводе необходимых исходных данных структура блок - схемы

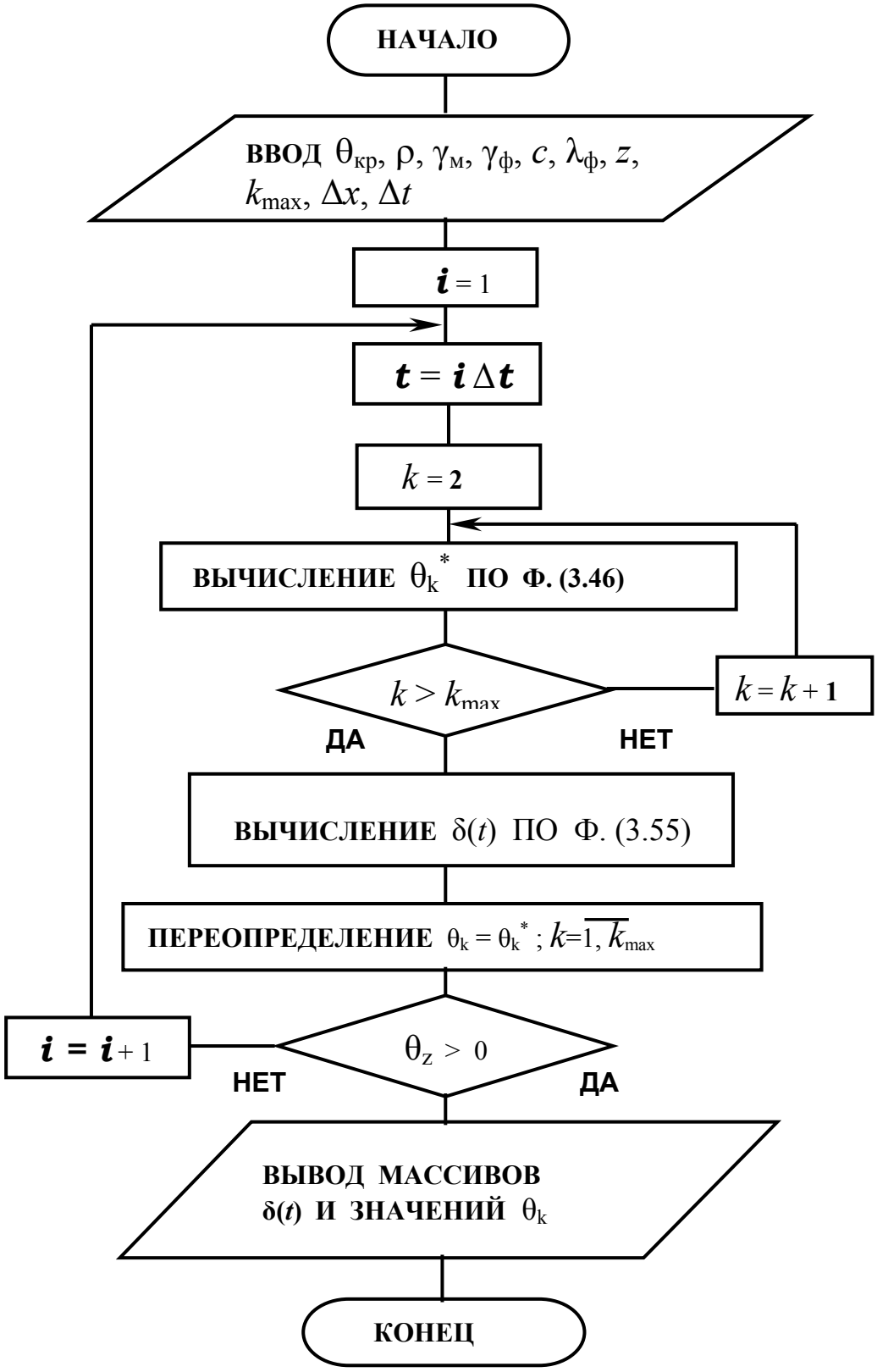

Рис. 3.19. Блок – схема алгоритма к рис. 3.18

алгоритма моделирования остаётся практически аналогичной той, которая показана на рис.3.19. Отличие заключается лишь в расчёте температур отдельных слоёв формы и оценке толщины корочки, принимающей здесь форму полого цилиндра (рис.3.20). Пусть наружный радиус отливки составляет $\boldsymbol{R}$, м. Толщина элементарного слоя формы – $\Delta \boldsymbol{r}$, м.

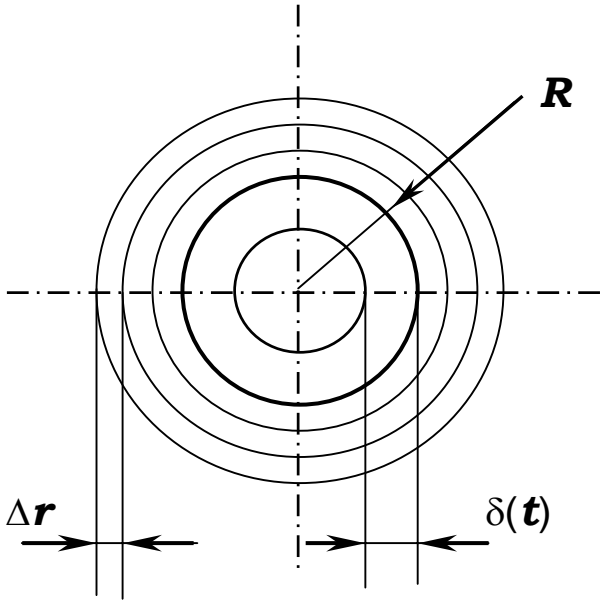

Рис.3.20. Схема формирования цилиндрического фронта затвердевания

К моменту времени $\boldsymbol{t}$ форма в своих кольцевидных слоях аккумулирует теплоту в количестве, кДж / м :

$$Q = \pi c \gamma_{\Phi} \times$$

$$\times \left\{ \left[(R+\Delta r)^2 - R^2 \right] \theta_1 + \left[(R+2\Delta r)^2 - (R+\Delta r)^2 \right] \theta_2 + \left[(R+3\Delta r)^2 - (R+2\Delta r)^2 \right] \theta_3 + \ldots \right\} =$$

$$= \pi c \gamma_{\Phi} \left\{ \left[(R+\Delta r)^2 - R^2 \right] \theta_1 + \sum_{k=2}^{n} \left[(R+k\,\Delta r)^2 - (R+(k-1)\,\Delta r)^2 \right] \theta_k \right\}, \qquad (3.56)$$

где величины θ_k рассчитываются по уравнению (3.53).

Это количество теплоты выделяет объём затвердевшего к данному моменту металла, кДж

$$Q = \pi \rho \gamma_M \left\{ R^2 - \left[R - \delta(t) \right]^2 \right\}, \qquad (3.57)$$

Раскрыв скобки в правой части последнего выражения и учитывая уравнение (3.56), получаем:

$$\delta(t)^2 - 2R\,\delta(t) + \frac{Q}{\pi\rho\gamma_M} = 0,$$

откуда текущая толщина слоя твёрдой фазы окончательно определяется формулой, м :

$$\delta(t) = R - \sqrt{R^2 - \frac{Q}{\pi\rho\gamma_M}}, \qquad (3.58)$$

Заметим, что если в уравнении (3.52) множитель 1 / ***r*** в правой части заменить на 2 / ***r*** , что справедливо для сферической системы координат, то аналогичным путём получим математическую базу для моделирования процесса формирования *сферического фронта* затвердевания.

Материалы настоящего раздела дают возможность исследовать характер влияния свойств материалов литейной формы и расплавов на скорость их затвердевания.

3.4.3. Моделирование процессов непрерывного литья

Существуют различные методики моделирования переходных процессов в системах непрерывного литья металлов и сплавов. На основании исследований, выполненных на кафедре МиЛП СЗТУ под научным руководством автора, разработана методика, при которой рассматривается тепловое и фазовое состояние поперечного сечения формирующегося слитка, фиксированное относительно тела слитка и движущееся вместе с ним во времени [14], [33] ... [36]. При этом предполагается, что вследствие интенсивных теплопотерь струи расплава, теплота перегрева последнего отсутствует, температурный градиент в жидкой фазе и на границе с твёрдой корочкой слитка равен нулю, а продольным теплопереносом пренебрегают. Принципиальной особенностью рассматриваемой методики является спо-

соб учёта выделения скрытой теплоты ρ, кДж/кг при затвердевании расплава.

Широко известен метод *эффективных теплоёмкостей*, при котором к действительному значению удельной теплоёмкости c, кДж/(кг · К) в области температур затвердевания прибавляют некоторую долю ρ. Этот способ недостаточно удобен для программирования и его трудно применить для чистых металлов и эвтектических сплавов.

Другой метод учёта выделения скрытой теплоты – метод *избыточных температур* – свободен от недостатков предыдущего и представляется более адекватным. Поэтому он был использован в разработках программ компьютерного моделирования процессов непрерывного литья.

Предложенная методика заключается в том, что расчёт температур в отдельных точках поперечного сечения слитка выполняется в две стадии. На первой стадии температуры определяются, например, по уравнениям (3.46) или (3.53) с учётом граничных условий по материалам раздела 3.4.1, уравнение (3.51). На второй стадии предусмотрен анализ состояния системы относительно температур затвердевания (для металлов и эвтектических сплавов) или точек ликвидуса и солидуса (для сплавов, затвердевающих в интервале температур). По результатам такого анализа к рассчитанным на первой стадии температурам вводятся поправки на выделение скрытой теплоты (рис. 3.20 и 3.21).

Детально механизм анализа описан в работе [14], с.53. 54.

Сущность его заключается в сравнении текущего значения расчётной температуры $\theta_к^*$ в каждой точке поперечного сечения слитка с температурами кристаллизации $\theta_{кр}$ (рис.3.21) и ликвидуса $\theta_л$ (рис.3.22). Если окажется, что расчётная температура на определённом шаге стала ниже этих температур, то в реальной системе должен происходить процесс выделения твёрдой фазы из расплава. В таком случае программа моделирования на данном шаге по времени Δt вводит поправки $\delta\theta_к$ к расчётным температурам (см. вертикальные стрелки на упомянутых рисунках). Такие поправки, выраженные в градусах Цельсия, полагаются пропорциональными приращению доли твёрдой фазы $\Psi(\theta)$ за шаг по времени:

$$\delta\theta_k = \frac{\rho}{c}\left[\psi(\theta_k^* + \delta\theta_k) - \psi(\theta_k)\right], \qquad (3.59)$$

Поправки вводятся до тех пор, пока сумма всех поправок для k – го элементарного слоя не достигнет значения, °С :

$$\sum_k \delta\,\theta_k = \frac{\rho}{c}\ . \qquad (3.60)$$

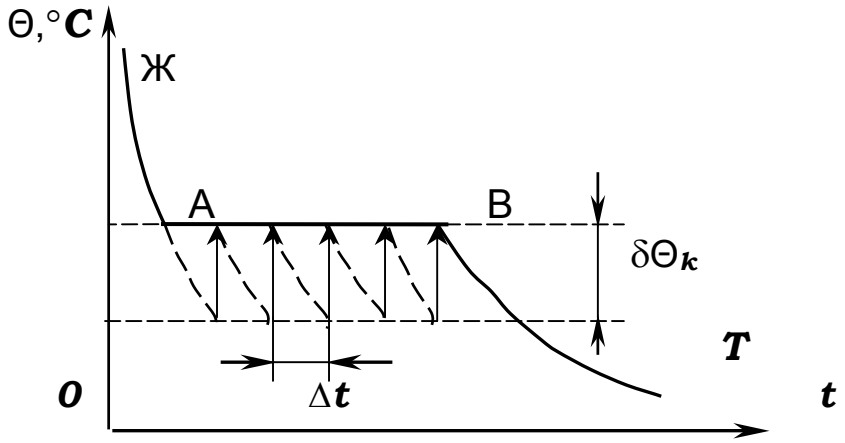

Рис. 3.21. Введение температурных поправок для металлов и эвтектических сплавов

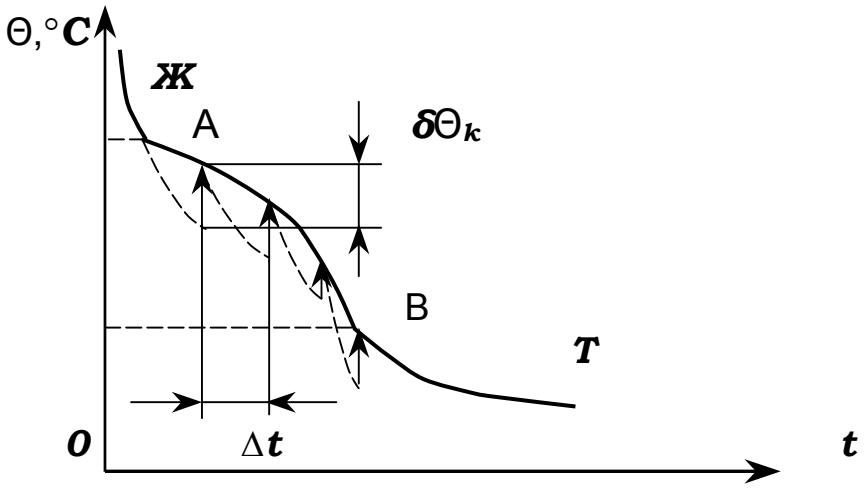

Рис. 3.22. Введение температурных поправок для сплавов, затвердевающих в интервале температур

Как следует из рис.3.21, поправки здесь вводятся с таким расчётом, чтобы после ввода очередной поправки вернуть значение расчётной температуры на *горизонтальную площадку* АВ термограммы, наличие которой является характерным признаком процесса затвердевания металлов и эвтектических сплавов при $\theta_{кр}$ = const. В отличие от этого для моделируемого сплава, затвердевающего в интервале температур (рис.3.22), каждая из поправок возвращает расчётную температуру на *линию ликвидуса* АВ, разделяющую области охлаждения жидкой (Ж) и твёрдой (Т) фаз.

В последнем случае следует различать особенности способов ввода температурных поправок для сплавов, затвердевающих в узком и широком интервалах температур.

Для сравнительно узкого интервала температур затвердевания (например, для стали ≈ 50 °С) линия ликвидуса АВ на рис.3.22 стремится к отрезку прямой. Поскольку температура θ в интервале между точками ликвидуса $\theta_л$ и солидуса $\theta_с$ может быть представлена в безразмерном выражении

$$X = \frac{\theta_Л - \theta}{\theta_Л - \theta_C}, \qquad (3.61)$$

то для узкого интервала температур затвердевания с достаточной степенью точности можно полагать, что доля твёрдой фазы $\psi(\theta)$ в двухфазной системе нарастает пропорционально $\Delta \boldsymbol{X}$, то есть – понижению величины $\boldsymbol{X}$ в этом интервале. Тогда

$$\delta\theta_k = \frac{\rho}{c}\left[\frac{\theta_Л - (\theta_k^* + \delta\theta_k)}{\theta_Л - \theta_C} - \frac{\theta_Л - \theta_k}{\theta_Л - \theta_C}\right], \qquad (3.62)$$

и искомая поправка составит, °С :

$$\delta\theta_k = (\theta_k - \theta_k^*) \Big/ \left[1 + (\theta_Л - \theta_C)\frac{c}{\rho}\right]. \qquad (3.63)$$

Для широкого интервала температур затвердевания (например, для бронзы ≈ 200 °С) допущение о прямолинейности линии ликвидуса неприемлемо. Поэтому линию ликвидуса диа-

граммы состояния следует аппроксимировать, например, полиномом и рассчитывать температурные поправки по известному «правилу отрезков». Пример результатов подобной процедуры приведен в работе [14], с.56.

На основании изложенных положений автором разработана серия программ численного моделирования процессов формирования непрерывнолитых слитков из металлов, эвтектических сплавов и сплавов общего вида при различных формах поперечного сечения: программа P40 – для плоских слитков с отношением ширины к толщине более трёх и слитков круглого сечения, программы P40sq и P40ra – для слитков прямоугольного (отношение ширины к толщине до трёх включительно) и квадратного сечений соответственно, а также ряд других программ с применением метода конечных разностей (см. [37] и материалы следующей главы настоящего учебного пособия).

Особенность принятых методик моделирования в двумерном пространстве состоит в том, что вместо общепринятых способов расчёта температуры в каждой точке поперечного сечения по температурам *четырёх* соседних точек применяется обычная и рассмотренная выше методика решения одномерной задачи. На рис.3.23 для случая симметричного охлаждения показана четверть поперечного сечения прямоугольного слитка при вычислении новых значений температур $\theta_{k,m}^*$ по температурам $\theta_{k,m}$ *двух* соседних точек с использованием уравнения (3.46). Здесь на каждом шаге по времени программно предусмотрен циклический процесс сканирования всех ячеек прямоугольной сетки сначала по вертикали (вдоль координаты $\boldsymbol{y}$) в пределах от $\boldsymbol{k} = 2$ до $\boldsymbol{k} = \boldsymbol{k}_c$ для каждого m (от $m = 2$ до $m = m_c$). Эта стадия процесса завершается введением температурных поправок на выделение скрытой теплоты по принципу, рассмотренному выше, и переопределением $\theta_{k,m}^* \to \theta_{k,m}$. Затем на том же шаге по времени процесс сканирования продолжается по горизонтали (вдоль координаты $\boldsymbol{x}$) в пределах от $m = 2$ до $m = m_c$ для каждого $\boldsymbol{k}$ (от $\boldsymbol{k} = 2$ до $\boldsymbol{k} = \boldsymbol{k}_c$), также с введением температурных поправок и переопределением $\theta_{k,m}^* \to \theta_{k,m}$ для подготовки к следующему шагу по времени.

Учёт граничных условий для $\boldsymbol{k} = 1$ и $m = 1$ совершается аналогично описанному уравнением (3.51). Именно граничные условия определяют направление переходного процесса тела

по температуре (нагревание или охлаждение) при неизменном виде расчётных формул.

Что касается расчёта температуры центрального θ_c (рис.3.17) или прилегающего к центру θ_{kc} (рис.3.16) слоя, то при симметричном теплопереносе может быть использована, например, формула

$$\theta_C^* = \theta_C + 4Fo(\theta_{PC} - \theta_C), \qquad (3.64)$$

где θ_{pc} – температура слоя, следующего за центральным в направлении к поверхности тела [38].

Для обеспечения условия гомохронности $Fo_x = Fo_y$ (одинакового масштаба времени по координатам x и y двумерного сечения слитка) необходимо выбирать одинаковые размеры ячеек сетки в двух взаимно перпендикулярных направлениях.

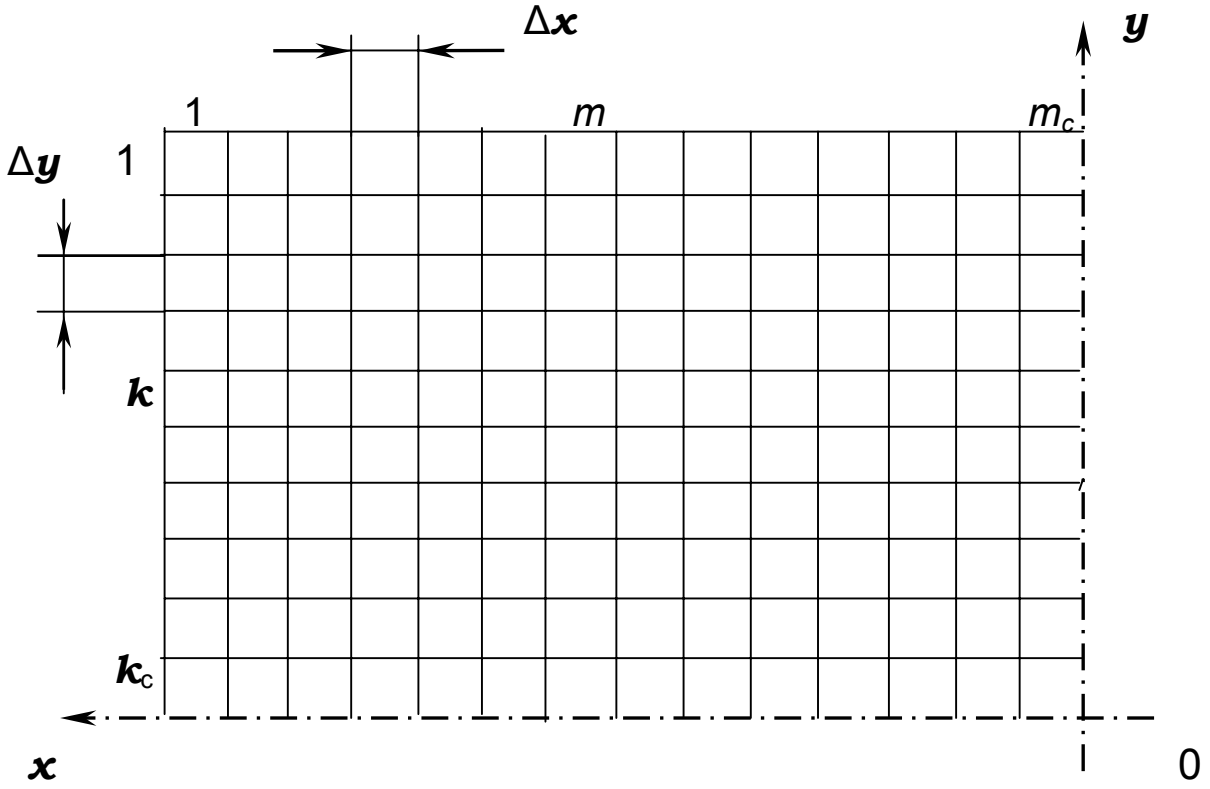

Рис.3.23. Сетка для расчёта процесса формирования слитка прямоугольного сечения

При несимметричном подводе или отводе теплоты необходимо выполнять сквозной расчёт температур всех элемен-

тарных слоёв, расположенных между противоположными поверхностями тела.

4. Элементы САПР литейной технологии

Проектирование этапов производства отливок (слитков) представляет собой достаточно сложный процесс, требующий специальных знаний и средств переработки необходимой информации. Эти знания студенты систематически приобретают при обучении в СЗТУ. В то же время, упомянутые средства в виде современных персональных компьютеров и их программного обеспечения призваны способствовать повышению качества технологических разработок и росту производительности труда конструкторов и технологов соответствующих служб литейных цехов и заводов. Проектирование металлургической и литейной технологии предусматривает разработку комплекта технической документации на каждую из операций технологического процесса производства литых изделий. В состав такой документации входят технологические инструкции и карты, результаты расчетов, чертежи моделей, формы, стержневых ящиков, модельных плит и т.п., включая чертёж самой проектируемой отливки. При проектировании литейной технологии следует стремиться обеспечить технологичность и качество конструкции литейной заготовки будущей детали механизма или машины, высокую производительность литейного оборудования, экономичность расходования материалов, высокую технико – экономическую эффективность производства в целом.

Системы автоматизированного проектирования (САПР) в области металлургии и литейного производства становятся неотъемлемой принадлежностью гибких автоматизированных производств (ГАП). Это обеспечивает быструю переналадку технологических линий, программно управляемых с помощью средств вычислительной техники (АСУ ТП, промышленных манипуляторов и роботов) на выпуск отливок новой конфигурации и массы, иной степени сложности, скорейшую отработку оптимальных режимов действия оборудования.

Согласно ГОСТ 23501.0 – 79 САПР является организационно – технической системой, выполняющей автоматизированное (то есть человеко – машинное, авт.) проектирование объектов и

состоящей из комплекса средств автоматизации проектирования, взаимосвязанного с подразделениями проектной организации. Объекты проектирования в САПР – это изделия (здесь – фасонные отливки и слитки), технологические процессы и организационно – технические системы.

На современном уровне комплекс технического обеспечения САПР строится на базе мощных серверов, связанных локальными вычислительными сетями с рядом рабочих станций. Первоначально использовались специальные операционные системы, управляющие программными пакетами такими, как, например ***I*** – Deas (США). Пакет предназначен для «твёрдотельного», то есть геометрического, моделирования, разработанного для проектирования моделей отливок сложных форм непосредственно на экране монитора с последующим изготовлением рабочих чертежей и автоматической разработкой программы для станков с числовым программным управлением. На таких станках затем изготовляются модели отливок, в том числе для нужд автомобилестроения, а также – судостроения (гребные винты сложного профиля и пр.). Впоследствии были разработаны версии ***I*** – Deas, работающие под управлением Windows NT, а затем и Windows 2000.

Внедрение САПР позволяет преодолеть известный «парадокс проектирования», когда требуется проектировать всё возрастающее количество всё более сложных объектов за всё боболее короткое время и силами сокращающегося штата конструкторов и технологов.

В процессе проектирования можно выделить три вида деятельности: изобретательство, инженерный анализ и принятие решений. Первый из них как творческая (интуитивная – эвристическая) деятельность человека по созданию нового объекта до конца не формализуется. Поэтому здесь компьютер выступает как «собеседник» проектанта и используется в диалоговом (интерактивном) режиме для осуществления оперативного просмотра на экране монитора больших количеств ранее найденных вариантов решения инженерно – технической задачи с целью поиска приемлемого варианта (возможного к доработке прототипа). Что же касается второго и третьего видов деятельности, то они поддаются формализации и поэтому в наиболь-

шей степени доступны к осуществлению с помощью технических средств САПР.

Конкретно проектирование технологических решений в рассматриваемой области включает в себя следующие основные задачи:

a) Расчёт оптимального состава плавильной шихты, обеспечивающего желаемый химический состав расплава и его минимальную стоимость;

b) Разработку технологического процесса плавки металла или специального литейного сплава;

c) Выбор оптимального состава формовочных и стержневых смесей;

d) Принятие рационального вида литейной формы, расположения отливки в ней и определения плоскости разъема формы;

e) Расчёт размера прибылей;

f) Определение мест подвода расплава и расчет литниковых систем;

g) Определение координат конечных точек и времени окончания процесса затвердевания расплава;

h) Расчёт глубины лунки жидкой фазы в непрерывнолитом слитке и выбор оптимальной скорости литья;

и ряд других задач.

Алгоритм функционирования системы автоматизированного проектирования (рис.4.1, с.131) процесса производства отливки предусматривает просмотр известных решений, хранящихся в базе данных (например – на основе Excel или Access) по результатам накопленного опыта, поиск прототипа или выявление необходимости разработки технических решений заново. При этом используют:

- для редактирования или создания текстовых документов текстовые процессоры Word или Tex;
- для выполнения математических расчетов – табличный процессор Excel, Visual Basic в различных его версиях и Mathcad или другие специализированные пакеты прикладных программ;
- для графических работ, кроме упомянутого выше ***I*** – Deas, – пакеты Autocad (с которым студенты знакомятся при изучении курса «Инженерная графика»), а также − Solid Works, Pro Engineer и пр.

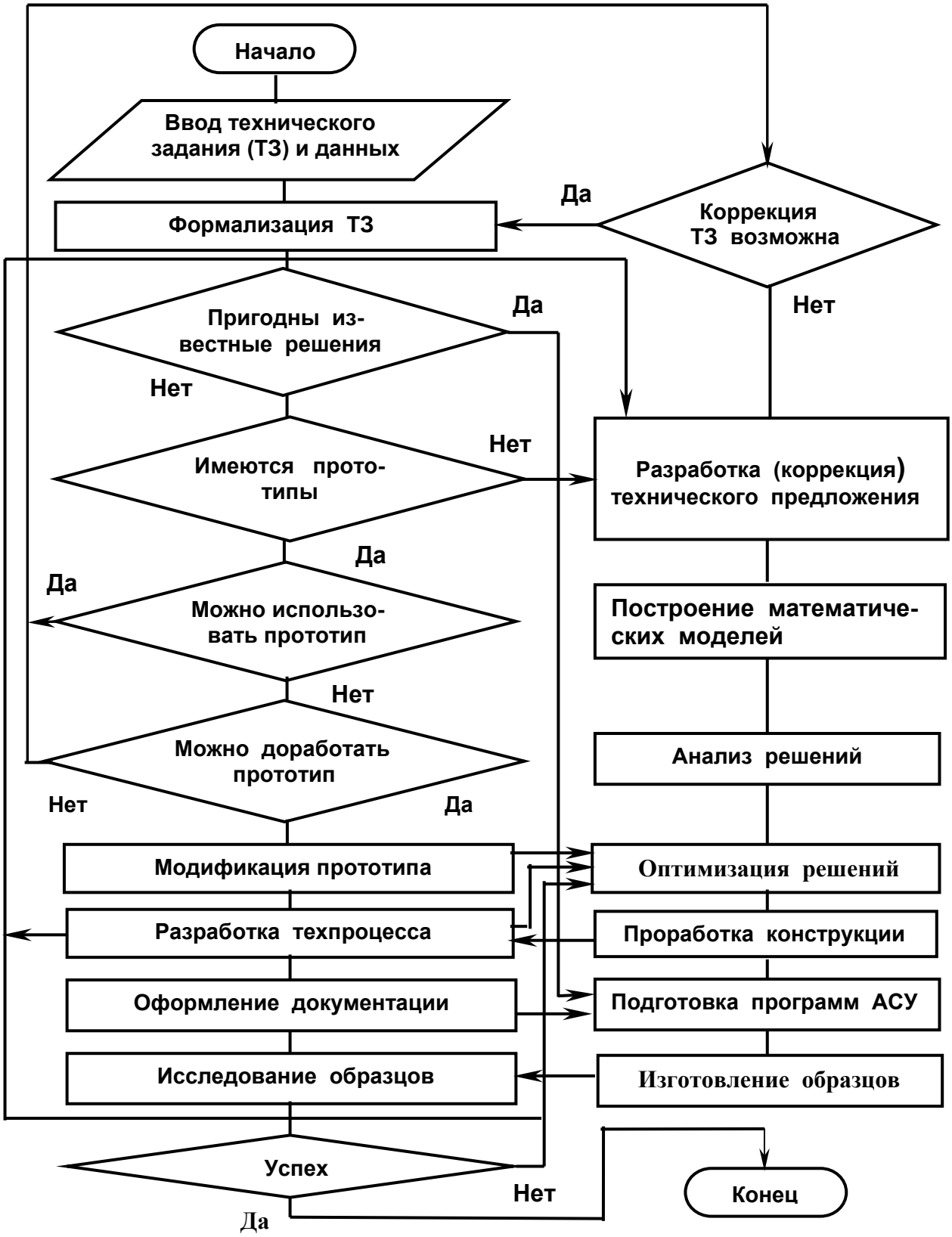

Рис. 4.1. Пример блок – схемы алгоритма САПР [11]

Важная роль при разработке новых объектов принадлежит компьютерному моделированию, которое во многих случаях подсказывает правильное решение, которое недоступно непосредственному вычислению по известным формулам.

Специальные расчёты, моделирование и оптимизацию выполняют, например, с использованием разработок автора и его коллег по кафедре металлургии и литейного производства (МиЛП) СЗТУ (табл. 4.1). Эти программы рассчитаны на ввод исходных данных в диалоговом (интерактивном) режиме и в дополнительных пояснениях не нуждаются.

Для решения многих задач моделирования процессов затвердевания расплавов при формировании отливок *сложных форм* получили распространение специальные коммерческие программные продукты, например зарубежные Pro Cast, Magma Soft и пр., а также отечественные – ПОЛИГОН [29], [30], [31].

Оформление технической документации должно удовлетворять требованиям ЕСКД и ГОСТ 2.423 -73.

Таблица 4.1. **Состав программного обеспечения САПР фонда алгоритмов и программ кафедры МиЛП СЗТУ**

Задача	Имя программы	Среда разработки
1. Расчёт оптимального состава плавильной шихты	P1	MS DOS (Turbo Basic) или Windows (Excel)
2. Численное моделирование процесса кислородно – аргонного рафинирования стали	P9VB	Windows (Visual Basic)
3. Оптимизация процесса освежения оборотной формовочной смеси по затратам на свежий кварцевый песок и качеству смеси	P10VB, P10MVB, P11VB	Windows (Visual Basic)

Продолжение табл. 4. 1.

4. Расчёт элементов литниковой системы (разработчик В.Т.Сенченко, канд. техн. наук, доц. кафедры МиЛП СЗТУ)	Litnik	MS DOS (Turbo Basic)
5. Расчёт элементов консольной литниковой системы	Console	MS DOS (Turbo Basic)
6. Расчёт активности компонентов шлака и равновесных с ним содержаний кислорода, фосфора и марганца в стали	P20	MS DOS (Fortran Power Station)
7. Расчёт окислительно – восстановительного потенциала атмосферы литейной формы	P21	MS DOS (Fortran Power Station)
8. Расчёт размеров усадочной раковины и прибыли (работа коллектива кафедры МиЛП СЗТУ)	Pribyl	MS DOS (Turbo Pascal)
9. Численное моделирование процесса формирования плоского фронта затвердевания металла в форме	P12	MS DOS (Turbo Basic)
10. Численное моделирование процесса формирования горизонтального стального слитка (2D и 2,5D задачи)	PGS4, PGS5	MS DOS (Fortran Power Station)
11. Численное моделирование процесса непрерывного литья плоских или цилиндрических слитков из металлов и сплавов с узким или широким диапазоном температур затвердевания	P40	MS DOS (Turbo Basic)
12. Численное моделирование процесса непрерывного литья слитков квадратного поперечного сечения	P40sq	MS DOS (Turbo Basic)

Окончание табл. 4. 1.

13. Численное моделирование процесса непрерывного литья слитков прямоугольного поперечного сечения	P40ra	MS DOS (Turbo Basic)
14. Численное моделирование процессов литья стальных слитков восьмигранного поперечного сечения	P5	MS DOS (Fortran Power Station)
15. Численное моделирование температурных и фазовых полей крупногабаритных стальных слябов, отливаемых непрерывным способом с учетом неравномерности условий охлаждения	P45	MS DOS (Fortran Power Station)

Материал данного раздела изучают выборочно и в индивидуальном порядке в соответствии с направлением профессиональной деятельности студента. Программы хранятся в фонде алгоритмов и программ кафедры металлургии и литейного производства СЗТУ и устанавливаются на компьютеры по месту проведения занятий руководителем. Для удобства использования эти программы заранее оттранслированы и представлены в форме выполняемых (exe) файлов. Они могут быть запущены в действие с помощью любого из навигаторов: Norton Commander, Volkov Commander, WinNavigator, Проводник (в современной операционной системе Windows.

Предметный указатель

А

В

Г

Д

Е

З

И

К

Л

М

О

П

Р

С

Т

У

Ф

Э

A

B

C

D

E

F

G

H

I

W

Литература

1. Рыжиков Ю.И. Решение научно-технических задач на персональном компьютере. – СПб.: Корона, 2000. – 272 с.: ил., табл.
2. Очков В.Ф. Mathcad 7 Pro для студентов и инженеров. – М.: Computer Press, 1998. – 381 с.: ил., табл.
3. Кудрявцев Е.М. Mathcad 2000. Символьное решение разнообразных задач. – М. : ДМК, 2001. – 386 с.: ил., табл.
4. Кетков Ю.Л. GW –, Turbo – и Quick Basic для IBM PC. – М.: Финансы и статистика, 1992. – 240 с.: ил., табл.
5. Очков В.Ф., Рахаев М.А. QBasic, Quick Basic, Basic Compiler. – М.: Финансы и статистика, 1995. – 368 с.: ил., табл.
6. Гарнаев А.Ю. Самоучитель VBA. Технология создания пользовательских приложений. – СПб.: BHV, 1999. – 504 с.: ил., табл.
7. Уэллс Э., Харшбаргер С. MS Excel 97. – М.: Русск. ред. Microsoft Press, 1998. – 510 с.: ил., табл.
8. Корнелл Г. Программирование в среде Visual BASIC 5. – Минск: ООО «Попури», 1998. – 608 с.: ил., табл.
9. Соха Дж., Рахмел Д., Нолл Д. Visual Basic. Версия 5.0. – Минск.: Попурри, 1998. – 320 с.: ил., табл.
10. Браун С. Visual Basic 6.- СПб - Харьков - Минск, 2001.- 421 с.: ил., табл.
11. Дембовский В. В. Методы исследования литейных процессов : Учеб. пособие. – Л. : СЗПИ, 1988. – 89 с.: ил., табл.
12. Дембовский В. В. Автоматизация литейных процессов. – Л.: Машиностроение, 1989. – 264 с.: ил., табл.
13. Технологические измерения и приборы в металлургии : Методические указания к выполнению практических работ / Сост. В.В. Дембовский. – СПб.: СЗТУ, 2000. – 15 с.
14. Дембовский В.В. Моделирование и оптимизация технологических систем и процессов. Математическое моделирование литейных процессов с применением ЭВМ / Конспект лекций. – Л. : СЗПИ, 1991. – 60 с.: ил., табл.
15. Расчеты металлургических процессов на ЭВМ / Рыжонков В.И., Падерин С.Н., Серов Г.В., Жидкова Л.К. – М.: Металлургия, 1987. – 231 с.: ил., табл.
16. Физико- химические расчёты электросталеплавильных процессов / Григорян В.А., Стомахин А.Я., Пономаренко А.Г. и др. – М.: Металлургия, 1989. – 288 с.: ил., табл.

17. Компьютерное моделирование освежения оборотной песчано - жидкостекольной смеси / Боровский Ю.Ф., Дембовский В.В., Иоффе М.А. и др. // Литейное производство. – 1998. – № 6. – С.34 – 35.
18. Оборот литейных материалов и оптимизация их состава / Боровский Ю.Ф., Дембовский В.В., Иоффе М.А. и др. // Материаловедение и технология обработки материалов. Сб. статей. – СПб. : СЗТУ, 2000. – С. 8 – 14.
19. Динамика оборотных материалов и оптимизация их состава / Боровский Ю.Ф., Дембовский В.В., Иоффе М.А., Шергин И.В. // Тез. докл. на научно – практическом семинаре «Литейное производство сегодня и завтра». – СПб.: Ассоциация литейщиков Санкт-Петербурга, 2000. – С. 54.
20. Расчет состава оборотных материалов при изготовлении отливок / Боровский Ю.Ф., Дембовский В.В., Иоффе М.А. и др. // Проблемы машиноведения и машиностроения. Сб. статей. Выпуск № 23. – СПб.: СЗТУ, 2001. – С. 160 – 164.
21. Белай Г.Е., Дембовский В.В., Соценко О.В. Организация металлургического эксперимента : Учебн. пособие. – М.: Металлургия, 1993. – 256 с.: ил., табл.
22. Tikhonov O.N. and Dembovsky V.V. Automatic Process Control in Ore Treatment and Metallurgy. Part 1. Process Dynamics. – Cairo.: General Egyptian Book Organization, 1973. – 448 pp.
23. Глинков Г.М., Климовицкий М.Д. Теоретические основы автоматического управления металлургическими процессами. – М.: Металлургия, 1985. – 304 с.: ил., табл.
24. Автоматизация уравления производством : Методические указания к выполнению практических работ / Сост. В.В.Дембовский. – СПб.: СЗТУ, 2000. – 16 с.
25. Дьяченко В.Ф. Основные понятия вычислительной математики. М.: Наука, 1977. 79 с.: ил., табл.
26. Мак – Адамс В.Х. Теплопередача. Пер с англ. – М.: Металлургиздат, 1961. – 686 с.: ил., табл.
27. Зенкевич О., Морган К. Конечные элементы и аппроксимация. Пер. с англ. – М.: Мир. – 1986. – 318 с.: ил., табл.
28. Математика и САПР // Сборник в двух книгах, пер. с франц. Книга 1. – М.: Мир, 1986. – 206 с.: ил., табл.
29. Тихомиров М.Д. Сравнение тепловых задач в системах моделирования литейных процессов ПОЛИГОН и ProCast. //

Труды ЦНИИМ. Вып. 2. – СПб.: НТЦ «Информтехника», 1996. – С. 32 – 37.

30. Компьютеризация и автоматизация процесса проектирования отливок и изготовления оснастки / Кузнецов В.П., Абрамов А.А., Тихомиров М.Д. и др. // Литейное производство. – 1997. – №4. – С. 13 – 14.
31. Тихомиров М.Д. Основные аспекты решения тепловой задачи при моделировании литейных процессов // Литейное производство. – 1998. – №4. – С. 17 – 18.
32. Михайлов А.М. Теоретические основы литейного производ-. ства: Курс лекций. Часть 2. – М. : МИСиС, 1977. – 198 с.
33. Дембовский В.В., Кокоулин Е.Л., Яценко А.А. Развитие методов численного моделирования процесса кристаллизации сплавов при непрепывном литье // Применение ЭВМ для разработки технологических процессов литья, проектирования оснастки и анализа качества отливок. Тез. зональной научно – технической конференции. – Ярославль.: Андроповский авиационный технологический институт, 1987. – С. 31 – 32.
34. Кокоулин Е.Л., Дембовский В.В. Механизм образования внутренних трещин в промежуточной зоне непрерывнолитой сортовой заготовки // Сталь. – 1989. – №1. – С. 43 – 45.
35. Дембовский В.В., Яценко А.А. Численное моделирование процесса затвердевания заготовки в графитовом кристаллизаторе // Известия вузов. Серия Чёрная металлургия. – 1987. – №2. – С. 78 – 79.
36. Компьютерное моделирование процессов непрерывного литья / Дембовский В.В., Зинин Ю.Н., Сенченко В.Т., Яценко А.А. // Тез. докл. на Всероссийском научно – практическом семинаре ”Литейное производтво вчера, сегодня и завтра”. – СПб.: ЛенАЛ, 2000. – С. 64.
37. Применение персональных компьютеров в металлургии и литейном производстве: методические указания к практическим занятиям, курсовым и дипломным работам / Сост. В.В.Дембовский, Ю.Н.Зинин, М.А.Иоффе, В.Т.Сенченко. – СПб.: СЗПИ, 1998. – 22 с.
38. Решение технологических задач ОМД на микро – ЭВМ // Колмогоров В.Л., Паршаков С.И., Буркин С.П. и др. – М.: Металлургия, 1993. – 320 с.

ПРИЛОЖЕНИЯ

Приложение 1. Программа (код) учебного проекта VBAProject1.xls

```
User Forml

'РЕШЕНИЕ КВАДРАТНОГО УРАВНЕНИЯ
'Ax^2 + Bx + C = 0
'VBAProjectl.xls
'Процедура, выполняемая при нажатии на [CommandButton2]
      Private Sub CommandButton2_Click()
'Описание переменных
Dim A As Single, B As Single
Dim C As Single
Dim Dis As Single
Dim Rel As Single, Re2 As Single
Dim Iml As Single, Im2 As Single
Dim Rl As Single, R2 As Single
'Описание окон ввода
A = Val(TextBoxA.Text)
B = Val(TextBoxB.Text)
C = Val(TextBoxC.Text)
'Определение дискриминанта
Dis = B^2 – 4*A*C
'Продолжение вычислений
'Вычисление корней
If Dis > 0 Then
'Вывод первого действительного корня
Rl = (– B + Sqr(Dis)) / 2 / A
TextBox5.Text. = Str(Rl)
'То же - второго действительного корня
R2 = (– B – Sqr(Dis)) / 2 / A
TextBox6.Text = Str(R2)
Else
'Вычисление вещественных частей корней
Rel = – B / 2 / A : Re2 = Rel
'Вычисление мнимых частей корней
Iml = Sqr(Abs(Dis)) / 2 / A : Im2 = – Iml
'Вывод  первого комплексного корня
TextBox5.Text = Str(Rel)  & " + i *"  &  Str(Iml)
'То же - второго комплексного  корня
TextBox6.Text = Str(Re2)  & " – i *" &  Str(Abs(Im2) )
End If
    'Конец  процедуры, выполняемой при нажатии  на  [CommandButton2]
    End Sub
    'Процедура, выполняемая при нажатии [CommandButton1]
Private Sub CommandButton1_Click()
    End
'Конец  процедуры, выполняемой  при  нажатии  на  [CommandButton1]
```

```
End Sub
```

Приложение 2. Программы учебного проекта VBAProject2.xls

```
UserForm1 _ 1

'ПОСТРОЕНИЕ ГРАФИКА ПО РЕЗУЛЬТАТАМ ВЫЧИСЛЕНИЙ
'Программа VBAProject2.xls
'Запуск процедуры вычислений,
'выполняемой при нажатии на [CommandButton1]
'Процедура заключается в подготовке
'исходных данных для построения
'графиков функций
'      Z1 = (X^2 – 2*Y) / (2*Y^2) при X==const
'      Z2 = exp(Z1)
'в диапазоне варьируемых значений X
Private Sub CommandButton1_Click()
'Описание переменных удвоенной точности,
'где Аргумент1 = X, Аргумент2Нач - начальное значение Y,
'Аргумент2Кон - конечное значение Y,
'Аргумент2Шаг - шаг варьирования Y
Dim Аргумент1 As Double
Dim Аргумент2Нач As Double
Dim Аргумент2Кон As Double
Dim Аргумент2Шаг As Double
'Переменная n (число строк в таблице Excel,
'отводимых для записи данных) целого типа описана в модуле
'Описание строковых переменных
Dim Формула1 As String
Dim Формула2 As String
'Задание удвоенной точности при чтении данных
 'в окнах ввода пользовательской формы 1
With UserForm1
Аргумент2Нач = CDbl(TextBox1.Text)
Аргумент2Кон = CDbl(TextBox2.Text)
Аргумент2Шаг = CDbl(TextBox3.Text)
Аргумент1 = CDbl(TextBox6.Text)
Формула1 = Trim(CStr(TextBox4.Text))
Формула2 = Trim(CStr(TextBox5.Text))
End With
'Очистка столбцов от A до D включительно
'в таблице Excel
```

```
Range("A:D").Clear
'Задание ширины этих столбцов
Range("A:D").ColumnWidth = 10
 'Форматирование данных в тех же столбцах по центру
Range("A:D").Select
With Selection
.HorizontalAlignment = xlCenter
.VerticalAlignment = xlCenter
.WrapText = True
End With
'Назначение ячеек таблицы для записи данных.
'В столбец A записывается Аргумент1=X;
'в столбец B - Аргумент2 = Y от его начального
'Аргумент2Нач до конечного Аргумент2Кон
'значения с шагом Аргумент2Шаг;
'в столбец C – Формула1 для вычисления Z1;
'в столбец D - Формула2 для вычисления Z2.
'Первая строка отводится под заголовки
Range("A1").Value = "Аргумент 1"
Range("B1").Value = "Аргумент 2"
Range("C1").Value = "Функция 1"
Range("D1").Value = "Функция 2"
'Во вторую строку столбцов A и B - исходные данные
Range("A2").Value = Аргумент1
Range("B2").Value = Аргумент2Нач
'Назначение диапазона записи данных с заданным шагом
Range("B2").Select
Selection.DataSeries  Rowcol := xlColumns,  _
Type:=xlLinear, Step:=Аргумент2Шаг, Stop:=Аргумент2Кон
'Внимание!
'Пробел и следом за ним знак подчёркивания означают перенос
'продолжения строки.
'Подсчёт числа строк диапазона
n = Range("B2").CurrentRegion.Rows.Count
Range(Cells(2, 1), Cells(n, 1)).Value = Аргумент1
'Задание ячейки для записи 1-й формулы
Range("C2").Formula = Формула1
'Автозаполнение формулой диапазона 3-го столбца (т.е.C)
'со 2-й по n-ю строки включительно
Range("C2").AutoFill  _
 Destination:=Range(Cells(2, 3), Cells( n, 3)),   _
 Type:=xlFillDefault
```

```
'То же для 2-й формулы
Range("D2").Formula = Формула2
Range("D2").AutoFill _
Destination:=Range(Cells(2, 4), Cells(n, 4)),  _
Type:=xlFillDelault
'Конец процедуры, выполняемой при нажатии на [CommandButton1]
End Sub
'Процедура закрытия пользовательской формы 2,
'выполняемая нажатием на [CommandButton3]
Private Sub CommandButton3_Click()
UserFormI.Hide
'Конец процедуры закрытия формы
End Sub
'Процедура инициализации формы, в результате
'которой нажатие на клавишу [Enter]
'запускает процедуру, а нажатие на [Escape]
'вызывает её сброс (нажимать дважды).
Private Sub UserForm_Initialize()
CommandButton1.Default = True
ComniandButton3 . Cancel = True
'Вызов формы 1
UserForm1.Show
'Конец процедуры инициализации формы
End Sub
'Запуск процедуры построения графика средствами Excel,
'осуществляемой нажатием на [CommandButton4]
Private Sub CommandButton4_Click()
'Назначение числа строк данных, определённого выше
n = ActiveSheet.Cells(1, 1).CurrentRegion.Rows.Count
'Очистка области построения графика
'Вызов Мастера диаграмм Excel
'задание параметров графика,
'включая надписи по его осям
ActiveSheet.ChartObjects.Delete
ActiveSheet.ChartObjects.Add(195, 30, 200, 190).Select
ActiveChart.ChartWizard Source:= _
    Range(CelIs(2, 2), Cells(n, 4)),  _
    Gallery:=xlLine, Format:=4, PlotBy:=xlColumns,  _
    CategoryLabels : =1, SeriesLabels : =0 , HasLegend:=False,  _
    Title:="Z1=f1(Y)  и  Z2=f2(Y)", _
    CategoryTitle:="Y",  _
    ValueTitle:="Zl,Z2"
'Конец процедуры построения графика End Sub
```

```
End Sub
```

Module

```
Dim n As Integer ' Определение n как переменной целого типа
```

Приложение 3. Коды форм и модуля учебного проекта Project1VB

Form1

```
‘Вычисления и построение графика средствами VB
‘Project1VB
‘Процедура, выполняемая при нажатии на Command2 [ПУСК]
‘Расчёт данных для графика
    Private Sub Command2_Click()
‘Описание данных и массивов общего типа см. в модуле
    Dim X As Integer
‘Описание окон ввода данных, где
‘Nmin – начальное значение N,
‘DN – шаг варьирования N,
‘Nmax – конечное значение N:
    Nmin = Val(Text1.Text)
    DN = Val(Text2.Text)
    Nmax = Val(Text3.Text)
    M = Val(Text4.Text)
‘Цикл вычисления корней R1, R2 и функций Y1, Y2
‘при различных значениях N
    For I = Nmin To Mmax Step DN
    N(I) = I
    If N(I) ^ 2 – 4 * M < 0 Then
    MsgBox “Недопустимое соотношение исходных данных !”
    Exit Sub
    End If
    R1(I) = (– N(I) + Sqr(N(I) ^ 2 – 4 * M)) / (2 * M)
    R2(I) = (– N(I) – Sqr(N(I) ^ 2 – 4 * M)) / (2 * M)
    Y1(I) = 3 * Exp(7 * R1(I)) : Y2(I) = Exp(R2(I))
    Next I
‘Конец процедуры,выполняемой при нажатии на Command2
    End Sub
‘Процедура, выполняемая при нажатии на Command3 [ГРАФИК]
‘Построение графика
    Private Sub Command3_Click()
    Form1.Hide ‘Закрытие первой формы
    Form2.Show ‘Открытие второй формы
```

```
    Form2.Picture1.Scale (– 1.25, 5) – (Nmax+1, – 1.25)  'Задание масштаба
    Form2.Picture1.DrawWidth = 2  'Задание толщины линий
    For  X = Nmin  To  Nmax
    Form2.Picture1.Line (0, 0) – (Nmax, 0),  QBColor(1)  'Ось абсцисс
    Form2.Picture1.Line (0, 0) – (0, 4),  QBColor(1)  'Ось ординат
    Next  X
'Цикл построения кривых  Y1 = f (R1)  и  Y2 = f (R2)
    For  I = 1  To  Nmax  Step  DN
    Form2.Picture1.Line  (N(I – 1), Y1(I – 1)) – (N(I), Y1(I)),  QBColor(12)
    Form2.Picture1.Line  (N(I – 1), Y2(I – 1)) – (N(I), Y2 (I)),  QBColor(10)
    Next  I
'Построение горизонтальных линий сетки
    For  X = 0.5  To  4  Step  0.5
    Form2.Picture1.Line  (–  0.1, X) – (Nmax, X),  QBColor(8)
    Next  X
'Построение вертикальных линий сетки
    For  X = 0.5  To  Nmax  Step  0.5
    Form2.Picture1.Line  (X, –0.05) – (X, 4),  QBColor(8)
    Next  X
'Оцифровка делений оси абсцисс
    For  X = Nmin  To  Nmax
    Form2.Picture1.CurrentX = X – 0.2
    Form2.Picture1.CurrentY =  – 0.3
    Form2.Picture1.Print  X
    Next  X
'Оцифровка делений оси ординат
    For  Z = 0.5  To  4  Step  0.5
    Form2.Picture1.CurrentX =  – 1
    Form2.Picture1.CurrentY =  Z + 0.15
    Form2.Picture1.Print  Z
    Next  Z
'Обозначения элементов графика
    v = 0.075*Nmax
    Form2.Picture1.CurrentX = v
    Form2.Picture1.CurrentY =  4.75
    Form2.Picture1.Print  "Обозначения"
    Form2.Picture1.Line  (v + 4, 4.5) – (v + 5, 4.5),  QBColor(12)
    Form2.Picture1.CurrentX =  6.25
    Form2.Picture1.CurrentY =  4.75
    Form2.Picture1.Print  "Y1;"
    Form2.Picture1.Line  (7, 4.5) – (8, 4.5),  QBColor(10)
    Form2.Picture1.CurrentX =  8.25
    Form2.Picture1.CurrentY =  4.75
    Form2.Picture1.Print  "Y2"
    Form2.Picture1.CurrentX =  Nmax – 1
    Form2.Picture1.CurrentY =  – 0.75
    Form2.Picture1.Print "N"
    Form2.Picture1.CurrentX =  – 0.75
    Form2.Picture1.CurrentY =  4.5
```

```
        Form2.Picture1.Print  "Y1, Y2"

    'Конец процедуры, выполняемоц при нажатии на Command3 [ГРАФИК]
        End Sub
    'Процедура, выполняемая при нажатии на Command1 [СБРОС]
        Private Sub Command1_Click()
        End
    'Конец процедуры, выполняемой при нажатии на Command1 [СБРОС]
        End Sub
```

Form2

```
    'Процедура, выполняемая при нажатии на Command1 [ВОЗВРАТ]
        Private Sub Command1_Click()
            Form2.Hide
            Form1.Show
        End Sub
    'Конец процедуры, выполняемой при нажатии на Command1 [ВОЗВРАТ]
```

Module1

```
    'Описание исходных данных и массивов,
    'доступных из любой процедуры и формы
        Public Nmin As Single, Nmax As Single
        Public DN As Single, M As Single
        Public R1(200) As Single, R2(200) As Single
        Public Y1(200) As Single, Y2(200) As Single
        Public N(200) As Integer
```

ОГЛАВЛЕНИЕ

Дембовский Владислав Владиславович

Компьютерные технологии в металлургии

и литейном производстве

Учебное пособие

Редактор И.Н.Кочугина

Сводный темплан 2002 г.

Лицензия № 020308 от 18.03.97.

Подписано в печать Формат 60 × 84 1 / 16

Б. Кн. - журн. П. л. 9,45. Б. л. 4,875. РТП РИО СЗТУ

Тираж . Заказ .

РИО СЗТУ, член Издательско - полиграфической ассоциации

вузов Санкт - Петербурга

191186, Санкт – Петербург, ул. Миллионная, 5