Ю. В. Триус
Черкаський національний університет
імені Богдана Хмельницького

Розв'язування екстремальних задач за допомогою пакету Matlab 6.5

Однією з важливих сфер використання систем комп'ютерної математики (СКМ) таких, як Mathcad, Matlab, Maple, Mathematica, у наукових дослідженнях і вищій школі є, на думку автора, розв'язування і дослідження екстремальних задач, що виникають у різних галузях людської діяльності від самої математики, до комп'ютерної техніки, економіки, управління тощо. Розуміння важливої ролі задач даного класу для розвитку сучасного постіндустріального суспільства обумовило те, що навчальні плани багатьох спеціальностей математичного, комп'ютерного, технічного та економічного напрямів підготовки містять дисципліни, основним предметом вивчення яких є оптимізаційні задачі і моделі. Це, зокрема, такі дисципліни, як „Методи оптимізації", „Математичне програмування" „Дослідження операцій", „Оптимізаційні задачі в економіці".

В роботі [7] нами було проаналізовано стан, проблеми і перспективи використання СКМ та систем обробки даних у ВНЗ 3-4 рівнів акредитації при викладанні математичних дисциплін, зроблено огляд можливостей найбільш популярних і потужних СКМ та систем обробки даних щодо розв'язування задач оптимізації, проведено їх порівняльний аналіз.

У дані статті буде розглянуто один з найкращих, за даними дослідження С. Стейнхауса [10], універсальний математичний пакет Matlab, проаналізовано його основні засоби, призначені для знаходження екстремумів функцій від однієї та багатьох змінних, а також для розв'язування задач лінійного і нелінійного програмування, а також наведено приклади їх використання для розв'язування конкретних задач. Це обумовлено тим, що:

– по-перше, не дивлячись на те, що МОН України визначило пакет Matlab як базовий для ВНЗ України, використання цього пакету при вивченні математичних дисциплін, за даними проведеного нами анкетування викладачів, є невиправдано низьким (7% з 70 респондентів);

– по-друге, пакет Matlab, за результатами нашого дослідження, має найбільш потужні і гнучки засоби розв'язування екстремальних задач, які дозволяють не лише одержати наближені розв'язки досить широкого класу задач оптимізації, але й завдяки прямому доступу до широкого набору параметрів, проводити всебічний аналіз ітераційного процес та методів, які при цьому використовуються;

– по-третє, як показав проведений нами аналіз, у навчально-методичній літературі, присвяченій СКМ, недостатньо уваги приділяється питанням застосування пакету Matlab для розв'язування екстремальних задач (див., наприклад, [2], [3], [8]).

1. Огляд можливостей СКМ щодо розв'язування екстремальних задач.

Коротко проаналізуємо основні можливості СКМ Mathcad, Matlab, Maple, Mathematica щодо розв'язування задач оптимізації, які забезпечуються досить потужними засобами, що або вбудовані у ядро цих систем, або входять до їх складу у вигляді додаткових модулів (пакетів розширення) і реалізують найбільш популярні методи оптимізації, зокрема, метод золотого перерізу і парабол для одновимірної оптимізації; симплексний метод Нелдера-Мілда, метод спряжених градієнтів, квазіньютонівські методи для задач багатовимірної нелінійної оптимізації; метод внутрішньої точки (метод Кармаркара) для розв'язування задач лінійного програмування великої розмірності тощо. Зазначимо, що в цих системах, як правило, при розв'язуванні конкретних задач реалізуються кілька методів оптимізації, які застосовуються в залежності від розмірності задачі, властивостей цільової функції чи особливостей наявних в задачі обмежень.

В таблиці 1 наведено дані про можливості СКМ Matlab, Maple, Mathematica, Mathcad щодо розв'язування деяких класів задач оптимізації, а в таблиці 2 – перелік вбудованих функцій і пакетів розширення деяких версій зазначених СКМ, які можна використовувати для розв'язування задач безумовної оптимізації, а також для розв'язування задач лінійного і нелінійного програмування [7].

Таблиця 1

Програма (версія) / **Задача оптимізації**	**Maple 7.0**	**Mathematica 4.1**	**Matlab 6.5**	**Mathcad 2001 Professional**
Лінійна (б/у)	+/+	+/+	м/м	+/+
Нелінійна (б/у)	+/+	+/-	м/м	+/+
Квадратична (б/у)	+/+	+/-	м/м	+/+

Примітка. В таблиці 1 використані наступні позначення:
„б" – задача безумовної оптимізації;
„у" – задача умовної оптимізації;
„м" – розв'язування задачі забезпечується додатковим модулем;
„+" – функція, яка забезпечує розв'язування задачі, вбудована в ядро програми;
„-" – розв'язування задачі не підтримується програмою.

Таблиця 2

Задача оптимізації / **Програма (версія)**	Задача одновимірної оптимізації	Задача безумовної оптимізації функції багатьох змінних	Задача лінійного програмування	Задача квадратичного програмування	Задача нелінійної умовної оптимізації
Matlab 6.5	Fminbnd (пакет Optimization Toolbox)	Fminsearch, Fminunc (пакет Optimization Toolbox)	Linprog (пакет Optimization Toolbox)	Quadprog (пакет Optimization Toolbox)	Fmincon (пакет Optimization Toolbox)
Maple 7.0	Minimize, Maximize, Extrema	Minimize, Maximize, Extrema	Extrema, Пакет розширення Simplex	Extrema	Extrema
Mathematica 4.1	FindMinimum	FindMinimum	ConstrainedMax, ConstrainedMin, LinearProgramming		
Mathcad 2001 Professional	Minimize, Maximize	Minimize, Maximize	Minimize, Maximize	Minimize, Maximize	Minimize, Maximize

2. Загальна характеристика пакету Matlab.

Пакет Matlab являє собою апробовану і надійну СКМ, яка призначена для розв'язування широкого кола математичних задач з поданням даних в універсальній матричній формі, яка запропонована фірмою Math Works Inc. (www.mathworks.com).

Matlab є універсальною інтегрованою СКМ, яка орієнтована на персональні комп'ютери класу IBM PC і Macintosh, робочі станції UNIX, і яка має потужні засоби діалогу, графіки і комплексної візуалізації, власну мову програмування, широкий спектр застосувань, включаючи опрацювання сигналів і зображень, проектування систем управління, природничі науки, фінанси та економіку, а також приладобудування. Відкрита архітектура дозволяє використовувати Matlab у поєднанні з іншими програмними продуктами для створення інструментів дослідження і розв'язування різноманітних задач.

Популярності системи Matlab сприяє пакет Simulink, за допомогою якого можна здійснювати імітаційне моделювання лінійних і нелінійних динамічних систем, а також багато інших пакетів розширення (Toolbox), які підсилюють математичні можливості системи, підвищують швидкість, ефективність і точність обчислень. До таких пакетів відноситься і Optimization Toolbox – пакет для розв'язування задач оптимізації, який реалізує основні методи одновимірної і багатовимірної оптимізації.

3. Використання пакету Matlab 6.5 для розв'язування задач оптимізації.

Розглянемо основні можливості використання пакету Matlab 6.5 для розв'язування таких задач оптимізації:

– одновимірної мінімізації;
– безумовної мінімізації нелінійних функцій;
– лінійного програмування;
– квадратичного програмування;
– умовної мінімізації нелінійних функцій.

Зауважимо, що всі функції, які використовуються в цьому пакеті, призначені для розв'язування задач мінімізації. Тому якщо необхідно розв'язати задачу максимізації, то досить перед цільовою функцією поставити знак "мінус" і скористатися однією з наявних функцій мінімізації.

3.1. Задача одновимірної мінімізації. Нехай задано задачу

$$f(x) \to \min,\ x \in [a; b]. \qquad (1)$$

Для знаходження точки локального мінімуму цільової функції $f(x)$ на заданому інтервалі $[a; b]$ і її значення у цій точці в пакеті Matlab 6.5 використовується функція fminbnd, яка має такий синтаксис:

```
x = fminbnd(fun,x1,x2);
x = fminbnd(fun,x1,x2,options);
x = fminbnd(fun,x1,x2,options,p1,p2,...);
[x,fval] = fminbnd(...);
[x,fval,exitflag] = fminbnd(...);
[x,fval,exitflag,output] = fminbnd(...),
```

при цьому при використанні:

- x = fminbnd(fun,x1,x2) – одержується точка x, яка є певним наближенням точки локального мінімуму функції, яка описана у змінній fun, на заданому інтервалі x1,x2;

- x = fminbnd(fun,x1,x2,options) – схожа на описану вище форму функції, але при цьому використовуються параметри з набору вектора параметрів options, який містить 34 параметри і які попередньо встановлюються за допомогою команди optimset. До таких параметрів відносяться, наприклад: tolX – величина ітераційної похибки, MaxFunEval – максимальна кількість обчислень значень функції, MaxIter – максимальна кількість ітерацій, Display – виведення додаткових відомостей про знаходження точки екстремуму за допомогою певних параметрів, зокрема, якщо встановлено параметр 'iter', то виводяться відомості про кожну ітерацію. Якщо треба використати параметри обчислень за замовчуванням, то замість options необхідно ввести [] (порожній масив);
- x = fminbnd(fun,xl,x2,options,p1,p2,...) – схожа з описаною вище формою функції, але до цільової функції передаються додаткові аргументи: p1, p2, ... ;
- [x,fval] = fminbnd(...) – додатково повертається значення цільової функції fval в точці мінімуму x;
- [x,fval,exitflag] = fminbnd(...) – додатково виводиться параметр exitflag, рівний 1, якщо задача розв'язана із заданою точністю tolX, і 0, якщо зроблено максимальну кількість ітерацій, яка задана за допомогою параметра MaxIter;
- [x,fval,exitflag,output] = fminbnd(...) – додатково виводиться структура output, що містить такі відомості, як кількість зроблених ітерацій (Iterations), кількість обчислень значень функції (funcCount), назва алгоритму, який використовувався при розв'язуванні (algorithm).

У наведених різних варіантах функції fminbnd використані такі позначення:
fun – рядок, який містить назву функції, що мінімізується;
x1, x2 – початок і кінець інтервалу, на якому шукається мінімум функції;
options – вектор параметрів обчислень;
p1, p2, ... – додаткові аргументи, які передаються до цільової функції.

Приклад 1. За допомогою пакету Matlab 6.5 знайти наближений розв'язок задачі одновимірної мінімізації функції $f(x) = 3x^4 + 5x^3 - 10x^2 + 6x$ на відрізку $[a; b] = [-4; 2]$.

Дії, які необхідно виконати для розв'язування поставленої задачі за допомогою функції fminbnd, подано на рис. 1.

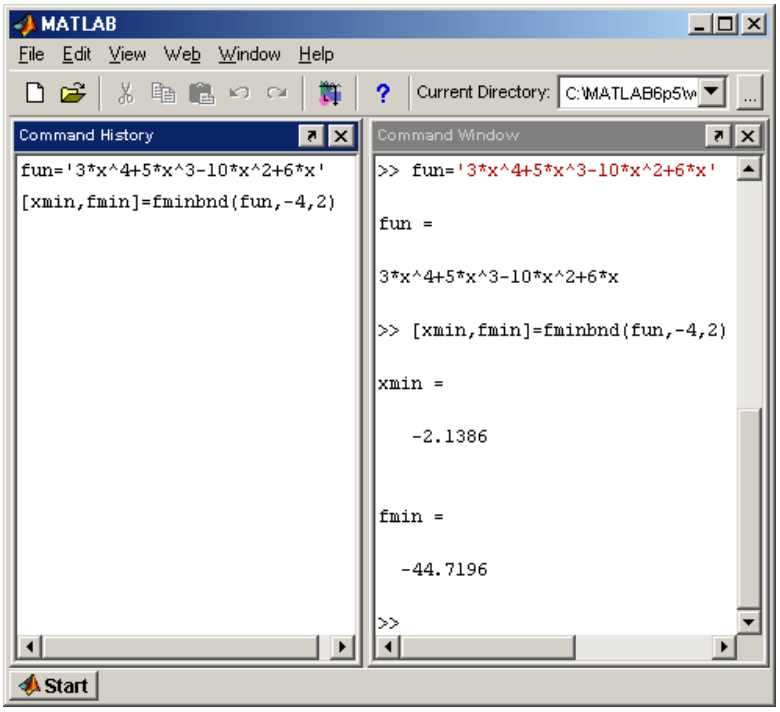

Рис. 1

В залежності від заданих параметрів функції fminbnd і вхідних даних, знаходження точки мінімуму відбувається за відомими методами одновимірної мінімізації: золотого перерізу і парабол (див., наприклад, [1]).

Наприклад, в результаті виконання у вікні Command window команди

```
>> [xmin,fmin,exitflag,output]=fminbnd(fun,-4,2,optimset('tolX',1.e-8,'display','iter'))
```

буде одержано

Func-count	x	f(x)	Procedure
1	-1.7082 -	38.8077	initial
2	-0.291796	-2.7047	golden
3	-2.58359	-34.813	golden
4	-1.9719 -	43.6941	parabolic
5	-2.08546	-44.6089	parabolic
6	-2.17372	-44.6691	parabolic
7	-2.13894	-44.7196	parabolic
8	-2.13815	-44.7196	parabolic

```
9          -2.13863      -44.7196      parabolic
10         -2.13863      -44.7196      parabolic
11         -2.13863      -44.7196      parabolic
12         -2.13863      -44.7196      parabolic
Optimization terminated successfully:
the current x satisfies the termination criteria using OPTIONS.TolX of 1.000000e-008
xmin =  -2.1386
fmin =   -44.7196
exitflag =     1
output =
        iterations: 12
        funcCount: 14
        algorithm: 'golden section search, parabolic interpolation'.
```

Одержаний результат досить детально характеризує ітераційних процес, при цьому цілком виправдано на початку цього процесу пакетом використано метод золотого перерізу для локалізації точки мінімуму, а потім для її уточнення використовується метод парабол, який ефективно працює в околі оптимальної точки.

На рис. 2 подано графічну інтерпретацію розв'язку поставленої задачі, одержану в результаті виконання наступних команд:

```
>> x= - 4:0.1:2;
>> y=3*x.^4+5*x.^3-10*x.^2+6*x
>>plot(x,y)
>> grid on.
```

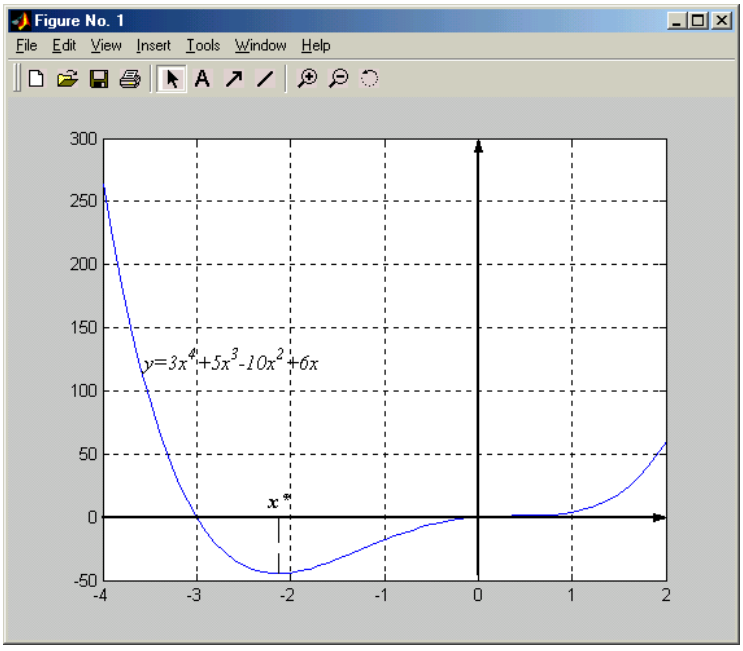

Рис. 2

Наведений приклад демонструє широкі можливості для дослідницької роботи студентів на лабораторних заняттях з використанням пакету Matlab 6.5.

3.2. Задача безумовної багатовимірної мінімізації. Для розв'язування задачі мінімізації нелінійної функції багатьох змінних на її області визначення:

$$f(x) \to \min,\ x \in D(f), \qquad (2)$$

де $x = (x_1, x_2, \ldots, x_n) \in R^n$, в пакеті Matlab 6.5 використовуються такі функції:

- *fminsearch*, яка реалізує симплексний метод Нелдера-Міда, який було запропоновано американськими вченими Спендлі, Текстом, Хімсвортом, розвинуто у роботах Нелдера і Міда (див., наприклад, [9]) і який відноситься до методів нульового порядку;
- *fminunc*, яка реалізує квазіньютонівські методи типу Давідона-Флетчера-Пауелла для мінімізації нелінійної диференційовної функції (див., наприклад, [4]) і входить до пакету Optimization Toolbox.

Функція *fminsearch* має такий синтаксис:

```
x = fminsearch(fun,x0);
x = fminsearch(fun,x0,options);
x = fminsearch(fun,x0,options,p1,p2,...);
[x,fval] = fminsearch(...);
[x,fval,exitflag] = fminsearch(...);
[x,fval,exitflag,output] = fminsearch(...),
```

при цьому при виконанні:

- x = fminsearch(fun,x0) – одержується вектор x, який є певним наближенням точки локального мінімуму функції fun в околі точки x0. Початкове наближення x0 може бути числом або відрізком при мінімізації функції однієї змінної, або вектором для функції багатьох змінних;
- x = fminsearch(fun,x0,options) – схожа на описану вище форму функції, але в ній використовується вектор параметрів options аналогічно тому, як і в функції fminbnd;
- x = fminsearch(fun,x0,options,p1,p2,...) – аналогічна до описаної вище функції, але до функції кількох змінних fun(x,p1,p2,...), що мінімізується, передаються додаткові аргументи p1, p2, ... Якщо треба використати параметри обчислень за замовчуванням, то замість options перед p1, p2 необхідно записати [];
- [x,fval] = fminsearch(...) – додатково виводиться значення цільової функції fval у знайденій точці x;
- [x,fval,exitflag] = fminsearch(...) – додатково виводиться параметр exitflag, який має додатне значення, якщо ітераційний процес завершився у відповідності до заданої точності обчислень, яка визначається за параметром tolX, від'ємне значення, якщо ітераційний процес не збігається до розв'язку, і 0, якщо було перевищено максимальну кількість ітерацій, яка визначається за параметром MaxIter;
- [x,fval,exitflag,output]= fminsearch(...) – додатково виводиться структура output, яка має такі поля:

  iterations – кількість виконаних ітерацій;
  funcCount – кількість обчислень значень цільової функції;
  algorithm – назва алгоритму, який було використано для розв'язування.

Функція *fminunc* має такий синтаксис:

```
x = fminunc(fun,x0);
x = fminunc(fun,x0,options);
x = fminunc(fun,x0,options,p1,p2,...);
[x,fval] = fminunc(...);
[x,fval,exitflag] = fminunc(...);
[x,fval,exitflag,output] = fminunc(...);
[x,fval,exitflag,output,grad] = fminunc(...);
[x,fval,exitflag,output,grad,hessian] = fminunc(...),
```

де x0 – початкове наближення, x – точка локального мінімуму, fval – значення цільової функції в цій точці, fun, options, exitflag, output, p1, p2 мають аналогічні значення, як і для функції fminsearch, grad – вектор градієнта цільової функції, яка описується в fun, hessian – гессіан, або матриця частинних похідних другого порядку цільової функції, значення яких можна вивести на екран.

Так, наприклад, в результаті виконання команди

```
[x,fval,exitflag,output,grad,hessian] = fminunc(fun,x0)
```

параметр grad одержить значення градієнта цільової функції в точці x, яка є результатом розв'язування поставленої задачі, а параметр hessian – гессіан функції в точці x. Але для цього треба спочатку виконати команди, за допомогою яких будуть активізовані відповідні параметри вектора options:

```
>>options = optimset('GradObj','on')
>>options = optimset('Hessian','on').
```

Приклад 2. За допомогою функцій *fminsearch* і *fminunc* пакету Matlab 6.5 знайти точку мінімуму функції Хіммельблау (Himmelblau)

$$f(x_1, x_2) = 100(x_2 - x_1^3)^2 + (1 - x_1)^2$$

при початковому наближенні $x^{(0)} = (-1; 2)$.

Відомо, що ця функція неопукла і одноекстремальна, дно яру якої задається кубічною параболою (рис. 3), при цьому $x^* = (1; 1)$ і $f(x^*) = 0$.

Для розв'язування задачі спочатку треба описати цільову функцію, наприклад, за допомогою команди:

```
>>fun='100*(x(2)-x(1)^3)^2+(1-x(l))^2'
```

і потім встановити необхідну точність обчислень, наприклад за допомогою команди:

```
>>options=optimset( 'tolX',l.e-6).
```

Для знаходження точки мінімуму за допомогою функції fminsearch треба набрати і виконати команду:

```
>> [xmin,fmin,exitflag,output]=fminsearch(fun,[-1 2]).
```

Після цього буде одержано результат:

```
xmin =    1.0000    1.0000
fmin =  4.7395e-010
exitflag =     1
output =
        iterations: 93
        funcCount: 169
        algorithm: 'Nelder-Mead simplex direct search'
```

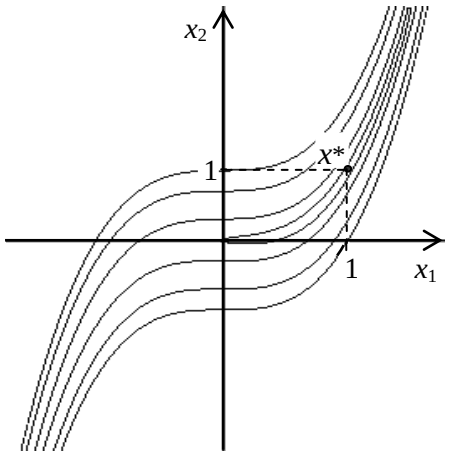

Рис. 3

Для розв'язування поставленої задачі за допомогою функції *fminunc* можна виконати, наприклад, команду:

>> [xmin,fmin,exitflag,output,grad,hessian]=fminunc(fun,[-1.9 2]).

Після цього буде одержано:

```
Warning: Gradient must be provided for trust-region method;
        using line-search method instead.
> In C:\MATLAB6P5\toolbox\optim\fminunc.m at line 211
        Optimization terminated successfully:
        Current search direction is a descent direction, and magnitude of
         directional derivative in search direction less than 2*options.TolFun
xmin =     0.9999    0.9998
fmin =   3.6765e-009
exitflag =      1
output =
        iterations: 23
        funcCount: 143
        stepsize: 1.0000
        firstorderopt: 0.0019
        algorithm: [1x38 char]
grad =     0.0019    -0.0009
hessian =      1.8022   -0.6005
              -0.6005    0.2003
```

Як видно з одержаних результатів, за допомогою функції *fminunc*, яка реалізує квазіньютонівський метод, наближений розв'язок знайдено за меншу кількість ітерацій, але при цьому вона 23 рази обраховує вектор градієнта і матрицю других похідних (гессіан), тоді як метод Нелдера-Міда, який реалізується функцією *fminsearch*, використовує лише відомості про значення цільової функції, які обчислюються 169 разів, що не на багато більше у порівнянні з 143 обчисленнями за функцією *fminunc.*

3.3. Задачі умовної багатовимірної мінімізації. Розглянемо функції, які входять до пакету Optimization Toolbox системи Matlab 6.5 і за допомогою яких можна розв'язувати задачі умовної оптимізації функцій багатьох змінних, зокрема задачі лінійного, квадратичного і нелінійного програмування: *linprog, quadprog, fmincon*.

Функція *linprog* призначена для розв'язування задачі лінійного програмування виду

$$f(x) = \sum_{j=1}^{n} c_j x_j = \langle c, x \rangle \to \min, \tag{3}$$

$$A \cdot x \le b, \tag{4}$$

$$Aeq \cdot x = beq, \tag{5}$$

$$lb \le x \le ub, \tag{6}$$

де c, x, b, beq, lb, ub – вектор-стовпчики, A, Aeq – прямокутні матриці, і яка має такий синтаксис:

```
x = linprog(c,A,b,Aeq,beq);
x = linprog(c,A,b,Aeq,beq,lb,ub);
x = linprog(c,A,b,Aeq,beq,lb,ub,x0);
x = linprog(c,A,b,Aeq,beq,lb,ub,x0,options);
[x,fval] = linprog(...);
[x,fval,exitflag] = linprog(...);
[x,fval,exitflag,output] = linprog(...);
[x,fval,exitflag,output,lambda] = linprog(...).
```

Розглянемо особливості параметрів exitflag, lambda, output, які використовуються в функції linprog. Так параметр exitflag приймає додатне значення, якщо ітераційний процес завершився у відповідності до заданої точності обчислень, від'ємне значення, якщо ітераційний процес не збігається до розв'язку, і 0, якщо було перевищено максимальну кількість ітерацій, яка визначається за параметром MaxIten, або максимальну кількість обчислень значень цільової функції, яка визначається за параметром MaxFunEvals.

Параметр lambda являє собою структуру з полями, які містять множники Лагранжа для кожної групи обмежень задачі в точці x, що є результатом розв'язування поставленої задачі:

- ineqlin – для обмежень-нерівностей,
- eqlin – для обмежень-рівнянь,
- upper – для прямих обмежень типу $x \le ub$,
- lower – для прямих обмежень типу $lb \le x$,

при цьому ненульові елементи векторів у полях параметра lambda відповідають активним обмеженням для знайденої точки x.

Зауважимо, що якщо в умові задачі деякі вхідні дані відсутні, то замість відповідних величин треба ставити []. Наприклад, якщо в умові задачі (3)-(6) відсутні обмеження-нерівності виду (4), то треба покласти A=[] і b=[].

Приклад 3. За допомогою функції *linprog* пакету Matlab 6.5 розв'язати задачу:

У хімічній лабораторії заводу потрібно виплавити новий сплав, що містить 30% свинцю, 25% цинку і 45% олова. Для цього використовують чотири сплави, вхідні дані про які наведено в таблиці 3.

Таблиця 3

Елементи	Вміст елементів у сплавах (%)			
	Сплав 1	Сплав 2	Сплав 3	Сплав 4
Свинець	24	15	36	42
Цинк	26	45	18	38
Олово	50	40	46	20
Ціна 1 кг сплаву (грн)	60	80	40	50

Визначити, яку кількість кожного сплаву потрібно витратити на кожний кілограм нового сплаву, щоб він був найдешевшим?

Математична модель задачі має такий вигляд:

$$f(x) = 60x_1 + 80x_2 + 40x_3 + 50x_4 \to \min,$$

$$\begin{cases} 0.24x_1 + 0.15x_2 + 0.36x_3 + 0.42x_4 = 0.3, \\ 0.26x_1 + 0.45x_2 + 0.18x_3 + 0.38x_4 = 0.25, \\ 0.5x_1 + 0.4x_2 + 0.46x_3 + 0.2x_4 = 0.45, \end{cases}$$

$$x_j \ge 0,\ j = \overline{1,4},$$

де x_j – кількість *j*-го сплаву, яку треба витратити на кожний кілограм нового сплаву $j = \overline{1,4}$.

Вхідні дані задачі подамо у матричному вигляді. Для цього у вікні Command window пакету Matlab треба виконати послідовно такі команди:

```
>> c=[60;80;40;50]
>> Aeq=[0.24 0.15 0.36 0.42; 0.26 0.45 0.18 0.38; 0.5 0.4 0.46 0.2]
>> beq=[0.3;0.25;0.45]
>> lb=zeros(4,1),
```

де функція zeros – створює масив з нульовими елементами.

Після цього треба набрати і виконати команду

```
>> [x,fval,exitflag,output,lambda] = linprog(c,[],[],Aeq,beq,lb),
```

при цьому буде одержано такий результат:

```
Optimization terminated successfully.
x =       0.0962  0.2308  0.6731  0.0000
fval =  51.1538
exitflag = 1
output =
        iterations: 5
        cgiterations: 0
        algorithm: 'lipsol'
lambda =
        ineqlin: [0x1 double]
        eqlin: [3x1 double]
        upper: [4x1 double]
        lower: [4x1 double]
>> lambda.upper
ans =     0     0     0     0
>>lambda.lower
ans =    0.0000    0.0000    0.0000   10.7692
>>lambda.eqlin
ans =   57.1795  -122.3077   -83.8462
>>lambda.ineqlin
ans =   Empty matrix: 0-by-1.
```

Таким чином, оптимальний план використання наявних сплавів при виготовленні одного кілограму нового сплаву наближено дорівнює

$X^* \approx (0.0962; 0.2308; 0.6731; 0.0000)$, або у відсотках $X^* \approx (9.62\%; 23.08\%; 67.31\%; 0\%)$,

при витратах $f(X^*) \approx 51.1538$ грн.

При розв'язуванні задачі використано метод внутрішньої точки LIPSOL – Linear Interior Point Solver (див., наприклад, Zhang Y., "Solving Large-Scale Linear Programs by Interior-Point Methods Under the MATLAB Environment," Technical Report TR96-01, Department of Mathematics and Statistics, University of Maryland, Baltimore County, Baltimore, MD, July 1995.), який застосовується, як правило, для розв'язування задач великої розмірності (large-scale).

Функція *quadprog* призначена для розв'язування задачі квадратичного програмування виду

$$f(x) = \frac{1}{2}\langle Hx, x\rangle + \langle c, x\rangle \to \min, \quad (7)$$

$$A \cdot x \le b, \quad (8)$$

$$Aeq \cdot x = beq, \quad (9)$$

$$lb \le x \le ub, \quad (10)$$

де H – симетрична невід'ємно визначена квадратна матриця, A, Aeq – прямокутні матриці, c, b, beq, lb, ub, x – вектор-стовпчики, і має такий синтаксис:

```
x = quadprog(H,c,A,b);
x = quadprog(H,c,A,b,Aeq,beq);
x = quadprog(H,c,A,b,Aeq,beq,lb,ub);
x = quadprog(H,c,A,b,Aeq,beq,lb,ub,x0);
x = quadprog(H,c,A,b,Aeq,beq,lb,ub,x0,options);
x = quadprog(H,c,A,b,Aeq,beq,lb,ub,x0,options,p1,p2,...);
[x,fval] = quadprog(...);
[x,fval,exitflag] = quadprog(...);
[x,fval,exitflag,output] = quadprog(...);
[x,fval,exitflag,output,lambda] = quadprog(...).
```

Приклад 4. За допомогою функції *quadprog* пакету Matlab 6.5 розв'язати задачу квадратичного програмування:

$$f(x) = x_1^2 + 4x_1x_2 + 6x_2^2 + 5x_1 + 8x_2 \to \min, \quad x \in X,$$

де $X = \{x \in R^2 \mid x_1 \ge 0,\ x_2 \ge 0,\ x_1 + x_2 \ge 3\}$, при початковому наближенні $x^{(0)} = (4; 4)$.

Перед розв'язуванням поставленої задачі її спочатку доцільно записати у вигляді (7)-(10):

$$f(x) = \frac{1}{2}\left\langle \begin{pmatrix} 2 & 4 \\ 4 & 12 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} 5 \\ 8 \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\rangle \to \min,$$

$$-x_1 - x_2 \le 3,$$

$$0 \le x_1,\ 0 \le x_2.$$

Потім у вікні Command Window пакету Matlab 6.5 для введення вхідних даних задачі треба виконати послідовно такі команди:

```
>> H = [2 4; 4 12]
>> c = [5; 8]
>> lb = zeros(2,1)
>> A = [-1 -1]
>> b=-3
>> x0=[4; 4]
```

Після цього набрати і виконати, наприклад, команду:

```
>>[x,fval,exitflag,output] = quadprog(H,c,A,b,[],[],lb,x0),
```

при цьому буде одержано такий результат:

```
Warning: Large-scale method does not currently solve this problem formulation,
switching to medium-scale method.
> In C:\MATLAB6P5\toolbox\optim\quadprog.m at line 213
Optimization terminated successfully.
x =       3.0000  0
fval =  24.0000
exitflag =  1
output =
        iterations: 2
        algorithm: 'medium-scale: active-set'
        firstorderopt: []
        cgiterations: [].
```

У даному випадку пакетом використано метод для розв'язування задач середньої розмірності (Medium-Scale Optimization), подібний до методу, описаному в роботі Gill, P. E. and W. Murray, and M.H. Wright, Practical Optimization, Academic Press, London, UK, 1981.

Зауважимо, що для розв'язування задач квадратичного програмування великої розмірності (Large-Scale Optimization), а також при відсутності обмежень-нерівностей, або при наявності лише обмежень-рівнянь, або при наявності лише прямих обмежень використовується метод, який базується на методі Interior-reflective Newton, описаному в роботі Coleman, T.F. and Y. Li, "A Reflective Newton Method for Minimizing a Quadratic Function Subject to Bounds on some of the Variables," SIAM Journal on Optimization, Vol. 6, Number 4, pp. 1040-1058, 1996.

Функція *fmincon* призначена для розв'язування задачі нелінійного програмування виду

$$f(x) \to \min, \quad (11)$$

$$c(x) \le 0, \quad (12)$$

$$ceq(x) = 0, \quad (13)$$

$$A \cdot x \le b, \quad (14)$$

$$Aeq \cdot x = beq, \quad (15)$$

$$lb \le x \le ub, \quad (16)$$

де $c(x), ceq(x)$ – вектор-функції, компоненти яких, як і функція $f(x)$, можуть бути нелінійними, x, b, beq, lb, ub – вектор-стовпчики, A, Aeq – прямокутні матриці, і яка має такий синтаксис:

```
x = fmincon(fun,x0,A,b);
x = fmincon(fun,x0,A,b,Aeq,beq);
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub);
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon);
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options);
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options,p1,p2, ...);
[x,fval] = fmincon(...);
[x,fval,exitflag] = fmincon(...);
[x,fval,exitflag,output] = fmincon(...);
[x,fval,exitflag,output,lambda] = fmincon(...);
[x,fval,exitflag,output,lambda,grad] = fmincon(...);
[x,fval,exitflag,output,lambda,grad,hessian] = fmincon(...).
```

Розглянемо особливості використання параметра nonlcon, за допомогою якого у функції fmincon визначаються нелінійні обмеження-нерівності виду (12) та обмеження-рівняння виду (13).

Функція nonlcon має в якості вхідного параметра вектор x, а повертає значення, як правило, двох векторів c і ceq. Вектор c містить значення нелінійних функцій, що відповідають лівим частинам обмежень-нерівностей, в точці x, а вектор ceq містить значення нелінійних функцій, що відповідають лівим частинам обмежень-рівнянь, в точці x. Функція nonlcon може бути визначена як функція користувача. Для цього, виконавши послідовність дій File\New\M-file, в текстовому редакторі Matlab 6.5 створюється m-файл, наприклад, з ім'ям nonlcon, який описує систему нелінійних обмежень-нерівностей і обмежень-рівнянь задачі виду (11)-(16).

Зауваження.

1. Якщо в умові задачі відсутні обмеження певного виду, то в m-файлі, в якому описуються нелінійні обмеження, треба записати відповідно c='' або ceq='', при цьому у заголовку функції повинні бути вказані обидва вихідні параметри [c,ceq].
2. Якщо в умові задачі є кілька обмежень-нерівностей або обмежень-рівностей, то для їх опису використовується матричний запис.

Приклад 5. За допомогою функції *fmincon* пакету Matlab 6.5 розв'язати задачу нелінійного програмування:

$$f(x) = x_1^3 - 6x_1^2 + 11x_1 + x_3 \to \min, \quad x \in X,$$

де

$$X = \{x \in R^3 \mid x_1^2 + x_2^2 - x_3^2 \le 0;\ x_1^2 + x_2^2 + x_3^2 \ge 4;\ x_1 \ge 0;\ x_2 \ge 0;\ 0 \le x_3 \le 5\},$$

при початкових наближення: а) $x^{(0)} = (10; 10; 10) \notin X$; б) $x^{(0)} = (4; 3; 5) \in X$.

При цьому наближеним розв'язком поставленої задачі є $x^* \approx (0; 1.414; 1.414)$, $f(x^*) \approx 1.414$.

Для того щоб розв'язати поставлену задачу треба створити m-файл, наприклад з ім'ям nonlcon (рис. 4), який описує систему функціональних обмежень задачі, а потім у вікні Command Window пакету Matlab 6.5 набрати і виконати, наприклад, команди:

```
>>fun='x(1)^3-6*x(1)^2+11*x(1)+x(3)'
>> [x,fval,exitflag,output]=fmincon(fun,[10;10;10],[],[],[],[],[0;0;0],[ ; ;5],@nonlcon).
```

В результаті буде одержано:

```
Optimization terminated: No feasible solution found.
 Search direction less than 2*options.TolX but constraints are not satisfied.
x =    5.0000    10.0000    10.0000
fval =    40.0000
exitflag =     -1
output =
      iterations: 1
      funcCount: 9
      stepsize: 1
      algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
      firstorderopt: 26.0000
      cgiterations: []
```

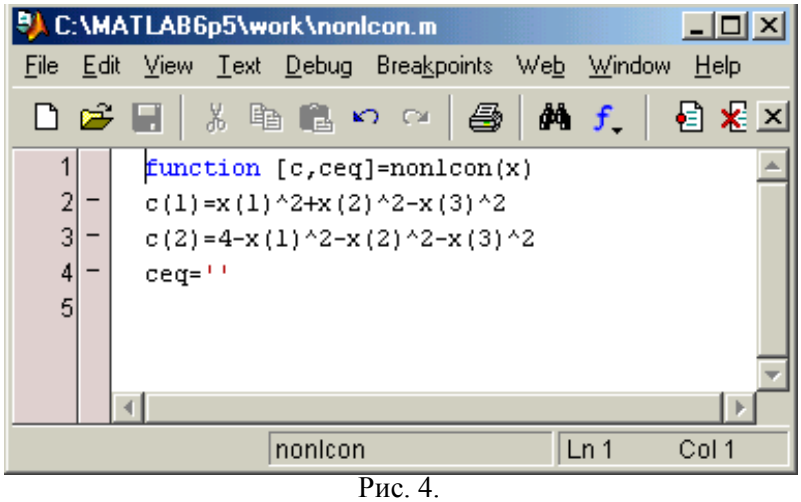

Рис. 4.

Розв'яжемо задачу при іншому значенні початкового наближення:

>> [x,fval, exitflag,output] = fmincon(fun,[4;3;5],[],[],[],[],[0;0;0],[; ;5],@nonlcon).

В результаті буде одержано:

Optimization terminated successfully:
First-order optimality measure less than options.TolFun and
maximum constraint violation is less than options.TolCon

Active Constraints: 1 3
x = 4 3 5
fval = 17
exitflag = 1
output =
iterations: 1
funcCount: 9
stepsize: 1
algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
firstorderopt: 0
cgiterations: []

Як видно з одержаних результатів, у обох випадках не було знайдено розв'язку, який би співпадав з розв'язком, що був знайдений за допомогою системи Mathcad. Це свідчить про те, що при розв'язуванні реальних оптимізаційних задач треба використовувати різні системи комп'ютерної математики і ретельно аналізувати одержані результати.

4. Останні роки на математичному факультеті Черкаського національного університету проводиться цілеспрямована поетапна робота по формуванню у студентів вмінь і навичок щодо використання СКМ у майбутній професійній діяльності. На першому етапі під час проведення обчислювальної практики (4 семестр) здійснюється ознайомлення студентів з основними можливостями пакетів Mathcad і Matlab на прикладах розв'язування задач математичного аналізу і лінійної алгебри. На другому етапі студенти використовують ці пакети при вивчені курсів „Методи обчислень" (6 семестри) і „Математичне моделювання" (6 семестр). На третьому етапі у межах спецкурсів і спецсемінарів розглядаються основи моделювання соціально-економічних процесів (8 семестр) з використанням СКМ і основи комп'ютерної математики (9 семестр). На заключному етапі при вивченні курсу „Методи оптимізації" (10 семестр) відбувається знайомство студентів з можливостями застосування СКМ Mathcad і Matlab для розв'язування екстремальних задач.

Тривалий час автором проводиться робота по створенню комп'ютерно орієнтованих методичних систем навчання таких дисциплін, як „Методи оптимізації" (для студентів математичних і комп'ютерних спеціальностей), „Математичне програмування" і „Дослідження операцій" (для студентів економічних спеціальностей). Результатом цієї роботи стало створення і впровадження у навчальний процес ВНЗ м. Черкаси навчально-методичних комплексів з цих дисциплін [5-7].

При цьому використання СКМ дозволило [7]:

- змінити акценти у підборі теоретичного матеріалу, зокрема приділити більше уваги методам оптимізації, які використовуються у СКМ;
- збільшити долю задач на побудову математичних моделей реальних оптимізаційних задач та їх дослідження за допомогою СКМ;
- більш широко використовувати графічні методи при розв'язуванні задач одно- і двохвимірної оптимізації;
- використовувати СКМ для розв'язування задач оптимізації класичними методами, зокрема на основі застосування необхідних і достатніх умов екстремуму та функції Лагранжа;
- для студентів математичних і комп'ютерних спеціальностей запровадити завдання на порівняння результатів, одержаних за допомогою чисельних методів оптимізації, реалізованих на одній з мов програмування, і за допомогою вбудованих засобів СКМ, та їх аналіз при різних вхідних даних, а також завдання на програмування в середовищах математичних пакетів чисельних методів оптимізації та їх дослідження;

- запровадити завдання на створення інтегрованих звітних документів про виконання лабораторних і розрахунково-графічних робіт з використання СКМ.

Як певний підсумок цієї роботи стало написання у співавторстві з М. І. Жаладком навчального посібника „Основи теорії і методів оптимізації”, який рекомендований МОН України для студентів математичних спеціальностей ВНЗ і в якому, зокрема, розглядаються основні функції та особливості використання пакетів Mathcad, Matlab, Mathematica і надбудови „Пошук рішення” MS Excel для розв’язування різних класів задач оптимізації.

Висновки

1. Широкі аналітичні, обчислювальні і графічні можливості сучасних математичних пакетів роблять їх одними з основних інструментів у професійній діяльності математика-аналітика, інженера, економіста-кібернетика тощо. Тому їх використання у навчальному процесі ВНЗ при вивченні математичних дисциплін дозволить підвищити рівень професійної підготовки студентів, рівень їх математичної та інформаційної культури, зробити майбутніх фахівців конкурентноспроможними на міжнародному ринку праці.

2. МОН України та ВНЗ необхідно здійснити реальні кроки для того, щоб як найшвидше ліцензійні математичні пакети в достатній кількості з’явилися на математичних кафедрах для забезпечення навчального процесу на математичних, природничих, комп’ютерних, технічних та економічних спеціальностях.

3. Викладачам математичних кафедр необхідно проводити цілеспрямовану роботу щодо створення комп’ютерно орієнтованих методичних систем навчання математичних дисциплін з широким використанням СКМ.

ЛІТЕРАТУРА

1. Васильев Ф.П. Численные методы решения экстремальных задач.– М.: Наука.– 1980. – 520 с.
2. Говорухин В., Цибулин В. Компьютер в математических исследованиях.–СПб.: Питер, 2001.– 624 с.
3. Дьяконов В. П. Компьютерная математика. Теория и практика. – М.: Нолидж, 2001.– 1296 с.
4. Поляк Б.Т. Введение в оптимизацию.– М.: Наука, 1983. – 384 с.
5. Триус Ю. В. Методична система навчання курсу “Основи теорії оптимізації”//Сучасні інформаційні технології в навчальному процесі: Зб. наук. пр./ Редкол.– К.: НПУ, 1997. – С.64-76.
6. Триус Ю. В., Онищенко Б. О. Комп’ютерно-орієнтована система навчання курсу “Математичні методи оптимізації”// Матеріали третьої Всеукраїнської конференції молодих науковців “Інформаційні технології в науці, освіті і техніці” (ІТОНТ-2002). – Черкаси, 2002.– С.155-156.
7. Триус Ю.В. Використання систем комп’ютерної математики при вивченні і розв’язуванні задач оптимізації // Проблеми сучасного підручника: Зб. наук. праць / Ред. кол.– К.: Педагогічна думка, 2004.– Вип.. 5.– Ч.ІІ.– С. 191-200.
8. Ульянченко О. В. Дослідження операцій в економіці: Підручник для студентів вузів/Харк. нац. аграр. ун-т ім. В.В. Докучаєва.– Харків: Гриф, 2002.– 580 с.
9. Nelder J.A., Mead R. A Simplex Method for Function Minimization, Computer J., Vol .7, pp. 308-313, 1965.
10. Stefan Steinhaus Comparison of mathematical programs for data analysis (Edition 4.4).– Munchen/Germany.– 60 p.: http://www.scientificweb.de/ncrunch/