Московский государственный технический университет
им. Н. Э. Баумана

В. Г. Алексеев, П. Н. Горюнов, Ю. И. Нестеров

МЕТОДЫ ОПТИМИЗАЦИИ КОНСТРУКТОРСКО-ТЕХНОЛОГИЧЕСКИХ РЕШЕНИЙ В УСЛОВИЯХ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ ЭЛЕКТРОННОЙ АППАРАТУРЫ

Издательство МГТУ им. Н. Э. Баумана
1994

Московский государственный технический университет им. Н.Э. Баумана

В.Г. АЛЕКСЕЕВ, П. Н. ГОРЮНОВ, Ю.И. НЕСТЕРОВ

МЕТОДЫ ОПТИМИЗАЦИИ КОНСТРУКТОРСКО-ТЕХНОЛОГИЧЕСКИХ РЕШЕНИЙ В УСЛОВИЯХ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ ЭЛЕКТРОННОЙ АППАРАТУРЫ

Методические указания

ИЗДАТЕЛЬСТВО МГТУ ИМ. Н.Э БАУМАНА
1994

ББК 30.2
А47

Рецензент Г.Р.Сагателян

А47 Алексеев В.Г., Горюнов П.Н., Нестеров Ю.И. Методы оптимизации конструкторско-технологических решений в условиях автоматизированного проектирования электронной аппаратуры: Методические указания. — М.: Изд-во МГТУ, 1993. - 32 с., ил.

Представлено построение альтернативных вариантов при проектировании конструкций ЭВС (РЭС) и технологических процессов их изготовления.

Для студентов, изучающих курс «Автоматизация проектирования электронной аппаратуры».

Ил. 12. Библиогр 3 назв.

ББК 30.2

Редакция заказной литературы
Виктор Григорьевич Алексеев
Павел Николаевич Горюнов
Юрий Иванович Нестеров

Методы оптимизации конструкторско-технологических решений в условиях автоматизированного проектирования электронной аппаратуры

Заведующая редакцией Н.Г.Ковалевская
Редактор Г.А.Нилова
Корректор Л. И. Малютина

Подписано в печать 12.08.94. Формат 60x84/16. Бумага тип. № 2. Печ.л.2,0, Усл.печ.л.1,86, Уч.-изд.л.1,72. Тираж 500 экз. Изд. № 27.
Заказ №467 С476

Заказ №467 С476
Издательство МГТУ, типография МГТУ.
107005, Москва Б-5,2-я Бауманская, 5.

ВВЕДЕНИЕ

Раздел «Методы оптимизации конструкторско-технологических решений в условиях автоматизированного проектирования электронной аппаратуры (ЭА)» входит как составная часть в дисциплину «Автоматизация конструкторско-технологического проектирования ЭВС (ЮС)». Цель раздела — выработка теоретических и практических навыков построения альтернативных вариантов при проектировании конструкций ЭВС (РЭС) и технологических процессов (ТП) их изготовления.

Задачами раздела являют, я приобретение знаний по методам оптимизации и их использования для анализа и синтеза, исследования и оптимизации конструкций ЭА и проектируемых ТП. На практических занятиях поданному разделу рассматриваются частные задачи анализа и синтеза конструкций ЭА и ТП производства ЭА.

АЛГОРИТМЫ ОПТИМАЛЬНОГО ПРОЕКТИРОВАНИЯ

Основной целью математической теории оптимального проектирования [1] является использование неизвестных объектов типа «черный ящик» или объектов, известных не полностью. «Черным ящиком» называют объект, для которого не имеется никакого математического описания. Это означает, что зависимость показателей качества F от управляющих параметров $x_j \ (j = \overline{1, n})$ для таких объектов неизвестна.

В процессе проектировании определяют, при каких значениях независимых переменных (аргументов) зависимая переменная (целевая функция) приобретает оптимальное значение. Поскольку в этом случае решается неклассическая задача оптимизации, когда неизвестна функциональная зависимость между критерием оптимизации и управляющими переменными, то оптимизируется не только целевая функция, но и сам способ оптимизации.

Указанную задачу решают, как правило, при проектировании новых объектов, для которых, естественно, нельзя дать достаточно совершенное теоретическое описание. Поэтому в процессе разработка и конструирования объекта используют множество аппроксимаций, среди которых нужно выявить наилучшие, позволяющие эффективно найти оптимальное решение.

Наличие случайных факторов, определяющих экстремум, придает задаче оптимизации в процессе проектирования стохастичность. Стохастические задачи оптимизации сложнее детерменированных, однако влияние случайных факторов сказывается главным образом на скорости поиска экстремума. Стохастичность задачи оптимизации еще более усиливается, если в качестве критерия оптимизации выступает надежность (стабильность, точность), показатели которой являются вероятностными.

Современная математическая теория оптимального проектирования, используемая для решения задач оптимизации независимо от их детерминированного или стохастического характера, классифицирует методы оптимального проектирования на две основные группы: линейные и нелинейные. Последние в свою очередь подразделяют на стохастические и динамические методы. Классификация методов оптимального проектирования представлена на рис. 1 [2].

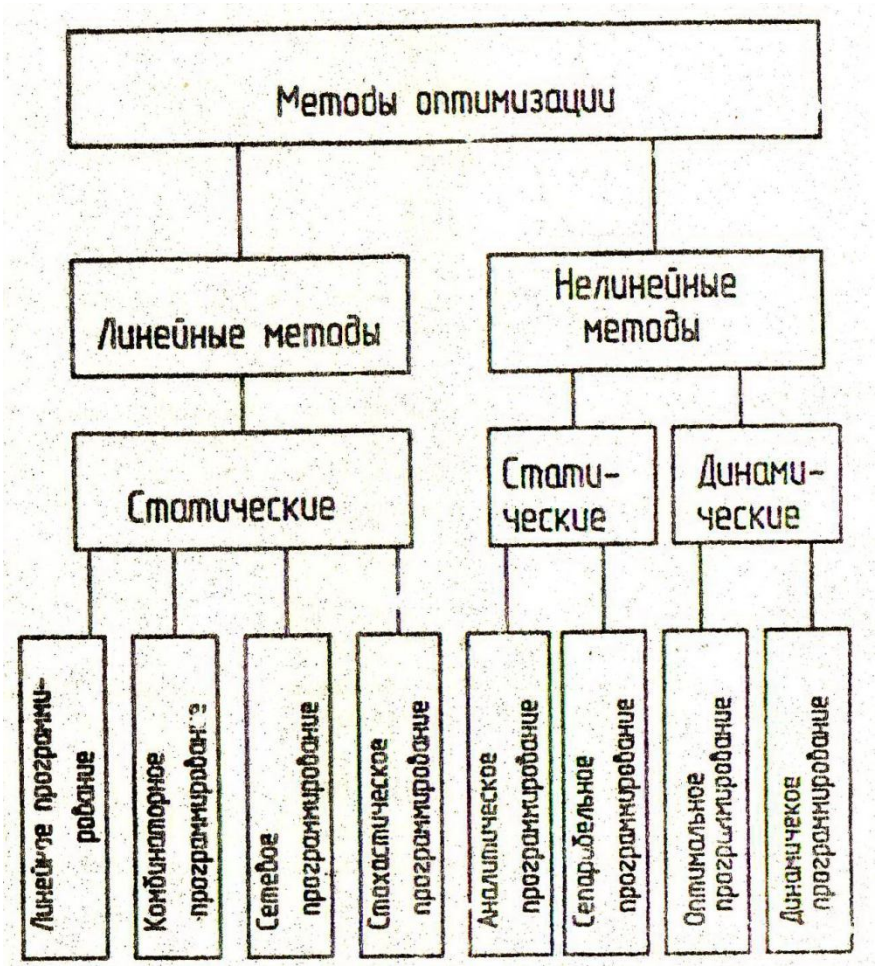


Рис. 1. Методы оптимизации, применяемые при проектировании

Для каждого метода оптимального проектирования разработан один или несколько алгоритмов, записываемых обычно в векторной форме:

$$\vec{r} = \vec{f}(\vec{X}, \vec{U}),$$

где $\vec{X}$ — n-мерный вектор с координатами $\{X_i\}$, $i = \overline{1, n}$;

$\vec{U}$ — r-мерный вектор с координатами $\{U_k\}$, $k = \overline{1, r}$;

$\vec{f}$ — m-мерный вектор с координатами $\{f_j\}$, $j = \overline{1, m}$.

Метод линейного программирования является одним из первых методов оптимального проектирования. Линейное программирование рассматривает линейные процессы, описываемые уравнением

$$\vec{Y} = \vec{A}\vec{X},$$

где $\vec{X} <=> \{x_i\}$ — n-мерный вектор независимой переменной, изменение которой влияет на выходной эффект (целевую функцию);

$\vec{Y} <=> \{y_i\}$ — m-мерный вектор критерия выходного эффекта (например, показатель качества изделия), представляющий собой матрицу воздействия i-го вида для получения j-го эффекта.

Задачи линейного программирования обычно записывают в виде

$$Q(X) = \sum_{i=1}^{n} a_i x_i => \min_{x \in S},$$

где S — множество допустимых значений переменных, определяемое системой

линейных неравенств:

$$S = \sum_{j=1}^{n} b_{ij} \geq c_i, (i = \overline{1, m} > n).$$

Задачу линейного программирования можно задать векторами

$$A = (\vec{a}_1, \vec{a}_n), c = (\vec{c}_1, \vec{c}_m)$$

и матрицей

$$B = \|b_{ij}\|, i = \overline{1, m}, j = \overline{1, n}.$$

Произвольный выбор неотрицательных чисел x≥0 называют планом. План, удовлетворяющий условию $\vec{Y} = \vec{A}\vec{X}$ при заданном выходном эффекте $\vec{Y}$ и заданных воздействиях $\vec{X}$, считают допустимым.

Допустимый план, экстремизирующийся каким-либо выбранным критерием $J(\vec{x})$ (стоимость, стабильность, точность и т.п.), является оптимальным. Для оптимального плана характеризующий критерий имеет вид

$$J(\vec{X}) = (\vec{C}, \vec{X}) <=> \sum_{i=1}^{n} c_i x_i => \max,$$

$$\vec{A}\vec{X} \leq \vec{Y}, \vec{X} \geq 0,$$

где $C <=> \{c_i\}, i = \overline{1, n}$ – параметр ограничения.

На практике для большинства реальных задач n>m , следовательно, имеется свобода, необходимая для оптимизации.

Существует несколько методов решения задачи линейного программирования. Простейший из них — симплексный (простой) метод, называемый иногда методом последовательного улучшения плана. В соответствии с этим методом описывают так называемый симплекс

$$(1, \vec{X}) = 1, \vec{X} \geq 0,$$

для которого решение задачи имеет вид

$$X_i = \text{sign}(c_i - \max c_i),$$

где sign-сигнум-функция, определяющая знак уравнения симплекса следующим образом:

$$\text{sign}(c) = \begin{cases} 1 \text{ при } c \geq 0; \\ 0 \text{ при } c < 0. \end{cases}$$

Если выходной эффект одинаков для всех элементов процесса, т.е.

$\{Y_j\} = \vec{Y} => \text{const}, j = \overline{1, m}$,
то уравнение $\vec{Y} = \vec{A}\vec{X}$ принимает вид $(\vec{a}, \vec{x}) = y$, а оптимальный план

$$x_i = \frac{y}{a_i} \text{sign}\left(\frac{c_i}{a_i} - \max \frac{c_i}{a_i}\right).$$

Алгоритм последовательного улучшения плана в общем случае достаточно сложный, поэтому симплекс-метод применяют лишь в тех случаях, когда не известен какой-либо другой метод, удобный для решения алгоритма на ЭВМ.

Метод комбинаторного программирования, являющийся разновидностью метода линейного программирования, разработан и применяется при решении задач существенно дискретного характера. Множество возможных решений таких экспериментальных задач состоит из комбинаций того или иного вида (перестановок, сочетаний)

последовательностей. Поэтому метод решения дискретных экстремальных задач и называют комбинированным.

Основная особенность метода комбинированного программирования состоит в том, что на выходной эффект накладывается ограничение $\vec{Y} \le Y_{\text{огр}}$ и, таким образом, все множество различных планов X разбивается на два подмножества:

допусти дых шансов, удовлетворяющих условию

$$\vec{A}\vec{X} \le Y_{\text{огр}};$$

недопустимых шансов, для которых

$$\vec{A}\vec{X} > Y_{\text{огр}}.$$

Оптимальным планом, как и при методе линейного программирования, является допустимый план, оптимизирующий выбранный критерий:

$$J(\vec{X}) = (\vec{C}, \vec{X}) => \max;$$

$$\vec{A}\vec{X} \le Y_{\text{огр}}, \vec{X} \in x.$$

Задачу комбинаторного программирования, в отличие от линейного программирования можно было бы решить простым перебором вариантов. К сожалению, если число рассматриваемых предложений составляет несколько десятков, нахождение оптимального числа простым перебором затруднительно.

Поэтому разработаны алгоритмы, существенно ускоряющие этот процесс путем выбора группы заведомо оптимальных планов, например, алгоритм плотной перестройки* алгоритм Гомори и др. [3].

Сетевым программированием называют раздел комбинированного программирования. Его применяют в том случае, когда дискретные величины (переменные) образуют двухиндексное множество

$$\vec{X} = \{x_{ij}\}, x_{ij} = \begin{cases} 1 \\ 0 \end{cases},$$

и оптимизируемой величиной является линейная функция вида

$$J(\vec{X}) = \sum_{i,j=1}^{n} c_{ij} x_{ij}.$$

Метод сетевого программирования геометрически можно интерпретировать в виде конечного точечного множества

$$\vec{A} = \{A_i\}, i = \overline{1, n},$$

представляющего собой множество дискретных двоичных переменных $\{x_{ij}\}$.

Элементы матрицы $\vec{C} = \| c_{ij} \|$ являются расстояниями между i-й и j-й точками, которые удовлетворяют следующим условиям:

$$c_{ij} = 0, c_{ij} > 0; c_{ij} = c_{ji}, c_{ij} + c_{jk} > c_{ik}.$$

Если некоторые из точек соединить между собой отрезками прямых, соответствующими в некотором масштабе расстояниям между точками, то получится сеть, состоящая из фрагментов — подмножеств точек, связанных между собой, и изолированных точек.

В случае решения сетевых задач оптимизируют критерий

$$J(\vec{X}) = \sum_{i,j=1}^{n} c_{ij} x_{ij},$$

при некоторых специальных ограничениях, аналитическая формулировка которых

зависит от вида конкретных задач. Ими же определяется алгоритм решения. Для задачи о кратчайшей связывающей требуется соединить все точки множества $\vec{A} = \{A_{ij}\}$ односвязной сетью S так, чтобы сумма длин элементов этой сети была минимальной. Ограничения при этом имеют вид

$$\sum_{i,j=1}^{n} x_{ij} \geq 1,$$

каждое означает, что любая точка связана, по крайней мере с одной другой, а для решения применяется алгоритм Прима.

Для задачи об оптимальном замкнутом маршруте, где требуется соединить все точки множества $\vec{A} = \{A_{ij}\}$ замкнутой непрерывной линией L, так, чтобы сумма длин элементов этой линии была минимальной, ограничивающие условия имеют вид:

$$\sum_{i,j=1}^{n} x_{ij} = 1,$$

каждое из которых означает, что любая точка односторонне связана с одной другой точкой, а для решения следует применять алгоритм Волгина.

Задачи, решаемые методом сетевого программирования, можно также решать комбинированным методом и методом линейного программирования как обобщающим методом. Все три описанных метода относятся к линейным методам оптимального проектирования, так как характеризующий критерий оптимального плана является линейной функцией.

К нелинейному программированию обычно относят задачи следующего типа: найти значение независимых переменных $x_1, \ldots, x_n$, при которых достигается наименьшее значение функции этих переменных $Q=Q(x_1, x_2, \ldots, x_n)$, называемой целевой функцией. На переменные $x_1, x_2, \ldots, x_n$ в общем случае могут быть наложены ограничения в виде неравенств

$$F_j(x_1, x_2, \ldots, x_n) \geq 0, \; j = \overline{1, s}.$$

При этом предполагают, что Q и F_j являются действительными нелинейными функциями действительных переменных и что известен алгоритм их вычисления. Получить решение задачи нелинейного программирования аналитическим путем в общем случае не дается, поэтому приходится применять методы поиска экстремума с помощью ЭВМ.

Существуют следующие методы нелинейного программирования (см. рис.1): статическое, не учитывающее развитие процесса программирования во времени (аналитическое, сепарабельное и стохастическое), и динамическое, учитывающее такое развитие (динамическое и оптимальное).

Метод аналитического программирования является обобщением метода линейного программирования, его применяют для оптимизации аналитических функций качества объекта, которые можно разложить в ряд, например ряд Тейлора в окрестности точки x_0 ограничиться линейными членами разложения:

$$Q(x) = Q(x_0) + \frac{\partial Q}{\partial x_1}\Big|_{x = x_0} (x_1 - x_1^{(0)}) + \ldots$$

$$+ \frac{\partial Q}{\partial X_n}\Big|_{x = X_0} (x_n - x_n^{(0)}),$$

где $x_n - x_n^{(0)} = \vec{U}$ — направление, в котором функция быстрее всего возрастает;
$Q(x_0)$ — значение функции в начальной точке x_0;

$\left\{\frac{\partial}{\partial x_i}\right\} = \vec{\nabla}$ — дифференциальный оператор Набла—Гамильтона.

Аналитически должны быть заданы целевая функции Q(x) и система функций y_{ji} , ограничивающих исследуемый процесс:

$$y_i = (\vec{X}) = 0;\ y_j = (\vec{X}) = 0;\ Y = \left\{y_i\right\}^m;\ Y = \left\{y_j\right\}^m;$$

где $i = \overline{1, n},\ j = \overline{1, m}.$

Метод аналитического программирования отличается от классического метода оптимизации наличием дополнительного условия неотрицательности разыскиваемого алана: $\vec{X} \geq 0.$.

Набор функций, соответствующих оператору Набла, называют градиентом целевой функции

$$\operatorname{grad}\varphi(\vec{x}) = \vec{\nabla}\vec{\varphi} = \left\{\frac{\partial\varphi}{\partial x_j}\right\}^n.$$

Точки $\vec{x}(v),\ v = \overline{1, s}$, в которых $\vec{\nabla}\vec{\varphi} = 0$, называют экстремальными функциями φ(x), а значения функции в этих точках $\varphi(v) = \varphi(\vec{x}(v)),\ v = \overline{1, s}$ — локальными экстремумами функций (локальные максимумы и минимумы, седловые точки, точки перегиба и т.д.)

Среди локальных экстремумов функции φ(x) необходимо выделять абсолютный или глобальный экстремум:

$$\varphi^* = \max_{v \in S}\left\{\varphi(v)\right\} \text{ или } \varphi^{**} = \min_{v \in S}\left\{\varphi(v)\right\}.$$

Многие функции, встречающиеся в задачах аналитического программирования, имеют большое число локальных экстремумов, которые, как правило, и обнаруживают поисковые методы, например» метод наискорейшего спуска.

Иногда задачу аналитического программирования можно свести к последовательности линейных задач, т.е. применить так называемое сепарабельное программирование,

Метод стохастического программирования является дальнейшим обобщением методов линейного и аналитического программирований для случая, когда набор ограничений зависит не только от набора контролируемых переменных $\vec{X}$, но и от набора случайных переменных $\vec{\varepsilon}$, имеющих заданную плотность распределения. Оптимизируемой величиной служит математическое ожидание заданной функции $f(\vec{X}, \vec{\varepsilon})$ и поэтому распределение деления.

$$J(\vec{X}) = \int f(\vec{X}, \vec{\varepsilon})\omega(\vec{\varepsilon})\, d\vec{\varepsilon}.$$

Модель метода дополняют следующими уравнениями для ограничений разыскиваемых планов:

$$Y(\vec{X}, \vec{\varepsilon}) = 0;\ \vec{X} \geq 0;$$

$$P(\vec{\varepsilon} < \vec{\varepsilon}^* < \vec{\varepsilon} + \Delta\vec{\varepsilon}) = \omega(\vec{\varepsilon})\, d\vec{\varepsilon}.$$

Задачи стохастического программирования обычно хорошо поддаются аналитическому исследованию, и только в некоторых случаях приходится прибегать к методам частичного перебора и поиска экстремума.

Все вышеописанные методы программирования относятся к статическим

методам, так как они не учитывают развитие процесса во времени. Этим недостатком не обладает метод динамического программирования Беллмана и более общий метод оптимального управления Понтрягина.

Согласно методу динамического программирования исследуемый объект характеризуется двумя наборами переменных:

$$\vec{X}_i\,(i=\overline{1,n}),\ \vec{U}_k\,(k=\overline{1,r}).$$

Набор переменных $\vec{X}_i$ характеризует состояние исследуемого объекта в i-й момент времени, а набор переменных $\vec{U}_i$ — планируемые действия в тот же момент. Процесс эволюции объекта является немарковским (с последействием). Эта зависимость может быть выражена следующими рекуррентными соотношениями:

$$\vec{X}_{i+1}=\vec{Y}_i\,(X_iU_i),\ i=\overline{1,n}.$$

Функции Y_i часто не поддаются аналитическому описанию, поэтому их задают в виде матриц. Планируемые действия не могут быть полностью произвольными, следовательно, их выбирают в некоторой допустимой области $U_i\in\Omega(\vec{X}_i),\ i=\overline{1,n}$.

Набор переменных $\vec{X}_i$, удовлетворяющих последним условиям, называют допустимым планом решения динамической задачи. Допустимый план, экстремизирующий заданную функцию конечного состояния объекта $J(\vec{X})=\varphi(X_{n+1})$, называют оптимальным.

Поскольку задачи динамического планирования являются задачами с большим числом переменных, был выдвинут принцип последовательной оптимизации (принцип Беллмана), позволяющий свести исходную n-шаговую задачу к последовательному решению n-одношаговых задач меньшей размерности и, кроме того, решать динамическую задачу последовательно в обратном времени, начиная с момента i=n.

Однако существуют задачи, например, связанные с максимизацией монотонной функции при определенных ограничениях, Которые можно решать методом прямой последовательной оптимизации, в частности методом последовательных приближений Пикара [3].

в фиксированном интервале времени $0 \le t \le t^*$. Оптимальный план воздействия на объект выбирают из некоторой допустимой области $\vec{U}(t)\in\Omega(\vec{X})$ и максимизируют линейный функционал от значений функций состояния объекта в конечный момент времени $\vec{F}(\vec{X})=\vec{F}=(\vec{C},\vec{X}_i^*)\Rightarrow\max$,

где $\vec{C}=\{c_i\}$ — параметр ограничения.

Общий принцип теории оптимального планирования (принцип Понтрягина) формулируется следующим образом: оптимальным является такое управление $\vec{U}^*$, которое максимизируется гамильтонианом $H=(\vec{Y},\vec{C})=[\vec{C}(\vec{U},\vec{\nabla}\vec{U})\vec{Y}]$, где $\vec{Y}(t)$ — функция состояния объекта; $\vec{\nabla}\vec{U}=\frac{\partial U}{\partial X_i}$ — оператор Набла-Гамильтона.

Из сказанного следует, что давление исходного реального объекта может быть представлено в виде $\vec{X}=(\vec{\nabla}\vec{U})H$.

Метод оптимального программирования, основанный на принципе Понтрягина, является наиболее общим методом оптимального проектирования, так как, в случае замены функции $\vec{U}(t)$ набором значений в дискретных точках

$\vec{U}_i = \vec{U}_h = \vec{U}_{ih}, i = \overline{1,n}$ задача максимизации $J(\vec{X})$ при ограничениях $\vec{U}(t) \in \Omega(\vec{X})$ становится задачей аналитического программирования. С учетом случайных воздействий на исследуемый объект задача оптимизации преобразуется в стохастическую.

МЕТОДЫ ПОИСКА ЭКСТРЕМУМА

Методы поиска экстремума можно разделить на локальные, когда отыскивается местный экстремум, и глобальные, когда отыскивается всеобщий экстремум. Классификация методов поиска экстремума приведена на рис. 2 [2].

Рассмотрим некоторые из методов.

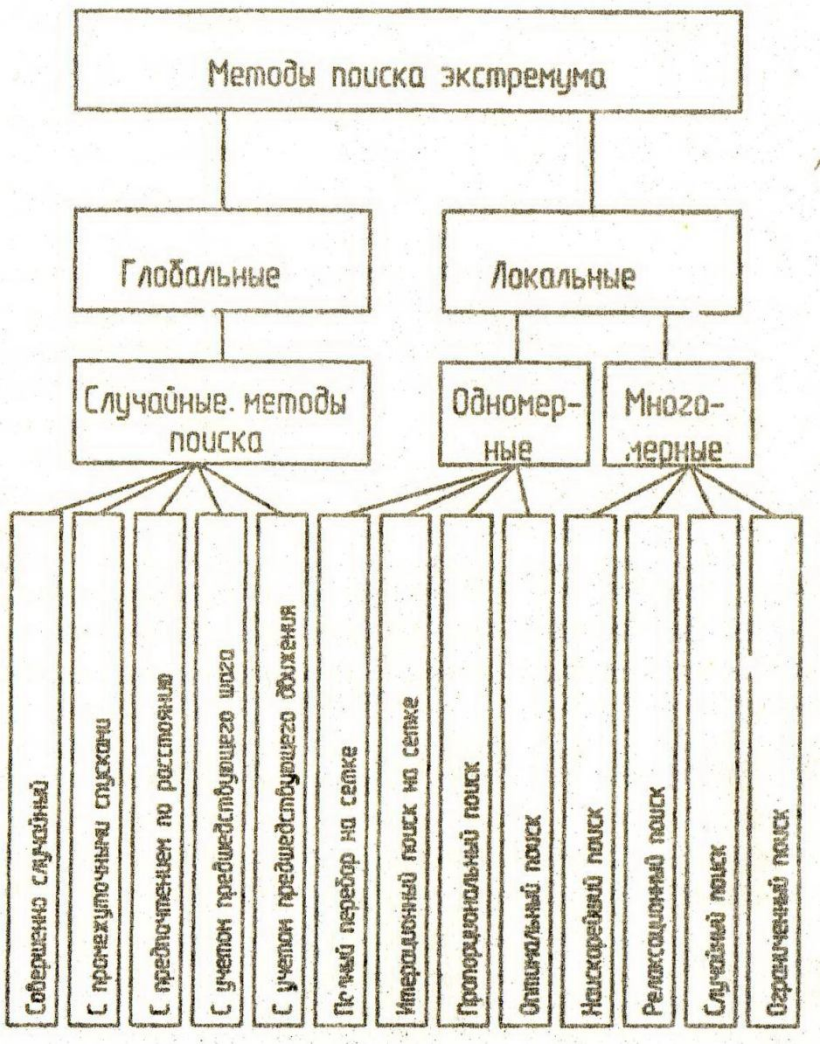
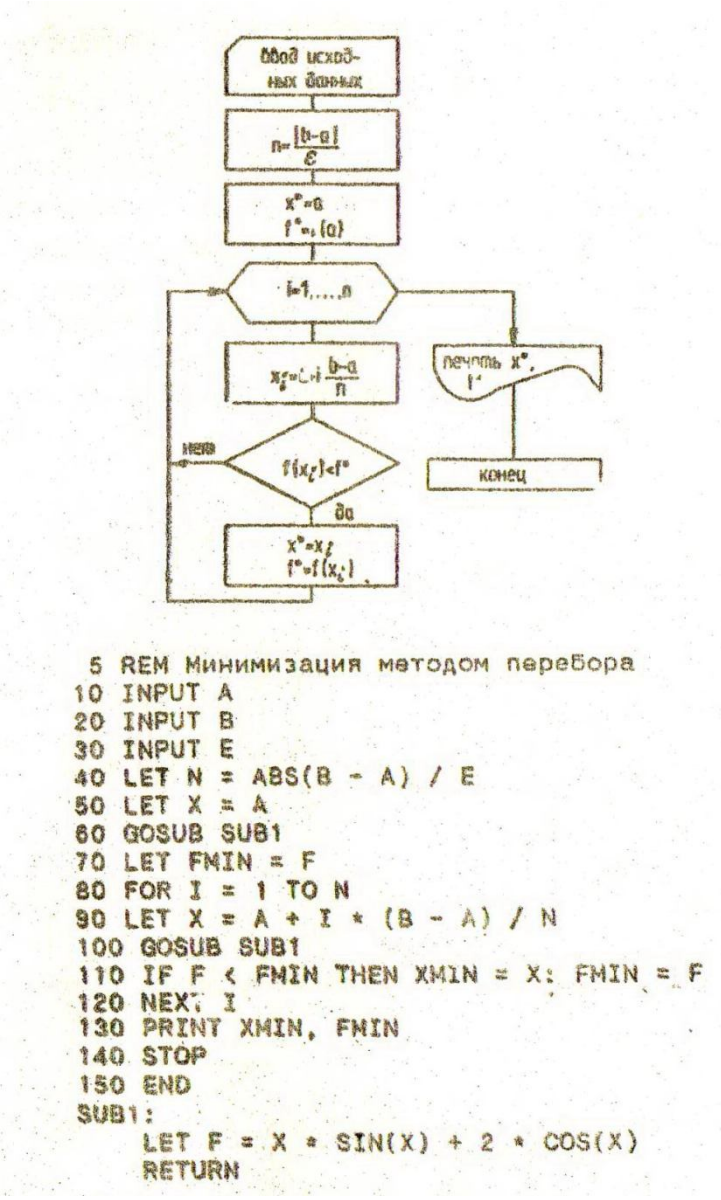


Рис. 2. Классификация методов поиска экстремума

Метода перебора

Метод перебора является простейшим из прямых методов минимизации. Пусть задана функция $f(x) \in \tilde{Q}[a;b]$ и требуется найти точку минимума x* этой функции на отрезке [a;b] с абсолютной погрешностью ε>0. Алгоритм минимизации методом перебора и текст программы представлены на рис.3.




```
 5 REM Минимизация методом перебора
10 INPUT A
20 INPUT B
30 INPUT E
40 LET N = ABS(B - A) / E
50 LET X = A
60 GOSUB SUB1
70 LET FMIN = F
80 FOR I = 1 TO N
90 LET X = A + I * (B - A) / N
100 GOSUB SUB1
110 IF F < FMIN THEN XMIN = X: FMIN = F
120 NEXT I
130 PRINT XMIN, FMIN
140 STOP
150 END
SUB1:
    LET F = X * SIN(X) + 2 * COS(X)
    RETURN
```

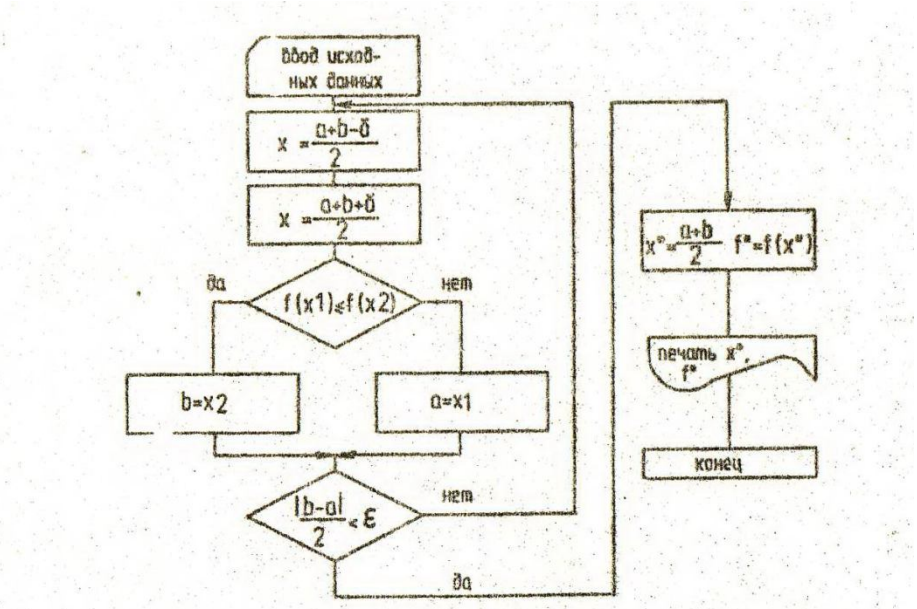
Рис. 3. Минимизация функции методом перебора

Методы минимизации, в которых точки x_i определяются в процессе поиска точки экстремума с помощью найденных ранее значений функции, называют последовательными.

Метод деления отрезка пополам

Метод деления отрезка пополам является простейшим последовательным методом. Он позволяет для любой функции $f(x) \in \tilde{Q}[a;b]$ построить последовательность вложенных отрезков $[a;b] \supset [a_1;b_1] \supset \ldots \supset [a_n;b_n]$, каждый из которых содержит точку минимума x*. Выбрав $\delta \in [0, 2\varepsilon]$, строят последовательности $\{a_n\}, \{b_n\}, \{x_1^{(n)}\}, \{x_2^{(n)}\}, r = 0, 1, \ldots$, где ε>0 — требуемая точность определения точки x*.

Текст программы и алгоритм приведены на рис. 4.




```
 5 REM Минимизация методом деления отрезка пополам
10 INPUT A
20 INPUT B
30 INPUT E
40 IF ABS(B - A) < 2 * E THEN 100
45 LET C = (A + B - E) / 2: LET D = (A + B + C) / 2
50 LET X = C: GOSUB SUB1: LET K = F
55 LET X = D: GOSUB SUB1: LET L = F
60 IF K > L THEN 90
80 LET B = D: GOTO 40
90 LET A = C: GOTO 40
100 LET X = (A + B) / 2: GOSUB SUB1
130 PRINT X, F: STOP
150 END
SUB1:
    LET F = X * SIN(X) + 2 * COS(X)
    RETURN
```

Рис. 4. Минимизация функции методом деления отрезка пополам

Метод золотого сечения

Метод золотого сечения является также последовательным методом минимизации, С учетом свойств золотого сечения, используют найденные значения приближений к точке минимума более рационально, что позволяет переходить к очередному отрезку, содержащему точку x* после вычисления одного, а не двух значений функции f(x). Деление отрезка на две неравные части осуществляемое так, что отношение длины всего отрезка к длине большей его части равно отношению длины большей части к длине меньшей, называют золотым сечением. Алгоритм приведен на рис. 5.

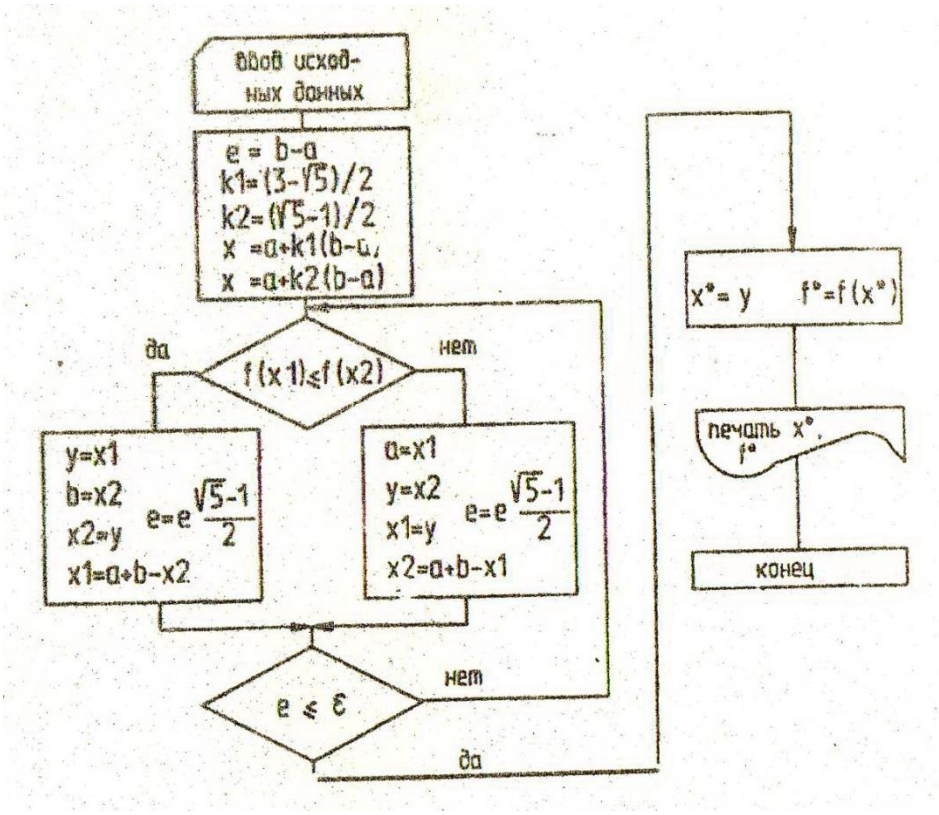


Рис. 5. Минимизация функции методом золотого сечения

В методах минимизации, рассмотренных выше, требуется, чтобы функция f(x) была унимодальной. Если функция этим свойством не обладает, то применение указанных методов приводит, вообще говоря, к неверному результату.

Метод ломаных

Метод ломаных является последовательным методом, рассчитанным па минимизацию функций, требование унимодальности, к которым не предъявляется. Алгоритм представлен на рис. 6.

Если вычисление или измерение производных функции f(x) не представляет больших затруднений, то можно применять непрямые методы, основанные на использовании производных f(x). Во многих случаях эти методы обеспечивают более быструю сходимость, чем прямые методы минимизации.

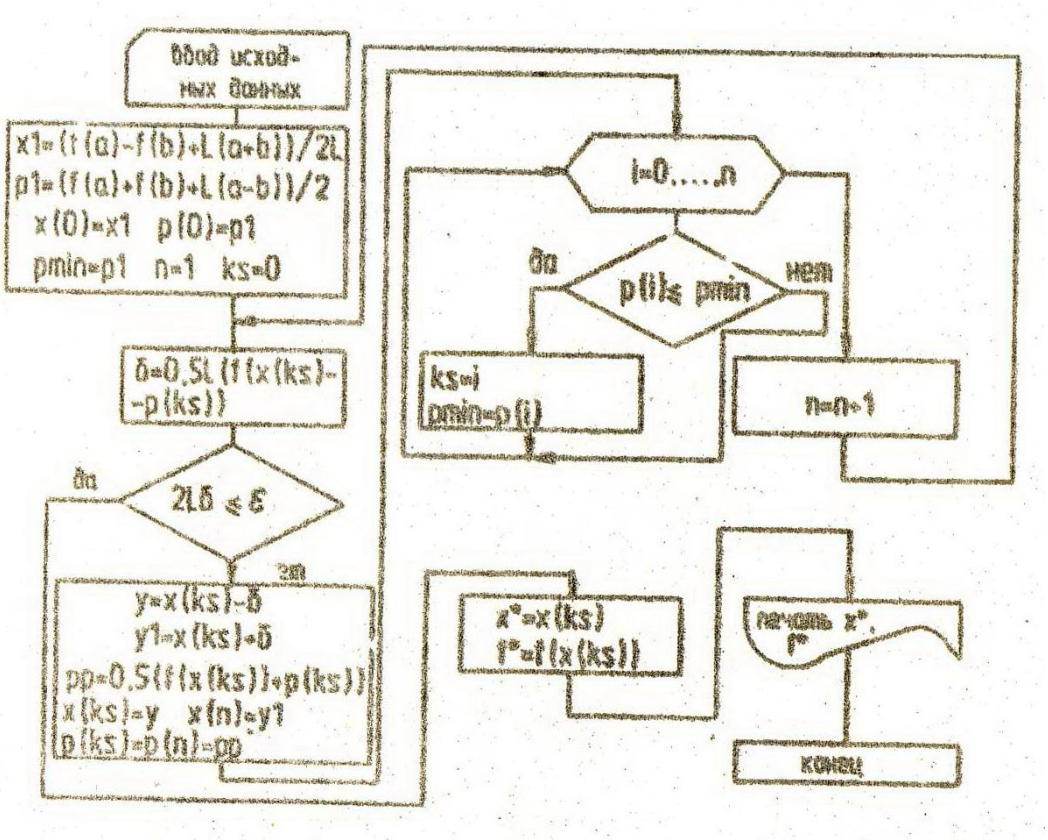


Рис. 6. Минимизация функции методом ломаных

Метод касательных

Метод касательных применяют для минимизации выпуклых дифференцируемых функций. Функцию f(x) называют выпуклой на отрезке [a,b], если $f(\alpha x' + (1-\alpha)x'') \le \alpha f(x') + (1-\alpha) f(x'')$ для произвольных x', $x'' \in [a;b]$ и $\alpha \in [0;1]$. Если f(x) — выпуклая дифференцируемая функция на отрезке [a,b], причем $f'(a) \cdot f'(b) < 0$, то можно построить последовательности $\{b_n\}$ и $\{c_n\}$ в соответствии с рекуррентными соотношениями

$$c_{n-1} = \frac{b_{n-1} \cdot f'(b_{n-1}) - a_{n-1} \cdot f'(a_{n-1}) + f(a_{n-1}) - f(b_{n-1})}{f'(b_n - 1) - f'(a_n - 1)},$$

где $a_n = a_{n-1}$, $b_n = c_{n-1}$ при $f'(c_{n-1}) \ge 0$,

$a_n = c_n - 1$, $b_n = b_n - 1$ при $f'(c_n - 1) < 0$.

После n шагов полагают $x^* \approx c_n$, $f^* \approx (c_n)$. Требуемая томность минимизации f(x) считается достигнутой, если производная $f'(c_n)$ достаточно близка к нулю, т.е. $|f'(c_n)| \le \varepsilon$, где ε – заданное число, характеризующее точность. Алгоритм приведен на рис. 7.

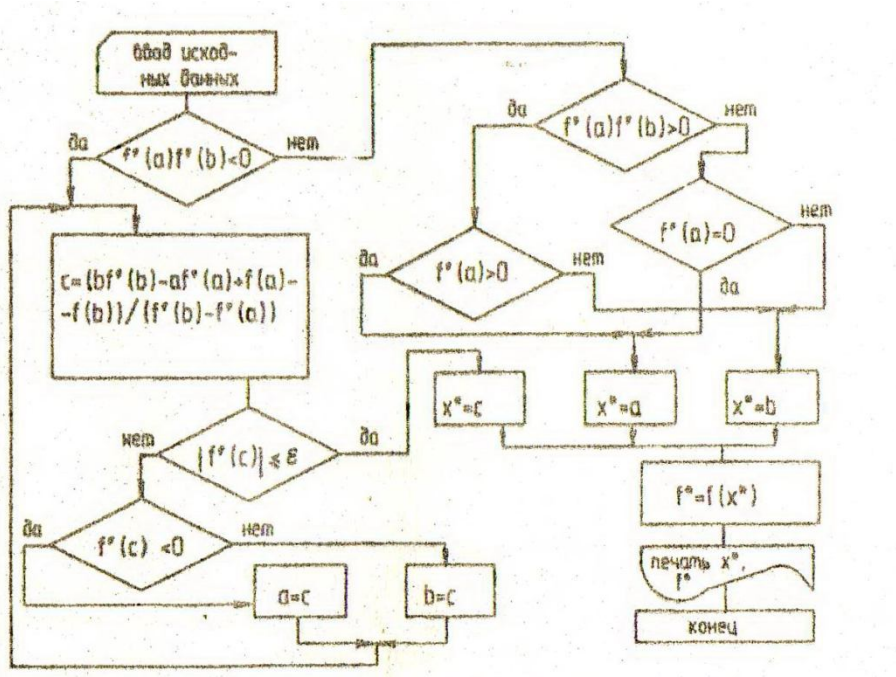


Рис. 7. Минимизация функции методом касательных

Метод Ньютона

Метод Ньютона, в котором используют не только первую, но и вторую производные функции f(x), при определенных условиях обеспечивает значительно более высокую, чем рассмотренные методы минимизации, скорость сходимости к точке минимума x*. Если f(x) — выпуклая дважды дифференцируемая функция, то выбрав начальное приближение x_0 можно достроить последовательность

$$x_n = x_{n-1} - \frac{f'(x_{n-1})}{f''(x_{n-1})}.$$

Считая неравенство $|f'(x_{n-1})| \le \varepsilon$ (ε – достаточно малое число) условием требуемой точности вычислений можно предположить что

$$x^* \approx x_n, f^* \approx f(x_n).$$

Рассмотрим далее методы поиска экстремума многомерных функций, основанные на вычислении производных.

Метод градиентного спуска

Пусть f(x) — выпуклая дифференцируемая функция и требуется найти, например, точку минимума x*, построим последовательности $x^{(k+1)} = x^{(k)} - \alpha_k f'(x^{(k)}), k = 0, 1, \ldots$, где величина α_k (параметрические шаги) выбрана достаточно малой для того, чтобы выполнялось условие $f(x^{(k+1)}) < f(x^{(k)})$. В качестве условий окончания вычислений обычно используют близость к нулю градиента $f'(x^{(k)})$, т.е. выполнение неравенств

$$\left| \frac{\partial f(x^{(k)})}{\partial x_i} \right| \le \varepsilon, \quad i = 1, 2, \ldots, n.$$

Алгоритм для метода градиентного спуска приведен на рис.8.

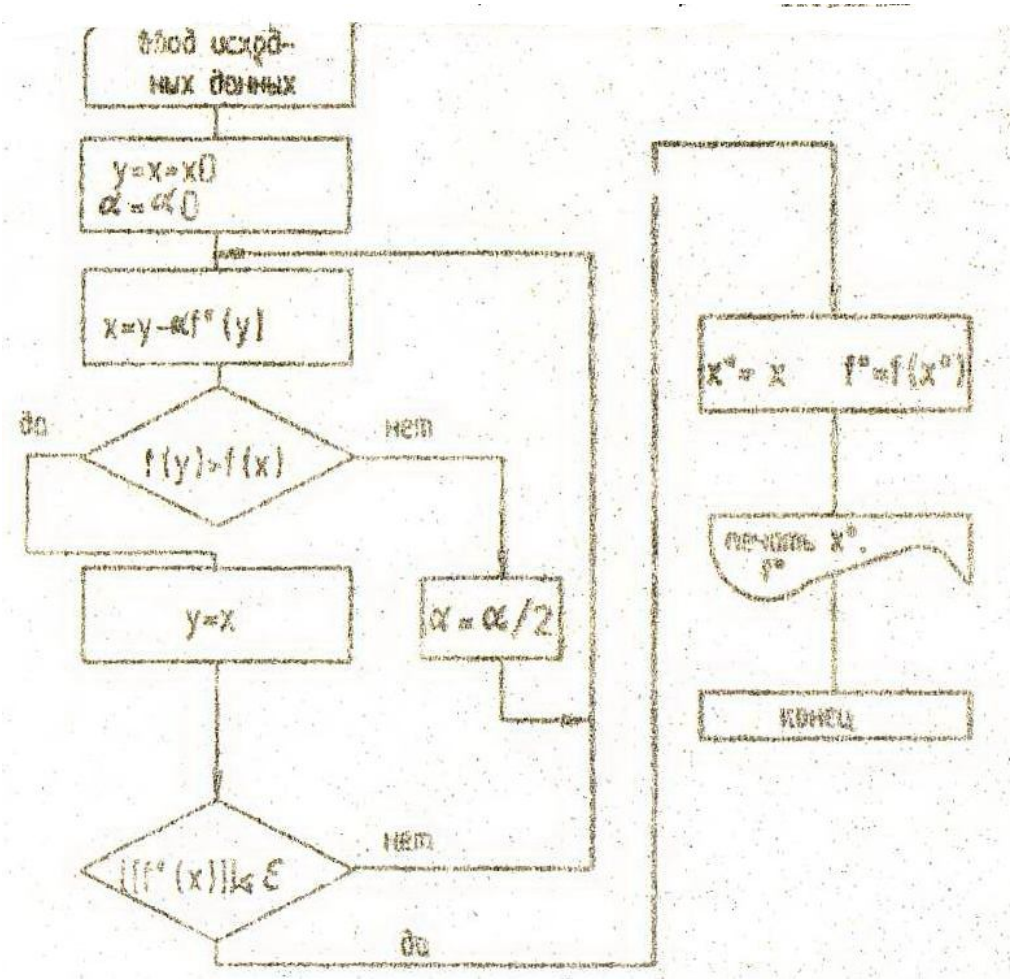


Рис. 8. Минимизация функции методом градиентного спуска

Метод наискорейшего спуска

Метод наискорейшего спуска отличается от метода градиентного спуска способом определения величины α_k, которую находят по условию $\Phi_k(\alpha_k) = \min \Phi_k(\alpha)$, где $\Phi_k(\alpha) = f[x^{(k)} - \alpha f'(x^{(k)})]$.

Алгоритм для метода наискорейшего спуска приведен на рис.9.

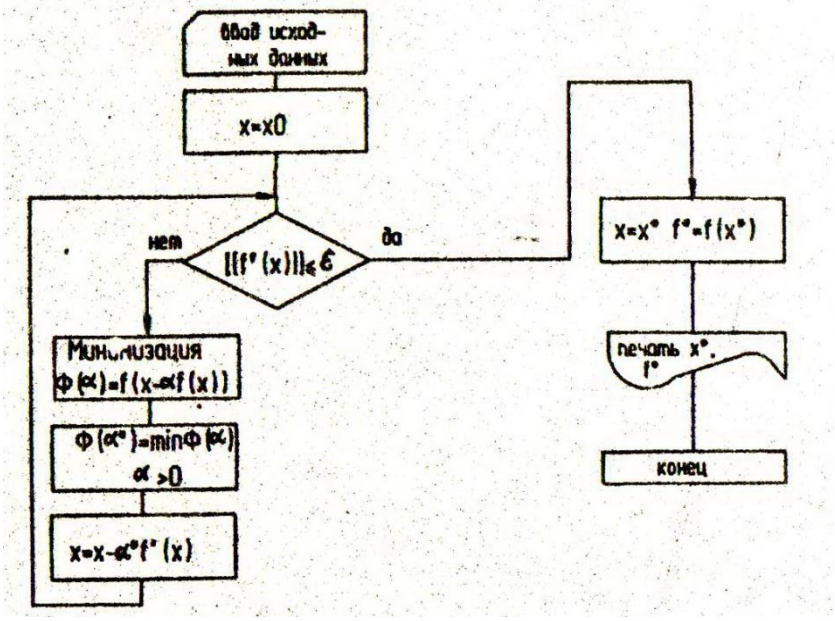


Рис. 9. Минимизация функции методом наискорейшего спуска

Если при построении последовательности приближений к точке минимума функции f(x) использовать информацию, содержащуюся в значениях не только первых, но и вторых производных f''(x), то при определенных условиях можно обеспечить более быструю сходимость этой последовательности, чем в градиентных методах.

Метод Ньютона для многомерных функций

Метод Ньютона применяется для безусловной минимизации выпуклых дважды дифференцируемых функций. В этом методе последовательность приближений $x^{(k)}$ к точке минимума функции f(x) строят с использованием первых и вторых производных следующим образом [3]: $x^{(k+1)} = x^{(k)} - [f''(x^{(k)})]^{-1} \cdot [f'(x^{(k)})]$, k=0, 1, … где $[f''(x^{(k)})]^{-1}$ — матрица, обратная матрице вторых производных функции f(x) в точке $x^{(k)}$.

Другие методы

Релаксационный поиск, или метод Гаусса—Зайделя, является итерационным многомерным локальным методом поиска, сущность которого состоит в том, что поиск экстремума проводят попеременно по всем аргументам. Недостатком релаксационного метода является безразличие к выбору направления, вследствие чего в релаксационном поиске может оказаться существенно больше шагов одномерного поиска, чем при градиентном.

Случайный локальный поиск отличается от градиентного или релаксационного поисков произвольным выбором направления.

Ограниченный поиск, или локальный многомерный поиск в условиях ограниченности зоны поиска, строго говоря, не является самостоятельным методом поиска.

Глобальные методы поиска отличаются тем, что вся область пространства параметров, в которой имеется минимум, покрывается множеством пробных точек в соответствии с выбранным законом. На основании полученной информаций строят глобальную модель поведения исследуемого технического объекта во всей области. Затем при помощи анализа этой модели указывают точку предполагаемого минимума или область ее расположения, которая меньше по сравнению с исходной областью. Далее процесс повторяют в этой области для уточнения положения минимума и т.д.

МЕТОДЫ РЕШЕНИЯ ЗАДАЧ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Задача линейного программирования (ЗЛП) состоит в минимизации линейной целевой функции $f(x) = f(x_1, \ldots, x_n) = \sum_{j=1}^{n} c_j x_j$, заданной системой линейных ограничений (равенств и (или) неравенств) на координаты $x_j \ (j = \overline{1, n})$.

Если ЗЛП содержит только ограничения равенства, то говорят, что она задана в каноническом виде

$$f(x) = \sum_{j=1}^{n} c_j x_j \to \min,$$

$$\sum_{j=1}^{n} a_j x_j = b_j, \ i = \overline{1, m}, \ x_j \geq 0.$$

Графические методы решения ЗЛП

Если ЗЛП содержит только две переменные и в ее условии нет ограничений

равенств, то такую задачу можно исследовать и решить графически.

Используя графический метод найти решение следующей ЗЛП:

$$f(x) = x_1 - 2x_2 \to \min$$
$$-x_1 + x_2 \le 0$$
$$2x_1 + x_2 \le 3$$
$$x_1, x_2 \ge 0$$

Изобразим на плоскости (x_1, x_2) допустимое множество данной задачи — многоугольник ABCD и одну из линий уровня $x_1 - 2x_2 = c$ с целевой функцией (рис. 10).

Направление убывания функции f(x) указывает вектор e = (-1;2) = -f'(x) — антиградиент.

Совершая параллельный перенос линии уровня вдоль направления e, находим ее крайнее положение. В этом положении прямая $x_1 - 2x_2 = c$ проходит через вершину B (1;1) многоугольника ABCD. Поэтому целевая функция f(x) принимает минимальное значение $f^* = f(1,1) = -1$. В общем случае ЗЛП может иметь и бесконечное множество решений.

Графический метод используют также для решения ЗЛП в каноническом виде с произвольным числом переменных x_j, если число свободных переменных системы уравнений не превосходит двух.

В примере [3], используя графический метод, найти решение следующей ЗЛП в каноническом виде

$$f(x) = x_1 + 9x_2 + 5x_3 + 3x_4 + 4x_5 + 14x_6 \to \min,$$

где

$$x_1 + x_4 = 20;$$
$$x_2 + x_5 = 50;$$
$$x_3 + x_6 = 30;$$
$$x_4 + x_5 + x_6 = 60;$$
$$x_j \ge 0;\ j = \overline{1,6}.$$

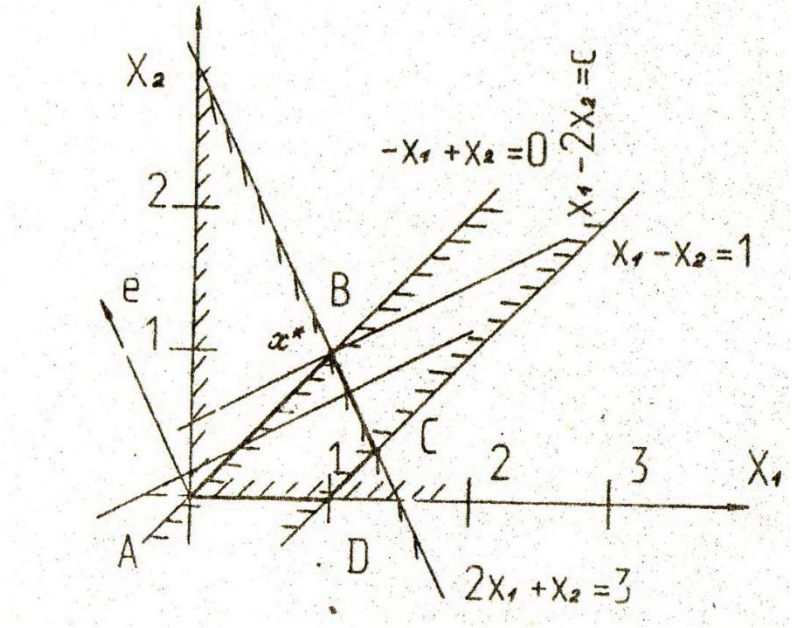


Рис. 10. Графический метод решения задачи линейного программирования

В данном случае матрица системы ограничений — равенств имеет вид

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Ее ранг r = 4 = m, причем минор, образованный первыми четырьмя столбцами, может быть выбран в качестве базисного. Число свободных переменных n-m=2, поэтому для решения задачи можно использовать графический метод.

Так, решив систему ограничений — равенств относительно базисных переменных $x_j,\ j = \overline{1,4}$, получим

$$\begin{cases} x_1 = -40 + x_5 + x_6; \\ x_2 = 50 - x_5; \\ x_3 = 30 - x_6; \\ x_4 = 60 - x_5 - x_6. \end{cases}$$

Исключив с помощью найденной системы переменные $x_1, \ldots x_4$ из выражения для целевой функции, находим $f(x)=740-7x_5+7x_6$. С учетом условия неотрицательности $x_j \geq 0$, $j = \overline{1,6}$:

$$f(x) = 740 - 7x_5 + 7x_6 \to \min,$$

где

$$x_5 + x_6 \geq 40;$$
$$x_5 \leq 50;$$
$$x_6 \leq 30;$$
$$x_5 + x_6 \leq 60;$$
$$x_5, x_6 \geq 0.$$

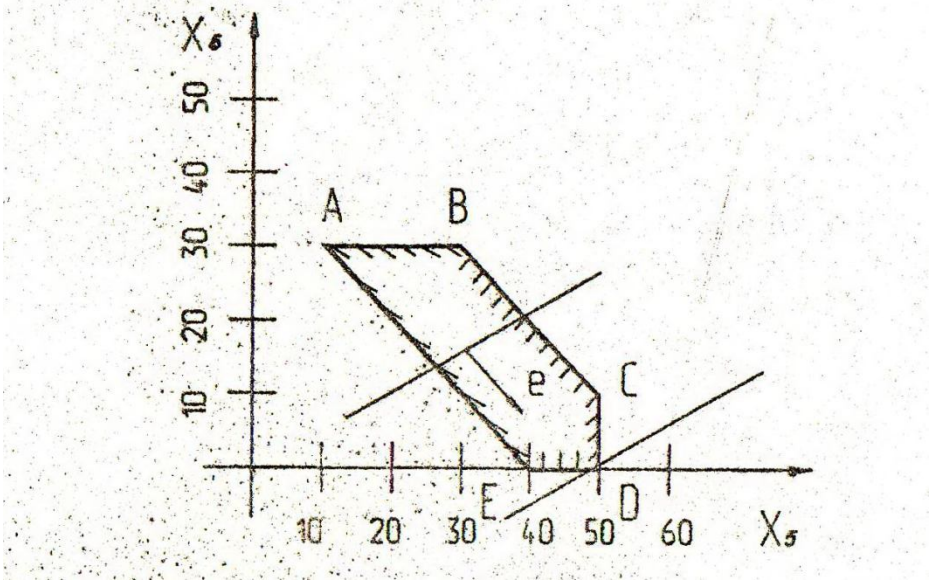


Рис. 11. Графический метод решения задачи линейного программирования

Допустимое множество U этой ЗЛП представляет собой многоугольник ABCDE (рис. 11). Перемещая линию уровня $-7x_5+7x_6=C$ функции f(x) по направлению вектора e=(7;-7), находим точку минимума f(x) – вершину D (50;0) многоугольника ABCDE. Подставив значения $x_5 = 50$, $x_6 = 0$ в систему равенств, окончательно находим x* = (10 , 0 , 30 , 10 , 50 , 0) , f* = 390.

Симплекс-метод решения ЗЛП

Общим методом решения произвольной ЗЛП является симплекс-метод, применение которого рассмотрим ниже.

Пример. Решить задачу линейного программирования симплекс методом, используя в качестве начальной угловой точки $x^{(0)}$.

$$f(x) = -5x_1 + 4x_2 - x_3 - 3x_4 - 5x_5 \to \min$$
$$3x_1 - x_2 + 2x_4 + x_5 = 5$$
$$2x_1 - 3x_2 + x_3 + 2x_4 + x_5 = 6$$
$$3x_1 - x_2 + x_3 + 3x_4 + 2x_5 = 9$$
$$x_j \geq 0;\ j = \overline{1,5}$$
$$x^{(0)} = (0, 0, 1, 2, 1)$$

Решение. Определим ранг матрицы и базовые элементы

$$A = \begin{pmatrix} 3 & -1 & 0 & 2 & 1 \\ 2 & -3 & 1 & 2 & 1 \\ 3 & -1 & 1 & 3 & 2 \end{pmatrix};\ r = 3.$$

Базовыми элементами являются x_3, x_4, x_5.

Выражаем базисные элементы через свободные:

$$\begin{cases} x_3 - x_1 - 2x_2 = 1 \\ x_4 + 2x_1 - 3x_2 = 2 \\ x_5 - x_1 + 5x_2 = 1 \end{cases} \Rightarrow$$

	x_1	x_2	
x_3	−1	−2	1
x_4	2	−3	1
x_5	−1	5	1
	−5	18	12

Координаты базисной точки (угловой точки)

$x^{(0)}$ = (0, 0, 1, 2, 1), так как все базисные элементы положительны, то данное базисное решение — допустимое и невырожденное. Исключаем базисные переменные из целевой функции:

$f(x) = -12 - 5x_1 + 18x_2$.

Строим симплекс-таблицу, среди P_j есть отрицательные (-5). Следовательно, данная угловая точка не является решением данной задачи линейного программирования.

В качестве опорного выбираем элемент $\alpha_{kl}^{(0)}$, находящийся на пересечении столбца x_1 и строки x_4. Заполняем новую симплекс-таблицу, пользуясь следующими правилами [3]:

поменять местами переменные x_k и x_l, а остальные переменные оставить на прежних местах;

на место опорного элемента поставить число $\frac{1}{\alpha_{kl}^{(0)}}$;

на остальных местах разрешающей строки записать соответствующие элементы исходной таблицы, деленные на опорный элемент;

на свободные места разрешающего столбца поставить со знаком «минус» соответствующие элементы исходной таблицы, деленные на опорный элемент;

оставшиеся свободные места в новой симплекс-таблице заполнить построчно следующим образом: из строки элементов исходной таблицы вычесть произведение ее элемента из разрешающего столбца на уже заполненную разрешающую строку новой таблицы

	x_4	x_2	
x_3	0,5	−3,5	2
x_1	0,5	−1,5	1
x_5	0,5	3,5	2
	2,5	10,5	17
	p_j		

Выразим базисные переменные x_1, x_3, x_5 через свободные x_2, x_4.

$$\begin{cases} x_1 = 1 - 0{,}5x_4 + 1{,}5x_2; \\ x_3 = 2 + 3{,}5x_2 - 0{,}5x_4; \\ x_5 = 2 - 0{,}5x_4 - 3{,}5x_2. \end{cases}$$

Затем преобразуем целевую функцию, исключая переменные x_1, x_3, x_5 к виду: $f(x) = -17 - 2{,}5x_4 + 10{,}5x_5$.

В нижней строке второй симплекс-таблицы все $p_j > 0$, дальнейшее уменьшение f(x) невозможно, поэтому $x^{(1)}$ (1, 0, 2, 0, 2) — точка экстремума (минимума),

$f^*(x) = f(x^{(1)}) = -17$.

Некоторые задачи нелинейного программирования можно успешно решать в том случае, если имеется возможность привести ах к задаче линейного программирования. В общем случае задача нелинейного программирования формулирует я следующим образом:

$$f(x) \to \min;$$
$$q_i(x) = b_i,\ i = 1, 2, \ldots, l$$
$$q_i(x) = b_i,\ i = l+1, \ldots, m,$$

где $f(x), q_i(x)$ — заданные (необязательно линейные) функции n-переменных.

Рассмотрим подход к решению следующей Задачи дрсбло-ли нейного программирования:

$$f(x) = \frac{-2x_1 + x_2}{x_1 + 2x_2 + 1} \to \min,$$

где

$$x_1 - 2x_2 \le 2;$$
$$2x_1 + x_2 + x_3 = 6;$$
$$x_j \ge 0;\ j = 0, 1, 2, 3.$$

Знаменатель $x_1 + 2x_2 + 1$ целевой функции положителен при всех x из допустимого множества U, так как $x_1, x_2 \ge 0$. Введя переменные $y_0 = l/(x_1 + 2x_2 + 1)$, $x_j = y_0 x_j$, $j = \overline{1, 3}$ получим следующую задачу линейного программирования:

$$\tilde{f}(y) = -2y_1 + 2y_2 \to \min,$$

где

$$y_1 - 2y_2 - 2y_0 \le 0;$$
$$2y_1 - y_2 + y_3 - 6y_0 = 0;$$
$$y_1 + 2y_2 + y_0 = 1;$$
$$y_j \ge 0;\ j = 0, 1, 2, 3.$$

Приведя эту задачу к каноническому виду и решив ее симплекс методом, находим: $y_0^* = \frac{1}{3};\ y_1^* = \frac{2}{3};\ y_2^* = 0;\ y_3^* = \frac{2}{3}$, откуда решение исходной, задачи будет иметь вид:

$$x^* = (2; 0; 2),\ f^* = \tilde{f}^* = -\frac{4}{3}.$$

МЕТОД ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Этот метод обычно используют для решения задач динамической оптимизации в дискретной форме.

Основная идея динамического программирования состоит в использовании рекуррентного соотношения, основанного на принципе оптимальности. Это соотношение позволяет каждый временной интервал (или шаг) оптимизировать независимо от других интервалов. Такой подход сокращает объем вычислений на несколько порядков и преобразует задачу совместного определения оптимальных переменных для всех интервалов в задачу последовательного определения оптимальных переменных управления на каждом временном интервале.

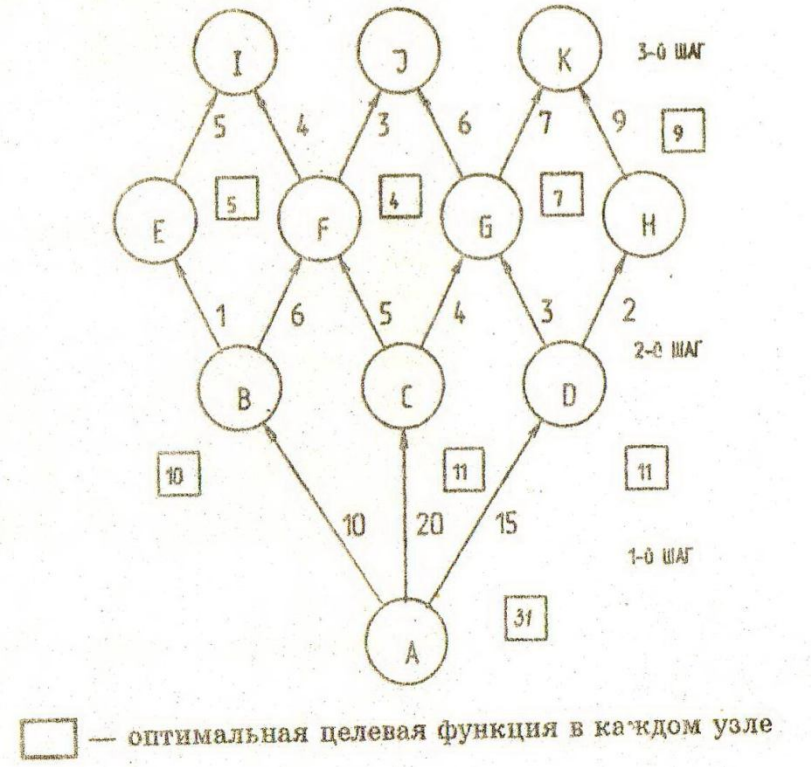


Рис. 12. Типичное дерево решений для трехшаговой задачи

На рис. 12 показано типичное дерево решений для трехшаговой задачи динамической оптимизации. Каждый узел описывает состояние технического объекта на данном шаге. С каждым узлом связаны определенные значения переменной состояния Y_j и узловой функции Ф.

Траектория представляет собой возможный набор переменных управления x_i. Таким образом, каждому узлу в конце траектории могут поставлены в соответствие определенные значения переменных состояния и целевой функции, которые получаются в результате учета, некоторого набора поправок, связанных с изменением переменных управление.

Предположим, что необходимо решить задачу по определению максимума, при этом прирост значения целевой функции на каждом отрезке траектории будет следующим:

1-й шаг: $\Delta\Phi^{(A \to B)} = 10$; $\Delta\Phi^{(A \to C)} = 20$; $\Delta\Phi^{(A \to D)} = 15$;

2-й шаг: $\Delta\Phi^{(B \to E)} = 1$; $\Delta\Phi^{(B \to F)} = 6$; $\Delta\Phi^{(C \to F)} = 5$;

$\Delta\Phi^{(C \to G)} = 4$; $\Delta\Phi^{(D \to G)} = 3$; $\Delta\Phi^{(D \to H)} = 2$;

3-й шаг: $\Delta\Phi^{(E \to I)} = 5$; $\Delta\Phi^{(F \to I)} = 4$; $\Delta\Phi^{(F \to J)} = 3$;

$\Delta\Phi^{(G \to J)} = 6$; $\Delta\Phi^{(G \to K)} = 7$; $\Delta\Phi^{(H \to K)} = 9$;

где $\Delta\Phi^{(I \to J)}$ - приращение целевой функции, связанное с движением $I \to J$. Построение дерева решений для задачи будем осуществлять, двигаясь в обратном направлении.

В соответствии с принципом оптимальности, чтобы весь процесс был оптимальным, каждый шаг должен быть оптимальным. На каждом шаге, начиная с 3-го, определяем максимальное значение целевой функции в каждом узле. На 2-м шаге будем пользоваться максимальными значениями целевой функции во входных узлах 3-го шага, т.е. узлов E, F, G, H. Траектории этого шага, максимизирующие целевую функцию Ф, можно записать так:

$(E \to I)$; $(F \to I)$; $(G \to K)$; $(H \to K)$.

Максимальные значения приращения целевой функции во входных узлах 3-го шага равны:

$$\Phi_{opt}(E) = \Delta\Phi^{(E \to I)} = 5;$$

$$\Phi_{opt}(F) = \Delta\Phi^{(F \to I)} = 4;$$

$$\Phi_{opt}(G) = \Delta\Phi^{(G \to K)} = 7;$$

$$\Phi_{opt}(H) = \Delta\Phi^{(H \to K)} = 9;$$

Для 2-го шага исходными данными является максимальные значения целевой функции в узлах E, F, G, H. Определим возможные траектории в узлах B, C, D . Для узла B

$$\Phi(B) = \Delta\Phi^{(B \to E)} + \Phi_{opt}(E) \text{ или}$$
$$\Phi(B) = \Delta\Phi^{(B \to F)} + \Phi_{opt}(F).$$

Максимальное значение $\Phi_{opt}(B)$ равно 10. Оптимальная траектория начинающаяся в узле B: $B \to F \to I$. Аналогично определим $\Phi_{opt}(C) = \Delta\Phi^{(C \to G)} + \Phi_{opt}(G) = 4 + 7 = 11.$

Оптимальная траектория, начинающаяся в узле C: $C \to G \to K$.Максимум целевой функции $\Phi_{opt}(A) = \Delta\Phi_{opt}^{(A \to C)} + \Delta\Phi_{opt}^{(C \to G)} + \Delta\Phi_{opt}^{(G \to K)} = 31$, что иллюстрирует принцип оптимальности.

ЛИТЕРАТУРА

1. Бегларян В.Х. Проектирование приборов, оптимальных по конструктивно-технологическим параметрам. М.: Машиностроение, 1997,177 с.

2. Первозванский А.Н. Поиск. М.:Наука, 1969.166 с.

3. Сборник задач по математике для втузов. Ч. 4. Методы стабилизации. Уравнения в частных производных. Интегральные уравнения: Учебное пособие/ Э.А.Вуколов, А.В.Ефимов, В.Н.Земсков и др., под ред. А.Ф. Ефимова. 2-е изд., перераб. М.:Наука, 1990. 340 с.

ОГЛАВЛЕНИЕ

Введение	4
Алгоритмы оптимального проектирования	5
Методы поиска экстремума	12
Метод перебора	12
Метод деления отрезка пополам	13
Метод золотого сечения	14
Метод ломаных	15
Метод касательных	16
Метод Ньютона	17
Метод градиентного спуска	17
Метод наискорейшего спуска	18
Метод Ньютона для многомерных функций	19
Другие методы	19
Методы решения задач линейного программирования	19
Графические методы решения ЗЛП	19
Симплекс-метод решения ЗЛП	22
Метод динамического программирования	24
Литература	26