МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Методичні вказівки

до лабораторних робіт з дисципліни
«Сучасні методи математичного моделювання та оптимізації»
для студентів усіх форм навчання спеціальності 125 «Кібербезпека та захист інформації» галузі знань 12 «Інформаційні технології»

2024

Методичні вказівки до лабораторних робіт з дисципліни «Сучасні методи математичного моделювання та оптимізації» для студентів усіх форм навчання спеціальності 125 «Кібербезпека та захист інформації» галузі знань 12 «Інформаційні технології» / Укл: Л.М.Карпуков, Т. О. Жорж. – Запоріжжя: НУ «Запорізька політехніка», 2024.- 62 с.

Наведено методи і алгоритми математичного моделювання і оптимізації, які реалізовано в прикладних пакетах MATLAB.

Укладачі: Л.М. Карпуков, професор, д.т.н.
Т. О. Жорж, зав.лаб.

Рецензент: С.М. Романенко, доцент, к.ф.-м.н.

Відповідальний за випуск: А.В. Коротун, доц., канд.фіз.-матем.наук

Затверджено
на засіданні кафедри
інформаційної безпеки та наноелектроніки
Протокол № 6 від 10.05.2024 р.

Рекомендовано до видання НМК факультету інформаційної безпеки та електронних комунікацій
Протокол № 7 від 14.05.2024 р

ЗМІСТ

ЛАБОРАТОРНА РОБОТА № 1

МЕТОДИ ОДНОВИМІРНОЇ МІНІМІЗАЦІЇ В СИСТЕМІ MATLAB

Мета роботи: Придбання практичних навичок програмування в середовищі MATLAB на прикладах вирішення завдань одномірної мінімізації чисельними методами.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Завдання одновимірної мінімізації полягає в знаходженні мінімуму функції $y = F(x)$ однієї змінної на інтервалі $x \in [a, b]$. Мінімуму відповідає точка

$$F(x^*) = \min F(x),\ x^* \in [a, b].$$

Інтервал $L = [a, b]$ пошуку мінімуму називається інтервалом невизначеності. На інтервалі L можлива присутність кількох локальних екстремумів, відповідних мінімуму. Найменший з них - це глобальний мінімум, решта - локальні мінімуми.

В системі MATLAB для одновимірного пошуку мінімуму призначена підпрограма **fminbnd** з пакета **Optimization Toolbox,** в якій на початковому етапі пошуку використовується метод золотого перетину, а в околицях мінімуму здійснюється перехід на метод квадратичної інтерполяції, що має більшу швидкість збіжності.

Метод золотого перетину. Метод відноситься до послідовних стратегій. Задається початковий інтервал невизначеності і необхідної точності. Алгоритм грунтується на розбитті інтервалу невизначеності в золотих пропорціях з подальшим аналізом значень функції у двох точках розбиття.

Точка золотого перетину визначається з наступної умови: *відношення довжини всього інтервалу до більшої частини дорівнює*

відношенню більшої частини до меншої частини. Чисельно ці відносини рівні $\left(1+\sqrt{5}\right)\Big/2 \cong 1.618$

На інтервалі $[a_0, b_0]$ є дві симетричні щодо його кінців точки y_0 і z_0 золотого перетину:

$$\frac{b_0 - a_0}{b_0 - y_0} = \frac{b_0 - y_0}{y_0 - a_0} = \frac{b_0 - a_0}{z_0 - a_0} = \frac{z_0 - a_0}{b_0 - z_0} = \frac{1+\sqrt{5}}{2} \cong 1.618,$$

тут точка y_0 виробляє золотий перетин відрізка $[a_0, z_0]$, а точка z_0 - відрізка $[y_0, b_0]$.

В якості точок обчислення функції вибираються точки золотого перетину. На кожній ітерації, крім першої, потрібно тільки одне нове обчислення функції. Пошук закінчується, якщо довжина поточного інтервалу невизначеності менше заданої величини.

Алгоритм пошуку мінімуму функції за методом золотого перетину зводиться до виконання наступних етапів.

1. Задається початковий інтервал невизначеності $L_0 = [a_0, b_0]$ та $\alpha > 0$ - - необхідна точність.
2. Задати $k = 0$.
3. Обчислити $y_k = a_0 + \frac{3-\sqrt{5}}{2}(b_0 - a_0),\ z_k = a_0 + b_0 - y_0$.
4. Обчислити $F(y_k), F(z_k)$
5. Якщо $F(y_k) \le F(z_k)$, то прийняти $a_{k+1} = a_k, b_{k+1} = z_k$ і $y_{k+1} = a_{k+1} + b_{k+1} - y_k, z_{k+1} = y_k$. Перейти до етапу 6.
6. Якщо $F(y_k) > F(z_k)$, прийняти $a_{k+1} = y_k, b_{k+1} = b_k$ і $y_{k+1} = z_k, z_{k+1} = a_{k+1} + b_{k+1} - z_k$
7. Обчислити $\Delta = \left|a_{k+1} - b_{k+1}\right|$ і перевірити умови закінчення пошуку. Якщо $\Delta \le \alpha$, процес пошуку припиняється і $x^* \in [a_{k+1} + b_{k+1}]$. За наближене рішення приймають середину останнього інтервалу $x^* \cong \frac{a_{k+1} + b_{k+1}}{2}$.

8. Якщо $\Delta > \alpha$, прийняти $k = k + 1$ і перейти до етапу 4.

Метод квадратичної інтерполяції. Метод квадратичної інтерполяції відноситься до послідовних стратегій. Задається початкова точка і за допомогою пробного кроку знаходяться три точки так, щоб вони були якомога ближче до шуканої точки мінімуму. В отриманих точках обчислюються значення функції. Потім будується інтерполяційний поліном другого ступеня, що проходить через ці три точки. За наближення точки мінімуму береться точка мінімуму полінома. Процес пошуку закінчується, коли отримана точка відрізняється від найкращої з трьох опорних точок не більше, ніж на задану величину.

Алгоритм пошуку точки мінімуму методом квадратичної інтерполяції зводиться до виконання наступних етапів.

1. Задати початкову точку x_1, величину кроку $\Delta x > 0$, $\varepsilon_1, \varepsilon_2$ - малі позитивні числа, що характеризують точність.
2. Обчислити $x_2 = x_1 + \Delta x$
3. Обчислити $f_1 = F(x_1),\ f_2 = F(x_2)$.
4. Якщо $F(x_1) > F(x_2)$, то прийняти $x_3 = x_1 + 2\Delta x$. Якщо $F(x_1) \le F(x_2)$, то прийняти $x_3 = x_1 - \Delta x$. Обчислити $f_3 = F(x_3)$.
5. Знайти $F_{\min} = \min(f_1, f_2, f_3),\ x_{\min} = x_i,\ F(x_i) = F_{\min}$.
6. Обчислити точку мінімуму інтерполяційного полінома, побудованого за трьома точкам

$$\bar{x} = \frac{1}{2}\frac{f_1 \cdot (x_2^2 - x_3^2) + f_2 \cdot (x_3^2 - x_1^2) + f_3 \cdot (x_1^2 - x_2^2)}{f_1 \cdot (x_2 - x_3) + f_2 \cdot (x_3 - x_1) + f_3 \cdot (x_1 - x_2)}$$

і величину $F(\bar{x})$. Якщо знаменник у формулі для $\bar{x}$ на деякій ітерації обертається в нуль, то результатом інтерполяції є пряма лінія. В цьому випадку рекомендується прийняти $x_1 = x_{\min}$ і перейти до кроку 2.

7. Перевірити виконання умов закінчення

$$\left|\frac{F_{\min} - F(\bar{x})}{F(\bar{x})}\right| < \varepsilon_1, \quad \left|\frac{x_{\min} - \bar{x}}{\bar{x}}\right| < \varepsilon_2 .$$

8. Якщо обидві умови виконані, то процедура закінчена і $x^* \cong \bar{x}$.

   Якщо хоча б одну з умов не виконано і $\bar{x} \in [x_1, x_3]$, вибрати найкращу точку ($x_{\min}$ або $\bar{x}$) і дві точки по обидва боки від неї. Перенумерувати ці точки в порядку зростання і перейти до етапу 6.

9. Якщо хоча б одну з умов не виконано і $\bar{x} \notin [x_1, x_3]$, то прийняти $x_1 = \bar{x}$ і перейти до етапу 2.

Функція fminbnd. Функція fminbnd - функція скалярної нелінійної мінімізації з обмеженнями виду x1 <x <x2. Алгоритм базується на методі золотого перетину і квадратичної (параболічної) інтерполяції. Варіанти запису звернення до функції:

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(...)

Тут *fun* - ім'я функції, що підлягає мінімізації;
x1, x2 - інтервал пошуку мінімуму;
options - набір опцій;
P1, P2, ..- додаткові аргументи P1, P2, і т.д., які передаються в цільову функцію fun.
Набір опцій *options* може містити наступні опції:
Display - рівень відображення:

'off' - висновок інформації відсутня, *'iter'* - висновок інформації про пошук рішення на кожній ітерації, *'final'* - висновок тільки підсумкової інформації;
TolX - допуск на допустиме відхилення по x;
MaxFunEvals - максимальне число обчислень функції;
MaxIter – максимальне допустиме число ітерацій.

Перераховані опції вказуються за допомогою команди *optimset.*
Вихідні параметри:

x- координата знайденого мінімуму;
fval - значення функції в точці мінімуму;
exitflag - індикатор результату пошуку мінімуму;
output - набір опцій підсумків мінімізації.
Параметр *exitflag* може набувати таких значень:
> 0 - функція сходиться до вирішення по x.
0 - максимальне число оцінок функції або ітерацій було перевищено
<0 - функція не сходиться до вирішення.
Набір опцій *output:*
iterations - число виконаних ітерацій;
funcCount - число оцінок функції;
algorithm - використовуваний алгоритм.

Приклади. Для мінімізації, наприклад, функції $f(x)=(x-3)^2-1$ на інтервалі [0, 5] необхідно представити цю функцію у вигляді m-файлу або в блокноті (Notepad), або в спеціальному редакторі M-file Editor системи MATLAB (кнопка New M-File на панелі команд) і зберегти в папі MATLAB \ work \:

```
>> function f = primer1(x)
>> f =(x-3).^2-1;
```

Можна построїти графік цієї функції та приблизно визначити по графіку точку мінімуму:

```
>> x=0:0.1:10;
>> y=primer1(x);
```

```
>> plot(x,y)
```

Варіанти звернення до функції мінімізації.
Варіант 1:

```
>> x=fminbnd('primer1',0,5)
x =
    3
```

По знайденому x в точці мінімуму знаходимо значення функції:

```
>> y=primer1(x)
y =
   -1
```

Варіант 2:

```
>> [x,vfal]=fminbnd('primer1',0,5)
x =
    3
vfal =
   -1
```

Варіант 3:

```
>> [x,vfal,exitflag]=fminbnd('primer1',0,5)
x =
    3
vfal =
   -1
exitflag =
    1
```

Варіант 4

```
>> options=optimset('Display','iter','tolX',0.001);
>>  [x,vfal]=fminbnd('primer1',0,5,options)
 Func-count     x          f(x)         Procedure
    1        1.90983     0.188471       initial
    2        3.09017    -0.991869       golden
    3        3.81966    -0.328157       golden
```

```
    4          3          -1    parabolic
    5       2.99967       -1       parabolic
    6       3.00033       -1       parabolic
Optimization terminated:
 the current x satisfies the termination criteria using OPTIONS.TolX of 1.000000e-003
x =
    3
vfal =
   -1
```

Варіант 5:

```
>> options=optimset('Display','iter','Maxiter',2);
>> [x,vfal]=fminbnd('primer1',0,5,options)
 Func-count    x          f(x)         Procedure
    1        1.90983    0.188471       initial
    2        3.09017   -0.991869        golden
    3        3.81966   -0.328157        golden
Exiting: Maximum number of iterations has been exceeded
        - increase MaxIter option.
        Current function value: -0.991869
x =
  3.090169943749474
vfal =
 -0.991869381244217
```

ЛАБОРАТОРНЕ ЗАВДАННЯ

Потрібно знайти мінімум функції в заданому інтервалі. Варіанти функцій наведені в таблиці 1.1.

Таблиця 1.1 - Варіанти функцій

№	Функція	Інтервал невизначеності	Початкова точка пошуку
1	$\frac{20}{x}\sin(x) + x^2$	[1; 4]	1

2	$0.4 \cdot \exp(\mathrm{x}) + \mathrm{x}^2$	[-2; 1]	-2
3	$4\ \exp(\mathrm{x}) + \mathrm{x}^2$	[-2; 1]	1
4	$\exp(x) + 1/x$	[0.1; 2]	2
5	$x + 1/\ln(x)$	[1.5; 4]	4
6	$0.6\ \mathrm{x} + \mathrm{x}/\ln(\mathrm{x})$	[1.5; 4]	4
7	$\mathrm{x}^2\ \exp(0.8\,\mathrm{x}) + 0.2$	[-1; 1]	1
8	$\frac{1}{\mathrm{x}} + \mathrm{x}\ln(\mathrm{x})$	[0.1; 4]	4
9	$\exp\left(\frac{1}{\mathrm{x}}\right) + \ln(2\,\mathrm{x})$	[0.5; 4]	4
10	$x - \ln(\ln(x))$	[0.5; 4]	4

Потрібно:

1. Скласти блок-схеми алгоритмів пошуку точки екстремуму функції методами золотого перетину і квадратичної інтерполяції.
2. Побудувати в системі MATLAB графік функції в заданому інтервалі і за графіком визначити координати екстремуму.
3. Знайти за допомогою функції *fminbnd* координати і значення функції в точці мінімуму. Здійснити 5 варіантів звернень до функції *fminbnd*, розглянутих в попередньому розділі.
4. Проаналізувати отримані результати і зробити висновки за досягнутою точністю та кількістю обчислень функції.
5. Дати письмові відповіді на контрольні питання.

ЗМІСТ ЗВІТУ

1. Мета роботи.
2. Формулювання завдання. Короткі теоретичні відомості.
3. Блок-схеми алгоритмів пошуку мінімуму.
4. Графічне представлення функції.
5. Лістинг програм.

6. Результати обчислень.
7. Висновки.
8. Відповіді на контрольні запитання.

Контрольні питання

1. Завдання одновимірної мінімізації.
2. Поняття інтервалу невизначеності.
3. Визначення локального і глобального екстремуму.
4. Як знайти координати максимуму функції за допомогою програми мінімізації?
5. Суть методу золотого перетину.
6. Умова визначення точки золотого перетину.
7. Суть методу квадратичної інтерполяції.
8. Точка мінімуму інтерполяційного полінома, побудованого за трьома точкам.
9. Набір опцій options.
10. Набір опцій output.
11. Значення параметра exitflag.

ЛАБОРАТОРНА РОБОТА № 2

МЕТОДИ БАГАТОВИМІРНОЇ БЕЗУМОВНОЇ МІНІМІЗАЦІЇ В СИСТЕМІ MATLAB

Мета роботи: Придбання практичних навичок програмування і оптимізації в середовищі MATLAB на прикладах вирішення завдань мінімізації багатовимірних функцій чисельними методами.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Завдання багатовимірної безумовної оптимізації функції f (x) багатьох змінних $X=x_1,..,x_n$ полягає в знаходженні точки X^* в n-вимірному просторі, в якій функція набуває мінімального значення, тобто потрібно знайти

$$f(X^*) = \min f(X),\ X^* \in X.$$

Чисельні методи знаходження мінімуму, як правило, складаються в побудові послідовності точок X_k, k=0,1,2…, задовольняють умові $f(X_0)>f(X_1)>\ldots>f(X_k)>\ldots>f(X^*)$. Методи побудови таких послідовностей називаються методами спуску. В цих методах точки послідовності X_k обчислюються за формулою:

$$X_{k+1} = X_k + \alpha_k S_k,$$

де S_k - вектор, що визначає напрямок спуску до мінімуму, α_k - довжина кроку в цьому напрямку.

В системі MATLAB для пошуку безумовного мінімуму багатовимірних функцій призначена підпрограма **fminsearch** з пакета **Optimization Toolbox**, в якій реалізовано симплексний метод Нелдера-Міда. Під симплексом розуміється n-мірний випуклий багатогранник, який має n + 1 вершину в n-вимірному просторі (для двовимірного простору це трикутник, для тривимірного - тетраедр).

Алгоритм методу Нелдера-Міда. Алгоритм заснований на повторюваних операціях відображення, стиснення і редукції симплексів.

Крок 1. Побудова початкового симплекса.

Задаються початкова точка X_0 і довжина ребра r симплекса. Уздовж координатних осей формуються інші вершини симплекса:

$$X_i = X_0 + re_i,\ i = 1,2,...,n,$$

де ei - одиничні вектори.

Крок 2.Операція відображення.

Для цього на к-й ітерації обчислюються значення цільової функції в кожній вершині симплекса. Визначається вершина X_{min} з мінімальним значенням функції f_{min} (найкраща вершина) і вершина X_{max} з максимальним значенням f_{max} функції (найгірша вершина). Визначається для k-й ітерації центр ваги симплекса, без урахування вершини X_{max}:

\

$$C_k = \frac{1}{n}\sum_i X_i,\ X_i \neq X_{max}.$$

Виробляється відображення гіршої вершини через центр ваги в напрямку $S_k=C_k-X_{max}$, що проходить через центр ваги і гіршу вершину. Визначається відображена точка:

$$X_{отр} = C_k + (C_k - X_{max}) = 2C_k - X_{max}.$$

Крок 3. Перевірка функції у відбитій точці і побудова нового симплекса.

Можливі три випадки:

а) Відбита точка краще кращої вершини:

$$f(X_{отр}) < f(X_{min}).$$

В цьому випадку проводиться операція розтягування симплекса в напрямку, що проходить через центр ваги і відбиту точку. Будується нова точка:

$$X_{рас} = C_k + \beta\left(X_{отр} - C_k\right),$$

де $\beta > 1$ - коефіцієнт розтягування.

Якщо $f(X_{рас}) < f(X_{отр})$, то найгірша вершина X_{max} симплекса заміщається точкою $X_{рас}$..

Якщо $f(X_{рас}) \geq f(X_{отр})$, то найгірша вершина X_{max} симплекса заміщується точкою $X_{отр}$.

а) Відбита точка краще поганої вершини, але гірше кращої:

$$f(X_{min}) \leq f(X_{отр}) < f(X_{max})$$

В цьому випадку застосовується операція стиснення симплекса. Визначається точка стиснення. Визначаються точки стиснення $X_{ст}$:

якщо $f(X_{max}) \leq f(X_{отр})$, то $X_{сж} = C_k + \gamma\left(X_{max} - C_k\right)$;

якщо $f(X_{max}) > f(X_{отр})$, то $X_{сж} = C_k + \gamma\left(X_{отр} - C_k\right)$;

де $\gamma < 1$ - коефіцієнт стиснення.

Найгірша вершина Xmax симплекса заміщується точкою Хст, якщо менше $f(X_{max})$ і менше в іншому випадку застосовується операція редукції симплекса.

Крок. 3. Редукція симплекса.

Проводиться зменшення розмірів симплекса. Новий симплекс будується навколо кращої вершини X_{min} шляхом зменшення у два рази довжини ребер, що з'єднують кращу вершину з іншими вершинами X_i симплекса. Точки нового симплекса:

$$X_{i\,нов} = \left(X_i + X_{min}\right)/2$$

Крок 4. Перевірка збіжності. Якщо збіжність не досягнуто, то перехід на крок 1.

Функція *fminsearch*. Функція *fmlnsearch* дозволяє знайти мінімум функції декількох змінних $y = f(x_1, x_2, ..., x_n)$ при відсутності обмежень на змінні.

Варіанти запису функції:

```
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(...)
```

Тут *fun* - ім'я функції, що підлягає мінімізації;

x0 - вектор змінних, що визначає початкову точку пошуку мінімуму;

options - набір опцій;

P1, P2, ..- додаткові аргументи *P1, P2,* і т.д., які передаються в цільову функцію *fun*.

Набір опцій options може містити опції:

Display - рівень відображення: *'off'* - висновок інформації відсутній, *'iter'* - висновок інформації про пошук рішення на кожній ітерації, '*final'* - висновок тільки підсумкової інформації;

TolX - допуск на допустиме відхилення по *x*;

MaxFunEvals - максимальне число обчислень функції;

MaxIter - максимальне допустиме число ітерацій.

Перераховані опції вказуються за допомогою команди optimset.

Вихідні параметри:

x- координата знайденого мінімуму;

fval - значення функції в точці мінімуму;

exitflag - індикатор результату пошуку мінімуму;

output - набір опцій підсумків мінімізації.

Параметр *exitflag* може набувати таких значень:

> 0 - функція сходиться до вирішення по x;

0 - максимальне число оцінок функції або ітерацій було перевищено;

<0 - функція не сходиться до вирішення.

Набір опцій output:

iterations - число виконаних ітерації;

funcCount - число оцінок функції;

algorithm - використовуваний алгоритм.

Приклади. Для мінімізації, наприклад, функції Розенброка,

$$y = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1\right)^2,$$

яка є однією з тестових функцій з початковою точкою [-1.2, 1.0], необхідно представити цю функцію у вигляді m-файлу або в блокноті *(Notepad)*, або в спеціальному редакторі *M-file Editor* системи MATLAB (кнопка New M-File на панелі команд) і зберегти в папці MATLAB \ *work* \.:

```
function y = rosenbr(x)
y = 100 * (x(2) - x(1)^2)^2 + (1 - x(1))^2;
```

Приклади звернення до функції мінімізації.

Варіант 1. Використовуються параметри за замовчуванням.

```
x0 = [-1.2,1];
[x,fval] = fminsearch('rosenbr',x0)
x =
   1.0000    1.0000
fval =
  8.1777e-010
```

Варіант 2. Використовуються параметри за замовчуванням. Виводиться підсумкова інформація.

```
[x,fval,exitflag,output]=fminsearch('rosenbr',[-1.2 1])
```

```
x =
    1.0000    1.0000
fval =
  8.1777e-010
exitflag =
     1
output =
    iterations: 85
     funcCount: 159
     algorithm: 'Nelder-Mead simplex direct search'
       message: [1x196 char]
```

Варіант 3. Використовуються параметри за замовчуванням за винятком числа ітерацій. Виводиться підсумкова інформація.

```
options=optimset('Display','iter','Maxiter',5);
[x,fval,exitflag,output]=fminsearch('rosenbr',[-1.2 1],options)
```

```
 Iteration   Func-count     min f(x)         Procedure
     0           1            24.2
     1           3            20.05        initial simplex
     2           5           5.1618         expand
     3           7           4.4978         reflect
     4           9           4.4978         contract outside
     5          11           4.38136          contract inside
Exiting: Maximum number of iterations has been exceeded
         - increase MaxIter option.
         Current function value: 4.381360
x =
   -1.0650    1.1000
fval =
    4.3814
exitflag =
     0
output =
    iterations: 5
     funcCount: 11
```

algorithm: 'Nelder-Mead simplex direct search'
message: [1x135 char]

Приклад мінімізації функції

$$z = 2x^2 - 1.05x^4 + 1/6x^6 + xy + y^2$$

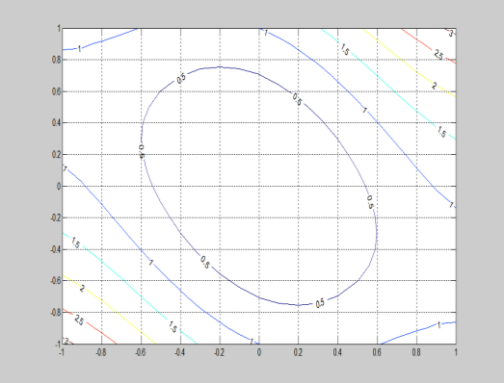
```
function z = flab2(x)
 z = 2*x(1)^2-1.05*x(1)^4+1/6*x(1)^6+x(1)*x(2)+x(2)^2;
```

Побудова контурного графіка функції:

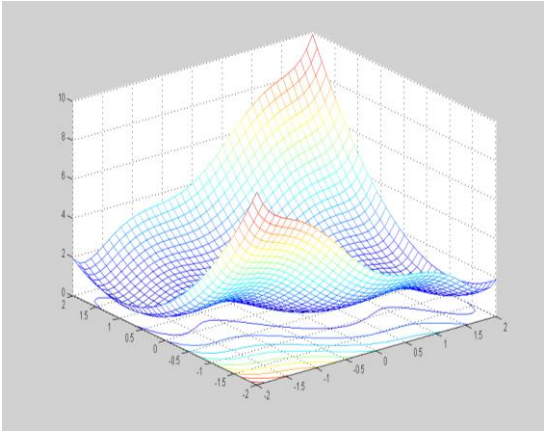
```
[x1,x2] = meshgrid(-1:.1:1);
Z = 2*x1.^2-1.05*x1.^4+1/6*x1.^6+x1.*x2+x2.^2;
[C,h] = contour(x1,x2,Z);
clabel(C,h);
grid on , hold on
```

Побудова тривимірного графіка з лініями рівного рівня:

```
[x1,x2] = meshgrid(-2:.1:2);
Z = 2*x1.^2-1.05*x1.^4+1/6*x1.^6+x1.*x2+x2.^2;
meshc(x1,x2,Z)
```



а)



б)

Рисунок 2.1 - Контурний (а) та тривимірний (б) графіки функції

Пошук мінімуму:

```
[x,fval,exitflag,output]=fminsearch('flab2',[0,0])
x =
    0     0
fval =
    0
exitflag =
    1
output =
   iterations: 4
    funcCount: 9
    algorithm: 'Nelder-Mead simplex direct search'
      message: [1x196 char]
```

Положення точки мінімуму на контурному графіку:

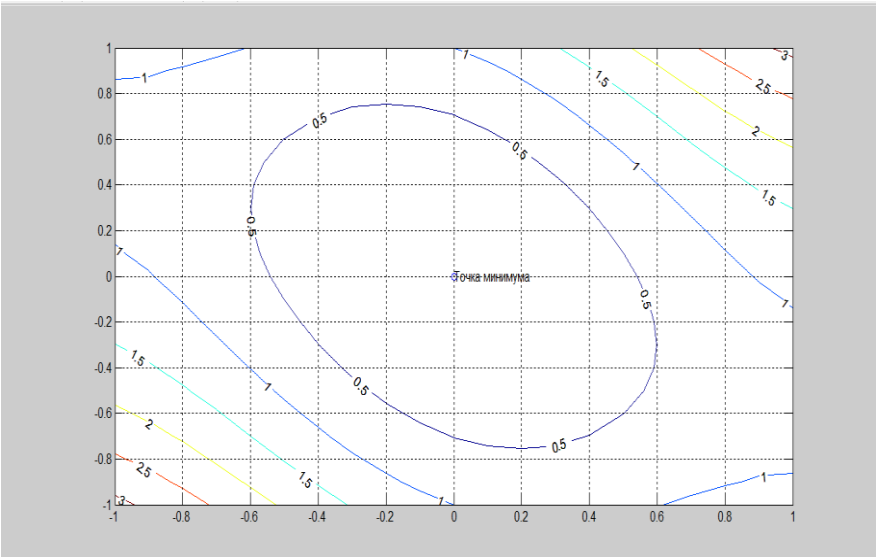


Рисунок 2.2 - Положення точки мінімуму на контурному графіку функції

ЛАБОРАТОРНЕ ЗАВДАННЯ

Потрібно знайти мінімум функції $f(x, y)$ в заданому інтервалі. Варіанти функцій наведені в таблиці 2.1.

Таблиця 2.1 - Варіанти функцій

№	Функція	Інтервал по x, y	Початкова точка пошуку
1	$x^2 + 12\sin(y)^2$	[-4; 6], [-2, 2]	5, 1

2	$14\sin(x)^2 + y^2$	[-2; 2], [-6; 6]	1.2, 6
3	$x^2 + 4\left(\cos(y) - 1\right)^2$	[-6; 8], [-4; 4]	5, 2
4	$(x - y)^2 + 14\sin(x)^2$	[-2; 2], [-4; 8]	1, 5
5	$(x + y)^2 + 12\sin(x)^2$	[-2; 2], [-5; 6]	2, 6
6	$(x - y)^2 + 14\sin(y)^2$	[-5; 8], [-2; 2]	6, 1
7	$(x - y)^2 + 3x^2 + 8\left(1 - \cos(y)\right)^2$	[-4; 4], [-2; 2]	4, 2
8	$(x - y)^2 + 2y^2 + 16\left(1 - \cos(x)\right)^2$	[-2; 2], [-4; 4]	2, 4
9	$2x^2 + 4\left(\cos(x - y) - 1\right)^2$	[-4; 6], [-4; 4]	5, 3
10	$2y^2 + 4\left(\cos(x - y) - 1\right)^2$	[-4; 4], [-4; 5]	3, 5

В ході виконання роботи потрібно:

1. Скласти блок-схему алгоритму Нелдера-Міда.
2. Побудувати контурний і тривимірний графіки функції.
3. Знайти за допомогою функції *fminsearch* координати і значення функції в точці мінімуму. Здійснити 3 варіанти звернень до функції, розглянутих в попередньому розділі.
4. Побудувати контурний графік функції і вказати положення мінімуму.
5. Проаналізувати отримані результати і зробити висновки за досягнутою точністю і кількістю обчислень функції.
6. Дати письмові відповіді на контрольні питання.

ЗМІСТ ЗВІТУ

1. Мета роботи.
2. Формулювання завдання. Короткі теоретичні відомості.
3. Блок-схема алгоритму Нелдера-Міда пошуку мінімуму.
4. Графічне представлення функції.
5. Лістинг програм.
6. Результати обчислень.

7. Висновки.
8. Відповіді на контрольні запитання.

КОНТРОЛЬНІ ПИТАННЯ

1. Завдання безумовної мінімізації функції багатьох змінних.
2. Суть методу спуску.
3. Визначення симплекса.
4. Побудова початкового симплекса в методі Нелдера-Міда.
5. Операція відображення в методі Нелдера-Міда.
6. Операція розтягування в методі Нелдера-Міда.
7. Операція редукції в методі Нелдера-Міда.
8. Основні варіанти звернення до функції *fminsearch.*
9. Набір опцій *options.*
10. Набір опцій *output.*
11. Значення параметра *exitflag.*

ЛАБОРАТОРНА РОБОТА № 3

МЕТОДИ БАГАТОВИМІРНОЇ МІНІМІЗАЦІЇ З ОБМЕЖЕННЯМИ В СИСТЕМІ MATLAB

Мета роботи: Придбання практичних навичок програмування і рішення оптимізаційних завдань в середовищі MATLAB на прикладах вирішення завдань умовної мінімізації багатовимірних функцій чисельними методами.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Завдання умовної мінімізації функції *f (x)* багатьох змінних $X=x_1,..,x_n$ полягає в знаходженні точки X * у n-вимірному просторі, в якій функція набуває мінімального значення при наявності обмежень типу рівностей і типу нерівностей, тобто потрібно знайти

$$f(X^*) = \min f(X),\ X^* \in X;$$
$$g_i(X) \le 0, i = 1,2..p;$$
$$h_i(X) = 0, i = 1,2,..q.$$

Методи умовної мінімізації діляться на прямі і непрямі.

Прямі методи оперують безпосередньо з цільовою функцією і є модифікації методів безумовної мінімізації, що дозволяють здійснювати спуск до точки мінімуму X* , не виходячи за межі допустимої області, яка визначається обмеженнями.

Непрямі методи засновані на перетворенні завдання умовної мінімізації. До них відноситься метод коефіцієнтів Лагранжа і методи штрафних функцій. Метод Лагранжа зводить задачу умовної мінімізації до вирішення системи нелінійних рівнянь. Методи штрафних функцій перетворять завдання умовної мінімізації в послідовність завдань безумовної мінімізації. Ці методи засновані на складанні допоміжної цільової функції, утвореної з цільової функції завдання f (X), до якої додаються з ваговими коефіцієнтами функції, що враховують обмеження. Процес мінімізації є ітераційний процес,

на кожному кроці якого вирішується завдання безумовної мінімізації при поступовій зміні від кроку до кроку вагових коефіцієнтів.

В системі MATLAB для пошуку умовного мінімуму багатовимірних функцій призначена підпрограма **fmincon** з пакета **Optimization Toolbox**, в якій реалізована комбінація прямих методів і множників Лагранжа.

Функція *fmincon*. Функція *fmincon* призначена для мінімізації функції багатьох змінних f (X) при наявності обмежень

$$AX< b,\ \ C\,X = d,\ \ l\le X\le u,\ g(X) \le 0,\ h(X) = 0.$$

Тут A, C - матриці;

l, u - вектори стовпці;

g (X), h (X) - векторні функції від вектора X.

Можливі такі звернення до функції:

x = fmincon(fun, x0, A, b)
x = fmincon(fun, x0, A, b, C, d)
x = fmincon(fun, x0, A, b, C, d, l, u)
x = fmincon(fun, x0, A, b, C, d, l, u, nonlcon)
x = fmincon(fun, x0, A, b, C, d, l, u, nonlcon, options)
x = fmincon(fun, x0, A, b, C, d, l, u, nonlcon, options, P1, P2, . . ., Pn)
[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(. . .)

Тут ряд параметрів і опцій аналогічні використовуваним в функціях *fminbnd* і *fminsearc*:

fun - ім'я функції, що підлягає мінімізації;

x0 - вектор змінних, що визначає початкову точку пошуку мінімуму;

options - набір опцій;

P1, P2, ..- додаткові аргументи *P1, P2*, і т.д., які передаються в цільову функцію fun.

Набір опцій *options* може містити опції:

Display - рівень відображення:

'off' - висновок інформації відсутній;

'iter' - висновок інформації про пошук рішення на кожній ітерації;

'final' - висновок тільки підсумкової інформації;

TolX - допуск на допустиме відхилення по x;

MaxFunEvals - максимальне число обчислень функції;

MaxIter - максимальне допустиме число ітерацій.

Перераховані опції вказуються за допомогою команди optimset.

Вихідні параметри:

x- координата знайденого мінімуму;

fval - значення функції в точці мінімуму;

exitflag - індикатор результату пошуку мінімуму;

output - набір опцій підсумків мінімізації.

Параметр *exitflag* може набувати таких значень:

> 0 - функція сходиться до вирішення по x.

0 - максимальне число оцінок функції або ітерацій було перевищено

<0 - функція не сходиться до вирішення.

Набір опцій *output:*

iterations - число виконаних ітерації;

funcCount - число оцінок функції;

algorithm - використовуваний алгоритм.

Додаткові параметри:

grad, hessian, lambda - висновок градієнта, матриці Гесса і множників Лагранжа у фінальній точці;

A, b, C, d, l, u лінійні обмеження, перелічені в постановці завдання мінімізації;

nonlcon - покажчик на нелінійні вектор-функції, що обчислюють обмеження типу нерівностей g (X) і рівності h (X). Ці векторні функції задаються парами; якщо обчислюється тільки один тип обмежень, то іншому присвоюється нуль. При відсутності деяких обмежень відповідні позиції заповнюються квадратними дужками [].

Приклад мінімізації функції з обмеженнями:

$$f(\bar{x}) = (x_1 + 4)^2 + (x_2 - 4)^2 \to \min$$
$$2x_1 - x_2 \le 2, \quad x_1 \ge 0, \quad x_2 \ge 0.$$

Побудова контурного графіка спільно з лінією, відповідною лінійному обмеженню:

```
[x1,x2] = meshgrid(-6:.1:6);
Z = (x1+4).^2+(x2-4).^3;
[C,h] = contour(x1,x2,Z);
clabel(C,h);
grid on , hold on
x3=2*x1-2;
plot(x1,x3,'-o')
```

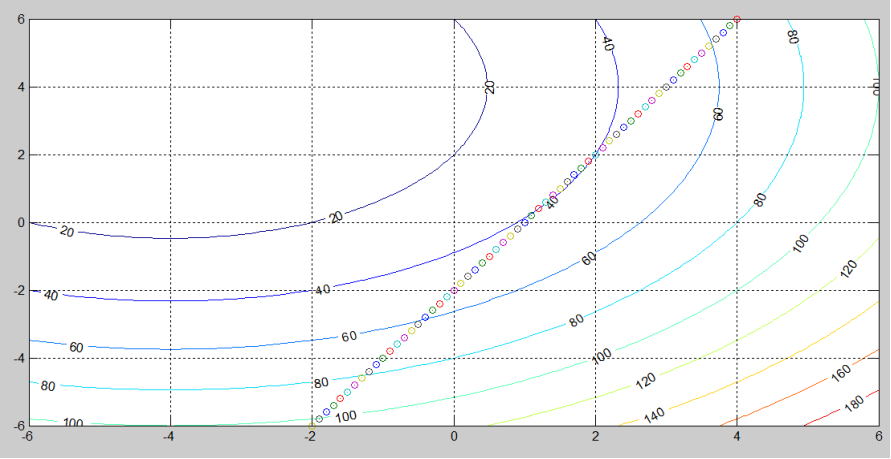


Рисунок 3.1 - Контурний графік функції і графік лінійного обмеження

Пошук мінімуму за допомогою функції fmincon.

Складання цільової функції:

function y = fun4c(x)
y =(x(1)+4)^2 + (x(2)-4)^2,

Складання обмежень.

Обмеження $2x_1 - x_2 \le 2$ виду AX<b:

A=[2, -1], b=[2].

Обмеження C X = d типу рівностей відсутні:

C=[ ], d=[ ].

Обмеження $x_1 \ge 0, x_2 \ge 0$ типу l≤X≤u:

l=[0,0],

верхні обмеження відсутні.

Варіанти вирішення завдання умовної мінімізації.

Варіант 1:

[x,fval] = fmincon('fun4c',[0,0],[2,-1],[2],[],[],[0,0])

x =

0 4.0000

fval =

16

Варіант 2:

[x,fval,exitflag,output] = fmincon('fun4c',[0,0],[2,-1],[2],[],[],[0,0])

x =

0 4.0000

```
fval =
    16
exitflag =
     1
output =
         iterations: 2
          funcCount: 9
       lssteplength: 1
           stepsize: 4.0000
          algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
      firstorderopt: 0
    constrviolation: 0
message: [1x791 char]
```

ЛАБОРАТОРНЕ ЗАВДАННЯ

Варіанти завдань із зазначенням функцій і обмежень, наведені в таблиці 3.1.

Таблиця 3.1 - Варіанти завдань

№	Функція	Інтервал по x, y	Початкова точка пошуку
1	$x^2 + 12\sin(y)^2$ $y - 2x \le -2,\ x \ge 0, y \ge 0$	[-4; 6], [-2, 2]	5, 1
2	$14\sin(x)^2 + y^2$ $y - 12x \le -2,\ x \ge 0, y \ge 0$	[-2; 2], [-6; 6]	1, 6
3	$x^2 + 4(\cos(y) - 1)^2$ $y - 2x \le -2,\ x \ge 0, y \ge 0$	[-6; 8], [-4; 4]	5, 2
4	$(x - y)^2 + 14\sin(x)^2$ $y - 12x \le -2,\ x \ge 0, y \ge 0$	[-2; 2], [-4; 8]	1, 5
5	$(x + y)^2 + 12\sin(x)^2$ $y - 6x \le -2,\ x \ge 0, y \ge 0$	[-2; 2], [-5; 6]	2, 6

6	$(x-y)^2+14\sin(y)^2$ $y-2x\le -2,\ x\ge 0, y\ge 0$	[-5; 8], [-2; 2]	6, 1
7	$(x-y)^2+3x^2+8(1-\cos(y))^2$ $y-2x\le -2,\ x\ge 0, y\ge 0$	[-4; 4], [-2; 2]	4, 2
8	$(x-y)^2+2y^2+16(1-\cos(x))^2$ $y-6x\le -2,\ x\ge 0, y\ge 0$	[-2; 2], [-4; 4]	2, 4
9	$2x^2+4(\cos(x-y)-1)^2$ $y-2x\le -2,\ x\ge 0, y\ge 0$	[-4; 6], [-4; 4]	5, 3
10	$2y^2+4(\cos(x-y)-1)^2$ $y-6x\le -2,\ x\ge 0, y\ge 0$	[-4; 4], [-4; 5]	3, 5

Потрібно:

1. Побудувати контурний графік функції спільно з лінійним обмеженням.
2. Вирішити задачу умовної мінімізації за допомогою функції fmincon. . Здійснити 2 варіанти звернень до функції, розглянутих у попередньому розділі.
3. На контурному графіку з лінією-обмеженням відзначити точку знайденого мінімуму.
4. Проаналізувати отримані результати і зробити висновки за досягнутою точністю і кількістю обчислень функції.
5. Дати письмові відповіді на контрольні питання.

ЗМІСТ ЗВІТУ

1. Мета роботи.
2. Формулювання завдання. Короткі теоретичні відомості.
3. Графічні представлення функції, обмежень, точки мінімуму.
4. Лістинг програм.
5. Результати обчислень.
6. Висновки.
8. Відповіді на контрольні запитання.

КОНТРОЛЬНІ ПИТАННЯ

1. Завдання умовної мінімізації функції багатьох змінних.
2. Суть прямих методів умовної мінімізації.
3. Суть непрямих методів умовної мінімізації.
4. Суть методу Лагранжа.
5. Суть методу штрафних функцій.
6. Види обмежень у функції *fmincon.*
7. Як в функцію *fmincon* вводиться опис обмежень $AX < b$, $C X = d$.
8. Як в функцію *fmincon* вводиться опис обмежень $l \le X \le u$.
9. Як в функцію *fmincon* вводиться опис нелінійних функцій - обмежень $g(X) \le 0$, $h(X) = 0$.
10. Основні варіанти звернення до функції *fmincon*.

ЛАБОРАТОРНА РОБОТА № 4

ГЕНЕТИЧНИЙ АЛГОРИТМ ОПТИМІЗАЦІЇ В СИСТЕМІ MATLAB

Мета роботи: ознайомитися з інтерфейсом і функціями програми Genetic Algorithm Toolbox системи Matlab, набути навичок роботи в системі.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Генетичні алгоритми (ГА) - це методи оптимізації, в яких моделюються еволюційні механізми, засновані на теорії природного відбору і генетичного успадкування.

В ГА використовується термінологія, запозичена з біології та генетики.

Оптимізація в ГА являє собою ітераційний процес, на кожній ітерації якого формується нове покоління популяції у вигляді сукупності нових особин. Під популяцією розуміється набір пробних рішень. Окреме пробне рішення, що характеризується вектором $X = x_1, x_2, .., x_n$, розглядається як особина.

Для оцінки якості пробних рішень використовується функція придатності (ФП). Кращому вирішенню відповідає більше значення ФП.

Оцінка особини - це значення ФП в точці n-мірного простору, що задається вектором X. Чим вище оцінка, тим краще пристосованість і живучість особини. Якщо спочатку оптимізаційна задача сформульована у вигляді задачі мінімізації функції f (X), то перехід до задачі максимізації функції придатності виконується по співвідношенню:

$$\max f(X) = \min\left[-f(X)\right].$$

Змінні вектора X кодуються, зазвичай у двійковому коді, і представляються у вигляді бітового рядка, довжиною, що забезпечує необхідну точність.

Пробне рішення, представлене в двійковому коді у вигляді сукупності бітових рядків, відповідних змінним, розглядається як хромосома, яка характеризує ознаки особини. Хромосома складається з генів у вигляді набору біт. Ген характеризує одна з ознак особини.

Сформована структура хромосоми визначає генотип особини, тобто точку в бітовому просторі, в свою чергу фенотип особини - це точка в просторі змінних X.

Для створення нового покоління застосовуються операції селекції, кросовера і мутації.

Селекція - відбір групи особин, які називаються батьками, для формування нащадків.

Кросовер (схрещування) складається в народженні нащадків від кожної батьківської пари. Суть його полягає в тому, що випадковим чином вибираються точки розриву, за якими батьківські хромосоми розриваються на частини. Потім батьківські хромосоми обмінюються частинами, утворюючи хромосоми нащадків.

Мутація - випадкова зміна одного або декількох розрядів коду хромосоми нащадка.

Нова популяція на кожній ітерації формується з нащадків кросовера і мутацій з кращими значеннями функції придатності, а також з елітних нащадків - копій батьків з найкращими з спостережуваних значеннями функції придатності.

ГА реалізований в системі MATLAB у вигляді функції **ga**, а також у вигляді інтерактивної програми Genetic Algorit**hm Toolbox.**

Інтерфейс програми Genetic Algorithm Toolbox. Для запуску програми слід в командному рядку MATLAB виконати команду ***gatool***, що викликає на екран основне вікно графічного інтерфейсу (див. Рис. 4.1).

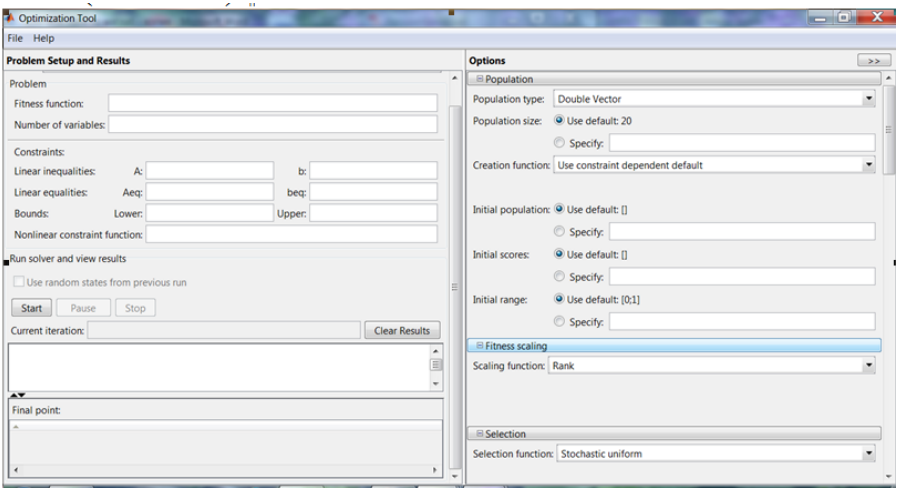


Рисунок 4.1 - Основне вікно графічного інтерфейсу ГА

Вікно на рисунку не містить довідкову систему, яка була закрита кнопкою <<.

У лівій частині вікна розташована панель **Problem Setup and Results**, справа - панель **Options.**

Панель Problem Setup and Results. Панель містить поля, в яких формулюється оптимізаційна задача і виводяться результати оптимізації:

- в полі *Fitness function* вказується функція, що оптимізується, у вигляді *@fitnessfun*, де *fitnessfun.m* - назва М-файлу, в якому попередньо слід описати функцію, що оптимізується;

- в полі *Number of variables* вказується число змінних вектора X;

В панелі *Constraints* задаються лінійні і нелінійні обмеження:

- в полі *Linear inequalities* задається лінійне обмеження нерівністю $A X \le b$, де A - матриця, b - вектор;

- в полі *Linear equalities* задаються лінійні обмеження рівністю $A X = b$, де A - матриця, b - вектор;

- в полі *Bounds* в векторному вигляді задаються нижнє і верхнє обмеження на змінні $L \le X \le U$;

- в полі *Nonlinear constraint function* задаються нелінійні функції обмежень.

Нелінійні обмеження задаються у вигляді вектор-функції обмежень типу нерівностей і вектор-функції обмежень типу рівностей. Відсутній тип обмежень відзначається квадратними дужками.

Наприклад, нехай задані нелінійні обмеження у вигляді нерівностей

$$2 + x_1 x_2 + x_1 - x_2 \le 0,$$
$$x_1 x_2 + 3 \le 0.$$

Цим обмеженням буде відповідати функція:

```
function [c, ceq] = ogran(x)
c = [2 + x(1)*x(2) + x(1) - x(2); -x(1)*x(2) + 3];
ceq = [];
```

В завданні обмеження ***ceq*** у вигляді рівностей відсутні, тому поля обмежень не заповнюються.

Панель *Run Solver* містить:

- керуючі елементи (кнопки *Start, Pause і Stop* для початку, припинення і повної зупинки роботи алгоритму);
- поля *Status and results,* в які виводяться результати роботи ГА.

На панелі *Status and results* відображаються такі елементи:

- повідомлення *GA terminated* - виконання алгоритму закінчено;
- значення функції придатності *(Fitness function)* в точці найкращого індивідуума для фінального покоління;
- причина зупинки алгоритму;
- координати кінцевої точки *(Final point).*

Під час роботи ГА можна змінити його параметри. Введення нових значень параметрів, що змінюють стан покоління, супроводжується повідомленням *Changes pending* на панелі *Status and Results*. При продовженні роботи зі зміненим станом покоління з'являється повідомлення *Changes applied.*

Панель Options. Панель призначена для настройки параметрів ГА. Панель (див. Рис. 4.2) містить 14 полів з назвами опцій. Активація опції здійснюється натисканням кнопки +, в результаті викликаються списки, що містять поля для введення і зміни параметрів.

Рисунок 4.2 - Опції настройки параметрів ГА

1. За допомогою кнопки **Population** (популяції) відкривається список опцій настройки популяцій.

Опція **Population type** визначає тип хромосом (спосіб представлення або кодування пробних рішень):

- *Double vector* (за замовчуванням) - вектор, складений з дійсних чисел формату *double* (64 біта, формат з плаваючою комою);

- *Bit string*- рядок двійкових символів;

- *Custom* - тип даних, визначається користувачем.

Опція **Population size** - розмір популяції (за замовчуванням дорівнює 20 хромосомам).

Опція **Creation function** (функція створення початкової популяції):

- *Uniform* - створює випадкову початкову популяцію з рівномірним розподілом (використовується за замовчанням);

- *Custom* - використання для користувача функції ініціалізації.

Опція ***Initial population*** (початкова популяція) дозволяє задати початкову популяцію *(Creation function* використовуватися в цьому випадку не буде). Введений в це поле масив повинен містити число рядків, який дорівнює кількості популяцій (*Population size)* і число стовпців, яке дорівнює кількості змінних *(Number of variables).*

Initial scores (початкові оцінки) - завдання оцінки особинам початкової популяції, за замовчуванням використовуються значення цільової функції.

Initial range (початковий діапазон) - завдання діапазону [Xmin; Xmax], в якому будуть генеруватися випадкові значення при створенні початкової популяції, за замовчуванням використовується діапазон [0, 1].

2. За допомогою кнопки ***Fitness Scaling*** (масштабування цільової функції) відкривається список способів перетворення початкових оцінок особин до вигляду, зручного для їх селекції:

- *Rank* (рангове масштабування) - за результатами первинних оцінок особин складається рейтинг, перший номер в рейтингу присвоюється кращій особини з найменшим значенням цільової функції (це масштабування використовується за замовчанням;

- *Proportional* (пропорційне масштабування) - задаються ймовірності пропорційно заданому числовому ряду для особин;

- *Top* (верхнє маштабування) - відбирається тільки частина особин з найкращими оцінками, їх число або частка від загальної кількості вказується у вигляді параметра;

- *Shift linear* (лінійний зсув) - є можливість вказати максимальну ймовірність найкращої особини.

- *Shift linear* - зміщене лінійне масштабування.

- *Custom* - спосіб масштабування визначається користувачем.

3. За допомогою кнопки ***Selection*** (опції селекції) відкривається список способів відбору батьківських особин на основі їх оцінок на етапі масштабування:

- *Stochastic uniform* (стохастична рівномірна селекція) - будується лінія, яка розбивається на відрізки довжиною, пропорційною оцінками батьків, потім здійснюється рух по цій прямій з постійним кроком, потрапляння кроку на відрізок лінії визначає вибір відповідного батька;

- *Remainder* (залишковий відбір) - в батьківську популяцію відбираються тільки найкращі хромосоми, у яких оцінки перевищують заданий поріг, інші хромосоми відкидаються;

- *Uniform* (рівномірний відбір) - батьки вибираються випадковим чином відповідно до заданого розподілу і з урахуванням кількості батьківських особин і їх ймовірностей

- *Roulette* (пропорційний відбір - «рулетка) - імітується рулетка, в якій розмір кожного сегмента пропорційний оцінці батька;

- *Tournament* (турнірна селекція) - випадково вибирається задане число особин, серед них на конкурсній основі вибираються кращі;

- *Custom* - дозволяє користувачеві задати власний спосіб селекції.

4. За допомогою кнопки Reproduction (опції відтворення) відкривається список способів створення нових особин:

- *Elite count* (число елітних особин) - вказується число кращих особин (за замовчуванням 2), які перейдуть в наступне покоління;

- *Crossover fraction* (частка особин для кросовера) - вказується частка особин (за замовчуванням 0,8), які будуть брати участь в схрещуванні.

5. За допомогою кнопки ***Mutation*** (опції мутації) відкривається список способів мутацій:

- *Gaussian* (гауссовий) - додається згідно з розподілом Гаусса невелике випадкове число до всіх компонентів кожного вектора-особини;

- *Uniform* (рівномірний) - відбирається випадковим чином частина компонентів хромосом, відібрані компоненти отримують випадкові значення з допустимого діапазону (використовується за замовчанням);

- *Adaptive feasible* - генерує набір напрямків в залежності від останніх найбільш вдалих і невдалих поколінь і з урахуванням накладених обмежень просувається уздовж всіх напрямків на різну довжину.

- *Custom* - використання способу мутації, що визначається користувачем.

6. За допомогою кнопки ***Crossover*** (опції схрещування) викликається список способів схрещування:

- *Scattered* (розподілений кросовер) - генерується випадковий двійковий вектор, який визначає компоненти хромосом батьківської пари, що переходять в дочірню хромосому (використовується за умовчанням);

- *Single point* (одноточковий кросовер) - генерується випадкове число - точку схрещування, по якій проводиться обмін компонентами пари хромосом - батьків для отримання нащадка;

- *Two point* (двоточковий кросовер) - генеруються два випадкових числа - точки схрещування для обміну компонентами хромосом батьківської пари при отриманні нащадка;

- *Intermediate* (проміжний кросовер) - нащадок P визначається за формулою $P = X + \alpha \nu (Y - X)$, де α - випадкове число з інтервалу [0, 1, ν - коефіцієнт схрещування, Y, X - батьківська пара;

- *Heuristic* (евристичний кросовер) - використовується евристична формула $P = Y + R (X - Y)$, за замовчуванням $R = 1,2$, тому нащадок розташовується близько до батьку чи матері X з найкращою оцінкою і на видаленні від батька Y з гіршого оцінкою;

- *Custom* - використання способу схрещування, що визначається користувачем.

7. За допомогою кнопки **Migration** (опції міграції) викликається список способів переміщення хромосом між субпопуляціями, які виявляються заданими в разі, якщо значення *Population size* являє собою вектор, з довжиною більше 1:

- *Direction* (напрямок міграції) - можливі варіанти: Forward- міграція вперед від попередньої субпопуляції до наступної (використовується за замовчанням), Both- міграція в обох напрямках;

- *Fraction* (частка) - визначає, скільки хромосом переміщаються між субпопуляціями (за замовчуванням - 0.2);

- *Interval* (інтервал) - визначає число поколінь, що визначає інтервал, з яким відбувається міграція (за замовчуванням - через кожні 20 поколінь).

8. За допомогою кнопки ***Algorithm settings*** (спеціальні параметри алгоритму) викликаються параметри, що визначають штраф при порушенні нелінійних обмежень:

- *Initial penalty* - початкове значення коефіцієнта штрафу;

- *Penalty factor* - множник коефіцієнта штрафу.

9. За допомогою кнопки ***Hybrid function*** (гібридні функції) викликається список функцій мінімізації, які можна застосувати для подальшого пошуку мінімуму після зупинки генетичного алгоритму.

10. За допомогою кнопки ***Stopping criteria*** (критерії зупину) викликається список ситуацій, при яких алгоритм робить зупинку:

- *Generations* - перевищення максимального числа поколінь;

- *Time limit* - перевищено ліміт часу на роботу алгоритму;

- *Fitness limit* - отримано з заданою точністю мінімальне значення ЦФ;

- *Stall generations* - перевищено кількість поколінь, які мало відрізняються;

- *Stall time limit* - перевищено ліміт часу, протягом якого не поліпшується оцінка ЦФ;

11. За допомогою кнопки ***Plot Functions*** (графічні функції) викликається список різновидів графіків, що ілюструють хід роботи алгоритму і показують досягнуті результати оптимізації.

12. За допомогою кнопки ***Output function*** можна включити висновок історії роботи алгоритму в окремому вікні (мітка *History to new window*) із заданим інтервалом поколінь, вказуються в полі *Interval,* а також можна вивести вихідну функцію, що задається користувачем в поле *Custom function.*

13. За допомогою кнопки ***Display to command window*** (відображення в командному вікні) викликається список даних, які відображаються в основному командному вікні при роботі алгоритму:

- *Off* - немає висновку в командне вікно;

- *Iterative* - висновок інформації по кожній ітерації працюючого алгоритму;

- *Diagnose* - висновок інформації по кожній ітерації і додаткових відомостях про можливі помилки та зміни параметрів алгоритму;

- *Final* - виводиться тільки причина зупинки і кінцевий результат.

14. За допомогою кнопки ***User function evaluation*** (оцінка функції користувача) відкривається список способів обчислення значень цільової функції і функцій обмежень: *in serial* - окремо; *vectorized* - одночасно; *in parallel* - паралельно в одному виклику.

Графічні можливості програми. Кнопкою ***Plot Functions*** відривається панель (див. Рис. 4.3) зі списком різновидів виведених на екран графіків (в деяких версіях графічного інтерфейсу програми ***gatool*** ця панель розташовується в лівій частині основного вікна):

- в полі *Plot interval* вказується число поколінь між черговим оновленням графіків;

- *Best fitness* (кращі значення) - графіки найкращого і середнього значення функції придатності для кожного покоління;

- *Best individual* - графік з номером і значенням кращої хромосоми в поколінні з найменшим значенням цільової функції;

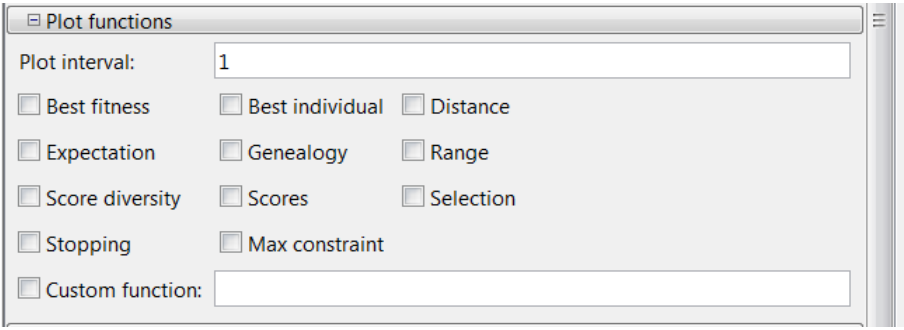


Рисунок 4.3 - Поле зі списком графіків

- *Distance* - графік залежності відстані між хромосомами популяції, що характеризує величину простору пошуку;

- *Expectation* (очікування) - графік придатності для кожного покоління, що характеризує очікуване число нащадків хромосоми в залежності від оцінки її придатності;

- *Genealogy* - графік, що показує походження кожної хромосоми (на екрані елітний відбір - чорний, схрещування - синій, мутація - червоний);

- *Range* - найкраще, найгірше і середнє значення придатності хромосом популяції;

- *Score diversity* - гістограма розподілу значень придатності хромосом;

- *Scores* - значення придатності для кожної хромосоми популяції;

- *Selection* - гістограма вибору батьківських хромосом;

- *Stopping* - відображення рівня виконання критеріїв зупинки ГА;

- *Custom* - відображення на графіку функції, зазначеної користувачем.

ЛАБОРАТОРНЕ ЗАВДАННЯ

В ході виконання роботи потрібно:

1. Виконати мінімізацію функцій, заданих в лабораторних роботах №1, 2 та 3.

2. Провести аналіз роботи ГА при мінімізації функції з лабораторної роботи №3:

Розмір популяції прийняти рівним 6, число поколінь прийняти рівним 10.

Мінімізувати функцію і вивести *графіки Best fitness і Best individual* для всіх видів кросовера за винятком призначеного для користувача. Графіки для двох видів кросовера, що забезпечили найкраще і найгірше рішення, привести в звіті;

Перебором способів селекції і мутації знайти найкращі рішення. Для обраних способів селекції, мутації і кросовера, що забезпечили найкращі рішення привести в звіті весь комплект графіків на панелі *Plots,* за винятком графіка *Max constraint.*

3. Проаналізувати отримані результати і зробити висновки за досягнутою точністю і кількістю обчислень функції.

7. Дати письмові відповіді на контрольні питання.

Зміст звіту

1. Мета роботи.
2. Формулювання завдання. Короткі теоретичні відомості.
3. Файли з цільовими функціями і обмеженнями.
4. Результати мінімізації функцій (перший пункт лабораторного завдання) з роздруківкою полів виведення результатів панелі *Run Solver.*
5. Результати мінімізації функцій (другий пункт лабораторного завданні).
6. Графіки способів кросовера, що забезпечили найгірше і найкраще рішення задачі мінімізації, з роздруківкою полів панелі *Run Solver*
7. Комплект графіків для обраних способів селекції, мутації, кросовера, з роздруківкою полів панелі *Run Solver*
8. Висновки по роботі.
9. Відповіді на контрольні запитання.

Контрольні питання

1. Що розуміється в ГА під функцією придатності?
2. Що розуміється в ГА під хромосомою і геном?
3. Що розуміється в ГА під громадянами та особиною?
4. Що розуміється в ГА під генотипом і фенотипом особини?
5. Що розуміється в ГА під селекцією?
6. Що розуміється в ГА під кросовером?
7. Що розуміється в ГА під мутацією?
8. Як формується нова популяція?
9. Які панелі є в графічному вікні програми gatool?
10. Призначення полів на панелі Problem Setup and Results.
11. Призначення полів на панелі Options.

12. Графічні можливості програми gatool.

ЛАБОРАТОРНА РОБОТА № 5

ПОБУДОВА НЕЧІТКОЇ ЕКСПЕРТНОЇ СИСТЕМИ

Мета роботи: придбання практичних навичок роботи з інтерактивною програмою Fuzzy Logic Toolbox в середовищі MATLAB на прикладі побудови нечіткої експертної системи.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Нечітка множина. Нехай X є базова множина (універсіум), x - його елемент. Тоді нечіткою підмножиною $\tilde{A}$ (або просто нечіткою множиною - НМ) множини X називається множина впорядкованих пар

$$\left\{x, \mu_{\tilde{A}}(x)\right\}, \forall x \in X,$$

де - $\mu_{\tilde{A}}(x)$ - функція приналежності нечіткої множини, значення якої обмежені відрізком [0,1].

Функція приналежності приписує кожному елементу $x \in X$ ступінь його приналежності до нечіткій множині $\tilde{A}$. При цьому можна виділити три випадки:

$\mu_{\tilde{A}}(x) = 1$ означає повну приналежність елемента *x* до нечіткій множині, $\tilde{A}$, тобто $x \in \tilde{A}$.

$\mu_{\tilde{A}}(x) = 0$ означає відсутність приналежності елемента x до нечіткої множини $\tilde{A}$.

$0 < \mu_{\tilde{A}}(x) < 1$ означає часткову приналежність елемента до нечіткої множини $\tilde{A}$.

Якщо X - кінцева множина, тобто $X = \{x_1, x_2, \cdots\cdots x_n\}$, то нечітка множина $\tilde{A} \subseteq X$ часто записується у вигляді:

$$\tilde{A} = \frac{\mu_{\tilde{A}}(x_1)}{x_1} + \frac{\mu_{\tilde{A}}(x_2)}{x_2} + \cdots \frac{\mu_{\tilde{A}}(x_n)}{x_n} = \sum_{i=1}^{n} \frac{\mu_{\tilde{A}}(x_i)}{x_i}.$$

Тут риса і знак плюс використовуються не як знаки арифметичних дій, а як роздільники для наочності запису.

Якщо X множина з нескінченним числом елементів, то НМ $\tilde{A} \subseteq X$ символічно записується у вигляді

$$A = \int_x \frac{\mu_A(x)}{x} dx.$$

Нечіткі числа - нечіткі змінні, які визначені на множині дійсних чисел X з функцією приналежності $\mu_{\tilde{A}}(x) \in [0,1]$. Наприклад, НМ, що відповідає поняттю «множина натуральних чисел, близьких числу 6» можна визначити наступним чином:

$$\tilde{A} = \frac{0,2}{3} + \frac{0,6}{4} + \frac{0,8}{5} + \frac{1,0}{6} + \frac{0,8}{7} + \frac{0,6}{8} + \frac{0,2}{9}$$

Лінгвістична змінна. Змінна, значеннями якої можуть бути слова або словосполучення природної або формальної мови, називається лінгвістичною змінною.

Наприклад, лінгвістичній змінній «температура в будинку» можна присвоїти наступні значення: «холодно», «комфортно», «жарко», Множина всіх значень лінгвістичної змінної називається **терм-множиною**, а окреме значення множини називається **термом.** Кожному значенню лінгвістичної змінної (терму) відповідає певна нечітка множина зі своєю функцією приналежності. Так, лінгвістичному значенню «холодно» може відповідати функція приналежності $\mu_{\text{холодно}}(x)$.

Лінгвістична змінна задається набором з п'яти складових:

{x, T, X,G,M},

де x - ім'я змінної;

T - терм-множина, кожен елемент якої, тобто терм, представляється як нечітка множина на базовій множині X;

G - синтаксичні правила, які породжують назву термів;

M - семантичні правила, які визначають функції належності нечітких термів, породжених синтаксичними правилами.

Приклад. Лінгвістичну змінну з ім'ям x "температура в будинку" можна зрівнити з:

- X=[5, 30] - базова множина у вигляді діапазону можливих температур;

- T={"холодно", "комфортно", "жарко} - терм-множина з функціями приналежності, наприклад:

$$\mu_{\text{холодно}}(x) = \frac{1}{1+\left(\frac{x-10}{7}\right)^{12}},$$

$$\mu_{\text{комфортно}}(x) = \frac{1}{1+\left(\frac{x-20}{4}\right)^{6}},$$

$$\mu_{\text{жарко}}(x) = \frac{1}{1+\left(\frac{x-30}{6}\right)^{8}};$$

- G - синтаксичні правила, що породжують нові терми з використанням квантіфікаторів «не», «дуже», «більш-менш" та інші;

- M - семантичні правила для обчислення функцій приналежності нових термів.

Приклади квантіфікаторів наведені в таблиці 5.1.

Таблиця 5.1 Приклади квантіфікаторів

Квантіфікатор	Функція приналежності

не	$1-\mu_t(x)$
дуже	$\mu_t(x)^2$
більш-менш	$\sqrt{\mu_t(x)}$

Графіки функцій приналежності термів "холодно", "комфортно",, "жарко", а також термів "більш-менш комфортно" і "дуже жарко" лінгвістичної змінної "температура в будинку" показані на рис. 5.1.

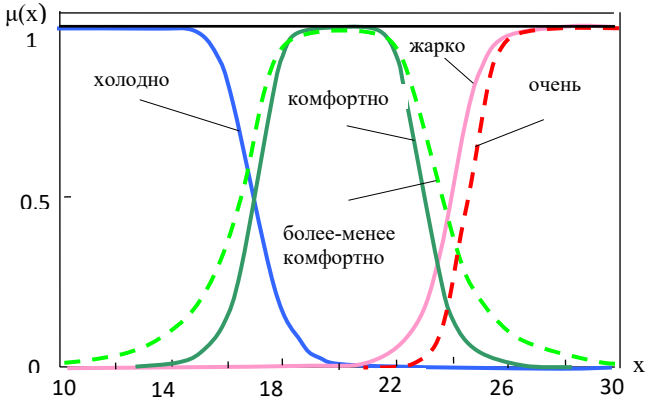


Рисунок 5.1 - Графіки функцій належності термів лінгвістичної змінної «температура в будинку»

Види функцій приналежності. Формально немає ніяких обмежень на вибір виду функцій приналежності. На практиці вважають за краще використовувати функції у вигляді простих аналітичних виразів. Наприклад, в системі Fuzzy Logic Toolbox реалізовані трикутна, трапецеїдальна, гауссова та інші функції.

Нечітка логічна система. На рис. 5.2 представлена структурна схема алгоритму нечіткої логічної системи.

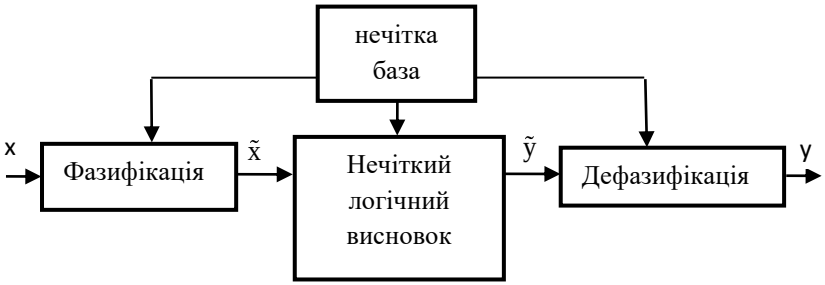


Рисунок 5.2 - Схема алгоритму системи нечіткого виведення

Алгоритм нечіткої логічної системи складається з наступних процедур:

1. Фазифікації вхідних змінних.
2. Нечіткий логічний висновок.
3. Дефазифікація вихідних змінних.

Нечітка логічна система оперує з лінгвістичними змінними, значення яких визначається термами-множинами. Логічні операції здійснюються з використанням бази знань, яка складається експертами. База знань нечіткої логічної системи полягає:

- з функцій приналежності, що показують ступінь відповідності реальних величин поняттям, визначеним термами-множинами,
- з продукційних правил, що встановлюють логічні зв'язки між вхідними та вихідними термами-множинами.

Процедура фазифікації - переклад точних значень вхідних змінних x в значення лінгвістичних змінних $\tilde{x}$.

Після завершення етапу фазифікації для всіх вхідних змінних повинні бути визначені конкретні значення функцій приналежності по кожному з лінгвістичних термів.

Нечіткий логічний висновок - перетворення терм-множин вхідних лінгвістичних змінних $\tilde{x}$ в терм-множини вихідних лінгвістичних змінних $\tilde{y}$, що здійснюється з використанням **нечіткої бази знань**, яка складається з продукційних правил, закладених експертами.

Умови і висновки продукційних правил формулюються на основі нечітких висловлювань щодо значень вхідних і вихідних лінгвістичних змінних.

Види нечітких висловлювань:

1. Висловлювання "x є A", де x - найменування лінгвістичної змінної, A - її значення, якому відповідає окремий лінгвістичний терм з базової терм-множини T лінгвістичної змінної $\tilde{x}$;

2. Висловлювання "x є ΔA", де Δ - квантіфікатор, відповідний таким словами, як: «*дуже*», «*більш*» або «*менш*», «*багато більше*» і іншим, які можуть бути отримані з використанням процедур G і M даної лінгвістичної змінної;

3. Складові висловлювання, побудовані з висловлювань видів 1 і 2 і нечітких логічних операцій в формі зв'язок: "І", "АБО", "ЯКЩО-ТО", "НЕ".

База правил нечіткої логічної системи являє собою кінцеву множину $П_1$, $П_2$, ... , $П_n$ правил нечітких продукцій, узгоджених щодо використовуваних в них лінгвістичних змінних. Найбільш часто база правил представляється в наступній формі:

$П_1$: якщо x є A_1, то y є B_1;
$П_2$: якщо x є A_2, то y є B_2;
……
$П_n$: якщо x є A_n, то y є B_n.

Тут нечітке висловлювання "x є A_k" є умова (передумова) даного правила нечіткої продукції, а нечітке висловлювання "y є B_k" - нечіткий висновок даного правила; A і B - нечіткі множини, які визначені на вхідний $\tilde{x}$ і вихідний $\tilde{y}$ лінгвістичних змінних.

Нечітке причинне відношення умови правила і висновку правила є нечітке відношення:

$$R_{A \to B} = A \to B$$

Тут «→» позначає нечітку імплікацію, під якою розуміється логічна операція перетворення відносини двох нечітких висловлювань A і B у нове нечітке висловлювання "якщо A, то B", ступінь істинності якого визначається функцією приналежності

$$\mu_R(x,y) = \mu_{A\to B}(x,y),$$

приймаючої будь-яке значення між 0 і 1.

Запропоновано різні способи знаходження функції $\mu_R(x,y)$ на основі відомих функцій приналежності $\mu_A(x), \mu_B(y)$, наприклад:

- правило типу minimum (правило Мамдані)

$$\mu_R(x,y) = \mu_{A\to B}(x,y) = \mu_A(x) \wedge \mu_B(y) = \min[\mu_A(x), \mu_B(y)];$$

- правило типу «перемноження» (правило Ларсена)

$$\mu_R(x,y) = \mu_{A\to B}(x,y) = \mu_A(x) \cdot \mu_B(y).$$

Тут символом « $\wedge$ » позначена операція взяття мінімуму.

В результаті обробки всіх правил нечіткої продукції по кожній вихідній змінній утворюється сукупність результуючих нечітких множин.

Композиція (об'єднання) всіх нечітких підмножин, призначених кожній змінній виводу. Композиція, як правило, здійснюється за допомогою логічних операцій *max* (максимум) або *sum* (сума). При операції *max* висновок результуючої нечіткої підмножини формується як поточковий максимум за всіма нечіткими підмножини, а при операції сума - як поточкова сума.

Процедура дефазифікація - це приведення до чіткості, тобто перетворення нечіткої множини по кожній змінній виведення в чітке число.

Використовуються різні способи дефазифікації, наприклад, в програмі Fuzzy Logic Toolbox закладені п'ять способів. На практиці найчастіше застосовується метод центру тяжіння (центроїдний метод):

$$z0 = \frac{\sum_{i=1}^{n} z_i \mu_c(z_i)}{\sum_{i=1}^{n} \mu_c(z_i)} \text{ - для дискретного варіанту;}$$

$$z0 = \frac{\int_{z\min}^{z\max} z\mu_c(z)dz}{\int_{z\min}^{z\max} \mu_c(z)dz} \text{ - для безперервного варіанту.}$$

Алгоритм нечіткого логічного висновку. Існуючі алгоритми нечіткого логічного висновку розрізняються способами імплікації, композиції, дефазифікації, що, природно, призводить до різних кінцевих результатів.

В програмі Fuzzy Logic Toolbox закладені алгоритми нечіткого логічного висновку Мамдані та Сугено. За замовчуванням використовується алгоритм Мамдані.

Алгоритм Мамдані нечіткого логічного висновку. Нехай базу знань становлять два продуктивні правила П1, П2, записані в такій формі:

$П_1$: якщо x є A_1 і y є B_1, то z є C_1;
$П_2$: якщо x є A_2 і y є B_2, то z є C_2;

де x і y- імена вхідних змінних;
z- ім'я змінної виводу;
A_1, A_2, B_1, B_2, C_1, C_2- деякі множини, задані функціями приналежності

$$\mu_{A_1}(x),\ \mu_{A_2}(x),\ \mu_{B_1}(y),\ \mu_{B_2}(y),\ \mu_{C_1}(z),\ \mu_{C_2}(z).$$

Необхідно визначити чітке значення z_0, відповідне чітким значенням x_0, y_0.

Ілюстрація до алгоритму приведена на рис. 5.3.

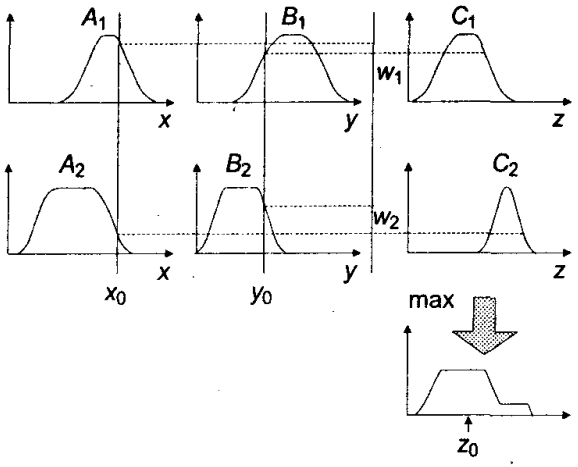


Рисунок 5.3 - Схема нечіткого виведення по Мамдані

Етапи алгоритму:

1. Введення нечіткості. Знаходяться ступені істинності для передумов кожного правила $\mu_{A_1}(x_0)$, $\mu_{A_2}(x_0)$, $\mu_{B_1}(y_0)$, $\mu_{B_2}(y_0)$.

2. Логічний висновок. Використовується формула нечіткої імплікації:

$$\mu_{A\to B}(x,y)=\mu_A(x)\wedge\mu_B(y)=\min[\mu_A(x),\mu_B(y)].$$

Відповідно до цієї формули для передумов кожного правила знаходяться рівні «відсікання» в результаті застосування операції minimum:

$$w_1=\mu_{A_1}(x0)\wedge\mu_{B_1}(y0),$$
$$w_2=\mu_{A_2}(x0)\wedge\mu_{B_2}(y0).$$

Потім визначаються усічені функції приналежності змінної виходу

$$\mu_{C_1'}(z) = \left(w_1 \wedge \mu_{C_1}(z)\right),$$
$$\mu_{C_2'}(z) = \left(w_2 \wedge \mu_{C_2}(z)\right).$$

3. Композиція. Проводиться об'єднання знайдених усічених функцій з використанням операції max, далі позначається як «$\vee$», що призводить до отримання підсумкового нечіткого підмножини для змінної виводу z з функцією приналежності

$$\mu_{C_\Sigma}(z) = \mu_{C_1'}(z) \vee \mu_{C_2'}(z) = \left(w_1 \wedge \mu_{C_1}(z)\right) \vee \left(w_2 \wedge \mu_{C_2}(z)\right).$$

4. Приведення до чіткості. Проводиться для знаходження z_0, наприклад, центроїдним методом.

ПРИКЛАД ПОБУДОВИ НЕЧІТКОЇ ЕКСПЕРТНОЇ СИСТЕМИ.

Побудова експертної системи на прикладі задачі про чайові. Завдання може бути описано наступними умовами:

1. Якщо обслуговування погане або їжа несмачна, то чайові - малі.
2. Якщо обслуговування хороше, то чайові - середні.
3. Якщо обслуговування відмінне або їжа смачна, то чайові - щедрі.

Якість обслуговування та їжі будемо оцінювати за 10-бальною шкалою (0- найгірша оцінка, 10 - найкраща).

Припустимо, що малі чайові складають 5% вартості обіду, середні - 15%, щедрі -25%.

Даної інформації досить для проектування нечіткої експертної системи. Ця система буде мати 2 входи - «сервіс» і «їжа», один вихід - «чайові», три правила типу «якщо ..., то ...» і по 3 значення, відповідно, 0 балів, 5 балів, 10 балів і 5% , 15%, 25% для центрів функцій приналежності входів і виходу.

Побудуємо систему, використовуючи алгоритм висновку Мамдані.

1. **Складання структури експертної системи**. Командою fuzzy запускається редактор системи нечіткого виведення Fuzzy Inference System (FIS-редактор).

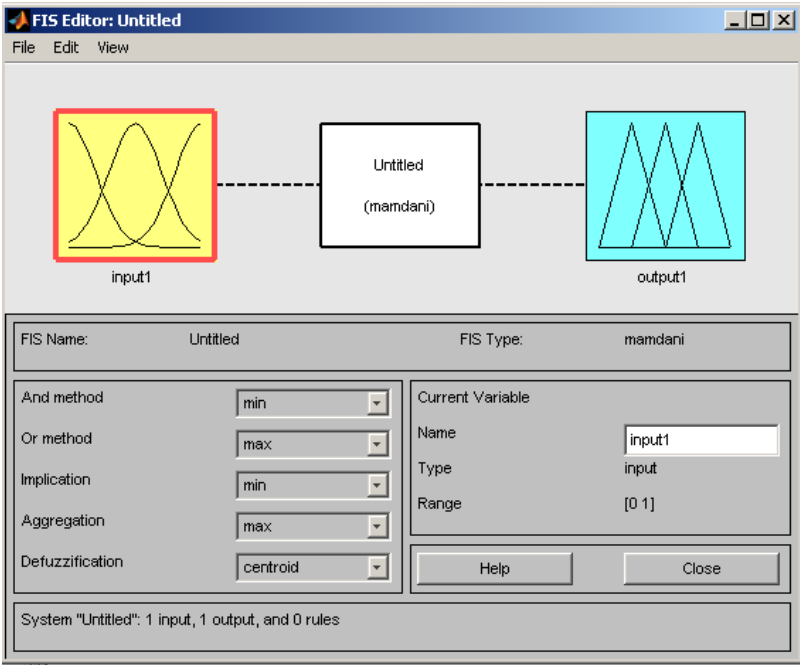


Рисунок 5.4 - Графічний інтерфейс редактора FIS

Для нечіткого логічного висновку скористаємося алгоритмом висновку Мамдані, тому не вносимо зміни в центральному білому блоці, де вписано назву алгоритму за замовчуванням - mamdani.

В системі повинно бути два входи і один вихід. Через пункт меню Edit / Add input додаємо в систему другий вхід - у вікні редактора з'являється другий жовтий блок з ім'ям input2.

Надаємо назви входів і виходу. Одноразовим клацанням лівої кнопки миші по блоку input1 активізуємо його. В поле його імені (праворуч в нижній частині вікна редактора) записуємо «сервіс». Завершується введення нового імені натисканням клавіші Enter.

Аналогічним чином встановлюється ім'я «їжа» блоку input2 і «чайові» - вихідного блоку output1.

Надаємо ім'я створюваній експертній системі, наприклад, «tip» (по-англійськи - чайові) через пункт меню File / Save to workspace as ... (Зберегти в робочому просторі як ....).

Вікно редактора після введених змін показано на рис.5.5.

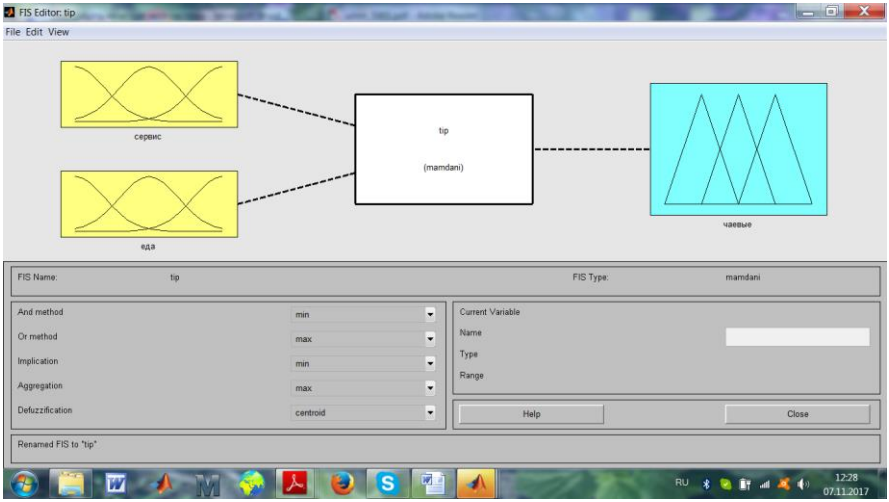


Рисунок 5.5 - Вид вікна редактора після завдання структури системи

В проектованій системі параметри, перераховані в лівій нижній частині вікна редактора, взяті за замовчуванням. За допомогою цих параметрів можна вибрати метод імплікації, композиції і дефазифікації. На будь-якому етапі проектування нечіткої системи в параметри можна внести необхідні корективи.

2. **Завдання функцій приналежності**. Виклик програми-редактора функцій належності можна здійснити подвійним клацанням лівої кнопки миші по блоку входу або виходу.

Подвійне клацання по блоку «сервіс» відкриває вікно редактора функцій належності. У полях Range і Display Range встановлюємо діапазон зміни і відображення цієї змінної - від 0 до 10 (балів), підтверджується клавішею Enter. Викликаємо через пункт

меню Edit / Add MF діалогове вікно і задаємо в ньому функції приналежності гауссового типу (gaussmf) із загальним числом 3. Після

натискання кнопки ОК відбудеться повернення у вікно редактора функцій належності. Активація функцій у вікні редактора виконується одноразовим клацанням по відповідному графіку (колір графіка змінюється на червоний). Мітки на графіку дозволяють змінювати його положення і розмах. Ім'я графіку можна привласнити в полі Name, а в поле Type змінити, при необхідності, тип функції приналежності. Послідовно активувавши три графіка змінної «сервіс», задамо їм імена «поганий», «хороший» і «відмінний». На рис. 5.6 представлено вид вікна редактора функцій належності з заданими функціями належності для змінної «сервіс».

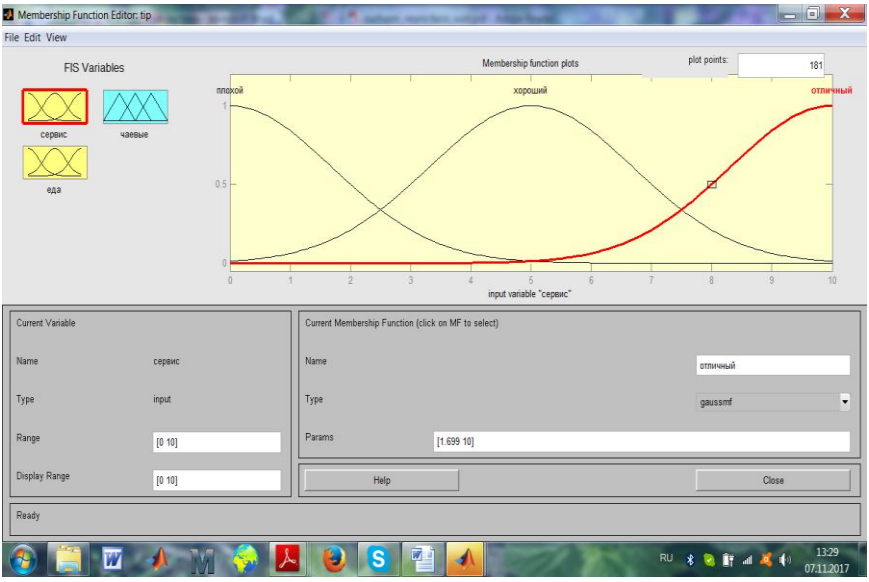


Рисунок 5.6 - Вид вікна редактора функцій належності для змінної «сервіс»

Клацанням лівої кнопки миші по значку «їжа» увійдемо в вікно редагування функцій приналежності для цієї змінної. Задаємо спочатку діапазон її зміни від 0 до 10, а потім, вступаючи як раніше,

задаємо дві функції приналежності трапецієподібні з параметрами, відповідно, [0 0 у середньому 1 3] і [7 9 10 10] і іменами «несмачна» і «смачна» .

Для вихідної змінної «чайові» потрібно вказати спочатку діапазон зміни - від 0 до 30, потім поставити три функції приналежності трикутної форми з іменами «малі», «середні», «щедрі» і з параметрами [0 0 10], [10 15 20], [20 30 30], відповідно.

2 **Конструювання правил.** Редактор правил продукцій системи нечіткого виведення може бути відкритий за допомогою головного меню редактора FIS командою меню Edit> Rules.

Редактор правил дозволяє задати логічні зв'язки для умов правила (перемикач Connection) і вага правила (поле введення Weight). Кнопки в нижній частині графічного інтерфейсу редактора правил служать для видалення виділеного у вікні правила (Delete rule), додавання створеного правила в базу правил (Add rule) і внесення змін в виділене в вікні правило (Change rule). В правому нижньому куту знаходяться кнопки виклику вбудованої довідкової системи MATLAB (Help) і кнопка закриття редактора правил (Close).

Для завдання правила необхідно виділити назви термів вхідних і вихідних лінгвістичних змінних, вказати логічний зв'язок між ними за допомогою перемикача Connection і додати правило в вікно правил кнопкою Add rule. Якщо деякий терм не входить в правило, то для нього слід вибрати значення "none". Якщо в умові правила використовується логічне заперечення деякого терма, то для цього терма слід зазначити відповідний прапорець з міткою "not".

Для проектованої системи набір правил буде виглядати наступним чином:

1. If (сервіс is поганий) or (їжа is несмачна) then (чайові is малі) (1)
2. If (сервіс is хороший) then (чайові is середні) (1)
3. If (сервіс is відмінний) or (їжа is смачна) then (чайові is щедрі) (1)

Закриємо вікно редактора правил і повернемося у вікно FIS-редактора. Побудову системи закінчено і можна почати експерименти по її вивченню.

4. Тестування спроектованої системи. Для перевірки роботи створеної системи потрібно відкрити (через пункт меню View / View rules ...) вікно перегляду правил і встановити значення змінних: сервіс

= 0 (тобто нікудишній), їжа = 10 (тобто смачна) . Отримана відповідь: чайові = 15 (тобто середні). Можна перевірити й інші варіанти.

Пункт меню *View / View surface* дозволяє побудувати тривимірний графік поверхні відгуку. За допомогою мишки графік можна повертати в різні боки. У відкритому графічному вікні, змінюючи імена змінних в полях введення X (input) і Y (input)), можна задати перегляд одновимірних залежностей, наприклад, «чайових» від «їжі». Для цього в поле введення X вказати «їжа», а в поле введення Y вибрати «none».

ЛАБОРАТОРНЕ ЗАВДАННЯ

Треба розробити нечітку експертну систему (ЕС). Завдання на розробку:

1. ЕС підготовки до іспиту.
2. ЕС по вибору теми для магістерської роботи.
3. ЕС щодо вибору місця роботи після закінчення університету.
4. ЕС, що рекомендує конфігурацію персонального комп'ютера.
5. ЕС пошуку несправностей в комп'ютері.
6. ЕС щодо вибору системи захисту інформації.
7. ЕС оцінки якості програмного забезпечення.
8. ЕС по вибору смартфона.
9. ЕС по вибору автомобіля.
10. ЕС, приймаюча рішення про формування бюджету сім'ї.

Вибір варіанту проводиться відповідно до бажання студента, на підставі його знань про предметну область.

В ході виконання роботи потрібно:

1. Ознайомитися з основними визначеннями нечіткої логіки.
2. Ознайомитися з можливостями і принципом роботи програми Fuzzy Logic Toolbox.
3. Розробити згідно з даними варіанту нечітку експертну систему.
4. Варіюючи способами імплікації, композиції (агрегації) і дефазафікації, простежити зміни точних значень вихідних змінних

при фіксованих значеннях вхідних змінних. Виконати не менше трьох варіантів розрахунків.

5. Письмово відповісти на контрольні питання.
6. Оформити звіт про виконану роботу.

ЗМІСТ ЗВІТУ

1. Мета роботи.
2. Формулювання завдання. Короткі теоретичні відомості.
3. Експертна система.
4. Результати моделювання.
5. Висновки по роботі.
6. Відповіді на контрольні запитання.

КОНТРОЛЬНІ ПИТАННЯ

1. Дайте визначення нечіткої множини. Яка його основна відмінність від звичайного (чіткого) ?
2. Що таке функція приналежності?
3. Як зазвичай записуються нечіткі множини?
4. Наведіть приклад нечіткої множини, відповідного поняттю «множина натуральних чисел, близьких числа 7».
5. Дайте визначення лінгвістичної змінної.
6. Складіть терм-множину для лінгвістичної змінної з ім'ям х "температура в будинку".
7. Поясніть на малюнку породження нових термів лінгвістичної змінної «температура в будинку» шляхом використання квантіфікаторов.
8. Схема алгоритму системи нечіткого виведення.
9. Що таке фазифікації?
10. Що таке нечіткий логічний висновок?
11. Наведіть приклад бази правил нечіткої логічної системи.
12. Що таке нечітка імплікація?
13. Що таке дефазифікація?

14. Центроїдний метод дефазифікації.
15. Метод логічного висновку в алгоритмі Мамдані.
16. Метод композиції в алгоритмі Мамдані.

ПЕРЕЛІК ПОСИЛАНЬ

1. Лазарєв Ю. Ф. Початок програмування в середовищі MatLAB: Навчальний посібник. - К.: НТУУ "КПІ", 2003. - 424 с .: іл.

2. Чуйко Г. П. Математичне моделювання систем і процесів : [навчальний посібник] / Г. П. Чуйко, О. В. Дворник, О. М. Яремчук. – Миколаїв : Вид-во ЧДУ імені Петра Могили, 2015. – 244 с.

3. Кононюк А.Ю. Нейроні мережі і генетичні алгоритми – К.:«Корнійчук», 2008. – 446 с.

4. Слепцов А.І., Зоденкампм М.А. Прийняття рішень в складних системах – К.: НПУ імені М. П. Драгоманова, 2007. – 148 с.