АКАДЕМИЯ НАУК СССР
СИБИРСКОЕ ОТДЕЛЕНИЕ
АВТОМЕТРИЯ

№ 5 1974

МЕТОДЫ ОБРАБОТКИ ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ

УДК 621.317.7.001.5

Ю. Л. РОЗОВ, М. Б. УЛИЦКИЙ
(Ленинград)

О ВЫБОРЕ ОПТИМАЛЬНОГО АЛГОРИТМА РАБОТЫ ПРЕОБРАЗОВАТЕЛЯ ИНФОРМАЦИИ, ПРЕДНАЗНАЧЕННОГО ДЛЯ ОПРЕДЕЛЕНИЯ ЗНАЧЕНИЙ МНОГОПАРАМЕТРИЧЕСКИХ ПРОЦЕССОВ ПО РЕЗУЛЬТАТАМ КОСВЕННЫХ ИЗМЕРЕНИЙ

1. При построении автоматизированных комплексов обработки экспериментальных данных в целом ряде случаев возникает задача создания вторичного преобразователя информации, предназначенного для вычисления значений исследуемого многопараметрического процесса на основании обработки результатов косвенных измерений. Такой преобразователь информации представляет собой, по существу, специализированное вычислительное устройство.

К подобным специализированным устройствам обычно предъявляются следующие требования: а) объем памяти устройства должен быть сравнительно невелик; б) устройство должно обладать высоким быстродействием; в) определение значений исследуемого процесса должно производиться с заданной допустимой погрешностью.

В соответствии с этими требованиями возможны различные постановки задачи выбора оптимального алгоритма работы устройства. Остановимся более подробно на рассмотрении возможных постановок задачи и соответствующих методах решения.

2. Пусть $f(x)$, вообще говоря,— неизвестная функция n переменных, $x=(x_1, x_2, \ldots, x_n)$, $x \in R_n$. Для определения вида функции $f(x)$ предварительно проводятся измерения ее значений на регулярной сетке аргумента и по данным измерений строится аппроксимирующая поверхность $\hat{f}(x)$. Задачей вторичного преобразователя комплекса обработки данных является вычисление значений функции $\hat{f}(x)$ по данным измерения аргумента x в любой точке.

Необходимо отметить, что в том случае, когда измерения значений исследуемой функции $f(x)$ произведены с существенными случайными погрешностями, целесообразно производить предварительное сглаживание результатов измерений. Для вычисления значений сглаженной функции $\tilde{f}(x)$ в точках исходной сетки, можно использовать рассмотренные в [1] алгоритмы сглаживания функций многих переменных. В этом случае допустимая погрешность аппроксимации при вычислении значений функции $\hat{f}(x)$ должна быть согласована с погрешностью, вносимой применением сглаживания с учетом подавления случайной составляющей. Формулы для подсчета систематической и случайной составляющих погрешности сглаженных значений приведены в [2].

Будем строить алгоритм работы вычислительного устройства на основании кусочно-полиномиальной аппроксимации результатов предварительных измерений (или их сглаженных значений) методом наименьших квадратов.

Учитывая перечисленные выше требования, предъявляемые к устройству, можно поставить задачу выбора его оптимальной работы как следующую задачу математического программирования:

$$\min_{m,l} V(m, l); \tag{1}$$

$$\min_{m,l} T(m, l); \tag{2}$$

$$\varepsilon_i(m_i, l) \leqslant \varepsilon_0; \quad (i=1, 2, \ldots, l). \tag{3}$$

Здесь l — число участков аппроксимации; $m=(m_1, m_2, \ldots, m_l)$ — вектор степеней аппроксимирующих полиномов; $V(m, l)$ — объем памяти вычислительного устройства; $T(m, l)$ — максимальное время вычисления значения функции (быстродействие устройства); $\varepsilon_i(m_i, l)$ — погрешность аппроксимации на i-м участке.

Кроме того, имеются ограничения, налагаемые применением метода наименьших квадратов, а именно, общее число точек на участке должно быть больше или равно степени аппроксимирующего полинома.

Очевидно, что степени аппроксимирующих полиномов m_i и число участков аппроксимации l являются величинами целыми, следовательно, задача (1)—(3) представляет собой задачу целочисленного программирования с двумя целевыми функциями.

Постановка задачи выбора оптимального алгоритма работы вычислительного устройства может отличаться от постановки (1)—(3). Так, например, могут быть поставлены условия минимизации объема памяти при ограничениях на быстродействие и точность аппроксимации или, напротив, условия минимизации быстродействия при ограничении на объем памяти. Обе эти постановки приводят к необходимости решения задачи целочисленного программирования с одной целевой функцией, причем предлагаемая ниже процедура оптимизации для решения задачи (1)—(3) полностью применима и в этих случаях.

Наконец, возможна постановка задачи выбора допустимого алгоритма работы устройства при ограничениях на объем памяти, быстродействие и точность аппроксимации. Такая задача не является задачей математического программирования, однако для ее решения будут полезны приведенные ниже зависимости перечисленных характеристик устройства от числа участков аппроксимации и степеней аппроксимирующих полиномов.

3. Рассмотрим подробнее метод решения поставленных задач оптимизации.

Проблеме выбора решения в случае нескольких целевых функций, или проблеме векторной оптимизации, посвящен ряд работ [3—6]. Сложность этой проблемы обусловливается, в первую очередь, противоречивостью критериев и необходимостью использования некоторой схемы компромисса, позволяющего гармонично учитывать все критерии. Следуя терминологии, принятой в теории исследования операций [3], сформулируем несколько возможных подходов к решению задачи (1)—(3).

В том случае, когда критерии (1) и (2) имеют одинаковую важность, целесообразно для построения обобщенного критерия воспользоваться «принципом справедливого компромисса» [3]. При этом справедливым считается такой компромисс, при котором относительный уровень снижения качества по одному из критериев не превосходит относительного уровня повышения качества по другому критерию. Ис-

пользование этого принципа приводит к необходимости решения следующей задачи:

$$\min_{m,l}\{\Phi_1(m, l) = \bar{V}(m, l)\,\bar{T}(m, l)\}, \tag{4}$$

$$\varepsilon_i(m_i, l) \leqslant \varepsilon_0 \quad (i=1, 2, \ldots, l). \tag{5}$$

Здесь $\bar{V}(m, l)$ и $\bar{T}(m, l)$ — нормализованные целевые функции. Необходимость нормализации вызвана тем, что целевые функции имеют различные масштабы измерения. Существует целый ряд способов нормализации [3], одним из которых является, например, следующий:

$$\bar{V}(m, l) = \frac{V(m, l)}{\min\limits_{m,l} V(m, l)}, \tag{6}$$

$$\bar{T}(m, l) = \frac{T(m, l)}{\min\limits_{m,l} T(m, l)}, \tag{7}$$

где минимумы функций $V(m, l)$ и $T(m, l)$ определяются с учетом ограничений (3).

Использование «принципа интегральной оптимальности» [3] приводит к необходимости минимизации функционала

$$\Phi_2(m, l) = \bar{V}(m, l) + \bar{T}(m, l) \tag{8}$$

при ограничениях (3).

В тех случаях, когда критерии (1) и (2) имеют различную степень важности, целесообразно ввести коэффициенты важности критериев λ_1 и λ_2. Это дает возможность при выборе решения отдавать предпочтение более важному критерию. Критерий (4) при этом принимает вид

$$\min_{m,l}\{\Phi_3(m, l) = \bar{V}^{\lambda_1}(m, l)\,T^{\lambda_2}(m, l)\}, \tag{9}$$

а критерий (8) можно записать в следующем виде:

$$\min_{m,l}\{\Phi_4(m, l) = \lambda_1\bar{V}(m, l) + \lambda_2\bar{T}(m, l)\}. \tag{10}$$

Рассмотрим теперь, чем определяются объем памяти вычислительного устройства и его быстродействие.

Объем памяти устройства определяется числом коэффициентов аппроксимирующих полиномов и числом координат граничных точек участков аппроксимации.

Общее число коэффициентов полинома степени m_i от n переменных равно числу линейно-независимых функций n переменных со степенями от 0 до m_i и равно

$$N_1(m_i) = \sum_{k=0}^{m_i} C_{n+k-1}^{n-1} = C_{n+m_i}^{n} = \frac{(n+m_i)!}{n!m_i!}. \tag{11}$$

Число координат граничных точек участков аппроксимации $N_2(l)$ зависит от способа разбиения на участки. Предлагаемый способ состоит в следующем. Значения функции $f(x)$ заданы на регулярной сетке аргумента, образующей в пространстве R_n гиперпараллелепипед D. Пусть степень аппроксимации фиксирована. Тогда, в случае если она пре-

вышает допустимую, проводится гиперплоскость, параллельная одной из граней гиперпараллелепипеда, разделяющая D на две равные части — D_1 и D_2. Затем на каждой из частей строится новая аппроксимирующая поверхность $\hat{f}(x)$. Поскольку можно провести n таких разделяющих гиперплоскостей, необходимо произвести последовательно n разбиений и выбрать ту пару участков, у которой максимальная по участкам D_1 и D_2 погрешность аппроксимации будет наименьшей. Затем, если это необходимо, такому разбиению подвергаются участки D_1 и D_2 и так далее. В этом случае величину $N_2(l)$ можно легко оценить сверху как

$$N_2(l) \leqslant n(l+1) \tag{12}$$

и объем памяти вычислительного устройства определить следующим образом:

$$V(m, l) = \sum_{i=1}^{l} C_{n+m_i}^{n} + n(l+1). \tag{13}$$

В качестве величины, определяющей быстродействие устройства, примем суммарное число операций умножения при вычислении значений функции $\hat{f}(x)$ в точках того участка, на котором степень аппроксимации максимальна *. Пусть

$$\max_{i \in [1,l]} m_i = \tilde{m}. \tag{14}$$

Число операций умножения при вычислении полинома степени $\tilde{m}$ от n переменных

$$N_3(\tilde{m}) = \sum_{k=0}^{\tilde{m}} kC_{n+k-1}^{n-1}. \tag{15}$$

Отсюда быстродействие устройства определится следующим образом:

$$T(m, l) = \tau \sum_{k=0}^{\tilde{m}} kC_{n+k-1}^{n-1}, \tag{16}$$

где τ — время выполнения одной операции умножения.

Точность аппроксимации на каждом участке будем характеризовать среднеквадратичной погрешностью, т. е.

$$\varepsilon_i(m_i, l) = \left[\frac{\sum_{j=1}^{N_i} [f(x^j) - \hat{f}(x^j)]^2}{N_i} \right]^{1/2}, \tag{17}$$

где N_i — общее число точек i-го участка ($i = 1, 2, \ldots, l$).

В качестве аппроксимирующих полиномов будем использовать ортонормированные решетчатые полиномы многих переменных [1]. Это позволит избежать необходимости решения системы линейных алгебраических уравнений при определении значений функции $\hat{f}(x)$, что при большом числе переменных затруднительно. (Это обстоятельство весь-

* Предполагается, что проектируемое вычислительное устройство обладает только оперативной памятью, поэтому операции обращения к внешним запоминающим устройствам исключены и операции умножения являются наиболее длительными.

ма существенно при реализации процедуры оптимизации на ЭВМ.) Значение аппроксимирующей функции в точке x^j i-го участка определится, таким образом, по следующим формулам:

$$\hat{f}(x^j) = \sum_{k=0}^{C_{n+m_i}^n} \alpha_k \varphi_k (x^j), \tag{18}$$

$$\alpha_k = \sum_{s=1}^{N_i} \rho(x^s)\, \varphi_k (x^s)\, f(x^s), \tag{19}$$

где $\varphi(x) = \left(\varphi_1(x), \varphi_2(x), \ldots, \varphi_{C_{n+m_i}^n}(x)\right)$ — вектор полиномов степеней от 0 до m_i от n переменных, ортонормированных с весом $\rho(x)$.

4. Как видно из выражений (13) и (15), объем памяти вычислительного устройства и его быстродействие являются нелинейными функциями от m и l. Следовательно, сформулированные выше задачи оптимизации алгоритма работы устройства представляют собой задачи нелинейного целочисленного программирования, для решения которых предлагается использовать эвристическую процедуру минимизации, основанную на методе штрафных функций. Следуя основной идее метода штрафов [7], введем в рассмотрение функцию

$$\Phi_i(m, l, \mu) = \Phi_i(m, l) + \mu \sum_{j=1}^{l} \delta_j [\varepsilon_j - \varepsilon_0] (\varepsilon_j - \varepsilon_0)^2, \tag{20}$$

где $\delta_j[z] = \begin{cases} 1, & z > 0; \\ 0, & z \leqslant 0; \end{cases}$ $\mu > 0$ — коэффициент штрафа.

Здесь $\Phi_i(m, l)$ $(i=1, 2, 3, 4)$ — одна из приведенных выше обобщенных целевых функций, выбор которой производится исследователем предварительно. Далее индекс i будем для простоты опускать. При $\mu \to \infty$ в силу свойств метода штрафов минимум функции $\Phi(m, l, \mu)$ стремится к решению поставленной задачи.

Поиск минимума функционала (20) может осуществляться только на множестве дискретных точек в пространстве (m, l), поэтому предлагается использовать для минимизации алгоритмы поиска экстремума с постоянным шагом (релейные алгоритмы). Предлагаемый алгоритм поиска имеет вид:

$$m_i^{k+1} = m_i^k - a_k \operatorname{sign} \widetilde{\Phi}'_{m_i}(m, l, \mu); \tag{21}$$

$$l^{k+1} = l^k - b_k \operatorname{sign} \widetilde{\Phi}'_l (m, l, \mu),$$

где $\widetilde{\Phi}'_{m_i}(m, l, \mu)$ и $\widetilde{\Phi}'_l(m, l, \mu)$ — оценки составляющих градиента функции $\Phi(m, l, \mu)$ по m_i и l соответственно. Эти оценки могут быть получены, например, с помощью конечных разностей функции $\Phi(m, l, \mu)$, т. е.

$$\widetilde{\Phi}'_{m_i}(m, l, \mu) = \frac{\Phi(m_1, \ldots, m_i + \Delta m_i, \ldots, m_l, l, \mu) - \Phi(m, l, \mu)}{\Delta m_i};$$

$$\widetilde{\Phi}'_l(m, l, \mu) = \frac{\Phi(m_1, \ldots, m_l, l + \Delta l, \mu) - \Phi(m, l, \mu)}{\Delta l}. \tag{22}$$

Напомним, что на каждом шаге при определении оценки градиента $\widetilde{\Phi}'(m, l, \mu)$ необходимо находить максимальную по всем участкам

степень аппроксимации в соответствии с формулой (14). Эту процедуру можно производить, например, методом перебора.

Характерным свойством релейных алгоритмов поиска является то, что процесс поиска экстремума выходит на некоторый предельный цикл в окрестности экстремума функционала (20), причем вид и положение предельного цикла зависят от параметров поиска a_k, b_k, Δm_i, Δl.

При постановке задачи оптимизации с одной целевой функцией полностью применима описанная процедура поиска экстремума с тем лишь отличием, что минимизирующий функционал (20) будет записываться несколько иначе. Так, в случае минимизации объема памяти устройства будем иметь

$$\Phi(m, l, \mu_1, \mu_2) = V(m, l) + \mu_1 \delta [T(m, l) - T_0] (T(m, l) - T_0)^2 + \\ + \mu_2 \sum_{j=1}^{l} \delta_j [\varepsilon_j - \varepsilon_0] (\varepsilon_j - \varepsilon_0)^2, \qquad (23)$$

а при минимизации быстродействия

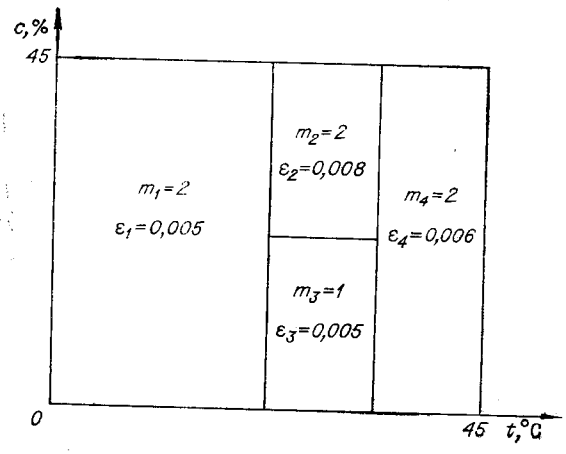
$$\Phi(m, l, \mu_1, \mu_2) = T(m, l) + \mu_1 \delta [V(m, l) - V_0] (V(m, l) - V_0)^2 + \\ + \mu_2 \sum_{j=1}^{l} \delta_j [\varepsilon_j - \varepsilon_0] (\varepsilon_j - \varepsilon_0)^2, \qquad (24)$$

где V_0 и T_0 — ограничения по объему памяти и быстродействию устройства.

Таким образом, в результате решения задачи выбора оптимального алгоритма работы вычислительного устройства гиперпараллелепипед будет разбит на оптимальное число участков $l_{опт}$, на каждом из которых будет определена оптимальная степень аппроксимации $m_{i\,опт}$, причем погрешность аппроксимации на каждом участке не будет превышать допустимой погрешности.

Пример. Работоспособность изложенного в статье метода была проверена при решении задачи выбора оптимального алгоритма работы преобразователя информации, предназначенного для обработки результатов электрохимических измерений. В качестве предварительных измерений была принята приведенная в [8] таблица результатов измерения удельной электропроводности серной кислоты $\varkappa$ в зависимости от концентрации c и температуры t, содержащая 540 значений $\varkappa$.

Критерий оптимальности алгоритма имел вид (10) при $\lambda_1 = \lambda_2 = \frac{1}{2}$.



Допустимая погрешность аппроксимации $\varepsilon_0=0{,}01$. Результат решения задачи приведен на рисунке. Как видно из рисунка, было получено 4 участка аппроксимации с соответствующими степенями аппроксимирующего полинома.

Необходимый объем памяти для хранения коэффициентов аппроксимации и координат граничных точек участков аппроксимации равен 600 бит. С учетом необходи-

мости запоминания программы вычислений можно рекомендовать использовать в данном случае ОЗУ емкостью 256×4 бит. Быстродействие устройства при этом будет составлять 30 мс, что хорошо согласуется со временем вывода результатов вычисления на регистрирующее устройство.

Задача решалась на ЭВМ М-220, имеется программа на языке АЛГОЛ-60.

ЛИТЕРАТУРА

1. В. Я. Катковник, М. Б. Улицкий. Ортонормированные решетчатые полиномы в задаче аппроксимации функций многих переменных.— В сб. «Механика и процессы управления. Вычислительная математика». Труды ЛПИ, Л., 1971, № 318.
2. В. Я. Катковник, М. Б. Улицкий. Об одном классе алгоритмов осреднения и численного дифференцирования функций многих переменных в задачах обработки данных эксперимента.— Труды ВНИИ. Л., Научприбор, 1972, № 1.
3. В. И. Борисов. Проблемы векторной оптимизации.— В сб. «Исследование операций». М., «Наука», 1972.
4. L. A. Zadeh. Optimality and Non-Scalarvalued Perfomence Criteria.— IEEE Trans., 1968, v. AC-8, N 1.
5. В. Л. Волкович. Методы принятия решений по множеству критериев оптимальности.— В сб. «Сложные системы управления», вып. 1. Киев, «Наукова думка», 1968.
6. Ю. А. Зак. Модели и методы построения компромиссных планов в задачах математического программирования с несколькими целевыми функциями.— Кибернетика, 1972, № 4.
7. А. Фиакко, Г. Мак-Кормик. Нелинейное программирование. Методы последовательной безусловной минимизации. М., «Мир», 1972.
8. H. E. Darling. Conductivity of Sulfuric Acid Solutions.—J. of Chem. and Engin. Data, 1964, v. 9, N 3.

Поступила в редакцию 21 февраля 1973 г.

УДК 517.948.32

В. П. ДИДЕНКО, Н. Н. КОЗЛОВ

(Киев)

РЕГУЛЯРИЗАЦИЯ ОДНОГО КЛАССА ЗАДАЧ ОБРАБОТКИ ИЗМЕРЕНИЙ

В работе рассматривается регуляризация некорректных задач в случае, когда областью определения оператора является негативное пространство. В частности, такие задачи возникают при синтезе оптимальных фильтров для оперативной обработки результатов измерений, когда характеристики входных сигналов определяются в процессе обработки [1]. Аналогичная ситуация имеет место и при идентификации объектов.

В этих конкретных условиях возникает необходимость в решении интегрального уравнения первого рода:

$$Au \equiv \int_0^T R(t, \tau)\, u(\tau)\, d\tau = f, \quad 0 \leqslant t \leqslant T.$$

Класс допустимых решений $u(t)$ может содержать обобщенные функции типа δ-функции Дирака, а также их производные, которые являются, как известно, элементами негативных пространств.