

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное  
образовательное учреждение высшего образования  
«Тульский государственный университет»

Институт прикладной математики и компьютерных наук  
Кафедра «Прикладная математика и информатика»

Утверждено на заседании кафедры  
«Прикладная математика и информатика»  
24 января 2022 г., протокол № 5

Заведующий кафедрой

 М.В. Грязев

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**  
**по выполнению практических (семинарских) занятий**  
**по дисциплине (модулю)**  
**«Методы оптимизации»**

**основной профессиональной образовательной программы**  
**высшего образования – программы бакалавриата**

по направлению подготовки  
**01.03.02 Прикладная математика и информатика**

с направленностью (профилем)  
**Прикладная математика и информатика**

Форма обучения: очная

Идентификационный номер образовательной программы: 010302-01-22

Тула 2022 год

**Разработчик методических указаний**

Горбачев Д.В., профессор каф. ПМийИ, д.ф.-м.н.



---

*(ФИО, должность, ученая степень, ученое звание)*

---

*(подпись)*

## Список тем

1. Вычисление производных и дифференциалов, решение одномерной прикладной гладкой экстремальной задачи
2. Решение прикладных гладких задач
3. Решение типовых гладких задач
4. Решение гладких задач с равенствами
5. Решение гладких задач с равенствами и неравенствами
6. Решение выпуклых задач
7. Решение задачи Конторовича, другие задачи линейного программирования прикладного характера
8. Решение задач симплекс-методом
9. Решение разных задач симплекс-методом
10. Решение задач двойственным симплекс-методом
11. Решение транспортной задачи
12. Решение задач квадратичного программирования, применение одномерных численных методов
13. Применение многомерных численных методов нелинейной оптимизации

Далее приводятся примеры с решениями основных задач и необходимые теоретические сведения.

## Введение

Целью освоения дисциплины «Методы оптимизации» является изучение теории конечномерных экстремальных (оптимизационных) задач на нахождение максимумов и минимумов (включая гладкие и выпуклые задачи, задачи линейного и квадратичного программирования), знакомство с численными методами их решения и приложениями, в том числе при помощи современных математических пакетов и открытых библиотек [1–12].

Задачами освоения дисциплины являются:

- изучение теории и методов решения конечномерных экстремальных задач;
- освоение базовых численных методов оптимизации;
- знакомство с прикладными задачами экстремального характера.

## Основные типы экстремальных задач. Методы их решения

Экстремальными (или оптимизационными) называются задачи на нахождение минимумов ( $\min$ ) и максимумов ( $\max$ ), объединяемых общим термином экстремум ( $\text{extr}$ ) (оптимальных решений). Подобные задачи настолько часто встречаются на практике, что термин «Оптимизация» (Optimization), наряду с терминами «Анализ» (Analysis) и «Геометрия» (Geometry), является основными математическими «брендами».

В этом практическом занятии повторяем уже изученные на курсе методов оптимизации приемы решения конечномерных задач.

Приведем простые примеры экстремальных задач.

1. Найти  $xe^{-x^2} \rightarrow \min$ . *Указание:*  $f' = 0$ ,  $f'' > 0$ .

2. Найти  $x_1^2 + x_2^2 - x_1x_2 \rightarrow \min$ ,  $0 \leq x_1 + x_2 \leq 1$ . *Указание:* свести к задаче внутри  $D$  и на ее границе.

3. Найти форму пожарного ведра максимального объема с заданной образующей. *Указание:* формализовать задачу и решить методом Лагранжа.

4. Найти расстояние между двумя треугольниками на плоскости, в пространстве. *Указание:* взять случайные треугольники и формализовать задачу.

5. Найти  $f(x) = -6x_1 - 8x_2 \rightarrow \min$  при условиях  $x_j \geq 0$ ,  $j = 1, \dots, 4$ ,

$$2x_1 + 5x_2 + x_3 = 20, \quad 12x_1 + 6x_2 + x_4 = 72.$$

*Указание:* решить как задачу линейного программирования.

## Основные типы экстремальных задач. Matlab/Maple

В этом практическом занятии вспоминаем основные численные методы решения конечномерных экстремальных задач, изучаемые в курсе «Методы оптимизации», выполняем простейшие примеры в пакетах Matlab и Maple. Примеры взять из предыдущего занятия.

1. Найти  $xe^{-x^2} \rightarrow \min$ .
2. Найти  $x_1^2 + x_2^2 - x_1x_2 \rightarrow \min, 0 \leq x_1 + x_2 \leq 1$ .
3. Найти форму пожарного ведра максимального объема с заданной образующей.
4. Найти расстояние между двумя треугольниками на плоскости, в пространстве.
5. Найти  $f(x) = -6x_1 - 8x_2 \rightarrow \min$  при условиях  $x_j \geq 0, j = 1, \dots, 4$ ,

$$2x_1 + 5x_2 + x_3 = 20, \quad 12x_1 + 6x_2 + x_4 = 72.$$

**Прямые методы решения  
конечномерных экстремальных задач.  
Одномерный случай**

Рассматривается конечномерная экстремальная задача

$$f(x) \rightarrow \min, \quad x \in D \subset \mathbb{R}^n.$$

Прямые методы используют только значения целевой функции. На практике они применяются, когда больше информации о  $f$  неизвестно или вычислить производные проблематично.

Для гарантированной сходимости в окрестности решения  $\hat{x} \in (a, b)$  целевая функция  $f(x)$  должна быть унимодальной, когда она слева точки  $\hat{x}$  невозрастает, а справа не убывает. Например, выпуклая в окрестности решения функция является унимодальной.

*Пример:*  $f(x) = x^2 + 3x(\ln x - 1)$ ,  $x \in [0.5, 1]$ .

*Метод половинного деления.* Полагаем

$$a'_k = \frac{a_{k-1} + b_{k-1} - \delta}{2}, \quad b'_k = \frac{a_{k-1} + b_{k-1} + \delta}{2},$$

и

$$[a_k, b_k] = \begin{cases} [a_{k-1}, b'_k], & \text{если } f(a'_k) < f(b'_k), \\ [a'_k, b_{k-1}] & \text{иначе.} \end{cases}$$

Здесь  $[a_0, b_0] = [a, b]$ ,  $\delta > 0$  — малый параметр метода.

Критерий окончания следующий:  $x^* = (a_N + b_N)/2$ , если  $\varepsilon_N = (b_N - a_N)/2 \leq \varepsilon$ , где  $\varepsilon$  — точность метода.

За  $N$  вычислений функции  $f$  интервал неопределенности уменьшается как  $O(2^{N/2}) = O((0.71)^N)$ .

*Метод золотого сечения.* Этот метод основан на понятии золотого сечения и является более эффективным в плане скорости уменьшения интервала неопределенности. Золотым сечением называется такое разбиение отрезка, когда отношение большей части к меньшей равно отношению длины отрезка к большей части. Нетрудно посчитать, что это отношение равно  $\varphi = \frac{1+\sqrt{5}}{2}$ .

Золотое сечение отрезка  $[a, b]$  осуществляют две точки:

$$\alpha = a + \frac{3 - \sqrt{5}}{2} (b - a), \quad \beta = a + b - \alpha,$$

причем  $\alpha$  одновременно является второй точкой золотого сечения отрезка  $[a, \beta]$ , а  $\beta$  — первой точкой отрезка  $[\alpha, b]$ .

Перед началом итерации полагаем  $[a_0, b_0] = [a, b]$ ,  $A_0 = f(\alpha_0)$ ,  $B_0 = f(\beta_0)$ . Далее

$$[a_k, b_k] = \begin{cases} [a_{k-1}, \beta_{k-1}], & \text{если } A_{k-1} < B_{k-1}, A_k = f(\alpha_k), B_k = A_{k-1}, \\ [\alpha_{k-1}, b_{k-1}] & \text{иначе, } A_k = B_{k-1}, B_k = f(\beta_k). \end{cases}$$

Здесь на каждом шаге требуется только одно вычисление функции  $f$ . Критерий окончания можно использовать такой же, как и в методе половинного деления.

За  $N$  вычислений функции  $f$  интервал неопределенности уменьшается как  $O(\varphi^{-N}) = O((0.62)^N)$ .

Оптимальным одномерным прямым методом в отмеченном выше плане является метод, основанный на последовательности чисел Фибоначчи. Его предлагается рассмотреть самостоятельно.

**Примеры решения задач при помощи Matlab и Maple.**  
Основные функции Matlab: `fminsearch`, `fminbnd`, `fminunc`; Maple: `Minimize/Maximize`.

На Maple реализовать методы деления пополам и золотого сечения.

**Прямые методы решения  
конечномерных экстремальных задач.  
Многомерный случай**

Рассмотрим многомерный случай функции  $f(x) = f(x_1, \dots, x_n)$ .

*Пример:* найти

$$\min 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

Эта функция называется банана-функцией Розенброка.

*Метод покоординатного спуска.* Пусть  $x_k$  — текущее приближение. Тогда  $j$ -ю координату  $x_{k+1,j}$  точки  $x_{k+1}$ ,  $j = 1, \dots, n$ , находят из решения одномерной задачи

$$f(x_{k1}, \dots, x_{k,j-1}, t, x_{k,j+1}, \dots, x_{kn}) \rightarrow \min_t, \quad x_{kj} = t^*.$$

*Метод деформируемого многогранника (Нелдера–Мида).* В этом случае вначале задаются  $n + 1$  векторов  $x_i$ ,  $i = 0, 1, \dots, n$ , так что векторы  $x_i - x_0$  при  $i > 0$  линейно независимы. Тогда точки  $x_i$  будут вершинами невырожденного многогранника (симплекса). Обычно берется правильный симплекс.

Пусть теперь  $\{x_i\}_{i=0}^n$  — текущий симплекс. Считаем, что точки  $x_i$  отсортированы в порядке неубывания значений  $f(x_i)$ :  $f(x_0) \leq f(x_1) \leq \dots \leq f(x_n)$ . Опишем очередную итерацию.

Точка  $x_n$  отбрасывается как самая неудачная. Далее вычисляем центр тяжести оставшихся точек  $\bar{x} = \frac{1}{n} \sum_{i \neq n} x_i$  и строим при помощи отражения пробную точку  $y = 2\bar{x} - x_n$ .

Возможны три случая.

1. Если  $f(x_0) \leq f(y) \leq f(x_n)$ , то пробная точка «удачная» и она заносится в новый набор вершин симплекса вместо  $x_n$ .

2. Если  $f(y) < f(x_0)$  (значение функции уменьшилось), тогда направление отражения считается «удачным» и выполняется попытка еще больше уменьшить значение за счет растяжения симплекса:  $y' = \bar{x} + \alpha(\bar{x} - x_n)$  ( $\alpha \approx 2$ ). Если  $f(y') < f(y)$ , то в набор вместо  $x_n$  добавляют  $y'$ , иначе  $y$ .

3. Если  $f(y) > f(x_{n-1})$ , то симплекс сжимают: при  $f(y) > f(x_n)$  точку вершину  $x_n$  заменяют точкой  $y' = \bar{x} - \beta(\bar{x} - x_n)$  ( $\beta \approx 1/2$ ), иначе точкой  $y' = \bar{x} + \beta(\bar{x} - x_n)$ .

Чтобы устранить накопление излишних деформаций после определенного числа итераций текущий симплекс заменяют правильным симплексом (операция восстановления), сохраняя при этом две лучшие точки текущего набора вершин.

**Примеры решения задач при помощи Matlab и Maple.**  
Основные функции Matlab: `fminsearch`, `fminbnd`, `fminunc`; Maple: `Minimize/Maximize`.

Возьмем пример из справки Matlab.

```
banana = @(x)100*(x(2)-x(1)^2)^2+(1-x(1))^2;  
[x,fval] = fminsearch(banana,[-1.2, 1])  
x =  
... 1.0000 ... 1.0000  
fval =  
... 8.1777e-010
```

## Методы решения при помощи производных. Градиентные методы

Рассматривается безусловная конечномерная экстремальная задача

$$f(x) \rightarrow \min, \quad x \in D \subset \mathbb{R}^n,$$

где  $f$  — гладкая функция и  $D$  — открытое множество. Например,

$$e^{x_1} (4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1) \rightarrow \min.$$

Общей чертой рассматриваемых далее методов является то, что итерации в них записываются следующим образом:

$$x_{k+1} = x_k - \tau_k e_k,$$

где  $e_k$  — вектор-направление спуска,  $\tau_k$  — шаг. Критерием окончания обычно является условие

$$|e_k| = O(\varepsilon) \Leftrightarrow |x_{k+1} - x_k| = O(\varepsilon).$$

**Градиентные методы.** Гладкая функция убывает наискорейшим образом в направлении антиградиента  $-f'$ . Таким образом, получаем, что

$$x_{k+1} = x_k - \tau_k g_k, \quad g_k = f'(x_k).$$

Такой способ организации итераций называется градиентным спуском. Критерием окончания итераций обычно является близость к нулю градиента:  $|g_k| \leq \varepsilon$ .

*Метод наискорейшего спуска.* В этом случае шаг определяется из решения одномерной экстремальной задачи

$$\tau_k = \operatorname{argmin}_{\tau > 0} f(x_k - \tau g_k).$$

Например, если  $f$  — квадратичная функция, то

$$\tau_k = \frac{(g_k, g_k)}{(Ag_k, g_k)}.$$

В общем случае для нахождения  $\tau_k$  можно воспользоваться одномерными численными методами поиска минимума.

*Метод сопряженных градиентов.* Этот метод связан с градиентным спуском, однако лишен некоторых отмеченных выше недостатков. В этом случае шаг  $\tau_k$  определяется как и выше, а направление спуска выбирается следующим образом:

$$e_k = g_k + \alpha_k e_{k-1}, \quad \alpha_k = \frac{g_k^2}{g_{k-1}^2}, \quad e_0 = g_0.$$

Название метода обусловлено тем, что в случае квадратичной функции направления  $e_k$  и  $e_l$  сопряжены относительно матрицы  $A$ :  $(Ae_k, e_l) = 0$  при  $k \neq l$ .

*Метод сопряженных градиентов для системы линейных уравнений.* Рассмотрим систему линейных уравнений  $Ax = b$  с положительно определенной матрицей  $A$ . Ее решение совпадает с точкой минимума квадратичной функции  $f(x) = \frac{1}{2}(Ax, x) - bx$ . Если применить к последней метод сопряженных градиентов, то получаем известный одноименный итерационный способ решения системы линейных уравнений. Предлагается выписать его самостоятельно.

### Примеры решения задач при помощи Matlab и Maple.

Основные функции Matlab: `fminsearch`, `fminbnd`, `fminunc`, `pcg`, `cgs`, `bicg`; Maple: `Minimize/Maximize`, для аналитического решения в простых случаях можно использовать функцию `extrema`.

Приведем пример решения системы уравнений методом сопряженных градиентов и стандартные функции минимизации.

```
A = [2 1; 1 2];
b = [1; 1];
norm(A\b - pcg(A, b))
ans =
    2.0015e-016

function f = objfun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);

x0 = [-1, 1];
[x, fval, exitflag, output] = fminunc(@objfun, x0)
x =
    0.5000    -1.0000
fval =
    3.6609e-015
exitflag =
    1
output =
    iterations: 8
    funcCount: 66
    stepsize: 1
    firstorderopt: 1.2284e-007
    algorithm: 'medium-scale: Quasi-Newton line search'
```

```
.....message: [1x441 char]

> f:=exp(x[1])*(4*x[1]^2+2*x[2]^2+4*x[1]*x[2]+2*x[2]+1);
> Digits:=20:
> Optimization[Minimize](f,initialpoint={x[1]=-1,x[2]=1});
.. [.27104977449742243528e-15, [x[1]=.49999999112458726133,
.. x[2]=-.99999998926842954843]]
```

## Методы решения при помощи производных. Метод Ньютона

В методе Ньютона используется гессиан функции  $f \in C^2$ . Итерации записываются следующим образом:

$$x_{k+1} = x_k - \tau_k e_k, \quad H e_k = g_k,$$

где  $H = f''(x_k)$  — гессиан.

В окрестности решения метод сходится очень быстро, а именно квадратично (сверхлинейно):  $\varepsilon_k = O(q^{2^k})$ ,  $q < 1$ , что гораздо быстрее любого из градиентных методов. Поэтому используют простые критерий окончания итераций, например,  $|x_{k+1} - x_k| \leq \varepsilon$ .

Шаг  $\tau_k$  необходим для сходимости метода, когда текущие приближения далеко от решения. Его можно выбирать как и в методе наискорейшего спуска.

Основной сложностью в методе Ньютона является необходимость на каждом шаге вычислять гессиан  $H$  и решать с ним систему линейных уравнений. Для упрощения вычислений часто применяют так называемые квази-методы Ньютона, в которых либо матрицу  $H$  вычисляют с пропусками итераций, либо для  $H$  применяются приближения, не использующие вторых производных, либо как-то аппроксимируют матрицу  $H^{-1}$ .

**Примеры решения задач при помощи Matlab и Maple.** Основные функции Matlab: `fminsearch`, `fminbnd`, `fminunc`, `pcg`, `cgs`, `bicg`; Maple: `Minimize/Maximize`, для аналитического решения в простых случаях можно использовать функцию `extrema`.

Приведем пример решения системы уравнений методом сопряженных градиентов и стандартные функции минимизации.

```
function f = objfun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);

x0 = [-1,1];
[x,fval,exitflag,output] = fminunc(@objfun,x0)
x =
    0.5000    -1.0000
fval =
    3.6609e-015
exitflag =
    1
```

```

output =
.....iterations: 8
.....funcCount: 66
.....stepsize: 1
.....firstorderopt: 1.2284e-007
.....algorithm: 'medium-scale: Quasi-Newton line search'
.....message: [1x441 char]

> f:=exp(x[1])*(4*x[1]^2+2*x[2]^2+4*x[1]*x[2]+2*x[2]+1);
> Digits:=20:
> Optimization[Minimize](f,initialpoint={x[1]=-1,x[2]=1});
.. [.27104977449742243528e-15, [x[1] = .49999999112458726133,
.. x[2] = -.99999998926842954843]]

```

## Задача линейного программирования. Симплекс-метод

Задача линейного программирования (LP-задача) заключается в нахождении экстремума линейной функции при линейных ограничениях. В векторном каноническом виде она записывается следующим образом:

$$f(x) = cx \rightarrow \min, \quad Ax = b, \quad x \in \mathbb{R}_+^n \quad (\text{или просто } x \geq 0),$$

где  $A \in \mathbb{R}^{m \times n}$  — матрица ограничений. К этому виду можно привести любую задачу линейного программирования.

*Пример.* Найти  $f(x) = -6x_1 - 8x_2 \rightarrow \min$  при условиях  $x_j \geq 0$ ,  $j = 1, \dots, 4$ ,

$$2x_1 + 5x_2 + x_3 = 20, \quad 12x_1 + 6x_2 + x_4 = 72.$$

Здесь  $c = (-6, -8, 0, 0)$ ,  $A = \begin{pmatrix} 2 & 5 & 1 & 0 \\ 12 & 6 & 0 & 1 \end{pmatrix}$ ,  $b = (20, 72)^T$ .

**Симплекс-метод.** Симплекс-метод подробно изучается в курсе методов-оптимизации, поэтому здесь для напоминания решим пример выше. Вначале применим метод искусственного базиса для нахождения начальной точки  $x_0$ . Для этого составляем вспомогательную задачу линейного программирования

$$x'_1 + x'_2 \rightarrow \min,$$

$$x'_1 + 2x_1 + 5x_2 + x_3 = 20, \quad x'_2 + 12x_1 + 6x_2 + x_4 = 72.$$

с начальной симплекс-таблицей

	$x_1$	$x_2$	$x_3$	$x_4$	
$x'_1$	2	5	1	0	20
$x'_2$	12	6	0	1	72
	-14	-11	-1	-1	-92

Дальнейшие преобразования стандартны:

	$x_1$	$x_2$	$x_4$			$x_1$	$x_2$	
$x_3$	2	5	0	20	$x_3$	2	5	20
$x'_2$	12	6	1	72	$x_4$	12	6	72
	-12	-6	-1	-72		0	0	0

Возвращаемся к исходной задаче:

$$\begin{array}{c|cc|c} & x_1 & x_2 & \\ \hline x_3 & 2 & 5 & 20 \\ x_4 & 12 & 6 & 72 \\ \hline & -6 & -8 & 0 \end{array} \Rightarrow \begin{array}{c|cc|c} & x_4 & x_2 & \\ \hline x_3 & -1/6 & 4 & 8 \\ x_1 & 1/12 & 1/2 & 6 \\ \hline & 1/2 & -5 & 36 \end{array} \Rightarrow$$

$$\begin{array}{c|cc|c} & x_4 & x_3 & \\ \hline x_2 & -1/24 & 1/4 & 2 \\ x_1 & 5/48 & -1/8 & 5 \\ \hline & 7/24 & 5/4 & 46 \end{array} \Rightarrow \hat{x} = (5, 2, 0, 0), \quad S_{\min} = -46.$$

**Примеры решения задач при помощи Matlab и Maple.**

Основные функции Matlab: linprog; Maple: minimize/maximize (пакет simplex), LPSolve.

Решим пример выше на Maple.

```

> obj := -6*x[1] - 8*x[2];
> cs := {2*x[1] + 5*x[2] + x[3] = 20, 12*x[1] + 6*x[2] + x[4] = 72};
> v := simplex[minimize](obj, cs, NONNEGATIVE);
.. v := {x[1] = 5, x[2] = 2, x[3] = 0, x[4] = 0}
> assign(v); obj;
.. -46

```

## Задача линейного программирования. Метод Кармаркара

Метод внутренней точки (Кармаркара) записывается для задачи в общей форме на максимум:  $cx \rightarrow \max, Ax \leq b$ .

Пусть  $x_0 \in D = \{x \in \mathbb{R}^n: Ax \leq b\}$  — начальная допустимая точка (в задаче линейного программирования  $D$  — выпуклый многогранник, возможно неограниченный или пустой). Ее определение в общем случае является нетривиальной задачей. Есть вариации метода, когда точка  $x_0$  произвольная и после нескольких начальных итераций она попадает в  $D$ .

Очередная  $k$ -я итерация выглядит следующим образом:

$$v_k = b - Ax_k, \quad D_v = \text{diag}(v_k),$$

$$Mh_x = c, \quad M = A^T D_v^{-2} A, \quad h_v = -Ah_x.$$

Если  $h_v \geq 0$ , то  $S_{\min} = -\infty$ , решений нет. Иначе полагаем

$$\alpha = \gamma \cdot \min \{-v_{kj}/h_{vj} : h_{vj} < 0, j = 1, \dots, n\},$$

где  $\gamma \in (0, 1]$  — параметр метода, и  $x_{k+1} = x_k + \alpha h_x$ .

Критерием остановки может случить выполнение неравенства  $|c(x_{k+1} - x_k)| / \max\{1, |cx_k|\} \leq \varepsilon$ .

**Примеры решения задач при помощи Matlab и Maple.**  
Основные функции Matlab: linprog; Maple: minimize/maximize (пакет simplex), LPSolve.

Решим пример выше на Matlab.

```
f = [-6; -8; 0; 0];
Aeq = [2.5 1 0; 12.6 0 1];
beq = [20; 72];
lb = zeros(4,1);
[x,fval,exitflag,output] = linprog(f, [], [], Aeq, beq, lb)
x =
    5.0000
    2.0000
    0.0000
    0.0000
fval =
   -46.0000
exitflag =
    1
```

```
output :=  
.....iterations: 6  
.....algorithm: 'large-scale: interior point'  
.....cgiterations: 0  
.....message: 'Optimization terminated.'  
.....constrviolation: 1.0658e-014
```

Еще задание: сгенерировать задачу для 500 переменных и сравнить скорости решения в Matlab и Maple.

## Задачи нелинейного программирования. Разные методы решения

Рассматриваются условные конечномерные непрерывные экстремальные задачи. Еще их называют задачами математического программирования. Общая такая задача ставится следующим образом:

$$f_0(x) \rightarrow \min, \quad x \in A \subset \mathbb{R}^n,$$

$$f_i(x) \leq 0, \quad i = 1, \dots, m', \quad f_i(x) = 0, \quad i = m' + 1, \dots, m.$$

Здесь  $A$  задается, как правило, геометрически. Множество допустимых элементов  $D$  является его пересечением  $A$  множеством решений неравенств и равенств. Если множество  $A$  — выпуклый многогранник, а функции  $f_i$  линейны, то получаем задачу линейного программирования.

В гладком случае необходимые условия экстремума в задаче математического программирования даются теоремой Лагранжа. В выпуклой задаче локальные экстремумы всегда являются глобальными. Необходимые и, что важно, одновременно достаточные условия глобального экстремума даются теоремой Куна–Таккера (или Каруша–Куна–Таккера).

- Примеры.* 1.  $x_1 - 2x_2 + 3x_3 \rightarrow \text{extr}, \quad x_1^2 + x_2^2 + x_3^2 = 1.$   
 2.  $x_1^2 - x_2^2 \rightarrow \min, \quad x_1 + x_2 \leq 1, \quad x_1, x_2 \geq 0.$

**Разные методы решения.** Разработано множество методов условной оптимизации, например, методы возможных направлений, условного и проекции градиента, штрафных и барьерных функций.

Особняком стоят задачи линейного, квадратичного и полуопределенного программирования, для которых известны относительно эффективные методы решения, использующие идеи из указанные методов (например, в SDP используется метод внутренней точки с логарифмическим барьером).

Кратко опишем некоторые из упомянутых методов. В большинстве из них используются стандартные итерации вида:  $x_{k+1} = x_k + \tau_k e_k$ . Только теперь новое приближение  $x_{k+1}$  для направления  $e_k$  может «выскочить» из области допустимых элементов  $D$ . Собственно, в зависимости от того, как корректируется точка  $x_{k+1}$  или направление  $e_k$ , чтобы  $x_{k+1} \in D$ , и получаются разные способы.

Наиболее интересен из них метод возможных направлений и его варианты (например, применяемый в Matlab алгоритм active-set). Однако его здесь сложно описать, он предлагается в качестве самостоятельного изучения.

В методе проекции градиента, используют направление  $e_k = f'(x_k)$ , и если точка  $x_{k+1} \notin D$ , то ее заменяют ближайшей к ней точкой из  $D$ . Задача определения ближайшей точки:  $|x - x_{k+1}| \rightarrow \min_{x \in D}$ , не является простой. Однако, если  $D$  относительно просто устроено, например, определяются линейными ограничениями (выпуклый многогранник) или является шаром, то ее можно решить достаточно эффективно.

Методу условного градиента применяется для задачи выпуклого программирования. Пусть  $\bar{x} = \operatorname{argmin}_{x \in D} f'(x_k)(x - x_k)$  и  $x_{k+1} = x_k + \tau_k(\bar{x} - x_k)$ , где  $\tau_k \in (0, 1)$ . В силу выпуклости  $D$  имеем  $x_{k+1} \in D$ . Шаг  $\tau_k$  берется как в методе наискорейшего спуска, только сверху обрезается единицей.

В методах штрафных и барьерных функций условная задача модифицируется специальным образом и решается как безусловная:

$$f_k(x) = f(x) + \varphi_k(x) \rightarrow \min, \quad x \in \mathbb{R}^n,$$

где вспомогательные функции  $\varphi_k$  с ростом  $k$  «заставляют» очередное приближение попасть в область  $D$  или не выходить из него. При этом их значения при  $x_k \in D$  стремятся к нулю для  $k \rightarrow \infty$ .

В методе штрафных  $\varphi_k$  выбираются таким образом, чтобы  $\varphi_k(x) \rightarrow +\infty$  при  $x \notin D$ . Например,

$$\varphi_k(x) = k \left( \sum_{i=1}^{m'} [f_i(x)]_+^2 + \sum_{i=m'+1}^m f_i^2(x) \right),$$

где  $[t]_+ = \max\{0, t\}$ .

В методе барьерных функций  $\varphi_k$  выбираются таким образом, чтобы при приближении точек  $x_k$  к границе области  $D$  значения  $\varphi_k$  росли к  $+\infty$ . Например,

$$\varphi_k(x) = \frac{1}{k} \sum_{i=1}^m |f_i(x)|^{-2}.$$

### Примеры решения задач при помощи Matlab и Maple.

Основные функции Matlab: fmincon, fminbnd, linprog, quadprog, lsqlin, lsqnonlin, lsqnonneg; Maple: Minimize/Maximize, LPSolve, LSSolve, NLPsolve, QPSolve, extrema (для аналитического решения).

Решим примеры.

```
f = @(x) (x(1)^2-x(2)^2);
A = [1 1]; b = [1];
x0 = [1; 1];
[x,fval] = fmincon(f, x0, A, b, [], [], zeros(2, 1))
```

```

Active inequalities (to within options.TolCon = 1e-006):
..lower.....upper.....ineqlin...ineqnonlin
.....1.....1
x.=
.....0
.....1
fval.=
.....-1

>.Optimization[Minimize](x[1]-2*x[2]+3*x[3],
...{x[1]^2+x[2]^2+x[3]^2=1});
..[-3.74165738677860560, . [x[1] .= -0.267261244906314,
..x[2] .= 0.534522478094199, .x[3] .= -0.801783728561298]]
>.Optimization[Maximize](x[1]-2*x[2]+3*x[3],
...{x[1]^2+x[2]^2+x[3]^2=1});
..[3.74165738677767390, . [x[1] .= 0.267261241912691,
..x[2] .= -0.534522483825382, .x[3] .= 0.801783725738073]]

>.v:=extrema(x[1]-2*x[2]+3*x[3], .{x[1]^2+x[2]^2+x[3]^2=1},
...{x[1],x[2],x[3]}, .'s'):
..latex(v);
..latex(s);

```

$$\left\{ \sqrt{14}, -\sqrt{14} \right\}$$

$$\left\{ \left\{ x_1 = -1/14 \sqrt{14}, x_2 = 1/7 \sqrt{14}, x_3 = -3/14 \sqrt{14} \right\}, \right.$$

$$\left. \left\{ x_1 = 1/14 \sqrt{14}, x_2 = -1/7 \sqrt{14}, x_3 = 3/14 \sqrt{14} \right\} \right\}$$

```

>.Optimization[Minimize](x[1]^2-x[2]^2, .{x[1]+x[2]<=1},
...assume=.nonnegative);
..[-1.00000000206520, . [x[1] .= 0., .x[2] .= 1.00000000103260]]

```

## Задачи квадратичного программирования

Задача квадратичного программирования заключается в нахождении минимума квадратичной функции при линейных ограничениях:

$$f(x) = \frac{1}{2} (Qx, x) + rx \rightarrow \min, \quad Ax \leq b, \quad x \geq 0.$$

Здесь  $Q$  — положительно определенная матрица. Тогда задача будет выпуклой и при невырожденности ограничений имеет абсолютный экстремум. Его можно найти из теоремы Куна–Таккера, которая приводит к следующей системе уравнений:

$$\sum_{i=1}^n q_{ij}x_i + \sum_{i=1}^m \lambda_i a_{ij} - \mu_j = 0, \quad \sum_{j=1}^n a_{ij}x_j + x_{n+i} = b_i,$$

$$\lambda_i x_{n+i} = 0, \quad \mu_j x_j = 0, \quad x, \lambda, \mu \geq 0,$$

где  $i = 1, \dots, m$ ,  $j = 1, \dots, n$ . Эту систему можно решить методом искусственного базиса при помощи симплекс-метода. При этом в силу условий дополняющей нежесткости нельзя включать в базисные переменные одновременно  $\lambda_i$  и  $x_{n+i}$  с одинаковым индексом  $i$ , а также  $\mu_j$  и  $x_j$  с одинаковым индексом  $j$ .

*Пример.* Найти расстояние от точки  $q = (1, 1, 2)$  до треугольника с вершинами  $p_0 = (-1, -1, -1)$ ,  $p_1 = (1, -2, 1)$ ,  $p_2 = (-2, 1, 0)$ .

Формализуем этот пример. Запишем произвольную точку внутри треугольника в барицентрических координатах:

$$p = (1 - t_1 - t_2)p_0 + t_1p_1 + t_2p_2, \quad t_1, t_2 \geq 0, \quad t_1 + t_2 \leq 1.$$

Отсюда следует, что

$$p = p_0 + t_1p_{10} + t_2p_{20}, \quad p_{i0} = p_i - p_0,$$

и квадрат расстояния от точки  $q$  до  $p$  будет равен

$$|p - q|^2 = |t_1p_{10} + t_2p_{20} + e|^2, \quad e = p_0 - q.$$

Раскрывая квадрат, получаем задачу квадратичного программирования

$$\frac{1}{2} (Qt, t) + rt + e^2 \rightarrow \min, \quad t = (t_1, t_2) \geq 0, \quad t_1 + t_2 \leq 1,$$

где  $Q = 2 \begin{pmatrix} p_{10}^2 & p_{10}p_{20} \\ p_{10}p_{20} & p_{20}^2 \end{pmatrix}$ ,  $r = 2(p_{10}e, p_{20}e)$ . Матрица ограничений  $A = (1 \ 1)$ , вектор  $b = (1)$  и не забываем про условие неотрицательности  $t \geq 0$ .

**Примеры решения задач при помощи Matlab и Maple.**  
 Функция Matlab: quadprog; Maple: QPSolve.

Решим пример.

```

q = [1, 1, 2];
p0 = [-1, -1, -1];
p1 = [1, -2, 1];
p2 = [-2, 1, 0];
p10 = p1 - p0;
p20 = p2 - p0;
e = p0 - q;
Q = [dot(p10,p10) dot(p10,p20); dot(p10,p20) dot(p20,p20)];
r = 2*[dot(p10,e) dot(p20,e)];
A = [1 1]; b = [1];
[x,fval] = quadprog(Q,r,A,b,[],[],zeros(2,1));
x
sqrt(fval+dot(e,e))
x =
    0.7368
    0.2632
ans =
    2.2005

> Q:=[[9,-2],[-2,6]]:
.. r:=[-16,-10]:
.. e2:=17:
> f:=1/2*(Q[1,1]*t1^2+2*Q[1,2]*t1*t2+Q[2,2]*t2^2)+
.. r[1]*t1+r[2]*t2+e2:
> v:=Optimization[QPSolve](f,{t1+t2<=1,t1>=0,t2>=0});
.. v := [4.8421052631579,
.. [t1 = 0.736842105263158, t2 = 0.263157894736842]]
> sqrt(v[1]);
.. 2.200478417

```

## Разные задачи.

### Дискретные экстремальные задачи

В дискретных задачах множество допустимых элементов  $D$  и функция  $f$  обладают свойством дискретности. Например, задача линейного программирования  $cx \rightarrow \max, Ax \leq b$  превращается в дискретную, если искать только целочисленные решения ( $D \subset \mathbb{Z}^n$ ).

Например, к задаче целочисленного линейного программирования сводится известная задача о рюкзаке: имеется  $n$  золотых слитков,  $j$ -й из которых имеет объем  $w_j$  и стоимость  $c_j$ . Также дан рюкзак объемом  $W$ . Спрашивается, как нужно упаковать рюкзак слитками, чтобы его стоимость была наибольшей. Если через  $x_j$  обозначить бинарную переменную, равную 0, если предмет не помещается и 1, если помещается, то получаем следующую экстремальную задачу:

$$cx \rightarrow \max, \quad wx \leq W, \quad x \in \{0, 1\}^n.$$

Задача о рюкзаке является NP-полной. Это означает, что несмотря на простоту формулировки, лучшие из алгоритмов (перебор, метод ветвей и границ, динамическое программирование), работают за экспоненциальное время.

Упомянем еще задачу об упаковке в контейнеры: имеется  $n$  предметов, каждый объемом  $w_j$ . Требуется упаковать их в минимальное число контейнеров, каждый из которых имеет вместимость  $W$ . Она также является NP-сложной. Тем не менее можно предложить следующий приближенный алгоритм (FFD): отсортировать предметы в порядке убывания объемов, для текущего предмета попытаться найти место в текущем наборе контейнеров, если это невозможно, то взять для этого новый контейнер. Оказывается это простой подход позволяет найти 0.22-приближенное решение.

#### Примеры решения задач при помощи Matlab и Maple.

Основная функция Matlab: `bintprog`; Maple: `LPSolve`.

Примеры задачи о рюкзаке и целочисленного программирования.

```
W = [15];
w = [6 4 3 2 5];
c = [5 3 1 3 6];
x = bintprog(-c, w, W);
```

```

x
c*x
x =
.....1
.....0
.....0
.....1
.....1
ans =
.....14

```

```

>.restart; with(Optimization):
>.LPSolve(2*x+5*y, {3*x-y=1, x-y<=5}, assume={nonnegative});
.. [0.666666666666667, [x = 0.333333333333333, y = 0.]]
>.LPSolve(2*x+5*y, {3*x-y=1, x-y<=5},
.. assume={nonnegative, integer});
.. [12, [x = 1, y = 2]]

```

```

W:=15:
>.w:=[6,4,3,2,5]:
>.c:=[5,3,1,3,6]:cost:=add(c[j]*x[j],j=1..5):
>.LPSolve(cost, {add(w[j]*x[j],j=1..5)<=W},
.. assume={nonnegative},maximize);
.. [22.5000000000000, [x[1] = 0., x[2] = 0., x[3] = 0.,
.. x[4] = 7.50000000000000, x[5] = 0.]]
>.LPSolve(cost, {add(w[j]*x[j],j=1..5)<=W},
.. assume={binary},maximize);
.. [14, [x[1] = 1, x[2] = 0, x[3] = 0, x[4] = 1, x[5] = 1]]

```

## Разные задачи. Задачи вариационного типа

В экстремальных задачах вариационного типа экстремум ищется среди функций  $x \in D \subset X$ , где  $X$  — нормированное пространство, например  $C^r$  или  $L_p$ .

*Задачи вариационного исчисления.* Классической задачей вариационного исчисления является задача нахождения экстремума интегрального функционала

$$f(x) = \int_{t_0}^{t_1} L(t, x, x') dt \rightarrow \text{extr}, \quad x(t_j) = x_j, \quad x \in C^1.$$

Необходимое условие экстремума заключается в выполнении на кривой  $\hat{x} \in \text{locextr}$  уравнения Эйлера

$$-\frac{d}{dt} L_{x'} + L_x = 0.$$

Например,

$$\int_0^1 t^2(x'^2 + x^2) dt \rightarrow \min, \quad x(0) = 0, \quad x(1) = \text{sh } 1.$$

Имеем

$$L_{x'} = 2t^2 x', \quad L_x = 2t^2 x$$

и уравнение Эйлера

$$-2(t^2 x')' + 2t^2 x = 0 \Leftrightarrow t^2 x'' + 2tx' - t^2 x = 0.$$

Решение этого уравнения

$$x(t) = \frac{C_1 \text{ch } t + C_2 \text{sh } t}{t}.$$

Из граничных условий находим

$$\hat{x}(1) = \frac{\text{sh } t}{t}.$$

Основывать численную реализацию на приближенном решении уравнения Эйлера достаточно сложно, поскольку имеем дело с граничной задачей, а не начальной задачей Коши (для которой есть эффективные методы, например, Рунге–Кутта). Поэтому применяют следующие прямые методы.

В методе Рунге приближенно полагают

$$x \approx x^n(t) = \sum_{i=0}^n c_i x_i, \quad c_0 = 1,$$

где  $x_0(t_j) = x_j$ ,  $x_i$  для  $i > 0$  — первые  $n$  функций полной линейно независимой полной на отрезке  $[t_0, t_1]$  системы функций,  $x_i(t_j) = 0$ . Тогда

$$I(x^n) = F(c_1, c_2, \dots, c_n) \rightarrow \min, \quad c \in \mathbb{R}^n.$$

Это конечномерная безусловная задача, которую можно решить одним из изученных методов. Тогда

$$x^*(t) = \sum_{i=0}^n c_i^* x_i(t).$$

Например, для задачи выше можно взять

$$x_0(t) = 1 + t(\operatorname{sh} 1 - 1), \quad x_i(t) = t^i(1 - t).$$

В методе Галеркина приближенное решение  $x^n$  ищется в таком же виде, как и для метода Рунге. Для него вычисляется невязка

$$F(t) = L(x^n),$$

где  $L$  — линейный оператор, отвечающий уравнению Эйлера. Далее требуется ортогональность невязки базисным функциям:

$$\int_{t_0}^{t_1} F(t) x_i(t) dt = 0, \quad i = 1, \dots, n.$$

В силу линейности уравнения Эйлера (оператора  $L$ ) имеем:

$$0 = \int_{t_0}^{t_1} F(t) x_i(t) dt = (e_0, x_i) + \sum_{i=1}^n c_i (e_i, x_i),$$

где

$$e_i = L(x_i), \quad i = 1, \dots, n.$$

Получили систему линейных уравнений, из которых находим  $c^*$  и затем по ним строим  $x^*$ .

*Задачи теории приближений.* Среди всего многообразия задач теории приближений рассмотрим задачи о наилучшем приближении в  $C$  и  $L_2$ :

$$\|f - p\|_C \rightarrow \min_{\deg p \leq n}, \quad \|f - p\|_2 \rightarrow \min_{\deg p \leq n},$$

где  $p$  — полиномы (например, алгебраические или тригонометрические).

*Примеры.* 1. Найти  $p^*(x)$  и погрешность приближения  $S_{\min}^*$  в задаче

$$\min_{\deg p \leq 5} \|\sin x - p(x)\|_{C[0, \pi/2]}.$$

2. Решить пример 1 для нормы  $L_2[-1, 1]$ .

Приведем алгоритм Ремеза решения задачи

$$\|f(x) - p(x)\|_{C[a, b]} \rightarrow \min,$$

где

$$p(t) = \sum_{j=0}^n p_j \varphi_j(t),$$

где  $\varphi_j$  — базисные функции. Он основан на теоремах об альтернансе и Валле Пуссена.

Пусть

$$a \leq x_0 < x_1 < \dots < x_{n+1} \leq b$$

— начальное приближение альтернанса (выбираемое, например, случайно или как нули многочленов Чебышева). Решаем относительно  $p_j$  и  $d$  систему линейных уравнений

$$p(x_j) + (-1)^i d = f(x_i), \quad i = 0, 1, \dots, n + 1.$$

Находим

$$x^* = \operatorname{argmax}_{x \in [a, b]} |f(x) - p(x)|, \quad D = S_{\max}^*.$$

Заменяем одну из точек  $x_i$  точкой  $x^*$  так, чтобы знак  $f(x) - p(x)$  в точках новой последовательности чередовался. Критерий окончания:  $|d| \approx D$ .

В случае  $L_2$  имеем

$$p = \sum_{j=0}^n c_j \varphi_j$$

и

$$\|f - p\|_2^2 = \|f\|_2^2 + \|p\|_2^2 - 2(f, p) = \sum_{i, j=0}^n c_i c_j (\varphi_i, \varphi_j) - 2 \sum_{j=0}^n c_j (f, \varphi_j) + \|f\|_2^2.$$

Отсюда получаем следующую квадратичную задачу для определения оптимальных коэффициентов  $c_j$ :

$$(Qc, c) - 2rc \rightarrow \min.$$

Здесь матрица  $Q$ , составленная из скалярных произведений  $(\varphi_i, \varphi_j)$  положительно определена, если система базисных функций  $\varphi_j$  линейно независима.

Если дополнительных ограничений на коэффициенты  $c_j$  нет, то они находятся из линейной системы уравнений  $Qc = r$  (в данном

случае необходимого и достаточного условия глобального экстремума). Оно наиболее просто решается в случае ортонормированного базиса  $\varphi_j$ , когда  $Q$  будет единичной матрицей. Сложнее решаются случаи, когда на  $c_j$  ставятся дополнительные ограничения, например, условие неотрицательности  $c_j \geq 0$ . Тогда получаем задачу квадратичного программирования.

**Примеры решения задач при помощи Matlab и Maple.**  
 Основные функции Matlab: fminimax, lsqlin, lsqnonlin, lsqnonneg;  
 Maple: LPSolve, LSSolve, пакет numapprox: minimax, remez.

```
L:=t^2*(x1^2+x^2): Lx1:=diff(L,x1): Lx:=diff(L,x):
> eq:=-diff(subs({x1=diff(x(t),t),x=x(t)},Lx1),t)+
..subs({x1=diff(x(t),t),x=x(t)},Lx);
..dsolve({eq,x(0)=1});
```

$$-4t \frac{d}{dt}x(t) - 2t^2 \frac{d^2}{dt^2}x(t) + 2t^2x(t)$$

$$x(t) = \begin{cases} 1 & t = 0 \\ \frac{\sinh(t)}{t} & \text{otherwise} \end{cases}$$

```
n:=3:
> x:=1+t*(sinh(1)-1)+add(c[i]*t^i*(1-t),i=1..n):
> x:=unapply(x,t):
> x(0): x(1);
.. 1
.. sinh(1)
> F:=simplify(evalf(int(t^2*(diff(x(t),t)^2+x(t)^2),t=0..1)));
.. F:=.5327968159e-1*c[1]+.3996934187e-1*c[2]+
.. .3051607398e-1*c[3]+.6551226620e-1*c[3]^2+
.. .1428571429*c[1]^2+.2119047616*c[1]*c[2]+
.. .8968254030e-1*c[2]^2+.1603174611*c[1]*c[3]+
.. .1484126972*c[2]*c[3]+.4373048414
> v:=Optimization[Minimize](F);
.. v:= [0.432332358616673, [c[1]]=-0.175327828640891,
.. c[2]=-0.00790819512851335,
.. c[3]=-0.00942025924841215]]
> assign(v[2]);
> Optimization[Maximize](abs(x(t)-sinh(t)/t),t=0..1);
.. [.675349482581744e-5, [t]=.126431670888261]]

> eq;
.. -4*t*(sinh(1)-1+c[1]*(1-t)-c[1]*t+2*c[2]*t*(1-t)-
.. c[2]*t^2+3*c[3]*t^2*(1-t)-c[3]*t^3)
.. -2*t^2*(-2*c[1]+2*c[2]*(1-t)-4*c[2]*t+6*c[3]*t*(1-t)-
.. 6*c[3]*t^2)+2*t^2*(1+t*(sinh(1)-1)+
.. c[1]*t*(1-t)+c[2]*t^2*(1-t)+c[3]*t^3*(1-t))
> sys:=evalf({seq(int(eq*t^i*(1-t),t=0..1),i=1..3)});
> v:=solve(sys);
```

```

..{c[1] = -.1753278897, c[2] = -.7908013149e-2,
..c[3] = -.9420391124e-2}

>.numapprox[minimax](x -> sin(x), 0..Pi/2, [5, 0], x -> 1,
.. 'maxerror');
..x -> .706460e-5 + (.9996899312 + (.21935459e-2 +
.. (-.1722387625 + (.609741353e-2 + .5721696436e-2*x)*x)*x)*x
>.maxerror;
.. .7072293573e-5

```

## Литература

- [1] Сухарев, А.Г. Курс методов оптимизации : учеб. пособие / А.Г.Сухарев, А.В. Тимохов, В.В. Федоров. — 2-е изд. — М. : Физмат, 2005. — 368с. : ил. — (Классический университетский учебник). — Библиогр. в конце кн. — ISBN 5-9221-0559-0 /в пер./ : 259.60.
- [2] Пантелеев, А.В. Методы оптимизации в примерах и задачах : учебное пособие для вузов / А.В. Пантелеев, Т.А. Летова. — 2-е изд., испр. — М. : Высш. шк., 2005. — 544с. : ил. — (Прикладная математика для ВТУЗов). — Библиогр. в конце кн. — ISBN 5-06-004137-9 /в пер./ : 235.98.
- [3] Измаилов, А.Ф. Численные методы оптимизации : учеб. пособие для вузов / А.Ф. Измаилов, М.В. Солодов. — М. : Физматлит, 2005. — 304с. — Библиогр. в конце кн. — ISBN 5-9221-0045-9 /в пер./ : 145.20.
- [4] Алексеев, В.М. Оптимальное управление : учебник для вузов / В.М. Алексеев, В.М. Тихомиров, С. В. Фомин. — 3-е изд., испр. и доп. — М. : ФИЗМАТЛИТ, 2007. — 408 с. : ил. — (Классический университетский учебник). — Библиогр. в конце кн. — ISBN 978-5-9221-0589-7 (в пер.) : 452.76.
- [5] Алексеев, В.М. Сборник задач по оптимизации. Теория. Примеры. Задачи : задачник для вузов / В. М. Алексеев, Э. М. Галеев, В. М. Тихомиров ; МГУ им. М. В. Ломоносова. — 3-е изд., испр. — М. : ФИЗМАТЛИТ, 2008. — 256 с. — (Классический университетский учебник). — Библиогр. в конце кн. — ISBN 978-5-9221-0992-5 (в пер.) : 311.85.
- [6] Струченков, В.И. Методы оптимизации. Основы теории, задачи, обучающие программы : учеб.пособие / В.И.Струченков. — М. : Экзамен, 2005. — 256с. : ил. — Библиогр.в конце кн. — ISBN 5-472-00465-9 : 72.20.
- [7] Васильев, Ф.П. Численные методы решения экстремальных задач : учеб. пособие для вузов / Ф. П. Васильев. — 2-е изд., перераб. и доп. — М. : Наука, 1988. — 549 с. : ил. — Библиогр. в конце кн. — ISBN /В пер./ : 1.60.
- [8] Дьяконов, В.П. MAPLE 9.5/10 в математике, физике и образовании / В.П.Дьяконов. — М. : СОЛОН-Пресс, 2006. — 720с. : ил. + 1опт.диск (CD ROM). — (Библиотека профессионала). — Библиогр. в конце кн. — ISBN 5-98003-258-4 : 448.35.
- [9] Алексеев, Е.Р. Решение задач вычислительной математики в пакетах Mathcad 12, MATLAB 7, Maple 9 / Алексеев Е.Р., Чеснокова О.В. — М. : NT Press, 2006. — 496с. : ил. — (Самоучитель). — Библиогр. в конце кн. — ISBN 5-477-00208-5 : 135.15.
- [10] Журнал вычислительной математики и математической физики. — М.: Наука.
- [11] Прикладная математика и механика : журнал. — М.: Наука.
- [12] Библиотека численного анализа НИВЦ МГУ: [http://num-anal.srcc.msu.ru/lib\\_na/libnal.htm](http://num-anal.srcc.msu.ru/lib_na/libnal.htm)