ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ ЗАДАЧ ЧИСЛЕННОГО АНАЛИЗА

С.В. Морозов, В.А. Семенов

Рассматриваются вопросы объектно-ориентированного программирования задач анализа функций одной и нескольких переменных. Строится классификация математических функций, составляющая практическую основу для объектной реализации единой иерархии функциональных классов. Обсуждаются возможности организации классов на языке Си++ и результаты временного тестирования различных функциональных реализаций.

1. Введение

В настоящее время объектно-ориентированный подход (ООП) находит применение в самых разнообразных областях программирования [1]. Предмет настоящей статьи составляет исследование возможностей применения ООП к программированию довольно неожиданной для подобных приложений области вычислительной математики — анализа функций одной и нескольких переменных. Теоретическое объектное исследование функциональных проблем и связанных с ними численных постановок задач интерполяции и приближения, дифференцирования, интегрирования, решения нелинейных уравнений и математического программирования обнаруживает важные преимущества объектной технологии, обуславливающие ее активное внедрение в данную область.

В работе строится единая математическая классификация функциональных объектов, которая может использоваться в качестве конструктивной основы для многочисленных программных реализаций. Рассматриваются различные способы организации функциональных классов. Целесообразность применения объектной технологии в данной области подтверждается результатами практической реализации функциональных классов на языке Си++ и их временного тестирования. Особое внимание при этом уделяется вопросам эффективности вычислений функций и их производных, являющимися базовыми операциями для большинства вычислительных методов.

2. Объектный анализ функциональных проблем

На наш взгляд функция (или функциональное отображение) как фундаментальное математическое понятие является одним из главных конструктивных объектов вычислительной математики. С функциональными объектами связаны практически все классические постановки нелинейных алгебраических и дифференциальных задач и применяемые для их решения вычислительные алгоритмы. При этом большинство численных методов могут быть реализованы именно как методы соответствующих функциональных классов. Обсудим такую возможность, следуя приведенному ниже краткому перечню наиболее характерных формулировок задач анализа [2, 3].

Задача интерполяции обычно ставится следующим образом. Пусть известны значения f_k некоторой функции $f(x)$ в точках x_k, $k=1, \ldots, n$. Требуется определить параметры приближающей функции $g(x, a)$ такие, что $g(x_k, a)=f_k$, $k=1, \ldots, n$.

Задача наилучшего приближения в гильбертовом пространстве $\mathbf{H}$ состоит в том, чтобы для заданного функционального элемента $f \in \mathbf{H}$ найти такой обобщенный многочлен g, являющийся линейной комбинацией $g_k \in \mathbf{H}$, $k=1, \ldots, n$, для которого отклонение $\|f - g\|$ было бы минимальным.

Задачи численного дифференцирования и интегрирования могут быть сформулированы как задачи вычисления производной и определенного интеграла с заданным порядком вычислительной погрешности по значениям функции f_k в точках x_k, $k=1, \ldots, n$.

Решение алгебраических уравнений, определяемых отображением $F(x)$: $\mathbf{D} \subset \mathbf{R}^n \to \mathbf{R}^n$, состоит в нахождении такого $x_0 \in \mathbf{D}$, для которого $F(x_0)=0$. Частным, но практически важным случаем задачи является решение линейных систем уравнений.

Задачи математического программирования в наиболее общей постановке формулируются как условная минимизация функционала на некоторой области $f(x)$: $\mathbf{D} \subset \mathbf{R}^n \to \mathbf{R} \to \min$, $g(x)$: $\mathbf{D} \subset \mathbf{R}^n \to \mathbf{R}^m \le 0$, $x \in \mathbf{D} \subset \mathbf{R}^n$. Приведенная постановка при конкретизации дополнительных свойств нелинейного функционала, ограничений и области определения естественным образом редуцируется к задачам безусловной оптимизации, а также к задачам линейного, квадратичного и выпуклого программирования.

Нетрудно видеть, что все основные виды задач формулируются с использованием следующего ограниченного множества понятий: функция, вектор, область пространства, сетка и сеточная функция. С данными понятиями мы будем связывать в дальнейшем эквивалентную объектную систему:

FunctionAnalysis = {Function, Domain, Mesh, MeshFunction}.

Объекты, соответствующие понятиям вектора и матрицы, традиционно будем относить к самостоятельному разделу линейной алгебры:

LinearAlgebra = {Vector, Matrix}.

Этими же объектами в основном оперируют и вычислительные методы, применяемые для их решения. Данное обстоятельство в значительной степени объясняется существенной взаимной редуцируемостью нелинейных задач.

В самом деле, результаты и методы теории интерполирования, кроме непосредственных приложений, применяются в численном анализе при конструировании формул дифференцирования и интегрирования, при построении сеточных и разностных аналогов уравнений математической физики. Поскольку возможна постановка таких задач с условиями обеспечения численной погрешности заданного порядка у конструируемых дифференциальных и квадратурных формул, такой подход обеспечивает желаемую общность и применяется в самых разнообразных ситуациях.

Решение задач приближения используется, в частности, при составлении стандартных программ вычисления элементарных и специальных функций, когда требуется обеспечить необходимую точность в заданном классе аппроксимирующих функций. Подобные вычисления часто основываются на известных разложениях в ряды, а вычисления специальных функций, кроме того, могут реализовываться и путем непосредственного применения квадратур.

Задачи решения систем нелинейных алгебраических уравнений возникают в различных приложениях, часто в связи с необходимостью решения разностных уравнений, полученных в результате дискретизации дифференциальных задач. Применяемые при этом численные методы, как правило, используют возможности линейной или более сложных аппроксимаций нелинейного отображения, основанных на его аналитическом или численном дифференцировании. Мера доверия к подобным преобразованиям контролируется специальным функционалом невязки, к задаче минимизации которого в конечном счете и сводится решение уравнений.

Близкие методы используются и при решении задач безусловной нелинейной оптимизации. Одним из непосредственных способов их решения является сведение к системе нелинейных уравнений, определяющей необходимые стационарные точки. Для решения задач условной оптимизации применяются более сложные подходы, включающие в себя дополнительные ступени алгоритмической редукции. Наиболее распространенными из них являются сведение исходной задачи к задачам безусловной оптимизации или сразу к задачам линейного или квадратичного программирования.

Таким образом, для постановки нелинейных проблем и их решения, выражающегося в основном во взаимной многоуровневой алгоритмической редукции, предлагаемый понятийный аппарат множеств FunctionAnalysis и LinearAlgebra является достаточным. Это обстоятельство позволяет полагать, что соответствующая программная реализация данной объектной системы, используемая в качестве математического инструментального ядра, позволит унифицировать разработку программных средств для решения многообразных вычислительных задач.

3. Математическая классификация функций

Предлагается следующая математическая классификация функциональных объектов, приведенная на рис. 1. Данная классификация может служить практической основой для реализации единой целостной иерархии функциональных классов.

Левая часть иерархии представляет формальную классификацию функций по размерности значения и аргумента, а правая — по их математическим свойствам.

Целесообразность такой организации объясняется прежде всего необходимостью определения для любого функционального объекта операции вычисления значения (в дальнейшем определяемого оператором ( )), что требует строгой определенности типов аргумента и возвращаемого значения. Данное требование обеспечивается разделением функций на скалярные и векторные классы и делением последних в свою очередь на функции одной и нескольких переменных.

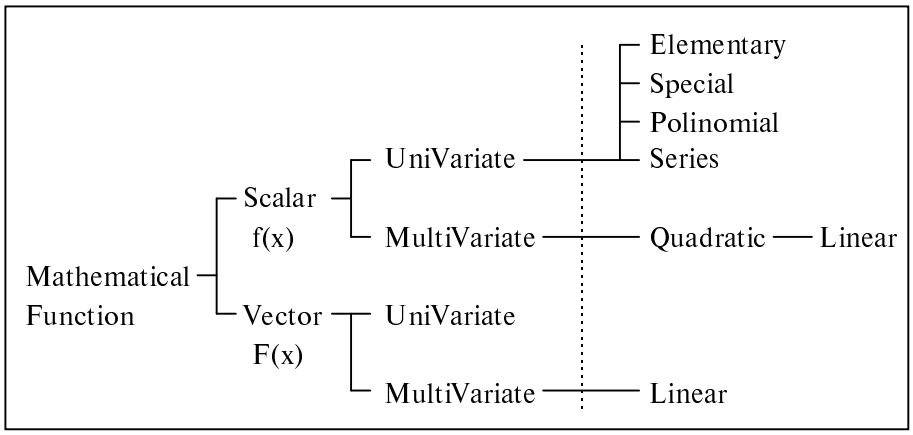

Рис. 1. Математическая классификация функций

Естественной может показаться попытка рассматривать скалярные функции как частный случай векторных, а функции одной переменной как функции нескольких переменных (объектная технология обеспечивает возможность такой реализации с использованием механизма перегрузки методов класса). Однако при этом неизбежна путаница с семантикой оператора ( ), обязательного для функциональных объектов. Другой важной мотивацией может служить и тот факт, что основные вычислительные методы внутренне организованы и предназначены только для решения определенных классов задач, связанных с функциональными объектами определенной размерности.

В самом деле, в функциональной иерархии, например, методы оптимизации могут инкапсулироваться только скалярными классами, определяющими минимизируемый функционал, но никак не векторными классами. Наоборот, методы решения систем нелинейных алгебраических уравнений могут быть определены только как методы векторных классов.

Существенным здесь является наблюдение, что несмотря на то, что задачи многомерной оптимизации, как и задачи решения нелинейных алгебраических уравнений, допускают алгоритмическую редукцию к соответствующим задачам одномерной оптимизации, сами многомерные методы в большинстве случаев не сводятся к соответствующим одномерным версиям. Обычно для функций одной переменной применяются специальные вычислительные подходы, учитывающие данный размерный фактор.

Правую часть иерархии будем строить путем принятия специальных математических свойств функций в качестве классификационных критериев. Результатом такого рассмотрения, в частности, может быть конкретизация скалярной функции одной переменной в виде множеств элементарных, специальных функций, полиномов, рядов, порождающих при объектной реализации соответствующие частные коллекции функциональных классов.

Важным с практической точки зрения может оказаться выделение квадратичных и линейных функционалов из скалярной функции многих переменных, которые определяют многочисленные постановки задач квадратичного и линейного программирования. Поскольку для решения задач таких классов разработаны специальные подходы, интересной вычислительной парадигмой становится возможность одновременного решения одной и той же численной задачи универсальным вычислительным методом, который инкапсулируется общим функциональным классом, и специальными методами, ориентированными на частные функциональные реализации.

Аналогичный смысл имеет и выделение линейных отображений из класса векторной функции многих переменных. В этом случае методы решения нелинейных алгебраических

уравнений, реализованные как методы класса нелинейного отображения, могут быть применены и к линейным системам уравнений, определяемым соответствующими матрицами.

Конечно, на практике для решения задач линейной алгебры применяются специальные методы. Тем не менее, общность универсальных подходов, реализованных уже на верхних уровнях функциональной иерархии и применимых ко всем частным постановкам, может оказаться не менее привлекательной, чем несколько более эффективная реализация методов линейной алгебры. Для итеративных методов различия в эффективности, по-видимому, будут самыми незначительными, поскольку затраты на линеаризацию отображения отсутствуют, а техника итерирования в обоих случаях приблизительно одинакова.

Рассмотрим более подробно иерархию классов скалярных функций одной переменной, поскольку большинство вычислительных процедур реализуется как методы данных классов. При этом будем следовать принятому математическому делению функций на элементарные, специальные функции, многочлены и ряды.

3.1. Элементарные функции

Группу элементарных функциональных классов составляют тригонометрические и гиперболические функции синуса, косинуса, тангенса, арксинуса, арккосинуса, арктангенса, степенная функция и ее частные случаи: константа, линейная и квадратичная функции, а также функции квадратного корня, экспоненты, натурального и десятичного логарифма (см. рис. 2).

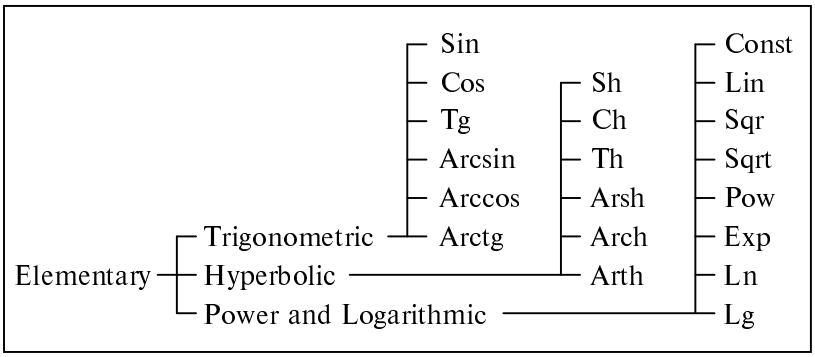

Рис. 2. Классификация элементарных функций

Важной особенностью объектной реализации элементарных функций является возможность вычисления значений всех производных и первообразной, аналитически выражаемых через элементарные функции. Данное свойство является чрезвычайно полезным при дифференцировании произвольных функций, так как правила дифференцирования допускают простую формализацию и могут быть программно поддержаны. Для непосредственного вычисления значений функций может использоваться разложение в ряд Тейлора либо специальная интерполяция, реализуемые методами соответствующих функциональных классов. Аналитические методы вычисления нулей и экстремумов элементарных функций, оказывающиеся тривиальными, тем не менее, могут применяться при тестировании соответствующих вычислительных процедур общего класса.

3.2. Специальные функции

Более глубокую классификацию допускают специальные функции [4]. В отличие от предыдущего случая, выделение специальных семейств функций может носить неформальный проблемно-ориентированный характер и использоваться при реализации общих математических свойств и вычислительных методов. Например, иногда семейства функций строятся как фундаментальные системы решений некоторых видов дифференциальных уравнений и в этом случае могут определять соответствующие дифференциальные уравнения и технологию их решения.

На рис. 3 и при дальнейшем рассмотрении будем использовать обозначения функций, близкие к общепринятым.

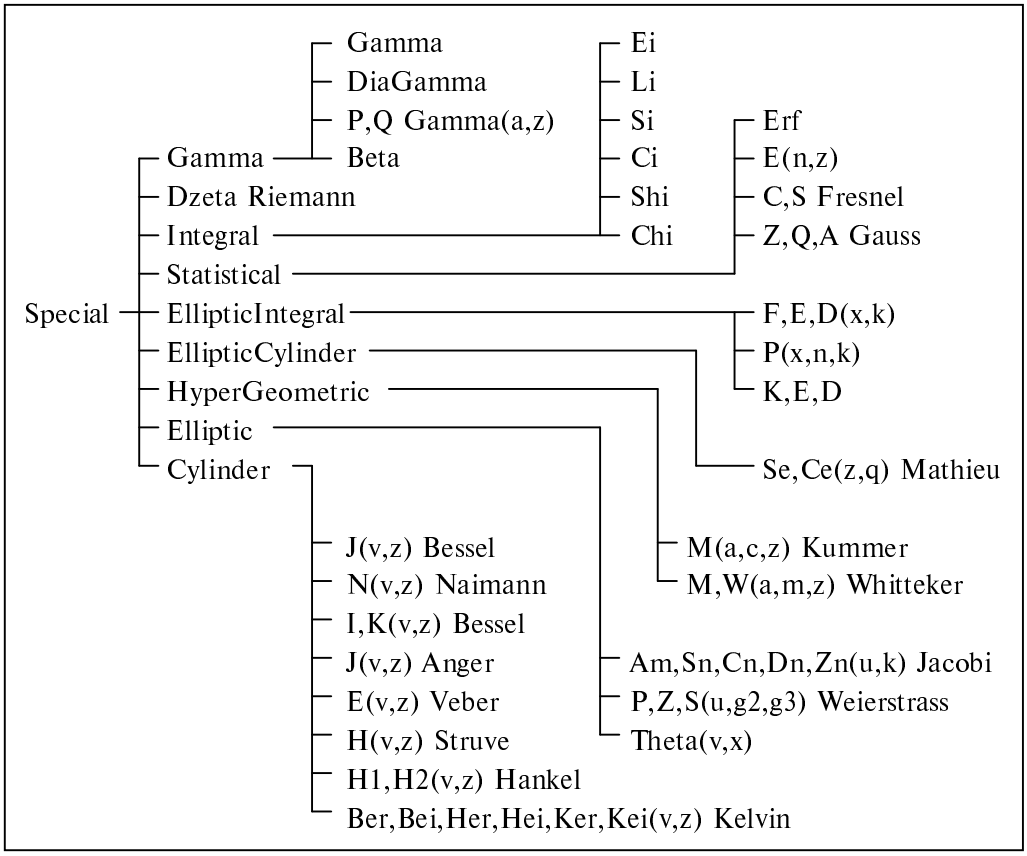

Рис. 3. Классификация специальных функций

Семейство гамма-функций составляют собственно сама гамма-функция Gamma, логарифмическая производная гамма-функции (или диагамма) DiaGamma, а также неполные гамма-функции P, Q и бета-функция Beta.

Семейство интегральных функций образуют интегральная показательная функция Ei, интегральный логарифм Li, интегральные тригонометрические и гиперболические функции синуса и косинуса Si, Ci и Shi, Chi соответственно.

К статистическому семейству функций следует отнести прежде всего интеграл вероятности Erf, обобщенную функцию ошибок E и интегралы Френеля C и S. Часто используются близкие с интегралом вероятности функции гауссовского распределения Z, Q и A.

Функциональный класс дзета-функции Римана имеет самостоятельное значение и может рассматриваться как целое функциональное семейство.

Семейство эллиптических интегралов составляют неполные эллиптические интегралы 1, 2, 3 родов в нормальной форме Лежандра F, E, P, D и полные эллиптические интегралы K, E, D. Заметим, что полные интегралы получаются в результате фиксирования предела интегрирования и могут рассматриваться как частные параметрические случаи неполных интегралов.

Класс эллиптических функций составляют амплитуда Якоби Am, ее тригонометрические варианты Sn, Cn, Dn, дзета-функция Якоби Zn, а также эллиптические функции Вейерштрасса P, Z, S и тэта-функция Teta. Поскольку эллиптические функции являются обратными к эллиптическим интегралам, одним из непосредственных путей реализации данных классов может стать определение их через операцию обращения произвольной функции, заданной в данном случае ссылкой на соответствующий класс эллиптического интеграла.

Класс цилиндрических функций составляют функции Бесселя 1, 2, 3 родов, а именно функция Бесселя J, функция Неймана N и функции Ганкеля H1, H2, модифицированные функции Бесселя I, K и функции Кельвина Ber, Bei, Her, Hei, Ker, Kei, а также связанные с ними функции Ангера J, Вебера E и Струве H.

Функции Матье Se, Ce образуют семейство функций эллиптического цилиндра.

Наконец, семейство гипергеометрических функций составляют функция Куммера M и функции Уиттекера M, W.

Важно заметить, что некоторые из перечисленных выше специальных функций являются функциями двух и более переменных и, следуя приведенной выше классификации, должны были бы располагаться в других ветвях иерархии. Тем не менее, обычно в приложениях исследуется их поведение в зависимости от одного основного аргумента при фиксировании других.

В этих случаях естественнее выглядит реализация, при которой дополнительные аргументы рассматриваются как внутренние параметры функционального семейства, а сами объекты являются функциями одной переменной. Более того, в ряде случаев функции одной переменной получаются в результате несложной редукции какой-либо функции нескольких переменных, и тогда возможно наследование частных классов от обобщенной функции.

Рассмотрим, например, неполные гамма-функции, являющиеся функциями двух переменных. При подходящим образом фиксированных значениях одной переменной можно получить многие специальные функции, в частности, интегральную показательную функцию, интегральный логарифм, интегральные синус и косинус, интеграл ошибок и интегралы Френеля. Аналогично, конфлюэнтные гипергеометрические функции содержат как частные случаи функции Бесселя, функции параболического цилиндра, неполные гамма-функции и полиномы Лаггера.

Несмотря на возможности более глубокого наследования специальных функциональных классов с использованием параметрической редукции, в дальнейшем мы ограничимся классификационными построениями, прежде всего, исходя из проблемной ориентации функциональных семейств.

К особенностям реализации специальных функциональных классов следует отнести возможности расширения множества наследуемых методов. Прежде всего, техника вычисления значений самой функции может строиться принципиально различным образом и представляться разными функциональными методами. Например, кроме разложения в ряд в окрестности какой-либо точки, возможно использование асимптотик при больших значениях аргумента, а также непосредственное применение квадратурных формул.

Важную возможность эффективного вычисления специальных функций предоставляет существование так называемых частных значений, определяемых аналитическим образом, но только в особых точках. Перегрузка оператора ( ) для множеств особых точек, иногда являющихся множествами натуральных или целых чисел, позволит упростить технику вычисления функций в таких случаях. Аналогично, значительно сократить ресурсы на вычисление функции в ряде случаев можно при использовании известных тождеств или функциональных уравнений, связывающих значения функций и их производных в точках при определенных значениях параметров.

В отличие от элементарных, определение нулей и экстремумов специальных функций, как правило, носит вычислительный характер. Поэтому для ее решения могут быть применены соответствующие численные процедуры родительского класса. Важную роль, тем не менее, при решении таких задач в глобальной постановке может сыграть информация о локализации таких точек, которая в ряде случаев известна и может использоваться общими методами базового класса.

3.3. Обобщенные полиномы

Важнейшую для численного анализа и многих приложений группу скалярных функций одной переменной составляют полиномиальные функции. Обычно полиномы строятся как линейные комбинации базисных функций или разложения по ним. Поскольку система базисных функций может быть выбрана достаточно произвольным образом, с рассматриваемыми полиномиальными объектами будем связывать и соответствующие им базисные системы, хотя возможно их представление и самостоятельными отдельными классами.

На рис. 4 приведена одна из возможных иерархий полиномиальных классов. В зависимости от того, какие признаки полиномов считать классификационными и, соответственно, какие данные и методы относить к наследуемым, структура иерархии может значительно видоизменяться. Поэтому будем следовать более существенной методической стороне, а не возможностям формального наследования.

Целесообразно разделить все многочлены на три класса, а именно на степенные, специальные и ортогональные системы. Степенные многочлены используются в основном для интерполяции, ортогональные системы — прежде всего для аппроксимации, специальные многочлены — для тех и других задач.

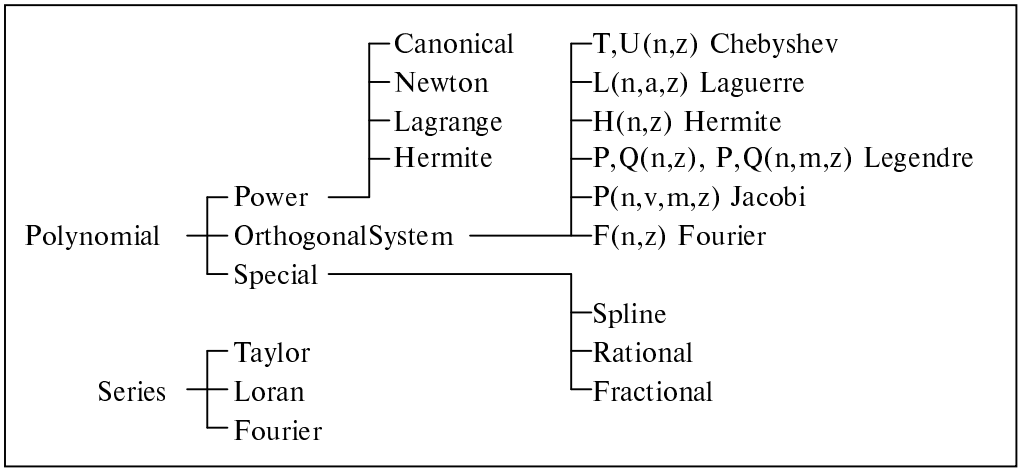

Рис. 4. Классификация полиномов и рядов

Наиболее простым примером многочленов, непосредственно появляющимся в разнообразных приложениях, является обычный степенной многочлен. Семейство степенных полиномов, используемых для алгебраического интерполирования, составляют также многочлены Ньютона, Лагранжа и многочлен Эрмита, применяемый при интерполяции с кратными узлами. Заметим, что все они по сути представляют один и тот же математический объект, но реализуют его различным образом в соответствии со способами организации интерполирующих методик.

Важную группу многочленов, используемых в задачах приближения функций, образуют так называемые ортогональные многочлены Чебышева, Лаггера, Эрмита, Лежандра и Якоби [5]. К этому же семейству можно отнести и тригонометрический многочлен Фурье, удовлетворяющий условиям ортогональности и находящийся в соответствии с общими методическими построениями данного семейства.

Наконец, к группе специальных многочленов отнесем кусочно-полиномиальные функции или сплайны, рациональные степенные функции и цепные дроби. Интерполяция сплайнами, прежде всего кубическими, и аппроксимация рациональными многочленами находят самое широкое практическое применение.

Обсудим особенности реализации методов рассматриваемых семейств функциональных классов, заметно отличающихся от предыдущих случаев. Поскольку с каждым из классов связывается соответствующая система базисных функций, задачи интерполяции и аппроксимации естественным образом реализуются как методы или конструкторы данных классов, осуществляющие необходимые преобразования по заданной сеточной функции. При этом в большинстве случаев удается оценить погрешность приближения через остаточные члены разложения, что особенно важно для пользователей и может быть поддержано соответствующими программными средствами.

Методы вычисления значений функций, а также методы дифференцирования и интегрирования реализуются аналитически, что, безусловно, важно для уменьшения погрешности. Для вычисления нулей и экстремумов все же требуется применение соответствующих численных методов. Однако для отыскания корней произвольного степенного многочлена возможно применение специальных численных методик, более эффективных, чем соответствующие наследуемые процедуры скалярного класса.

Для вычисления значений ортогональных многочленов, а также при их дифференцировании и интегрировании, могут использоваться известные рекуррентные соотношения между полиномами близких порядков. Нули и экстремумы таких функций, как правило, выражаются аналитическим образом. Последнее обстоятельство значительно упрощает конструирование многочленов, обладающих так называемым экстремальным свойством и обеспечивающих минимум ошибки при заданном порядке интерполирующего полинома.

Для классов ортогональных многочленов репертуар методов может быть значительно расширен. В частности, для каждого ортогонального семейства возможны определение весовой функции, вычисление коэффициентов по полиному, заданному в другом ортогональном базисе, вычисление старших коэффициентов в степенном базисе, которое реализуется по известным явным формулам. Важной особенностью ортогональных семейств является то, что они удовлетворяют линейным дифференциальным уравнениям второго порядка с переменными коэффициентами, к которым приводят различные физические постановки. Поэтому реализация таких классов может включать методы конструирования разрешимых дифференциальных уравнений и их решения.

3.4. Функциональные ряды

В заключение рассмотрим функциональную иерархию рядов, приведенную на рис. 4.

Организация классов, реализующих ряды Тейлора, Лорана и Фурье, в основном совпадает с классами соответствующих многочленов. Главным отличием является способ представления коэффициентов ряда. Если для произвольных многочленов необходимо явное хранение коэффициентов, то для рядов достаточно иметь лишь функциональные методы их определения, причем возможно с использованием известных рекуррентных соотношений. Поскольку необходимая операция доступа к коэффициентам многочлена не определена для рядов, а техника вычисления коэффициентов и элементов ряда не распространяется на многочлены, применение наследования близких функциональных объектов здесь кажется неоправданным и поэтому для рядов строится самостоятельная иерархия.

4. Особенности реализации функциональных классов

Рассмотрим организацию интерфейсов абстрактных классов, образующих верхний уровень иерархии скалярных функций.

Согласно приведенным в предыдущем разделе соображениям реализуются отдельные иерархии для функций одной и нескольких переменных, вершинами которых являются абстрактные классы ScalarUniVariateFunction и ScalarMultiVariateFunction соответственно. Их интерфейсы включают следующие методы:

— оператор ( ) вычисления функции, который является чисто виртуальным и должен переопределяться во всех конкретных функциональных классах;

— дифференцирование (вычисление первой, второй и n-ой производных функции одной переменной, градиента и гессиана функции нескольких переменных);

— интегрирование (вычисление квадратур и кубатур);

— поиск нулей и экстремумов для функции одной переменной;

— условная и безусловная оптимизация для функции нескольких переменных.

На данном уровне иерархии появляются только универсальные реализации вышеперечисленных методов. При необходимости они могут переопределяться в частных функциональных классах с учетом их конкретных особенностей. Также следует заметить, что репертуар методов для конкретных функций может быть значительно расширен. Например, методы интерполяции и аппроксимации естественным образом реализуются как методы классов многочленов.

Таким образом, для самой простой реализации некоторой пользовательской функции достаточно построить новый класс, являющийся наследником одного из уже имеющихся в функциональной иерархии, и переопределить оператор ( ). В тех случаях, когда требования к точности и эффективности функционального класса достаточно велики, возможно переопределение и других методов. В качестве примера приведем реализацию элементарной тригонометрической функции sec(x) на языке Си++:

```
typedef float Value;
class Secant : public ScalarUniVariateFunction
{
public:
  Value operator() (Value x) { return 1/Cos(x); }
  Value Diff1(Value x) { return Tg(x)/Cos(x); }
  Value Diff2(Value x)
```

```
    { Value v=Tg(x); return (2*v*v+1)/Cos(x); }
  Value Intgr(Value a, Value b)
    { return Ln((Tg(b)+1/Cos(b))/(Tg(a)+1/Cos(a))); }
};
```

В данном случае первая и вторая производные, а также интеграл функции sec(x) будут вычисляться аналитически. Аналогичным образом можно было бы реализовать и методы вычисления экстремумов и нулей функции. Поскольку методы вычисления производных старше второй, нулей и экстремумов в классе Secant не переопределены, то, следовательно, будут использоваться их универсальные численные реализации, наследуемые от базового класса.

5. Класс интерпретатора функций

Описываемая иерархия предоставляет пользователям еще одну возможность реализации функций — их записи в виде выражений, использующих операции непосредственно над функциональными объектами. При обработке подобного выражения происходит не вычисление его значения, а конструирование объекта одного из специальных классов интерпретаторов функций. В настоящий момент реализованы интерпретаторы скалярных функций одной и нескольких переменных: ScalarUniVariateFunctionInterpretator и ScalarMultiVariateFunctionInterpretator, являющиеся наследниками соответствующих абстрактных функциональных классов.

Вычисление функционального выражения осуществляется с помощью оператора ( ) путем интерпретации формы внутреннего представления данного выражения в виде двоичного дерева [6]. Выражение преобразуется во внутреннюю форму при конструировании соответствующего объекта класса интерпретатора функций по следующим правилам:

— любая функция является листом дерева;

— операции порождают в порядке, определяемом правилами алгебры, двоичное дерево, в котором левая ветвь соответствует первому операнду, а правая — второму операнду.

Была также реализована вторая версия классов интерпретаторов функций, где в качестве внутренней формы функциональных выражений использовалось линейное представление дерева в виде матрицы троек «операция — операнд 1 — операнд 2». Операции в матрице располагаются в том же порядке, как и в постфиксной записи выражения [6].

Самое главное достоинство функциональных интерпретаторов — это возможность с их помощью аналитически вычислять производные, не указывая выражений для их подсчета. Для этого лишь достаточно иметь полиморфные методы аналитического дифференцирования во всех классах, реализующих функции-операнды. В данном случае вычисление производных осуществляется по правилам дифференцирования суммы, разности, произведения, частного, композиции двух функций, а также обратной функции. Все эти правила допускают простую формализацию и могут быть программно поддержаны внутри функционального интерпретатора.

В отличие от дифференцирования, интегрирование, а также поиск нулей и экстремумов функциональных выражений осуществляется с помощью численных методов, так как их аналитическая реализация на основе формальных правил невозможна. Следует заметить, что для суммы и разности функций возможно применение правил интегрирования, а для произведения и частного — правил нахождения нулей. Однако в общем случае частичная аналитическая реализация данных методов вряд ли улучшит точность результата, а будет гораздо менее эффективна по быстродействию, чем соответствующая численная реализация.

Запись функционального выражения похожа на запись обычного математического выражения, однако есть несколько существенных отличий, о которых следует особо упомянуть:

— названия стандартных элементарных функций соответствуют принятым в математике, но начинаются с заглавной буквы: Exp, Ln, Sin, Cos, Tg, Arcsin, Arccos и т. д. (см. рис. 2), правила записи некоторых функций приведены в табл. 1;

— простой аргумент функции в выражении не указывается, например, *exp*(*x*) запишется как Exp;

— если аргумент является элементом вектора, то его индекс указывается после имени функции в квадратных скобках, например, $sin(x_2)$ запишется как Sin[2] (следует помнить, что нумерация векторных элементов начинается с нуля);

— сложный аргумент представляется с помощью операции композиции функций, например, $cos(2 \cdot x)$ запишется как Cos(2*Lin);

— в выражениях наряду с именами элементарных функций могут участвовать действительные константы и переменные, причем последние воспринимаются как константы, значения которых равны значениям соответствующих переменных в момент конструирования функционального интерпретатора;

— правила записи операций указаны в табл. 2.

Таблица 1

Правила записи некоторых стандартных функций

Функция	Запись в выражении
$f(x) = x$	Lin
$f(x) = x^2$	Sqr
$f(x) = \sqrt{x}$	Sqrt
$f(x) = x^y$	(Pow^y)

Таблица 2

Сводка функциональных операций

Операция	Название	Соответствующая математическая запись
−f	унарный минус	$-f(x)$
f+g	сложение	$f(x) + g(x)$
f−g	вычитание	$f(x) - g(x)$
f*g	умножение	$f(x) \cdot g(x)$
f/g	деление	$f(x) / g(x)$
f(g)	композиция	$f(g(x))$
!f	производная	$f'(x)$
f%a	интеграл	$\int_a^x f(t)\ dt$
~f	обратная функция	arc $f(x)$

В качестве примера использования классов интерпретаторов приведем реализацию с их помощью следующих функций:

— секанса;

— полинома пятой степени, представленного схемой Горнера;

— функции Розенброка двух переменных [7];
— тригонометрической функции двух переменных [7].

```
// секанс
ScalarUniVariateFunctionInterpretator func1=1/Cos;
// полином пятой степени
ScalarUniVariateFunctionInterpretator func2=
  (Lin*(Lin*(Lin*(Lin*(Lin+1)+1)+1)+1)+1);
// функция Розенброка
ScalarMultiVariateFunctionInterpretator func3=
  Sqr(1-Lin[0])+100*Sqr(Lin[1]-Sqr[0]);
// тригонометрическая функция
ScalarMultiVariateFunctionInterpretator func4=
  Sqr(2*Sin[0]+Cos[0]-Cos[1])+Sqr(2*Sin[1]+3*Cos[1]-Cos[0]-2);
```

После конструирования объектов классов интерпретаторов функций можно проводить их вычисление, дифференцирование, интегрирование и т. п.

6. Временное тестирование функциональных реализаций

Сравним различные функциональные реализации по времени выполнения базовых для большинства нелинейных задач операций вычисления функций и их производных.

Для тестирования были выбраны четыре функции, описанные в разделе 5. Сравнению подлежали следующие способы их реализации:

— в качестве самостоятельного класса, принадлежащего функциональной иерархии;

— путем записи функционального выражения, то есть с помощью класса интерпретатора функций, причем использовались обе версии интерпретаторов как с применением дерева в качестве внутренней формы представления выражения, так и с применением матрицы;

— традиционная процедурная реализация функции.

Таблица 3

Результаты тестирования операции вычисления функций (в секундах)

Способ реализации	Функция			
	секанс	полином	Розенброка	тригонометрическая
Класс	2,76	0,36	0,49	11,86
Выражение (дерево)	3,09	8,54	2,24	32,44
Выражение (матрица)	3,88	3,87	3,04	22,14
Процедура	3,16	0,36	0,26	11,61

Все вышеперечисленные способы предоставляют возможность вычисления значений производных по аналитическим выражениям. Кроме того, имеется возможность численного дифференцирования с помощью методов, реализованных в базовых функциональных классах.

Тестирование проводилось на рабочей станции Sun SPARCstation 1+ с тактовой частотой 20 МГц с использованием компилятора GNU C++ 2.5.8 (опции −O2, −ansi, −pedantic). Для подсчета времени использовалась стандартная функция ОС UNIX times с разрешающей способностью 0,01 с.

Результаты тестирования сведены в табл. 3 и 4, в которых перечислены способы реализации функций и приведены соответствующие им данные о времени выполнения 100 тыс. операций вычисления функций (табл. 3) или 100 тыс. операций вычисления их первых производных (табл. 4).

Сравнивая полученные результаты, можно заметить, что в большинстве случаев реализация функции в виде выражения оказывается неэффективной. Различия особенно существенны в случаях, когда арифметическая сложность выражения невелика и внутри него используются только простейшие функции: константа, линейная и квадратичная, как, например, в функции Розенброка и полиноме. Но обычно в подобных случаях очень легко реализуются программы аналитического дифференцирования, так как производные здесь выражаются простой математической формулой. Следовательно, такие функции лучше реализовать в качестве класса с переопределенным оператором ( ) и переопределенными методами дифференцирования.

Таблица 4

Результаты тестирования операции вычисления первой производной (в секундах)

Способ реализации	Функция			
	секанс	полином	Розенброка	тригонометрическая
Класс (аналит.)	7,25	0,33	0,86	12,33
Класс (числен.)	7,51	1,90	5,90	51,35
Выражение (дерево)	6,65	12,48	37,30	121,25
Выражение (матрица)	8,37	6,61	9,09	51,28
Процедура	7,69	0,33	0,30	11,70

Если же реализовать метод аналитического дифференцирования слишком трудно, например, когда функция выражается очень сложной математической формулой, остаются только два способа вычисления производной: с помощью численного метода и путем интерпретации функционального выражения. В последнем случае производная будет вычислена с максимально возможной точностью, хотя и с несколько большими временными затратами. Все эти факты необходимо учитывать при выборе способа реализации функции.

Таким образом, рассмотрены основные возможности применения объектной технологии к программированию задач численного анализа. Представление математических функций в виде единой целостной иерархии функциональных классов позволяет провести неформальную объектную систематизацию важных разделов вычислительной математики и, главное, унифицировать и существенно упростить разработку и использование программ численного анализа. Хотя в работе основное внимание уделяется вопросам вычисления функций и их производных, основные результаты могут быть использованы при реализации других численных методов, инкапсулируемых соответствующими функциональными классами.

Работа поддержана Российским Фондом фундаментальных исследований (грант 95-01-01239).

ЛИТЕРАТУРА

1. Proceedings of the Conference on Object-Oriented Programming: Systems, Languages, and Applications, ACM, 1994.
2. Бахвалов Н.С., Жидков Н.П., Кобельков Г.М. Численные методы. — М.: Наука, 1987.
3. Самарский А.А., Гулин А.В. Численные методы. — М.: Наука, 1989.
4. Янке Е., Эмде Ф., Леш Ф. Специальные функции. Формулы, графики, таблицы: Пер. с нем. — М.: Наука, 1977.
5. Лаврентьев М.А., Шабат Б.В. Методы теории функций комплексного переменного. — М.: Наука, 1987.
6. Грис Д. Конструирование компиляторов для цифровых вычислительных машин: Пер. с англ. — М.: Мир, 1975.
7. Дэннис Дж.-мл., Шнабель Р. Численные методы безусловной оптимизации и решения нелинейных уравнений: Пер. с англ. — М.: Мир, 1988.