Введение

***Длина дуги.*** Численное интегрирование задачи Коши для жестких систем обыкновенных дифференциальных уравнений (ОДУ) является одной из очень трудных задач вычислительной математики. Формально эта задача имеет следующий вид:

Здесь – скаляр, а и – векторные функции, а – размерность системы.

Задачу традиционно считают жесткой, если хотя бы в части отрезка . Решению жестких задач посвящена обширная литература, наиболее подробный обзор которой дан в монографии [*Хайрер, Ваннер*].

Различают два типа прикладных задач. Первый тип – это так называемые большие, или многопроцессные, задачи. Они описываются системами уравнений в частных производных, к которым подключена одна или несколько систем ОДУ. Примером могут служить задачи горения и взрыва. В них процесс горения, то есть реакции химических веществ, описывается системой уравнений химической кинетики; это система ОДУ. Выделяющееся тепло приводит к движению вещества. Это движение описывается уравнениями газодинамики, то есть системой уравнений в частных производных.

Расчет уравнений в частных производных гораздо более трудоемок, чем решение ОДУ. Поэтому в больших задачах шаг по времени определяется требованиями методов решения уравнений в частных производных. Этот же шаг вычислитель вынужден использовать для решения сопутствующей системы ОДУ. В этом случае естественным аргументом для решения задачи (1) является время .

Второй класс задач содержит только систему ОДУ. Для этого класса задач оказывается выгоднее выбрать другой аргумент – длину дуги интегральной кривой в многомерном пространстве. Длина дуги определяется соотношением

Если выбран новый аргумент , то старый аргумент становится его функцией:

таким образом, число неизвестных функций становится . Формально новую функцию можно включить в систему (1), добавив туда уравнение

Напомним, что расширенная система ОДУ содержит в правых частях аргумент Его в правых частях можно заменить на новую функцию: Тогда расширенная система формально не будет содержать аргумента в правых частях, то есть станет автономной.

Перейдем к новому аргументу Для этого в расширенной системе (1) заменим на с помощью соотношения (2). Получим следующую систему размерности :

При аргументе сумма квадратов правых частей (4) равна , то есть норма правой части никогда не бывает большой. Это облегчает численное интегрирование системы. Такой прием полезен даже для нежестких ОДУ, а для жестких он кардинально облегчает решение задачи. Переход к длине дуги и различные преимущества этого метода подробно описаны в монографии [*Кузнецов*]. Заметим, что правые части (4) не содержат аргумента , то есть эта система является автономной.

Переход к длине дуги практически не увеличивает трудоемкость одного шага численных расчетов. Практически всегда основное время расчетов уходит на вычисление правых частей , которые могут быть достаточно сложными функциями своих аргументов. Переход от (4) к (1) включает лишь несколько дополнительных арифметических операций, трудоемкость которых невелика. Размерность системы увеличивается на , что так же мало существенно для прикладных задач, где обычно довольно велико.

***Выбор шага.*** Расчет с постоянным шагом по времени  или по длине дуги  обычно невыгоден. Шаг целесообразно уменьшать там, где решение быстро меняется, то есть правые части ОДУ велики. На участках слабого изменения решения шаг можно увеличивать. В [1] подробно описаны алгоритмы автоматического выбора шага, принятые в мировой литературе. Традиционно используют два основных метода выбора шага. В первом методе каждый шаг выполняют по некоторой схеме -го порядка точности, в которую вложена схема -го порядка точности. Результат вложенной схемы берут в качестве ответа. По разности результатов двух схем выбирают величину следующего шага. Во втором методе шаг рассчитывают повторно, разбив его на два шага величиной . По разности этих расчетов вычисляют новый шаг.

На основе этих методов написано много пакетов программ. Большинство из них хорошо работают на нежестких задачах. Однако проверка этих пакетов на тестах с известными точными решениями показывает, что реальная точность расчетов лишь по порядку величины близка к запросу пользователя. Она обычно оказывается в несколько раз хуже или лучше. Поэтому в расчетах прикладных задач, где ответ неизвестен, пользователь не может быть вполне уверен в достижении требуемой точности.

Намного хуже ситуация для случая жестких задач. Для них расчеты на тестах показали [], что реальная точность может быть в раз хуже заявленной. Это относится даже к таким тщательно выверенным программам, как пакеты Гира или программы Дормана-Принса dopri5. Кроме того, на жестких задачах возможны “срывы” шага: иногда на участках слабо меняющегося решения программа без видимых причин уменьшает шаг в раз. Затем шаг постепенно увеличивается, но снова срывается; это может повторяться много раз. В [*Пошивайло, НН, Белов*] был предложен принципиально другой алгоритм автоматического выбора шага. Он основан на использовании длины дуги и кривизны интегральной кривой в многомерном пространстве. Интуитивно понятно, что чем больше кривизна , тем меньше должен быть шаг . Но каким должен быть алгоритм, связывающий эти две величины?

***Оптимальный шаг.*** Напомним определение кривизны кривой в -мерном пространстве с координатами . Касательная к этой кривой определяется через производную . Деля на длину этого вектора, получим единичный вектор направления касательной:

Кривизна определяется как производная вектора направления касательной по длине дуги:

Таким образом, кривизна является вектором даже в случае плоской кривой, то есть одного ОДУ; это радиус-вектор окружности, имеющей касание второго порядка с кривой . Наряду с этим говорят о скалярной кривизне , которая равна величине радиуса этой окружности.

В ранних работах [*Пошивайло*] с помощью многих численных экспериментов была подобрана неплохая эвристическая закономерность: . Она позволила провести расчеты некоторых прикладных задач, например – образование окислов фосфора и серы при горении различных топлив.

Задача теоретического нахождения зависимости крайне сложна. Из общих соображений понятно, что результат должен зависеть от того, какая именно разностная схема используется для интегрирования системы ОДУ. Однако удалось найти такой случай [*диф. ур.*], для которого теоретически обосновывается формула выбора оптимального шага. Пусть схема интегрирования имеет точность ; это может быть явная или неявная схема Эйлера или явно-неявная L1-устойчивая схема Розенброка. Тогда

Качественный вид этой формулы совпадает с ранее найденным эвристическим видом, а оптимальный показатель степени лишь слабо отличается от эвристического.

Как часто оказывается, “чистая” теоретическая формула (7) для своего практического применения требует “кухонных” поправок.

Во-первых, ясно, что при она дает . Формально это правильно: если на некотором участке кривой , этот участок есть прямая, то есть является линейной функцией. А для линейной функции любая численная схема дает точный ответ при сколь угодно большом шаге.

На практике лишь в отдельных точках кривой (например, в точках перегиба), а попадание узла расчетной сетки именно в эту точку имеет нулевую вероятность. Однако возможно попадание счетного узла в малую окрестность точки перегиба и получение неприемлемо большого шага . Надо ввести такую поправку, чтобы в любом случае число интервалов было не меньше некоторого разумного .

Во-вторых, надо дать разумное определение константы в (6). В нее должна быть включена некоторая интегральная нормировка по длине дуги, обеспечивающая желательное количество интервалов сетки .

Исходя из этого, в [*дифф ур*] был предложен следующий алгоритм. Пусть требуется решить задачу (4) на отрезке с априорно заданными , . Несколько видоизменим формулу (7) с учетом сделанных замечаний:

В точках с очень малой кривизной она дает . В точках с очень большой кривизной формула (8) переходит в (7); при этом константа такова, что . При любых не может получиться или . Это свидетельствует о разумности формулы (8).

Послать Белову чтобы он внес правки и поставил лит-ру.

Формулы тестов с синусом и тангенсом.

Рисунки: с разными лямбда ERK1, ERK2, ERK4. + для лямбда 10^4 ERK1+ERK2+ERK4 из одной точки.

Графики решения для трех лямбд в одном масштабе. Тоненькой линией вся кривая, маркерами или жирной – расчетный участок, где кривизна от 1 до 1.