УДК 681.5.015:004.681

ІВАНЧЕНКО Євгенія Вікторівна,
КОЗЮРА Валерій Дмитрович,
ХОРОШКО Володимир Олексійович

ОЦІНКА ПОКАЗНИКІВ ЯКОСТІ СИСТЕМИ УПРАВЛІННЯ ЗАХИСТОМ ІНФОРМАЦІЙНИХ РЕСУРСІВ

Частина 1. Визначення оцінок показників якості систем управління

Постановка проблеми. Завдання визначення оптимальних показників якості систем управління захистом інформаційних ресурсів автоматизованих систем є однією з найважливіших проблем проектування комплексних систем захисту інформації. Це зумовлено складністю подібних систем, наявністю безлічі варійованих параметрів, складністю обчислення показників якості. Крім того, визначувані показники якості повинні не лише забезпечувати оптимальність цільової функції, але і стійкість функціонування системи захисту в широкому діапазоні зовнішніх несприятливих впливів. У роботі [1] пропонується як модель системи управління захистом використовувати положення теорії систем автоматичного управління, в якій показники задаються у вигляді інтегральних квадратичних оцінок (ІКО) помилки управління [2; 3]. Проблема полягає в тому, що існуючі методи обчислення ІКО [4–7] не враховують помилки визначення показників якості, а також векторний характер цих показників.

Метою роботи є рішення завдань (розробка алгоритмів), що становлять проблему оптимізації стійких систем управління захистом при використанні векторних цільових функцій.

Виклад основного матеріалу. ІКО помилки управління має вигляд

$$I = \int_0^\infty e^2(t)dt. \tag{1}$$

Помилку управління e(t) можна визначити як зважену суму похідних відхилення z(t) = y(∞) – y(t) керованої величини y(t) від усталеного значення y(∞) = 1 за допомогою вагових коефіцієнтів $w_k, k = \overline{0,l};\ w_0 = 1$:

$$e(t) = \sum_{k=0}^{l} \omega_k \cdot z^{(l-k)}(t). \qquad (2)$$

Нехай керованій величині $y(t)$ відповідає передавальна функція

$$W(s) = \frac{\beta(s)}{\alpha(s)}, \alpha(s) = \sum_{i=0}^{n} \alpha_i s^{n-i};\ \beta(s) = \sum_{j=0}^{m} \beta_j s^{m-j};\ \alpha_n = \beta_m. \qquad (3)$$

Перетворення Лапласа $z_k(s)$ для похідних $z^{(k)}(t)$ у виразенні (2) виходить шляхом застосування теореми про диференціювання оригіналу з початковими умовами $z(0) = 1;\ z^{(k)}(0) = 0, k = \overline{1,l}$:

$$z_0(s) = \frac{1 - W(s)}{s};\ z_k(s) = -s^{k-l} \cdot W(s), k = \overline{1,l}; l \le n - m.$$

Перетворюючи рівність (2) за Лапласом і вводячи багаточлен з ваговими коефіцієнтами

$$\omega(s) = \sum_{k=0}^{l} \omega_k \cdot s^{l-k}, \qquad (4)$$

отримаємо зображення помилки:

$$E(s) = \frac{1 - W(s) \cdot \omega(s)}{s}. \qquad (5)$$

Враховуючи вирази (3) для передавальної функції і визначаючи через

$$\delta(s) = \frac{\alpha(s) - \beta(s) \cdot \omega(s)}{s}, \qquad (6)$$

зображення помилки представимо у вигляді відношення двох багаточленів:

$$E(s) = \frac{\delta(s)}{\alpha(s)}. \qquad (7)$$

У загальному випадку помилку можна визначити як різницю еталонної функції $y_e(t)$ і керованої величини $y(t)$: $e(t) = y_e(t) - y(t)$.

Передавальну функцію еталонної функції задамо як $W_e(s) = 1/\omega(s)$, тоді зображення помилки матиме вигляд

$$E(s) = \frac{W_e(s) - W(s)}{s} \qquad (8)$$

Після перетворень приходимо до рівності $E(s) = \delta(s)/\alpha(s)/\omega(s)$. Для визначення еталонної функції виберемо стандартну перехідну функцію $y_0(t)$ бажаної форми з відомими значеннями часу встановлення τ_0 і коефіцієнтами передавальної функції $W_0(s) = 1/\gamma(s)$ [2; 3]:

$$\gamma(s) = \sum_{k=0}^{i} \gamma_k \cdot s^{i-k}, \gamma_0 = 1.$$

Задаючи значення часу встановлення еталонної функції τ_e, вирахуємо вагові коефіцієнти $\omega_k = \mu^{l-k} \cdot \gamma_k, k = \overline{0, l}$, де $\mu = \tau_e/\tau_0$. За цими формулами і виразами (6), (7) складемо алгоритм формування зображення помилки.

Алгоритм 1. Вхідні параметри: τ_e – час встановлення еталонної функції, τ_0 – час встановлення стандартного процесу, γ – масив коефіцієнтів стандартної перехідної функції, α і β – масиви коефіцієнтів знаменника і чисельника передавальної функції. Вихідні параметри: δ – масив коефіцієнтів чисельника зображення помилки.

Крок 1. Покласти $l \coloneqq \dim\gamma - 1; n \coloneqq \dim\alpha - 1; \mu \coloneqq \tau_e/\tau_0; \eta \coloneqq 1; k \coloneqq l$.

Крок 2. Вирахувати $\omega_k \coloneqq \gamma_k \cdot \eta;\ \eta \coloneqq \eta \cdot \mu$.

Крок 3. Якщо $k > 0$, покласти $k \coloneqq k - 1$ і перейти до кроку 2.

Крок 4. Вирахувати згортку двох векторів $c \coloneqq \beta \otimes \omega$.

Крок 5. Вирахувати $\delta \coloneqq \alpha - c$.

Крок 6. Кінець.

Для обчислення ІКО (1) на основі перетворення помилки $E(s)$ за теоремою Парсеваля запишемо

$$I = \frac{1}{2\pi j} \int_{-j\infty}^{j\infty} E(s) \cdot E(-s) ds. \qquad (9)$$

Представимо зображення помилки рівністю

$$E(s) = \frac{\delta(s)}{\alpha(s)}, \qquad (10)$$

де $\alpha(s)$ – багаточлен степеня n, усі корені якого лежать в лівій напівплощині, $\beta(s)$ – багаточлен степеня n – 1. Тоді

$$I = \frac{1}{2\pi j} \int_{-j\infty}^{j\infty} \frac{\delta(s) \cdot \delta(-s)}{\alpha(s) \cdot \alpha(-s)} ds. \qquad (11)$$

Позначимо через $\delta(s^2) = \beta(s) \cdot \beta(-s)$ багаточлен парних степенів і перепишемо (11):

$$I = \frac{1}{2\pi j} \int_{-j\infty}^{j\infty} \frac{\delta(s^2)}{\alpha(s)\cdot\alpha(-s)} ds. \quad (12)$$

Нехай в зображенні (10)

$$\alpha(s) = \sum_{i=0}^{n} \alpha_i s^i; \ \beta(s) = \sum_{i=0}^{n-1} \beta_i s^i. \quad (13)$$

Вважаючи $m = n - 1$, багаточлен парних степенів (12) представимо у виді

$$\delta(s^2) = \sum_{k=0}^{m} \delta_k s^{2k}.$$

Інтеграл (12) вирахуємо за модифікованою формулою А. М. Каца [5]:

$$I = \frac{1}{2\pi j} \int_{-j\infty}^{j\infty} \frac{\delta(s^2)}{\alpha(s)\cdot\alpha(-s)} ds = \frac{(-1)^{n-1}}{2\alpha_n} \cdot \frac{G}{H}, \quad (14)$$

де H – визначник Гурвиця для багаточлена $\alpha(s)$, а G – визначник, отриманий з H заміною останнього рядка на рядок з коефіцієнтів багаточлена $\delta(s^2)$:

$$H = \begin{vmatrix} \alpha_0 & \alpha_2 & \alpha_4 & \alpha_6 & \cdots & 0 \\ 0 & \alpha_1 & \alpha_3 & \alpha_5 & \cdots & 0 \\ 0 & \alpha_0 & \alpha_2 & \alpha_4 & \cdots & 0 \\ 0 & 0 & \alpha_1 & \alpha_3 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & \cdots & \alpha_{n-1} \end{vmatrix}; G = \begin{vmatrix} \alpha_0 & \alpha_2 & \alpha_4 & \alpha_6 & \cdots & 0 \\ 0 & \alpha_1 & \alpha_3 & \alpha_5 & \cdots & 0 \\ 0 & \alpha_0 & \alpha_2 & \alpha_4 & \cdots & 0 \\ 0 & 0 & \alpha_1 & \alpha_3 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \delta_0 & \delta_1 & \delta_2 & \delta_3 & \cdots & \delta_{n-1} \end{vmatrix} \quad (15)$$

Формула (14) відрізняється від формули Каца записом багаточленів (13) у порядку зростання степеня змінної s, що дозволяє спростити алгоритм обчислення ІКО. Вважаючи $\alpha_i^{(0)} = \alpha_i, i = \overline{0:2:n}$; $\alpha_i^{(1)} = \alpha_i, i = \overline{1:2:n}$ (тут запис виду $i = \overline{k:2:n}$ означає послідовність цілих чисел, яка починається з k, збільшується з кроком 2 і не перевершує n), перетворимо визначники (15) до трикутного виду:

$$\lambda^{(1)} = \frac{\delta_0}{\alpha_0}; \quad \delta_j^{(1)} = \delta_j - \lambda^{(1)} \cdot \alpha_i, \qquad (i,j) = (2,1),(4,2),(6,3),\ldots; \quad (16)$$

$$\gamma^{(k)} = \frac{\alpha_{k-2}^{(k-2)}}{\alpha_{k-1}^{(k-1)}};\ \ \lambda^{(k)} = \frac{\delta_{k-1}^{(k-1)}}{\alpha_{k-1}^{(k-1)}},\ \ k = \overline{2, n-1}; \tag{17}$$

$$\alpha_i^{(k)} = \alpha_i^{(k-2)} - \gamma^{(k)} \cdot \alpha_{i+1}^{(k-1)};\ \ \delta_j^{(k)} = \delta_j^{(k-1)} - \lambda^{(k)} \cdot \alpha_{i+1}^{(k-1)}, \\ (i,j) = (k,k), (k+2, k+1), (k+4, k+2), \ldots \tag{18}$$

У цьому випадку ІКО (15) обчислюється за формулою

$$I = \frac{(-1)^{n-1} \cdot \delta_{n-1}^{(n-1)}}{2 \cdot \alpha_n \cdot \alpha_{n-1}^{(n-1)}}. \tag{19}$$

Алгоритм обчислення ІКО за формулою Каца виглядає так.

Алгоритм 2. Вхідні параметри: α і β – масиви коефіцієнтів знаменника і чисельника перетворення помилки. Вихідний параметр: I – ІКО.

Крок 1. Покласти $n \coloneqq \dim\alpha - 1; m \coloneqq n - 1;\ \delta \coloneqq PolPow2(\beta)$. ($PolPow2$ – функція, що обчислює масив коефіцієнтів багаточлена парних степенів $\delta(s^2)$).

Крок 2. Вирахувати $\lambda \coloneqq \delta_0/\alpha_0$ і покласти $j \coloneqq 1; i \coloneqq 2$.

Крок 3. Вирахувати $\delta_j \coloneqq \delta_j - \lambda \cdot \alpha_i$ і покласти $j \coloneqq j + 1$.

Крок 4. Якщо $i < m$, покласти $i \coloneqq i + 2$ і перейти до кроку 3.

Крок 5. Покласти $s \coloneqq 1; k \coloneqq 2$.

Крок 6. Вирахувати $\gamma \coloneqq \alpha_{k-2}/\alpha_{k-1}$; $\lambda \coloneqq \delta_{k-1}/\alpha_{k-1}$, і покласти $s \coloneqq -s; j \coloneqq k; i \coloneqq k$.

Крок 7. Вирахувати $\alpha_i \coloneqq \alpha_i - \gamma \cdot \alpha_{i+1};\ \delta_j \coloneqq \delta_j - \lambda \cdot \alpha_{i+1}$ і покласти $j \coloneqq j + 1$.

Крок 8. Якщо $i < m$, покласти $i \coloneqq i + 2$ і перейти до кроку 7.

Крок 9. Якщо $k < m$, покласти $k \coloneqq k + 1$ і перейти до кроку 6.

Крок 10. Вирахувати $I \coloneqq \delta_{n-1}/(2 \cdot \alpha_{n-1} \cdot \alpha_n)$.

Крок 11. Якщо $s < 0$, покласти $I \coloneqq -I$.

Крок 12. Кінець.

Інтеграл (11) можна вирахувати і методом Острьома за допомогою ітераційної процедури [7]. Нехай в зображенні помилки (10) багаточлени мають вигляд

$$\alpha(s) = \sum_{i=0}^{n} \alpha_i s^{n-i};\ \beta(s) = \sum_{i=0}^{n-1} \beta_i s^{n-i}. \tag{20}$$

Вважаючи

$\alpha_i^{(0)} = \alpha_i, i = \overline{0:2:n};\ \alpha_i^{(1)} = \alpha_i, i = \overline{1:2:n}, \beta_i^{(0)} = \beta_i, i = \overline{0:2:n-1};\ \beta_i^{(1)} = \beta_i, i = \overline{1:2:n-1}$,

вирахуємо $\gamma^{(k)}$ і $\lambda^{(k)}$ для $k = \overline{1,n}$:

$$\gamma^{(k-1)} = \frac{\alpha_{k-2}^{(k-2)}}{\alpha_{k-1}^{(k-1)}};\ \lambda^{(k-1)} = \frac{\beta_{k-2}^{(k-2)}}{\alpha_{k-1}^{(k-1)}},\ \ k = \overline{2,n-1}; \tag{21}$$

$$\alpha_i^{(k)} = \alpha_i^{(k-2)} - \gamma^{(k-1)} \cdot \alpha_{i+1}^{(k-1)};\ \beta_i^{(k)} = \beta_i^{(k-2)} - \lambda^{(k-1)} \cdot \alpha_{i+1}^{(k-1)}, i = \overline{k:2:n-1} \tag{22}$$

$$\gamma^{(k)} = \frac{\alpha_{k-1}^{(k-1)}}{\alpha_k^{(k)}};\ \lambda^{(k)} = \frac{\beta_{k-1}^{(k-1)}}{\alpha_k^{(k)}},\ \ k = \overline{n-1,n}. \tag{23}$$

Значення ІКО (11) обчислюється за формулою

$$I = 0{,}5 \cdot \sum_{k=1}^{n} \frac{\left(\lambda^{(k)}\right)^2}{\gamma^{(k)}}. \tag{24}$$

Алгоритм 3. Вхідні та вихідні параметри аналогічні алгоритму 2.
Крок 1. Покласти $n := \dim\alpha - 1; m := n - 1;\ I := 0; k := 2.$
Крок 2. Вирахувати $\gamma := \alpha_{k-2}/\alpha_{k-1}$, $\lambda := \beta_{k-2}/\alpha_{k-1}$, $I := I +\ \lambda^2/\gamma$ і покласти $i := k$.
Крок 3. Вирахувати $\alpha_i := \alpha_i - \gamma \cdot \alpha_{i+1};\ \beta_i := \beta_i - \lambda \cdot \alpha_{i+1}$.
Крок 4. Якщо $i < m$, покласти $i := i + 2$ і перейти до кроку 3.
Крок 5. Якщо $k < m$, покласти $k := k + 1$ і перейти до кроку 2.
Крок 6. Покласти $k := m$.
Крок 7. Вирахувати $\gamma := \alpha_{k-1}/\alpha_k$, $\lambda := \beta_{k-1}/\alpha_k$, $I := I +\ \lambda^2/\gamma$.
Крок 8. Якщо $k < n$, покласти $k := k + 1$ і перейти до кроку 7.
Крок 9. Вирахувати $I := 0{,}5 \cdot I$.
Крок 10. Кінець.

Вхідні параметри алгоритмів 2 і 3 обчислюються за алгоритмом 1.

Необхідна умова стійкості системи управління захистом полягає у вимозі позитивності усіх коефіцієнтів $\alpha_i > 0, i = \overline{0,n}$ [2; 3].

Вважаючи $\alpha_i^{(0)} = \alpha_i, i = \overline{0:2:n};\ \alpha_i^{(1)} = \alpha_i, i = \overline{1:2:n}$, критерій стійкості Рауса-Гурвиця зведемо до вирахувань за формулами

$$\gamma^{(k)} = \frac{\alpha_{k-2}^{(k-2)}}{\alpha_{k-1}^{(k-1)}};\ \alpha_i^{(k)} = \alpha_i^{(k-2)} - \gamma^{(k)} \cdot \alpha_{i+1}^{(k-1)},\ \ i = \overline{k:2:n-1};\ k = \overline{2,n-1}. \tag{25}$$

Введемо позначення елементів визначника Рауса-Гурвиця:

$$\rho_0 = \alpha_0, \rho_1 = \alpha_1, \rho_k = \alpha_k^k, k = \overline{2, n-1}, \rho_n = \alpha_n. \qquad (26)$$

Алгоритм 4. Вхідний параметр: α – масив коефіцієнтів характеристичного багаточлена. Вихідний параметр: ρ – масив коефіцієнтів визначника Рауса-Гурвиця.

Крок 1. Покласти $n \coloneqq \dim \alpha - 1; m \coloneqq n - 1;\ \rho \coloneqq \alpha; k \coloneqq 2$.
Крок 2. Вирахувати $\gamma \coloneqq \rho_{k-2}/\rho_{k-1}$ і покласти $i \coloneqq k$.
Крок 3. Вирахувати $\rho_i \coloneqq \rho_i - \gamma \cdot \rho_{i+1}$.
Крок 4. Якщо $i < m$, покласти $i \coloneqq i + 2$ і перейти до кроку 3.
Крок 5. Якщо $k < m$, покласти $k \coloneqq k + 1$ і перейти до кроку 2.
Крок 6. Кінець.

Необхідними і достатніми умовами стійкості системи управління захистом є умови $\alpha_i > 0, i = \overline{0, n}$, а також $\rho_k > 0, k = \overline{2, n-1}$. З урахуванням цих умов і алгоритму 4 розробимо алгоритм визначення стійкості системи управління за критерієм Рауса-Гурвиця.

Алгоритм 5. Вхідний параметр: α – масив коефіцієнтів характеристичного багаточлена. Вихідний параметр: B – ознака стійкості системи.

Крок 1. Покласти $n \coloneqq \dim \alpha - 1; B \coloneqq 1;\ i \coloneqq 0$.
Крок 2. Якщо $\alpha_i \leq 0$, покласти $B \coloneqq 0$ і перейти до кроку 10.
Крок 3. Якщо $i < n$, покласти $i \coloneqq i + 1$ і перейти до кроку 2.
Крок 4. Покласти $m \coloneqq n - 1;\ \rho \coloneqq \alpha; k \coloneqq 2$.
Крок 5. Вирахувати $\gamma \coloneqq \rho_{k-2}/\rho_{k-1}$ і покласти $i \coloneqq k$.
Крок 6. Вирахувати $\rho_i \coloneqq \rho_i - \gamma \cdot \rho_{i+1}$.
Крок 7. Якщо $i < m$, покласти $i \coloneqq i + 2$ і перейти до кроку 6.
Крок 8. Якщо $\rho_k \leq 0$, покласти $B \coloneqq 0$ і перейти до кроку 10.
Крок 9. Якщо $k < m$, покласти $k \coloneqq k + 1$ і перейти до кроку 5.
Крок 10. Кінець.

Нехай коефіцієнти характеристичного багаточлена-функції вектора варійованих параметрів системи $\alpha_i = \alpha_i(\boldsymbol{x}), i = \overline{0, n}; \boldsymbol{x} \in R^p$. Для того, щоб система була стійкою, необхідно і достатньо, щоб:

$$\alpha_i(\boldsymbol{x}) > 0, i = \overline{0, n};\ \rho_k(\boldsymbol{x}) > 0, k = \overline{2, n-1} \qquad (27)$$

Ступінь порушення першої групи нерівностей представимо однією штрафною функцією

$$P(\boldsymbol{x}) = \sum_{i=0}^{n} \max\{-\alpha_i(\boldsymbol{x}), 0\} \qquad (28)$$

Нерівностям (27) відповідають області виконання обмежень:

$$\Omega_1 = \{\boldsymbol{x} | \alpha_i(\boldsymbol{x}) > 0, i = \overline{0,n}\};\ \Omega_k = \{\boldsymbol{x} | \rho_k(\boldsymbol{x}) > 0\}, k = \overline{2, n-1}. \quad (29)$$

Складемо з них систему областей: $D_1 = \Omega_1, D_k = D_{k-1} \cap \Omega_k, k = \overline{2, n-1}$, з яких за допомогою різниць множин настроїмо області рівнів обмежень:

$$H_0 = R^p \backslash D_1;\ H_k = D_k \backslash D_{k+1}, k = \overline{1, n-1};\ H_{n-1} = D_{n-1}. \quad (30)$$

Для переходу в область стійкості пропонується двовимірна векторна штрафна функція:

$$F(\boldsymbol{x}) = \begin{cases} (0; P(\boldsymbol{x})), & \boldsymbol{x} \in H_0; \\ (k; -\rho_{k+1}(\boldsymbol{x})), & \boldsymbol{x} \in H_k, k = \overline{1, n-2}; \\ (n-1; 0), & \boldsymbol{x} \in H_{n-1}. \end{cases} \quad (31)$$

Перша компонента відображає приналежність аргументу певної області і називається *функцією рівня*. Друга є штрафом за порушення обмеження і називається *функцією штрафу*. При задоволенні усіх обмежень (третя компонента) функція рівня набуває максимального значення $n-1$ і функція штрафу обнуляється.

Алгоритм 6. Вхідний параметр: α – масив коефіцієнтів характеристичного багаточлена. Вихідні параметри: B – ознака стійкості системи; F – значення векторної штрафної функції.
Крок 1. Покласти $n \coloneqq \dim\alpha - 1; B \coloneqq 1;\ i \coloneqq 0; h \coloneqq 0; P \coloneqq 0$.
Крок 2. Якщо $\alpha_i \le 0$, вирахувати $P \coloneqq P - \alpha_i$ і покласти $B \coloneqq 0$.
Крок 3. Якщо $i < n$, покласти $i \coloneqq i + 1$ і перейти до кроку 2.
Крок 4. Якщо $B = 0$, покласти $F \coloneqq (h, P)$ і перейти до кроку 12.
Крок 5. Покласти $m \coloneqq n - 1;\ \rho \coloneqq \alpha; k \coloneqq 2$.
Крок 6. Вирахувати $\gamma \coloneqq \rho_{k-2}/\rho_{k-1}$ і покласти $h \coloneqq h + 1, i \coloneqq k$.
Крок 7. Вирахувати $\rho_i \coloneqq \rho_i - \gamma \cdot \rho_{i+1}$.
Крок 8. Якщо $i < m$, покласти $i \coloneqq i + 2$ і перейти до кроку 7.
Крок 9. Якщо $\rho_k \le 0$, покласти $F \coloneqq (h, -\rho_k);\ B \coloneqq 0$ і перейти до кроку 12.
Крок 10. Якщо $k < m$, покласти $k \coloneqq k + 1$ і перейти до кроку 6.
Крок 11. Вирахувати $F \coloneqq (h + 1, 0)$.
Крок 12. Кінець.

У разі оцінки векторної цільової функції вважатимемо, що передавальна функція (3) залежить від вектора варійованих параметрів $\boldsymbol{x}$:

$$W(\boldsymbol{x}, s) = \frac{\beta(\boldsymbol{x}, s)}{\alpha(\boldsymbol{x}, s)}, \alpha(\boldsymbol{x}, s) = \sum_{i=0}^{n} \alpha_i(\boldsymbol{x}) s^{n-i}; \quad (32)$$

$$\beta(\boldsymbol{x},s)=\sum_{j=0}^{m}\beta_j(\boldsymbol{x})s^{m-j};\ \alpha_n(\boldsymbol{x})=\beta_m(\boldsymbol{x}).$$

Задаючи ваговий багаточлен (4) і визначаючи багаточлен (6)

$$\delta(\boldsymbol{x},s)=\frac{\alpha(\boldsymbol{x},s)-\beta(\boldsymbol{x},s)\cdot\omega(s)}{s},\tag{33}$$

ІКО представимо як функцію варійованих параметрів:

$$I(\boldsymbol{x})=\frac{1}{2\pi j}\int_{-j\infty}^{j\infty}\frac{\delta(\boldsymbol{x},s)\cdot\delta(\boldsymbol{x},-s)}{\alpha(\boldsymbol{x},s)\cdot\alpha(\boldsymbol{x},-s)}ds.\tag{34}$$

При кожному значенні вектора $\boldsymbol{x}$ з області стійкості системи значення цієї функції можна вирахувати за алгоритмами 2 і 3. Перевизначимо векторну штрафну функцію (31) у векторну цільову функцію, доповнивши її значенням ІКО (34) в області стійкості:

$$F(\boldsymbol{x})=\begin{cases}(0;P(\boldsymbol{x})), & \boldsymbol{x}\in H_0;\\ \left(k;-\rho_{k+1}(\boldsymbol{x})\right), & \boldsymbol{x}\in H_k, k=\overline{1,n-2};\\ (n-1;I(\boldsymbol{x})), & \boldsymbol{x}\in H_{n-1}.\end{cases}\tag{35}$$

Модифікуємо алгоритми 2 і 3 відповідно до алгоритму 6 для обчислення векторної цільової функції (35) [6].

Алгоритм 7. Вхідні параметри: α і β – масиви коефіцієнтів знаменника і чисельника перетворення помилки. Вихідні параметри: F – значення векторної цільової функції; B – ознака стійкості системи.

Крок 1. Покласти $n\coloneqq \dim\alpha-1; B\coloneqq 1;\ i\coloneqq 0;h\coloneqq 0;P\coloneqq 0.$

Крок 2. Якщо $\alpha_i\le 0$, вирахувати $P\coloneqq P-\alpha_i$ і покласти $B\coloneqq 0.$

Крок 3. Якщо $i<n$, покласти $i\coloneqq i+1$ і перейти до кроку 2.

Крок 4. Якщо $B=0$, покласти $F\coloneqq(h,P)$ і перейти до кроку 17.

Крок 5. Вирахувати $\delta\coloneqq PolPow2(\beta);\lambda\coloneqq\delta_0/\alpha_0$ і покласти $m\coloneqq n-1;\ j\coloneqq 1;i\coloneqq 2.$

Крок 6. Вирахувати $\delta_j\coloneqq\delta_j-\lambda\cdot\alpha_i$ і покласти $j\coloneqq j+1.$

Крок 7. Якщо $i<m$, покласти $i\coloneqq i+2$ і перейти до кроку 6.

Крок 8. Покласти $s\coloneqq 1;k\coloneqq 2.$

Крок 9. Вирахувати $\gamma\coloneqq\alpha_{k-2}/\alpha_{k-1};\ \lambda\coloneqq\delta_{k-1}/\alpha_{k-1}$ і покласти $h\coloneqq h+1;s\coloneqq -k;j\coloneqq k;i\coloneqq k.$

Крок 10. Вирахувати $\alpha_i\coloneqq\alpha_i-\gamma\cdot\alpha_{i+1};\ \delta_j\coloneqq\delta_j-\lambda\cdot\alpha_{i+1}$ і покласти $j\coloneqq j+1.$

Крок 11. Якщо $i < m$, покласти $i := i + 2$ і перейти до кроку 10.
Крок 12. Якщо $\alpha_k \leq 0$, покласти $F := (h, -\alpha_k)$; $B := 0$ і перейти до кроку 17.
Крок 13. Якщо $k < m$, покласти $k := k + 1$ і перейти до кроку 9.
Крок 14. Вирахувати $I := \delta_{n-1}/(2 \cdot \alpha_{n-1} \cdot \alpha_n)$.
Крок 15. Якщо $s < 0$, покласти $I := -I$.
Крок 16. Вирахувати $F := (h + 1, I)$.
Крок 17. Кінець.

Алгоритм розрахунку векторної цільової функції за Острьомом.

Алгоритм 8. Вхідні та вихідні параметри аналогічні алгоритму 7.
Крок 1. Покласти $n := \dim\alpha - 1; B := 1;\ i := 0; h := 0; P := 0$.
Крок 2. Якщо $\alpha_i \leq 0$, вирахувати $P := P - \alpha_i$ і покласти $B := 0$.
Крок 3. Якщо $i < n$, покласти $i := i + 1$ і перейти до кроку 2.
Крок 4. Якщо $B = 0$, покласти $F := (h, P)$ і перейти до кроку 15.
Крок 5. Покласти $m := n - 1; I := 0; k := 2$.
Крок 6. Вирахувати $\gamma := \alpha_{k-2}/\alpha_{k-1}$; $\lambda := \beta_{k-2}/\alpha_{k-1}; I := I + \lambda^2/\gamma$ і покласти $h := h + 1; i := k$.
Крок 7. Вирахувати $\alpha_i := \alpha_i - \gamma \cdot \alpha_{i+1}$; $\beta_i := \beta_i - \lambda \cdot \alpha_{i+1}$.
Крок 8. Якщо $i < m$, покласти $i := i + 2$ і перейти до кроку 7.
Крок 9. Якщо $\alpha_k \leq 0$, покласти $F := (h, -\alpha_k); B := 0$ і перейти до кроку 15.
Крок 10. Якщо $k < m$, покласти $k := k + 1$ і перейти до кроку 6.
Крок 11. Покласти $k := m$.
Крок 12. Вирахувати $\gamma := \alpha_{k-1}/\alpha_k$; $\lambda := \beta_{k-1}/\alpha_k; I := I + \lambda^2/\gamma$.
Крок 13. Якщо $k < n$, покласти $k := k + 1$ і перейти до кроку 12.
Крок 14. Вирахувати $F := (h + 1, 0{,}5 \cdot I)$.
Крок 15. Кінець.

Висновки. На основі моделі уявлення системи управління захистом інформаційних ресурсів у вигляді системи автоматичного управління:

1. Запропонований спосіб формування інтегральних квадратичних оцінок (ІКО) помилки управління, що враховує вагові коефіцієнти оцінок за бажаним часом встановлення і стандартні передавальні функції.

2. Розроблені алгоритми обчислення ІКО за модифікованою формулою Каца і методом Острьома для систем управління довільного порядку, включаючи векторне уявлення цільової функції системи захисту.

3. Запропонована векторна штрафна функція і розроблений алгоритм її обчислення для відображення ступеня порушення умов стійкості параметрів системи захисту за критерієм Рауса-Гурвиця.

Частина 2. Покроковий підхід до мінімізації помилок оцінок систем управління

Постановка проблеми. Завдання визначення оптимальних показників якості систем управління захистом інформаційних ресурсів автоматизованих систем є однією з найважливіших проблем проектування комплексних систем захисту інформації. Це зумовлено складністю подібних систем, наявністю безлічі варійованих параметрів, складністю обчислення показників якості. Такі показники задаються у вигляді інтегральних квадратичних оцінок (ІКО) помилки системи управління захистом [1]. Проблема полягає в тому, що існуючі чисельні методи їх мінімізації такі, як методи Хука-Дживса і Нелдера-Міда [3; 8–10], орієнтовані на єдиний (скалярний) параметр, що впливає на ІКО, тоді як в реальних системах таких параметрів може бути декілька і в сукупності вони утворюють векторний параметр. Здійснювати оптимізацію одночасно за всіма параметрами дуже складно, тому перспективним вважається покроковий підхід [6; 11], коли оптимізація виконується послідовно за параметрами. При цьому повинні неухильно задовольнятися обмеження на стійкість системи шляхом оптимізації векторної цільової функції, алгоритми обчислення якої розроблені в частині 1 статті [1].

Метою роботи є розробка методів покрокового підходу до мінімізації інтегральної квадратичної оцінки помилки системи управління захистом з областю визначення, обмеженій умовами стійкості.

Виклад основного матеріалу. Нехай передавальна функція системи управління залежить від вектора варійованих параметрів $\boldsymbol{x} \in R^p$:

$$W(\boldsymbol{x},s) = \frac{\beta(\boldsymbol{x},s)}{\alpha(\boldsymbol{x},s)}; \alpha(\boldsymbol{x},s) = \sum_{i=0}^{n} \alpha_i(\boldsymbol{x}) s^{n-i};\ \beta(\boldsymbol{x},s) = \sum_{i=0}^{m} \beta_i(\boldsymbol{x}) s^{m-i};\ \alpha_n = \beta_m \quad (36)$$

Їй відповідає перехідна функція $y(\boldsymbol{x},t)\colon y(\boldsymbol{x},0)) = 0; y(\boldsymbol{x},\infty) = 1$. Помилка управління визначається як зважена сума відхилення $z(\boldsymbol{x},t) = y(\boldsymbol{x},\infty) - y(\boldsymbol{x},t)$ і її похідних $z_t^{(k)}(\boldsymbol{x},t)$:

$$e(\boldsymbol{x},t) = \sum_{k=0}^{l} w_k \cdot z_t^{(l-k)}(\boldsymbol{x},t), \quad (37)$$

де $w_k, k = \overline{0,l};\ l \le n - m;\ w_0 = 1$ – вагові коефіцієнти.

Як показано в [1], інтегральна квадратична оцінка (ІКО) помилки управління системою обчислюється за формулою:

$$I(\boldsymbol{x}) = \frac{1}{2\pi j} \int_{-j\infty}^{j\infty} \frac{\delta(\boldsymbol{x},s) \cdot \delta(\boldsymbol{x},-s)}{\alpha(\boldsymbol{x},s) \cdot \alpha(\boldsymbol{x},-s)} ds, \quad (38)$$

де $\delta(\boldsymbol{x},s) = [\alpha(\boldsymbol{x},s) - \beta(\boldsymbol{x},s) \cdot w(s)]/s$; $w(s) = \sum_{k=0}^{l} w_k \cdot s^{l-k}$.

ІКО визначається в області стійкості $D \subset R^p$, яка з використанням елементів першого стовпця визначника Рауса $\rho_k = \alpha_k^k, k = \overline{0,n}$, задається умовами

$$\alpha_i(\boldsymbol{x}) > 0, i = \overline{0,n};\ \rho_k(\boldsymbol{x}) > 0, k = \overline{2,n-1} \quad (39)$$

Цим нерівностям відповідають області виконання обмежень

$$\Omega_1 = \{\boldsymbol{x}|\alpha_i(\boldsymbol{x}) > 0, i = \overline{0,n}\};\ \Omega_k = \{\boldsymbol{x}|\rho_k(\boldsymbol{x}) > 0\}, k = \overline{2,n-1}. \quad (40)$$

Складемо з них систему квазідопустимих областей $D_1 = \Omega_1, D_k = D_{k-1} \cap \Omega_k, k = \overline{2,n-1}$, з яких за допомогою різниць множин побудуємо області рівнів обмежень

$$H_0 = R^p \backslash D_1;\ H_k = D_k \backslash D_{k+1}, k = \overline{1,n-1};\ H_{n-1} = D_{n-1}. \quad (41)$$

Ступінь порушення першої групи нерівностей (4) зобразимо штрафною функцією

$$P(\boldsymbol{x}) = \sum_{i=0}^{n} \max\{-\alpha_i(\boldsymbol{x}), 0\}. \quad (42)$$

Для оцінки ступеня порушення усіх умов стійкості системи управління в [1] сформована векторна цільова функція:

$$F(\boldsymbol{x}) = \begin{cases} (0; P(\boldsymbol{x})), & \boldsymbol{x} \in H_0; \\ (k; -\rho_{k+1}(\boldsymbol{x})), & \boldsymbol{x} \in H_k, k = \overline{1,n-2}; \\ (n-1; I(\boldsymbol{x})), & \boldsymbol{x} \in H_{n-1} \end{cases} \quad (43)$$

і розроблений алгоритм її обчислення. Властивості цієї функції зумовлені властивостями областей D_k, які утворюють систему вкладених областей: $D_{n-1} \subseteq D_{n-2} \subseteq \cdots \subseteq D_1 \subseteq R^p$. Оскільки $D_k \subset \Omega_j, j = \overline{1,k}$, то чим більше значення індексу квазідопустимої області k, тим більше виконано в ній умов (4). У області D_{n-1} мають бути виконані усі умови стійкості. Області рівнів обмежень (6) мають властивості $H_k \subset D_k, k = \overline{1,n-1};\ H_j \cap H_k = \emptyset, j \neq k; D = H_{n-1}$, тому області рівнів ділять увесь простір варійованих параметрів на n областей: $R^p = H_0 \cup H_1 \cup \ldots \cup H_{n-1}$.

Перша складова (43) – функція рівня – відображає приналежність її аргументів певної області рівня і являє собою лічильник числа виконаних обмежень, вона кусочно-постійна і може набувати значень від 0 до *n* – 1. Друга складова – функція штрафу – представляє поза області стійкості штрафну функцію порушеного обмеження. У області стійкості ця складова співпадає з ІКО. Векторна функція (43) враховує особливості завдання оптимізації – ієрархію обмежень в процесі обчислення ІКО.

Розробимо покроковий підхід мінімізації ІКО за допомогою функції (43), яка враховує умови стійкості (39). Нехай $\boldsymbol{x} \in H_0$, тоді в силу (41) $\exists i \in \{\overline{0,n}\}, \alpha_i(\boldsymbol{x}) \leq 0$. У області H_0 мірою цього порушення обмеження є функція штрафу (42), яка убуває у напрямі межі наступної області H_1, в якій має місце нерівність $\rho_2(\boldsymbol{x}) \leq 0$. Мірою цього порушення обмеження служить функція штрафу $-\rho_2(\boldsymbol{x})$, що убуває у напрямі межі області H_2 і т. д. У будь-якій непорожній області рівня $H_k, k = \overline{1, n-2}$ мірою порушення обмеження є відповідна функція штрафу $-\rho_{k+1}(\boldsymbol{x})$. Крок переходу з області H_k в область H_{k+1} полягає в мінімізації функції штрафу (43) в області H_k. При такій мінімізації чергове обмеження задовольняється, функція (43) зростає і починається наступний крок методу. Враховуючи обмежену кількість областей рівнів, перехід з будь-якої початкової точки в область стійкості виконається не більше ніж за *n* – 1 кроків.

Для реалізації покрокового підходу оптимізуватимемо векторну функцію $F(\boldsymbol{x})$ з урахуванням пріоритету її складових: першу складову, що має пріоритет, необхідно максимізувати, а другу складову – мінімізувати. Завдання оптимізації векторної цільової функції (43) з урахуванням пріоритету її складових позначимо $vecopt\, F(\boldsymbol{x})$. Для визначення кращих значень векторної функції застосуємо операцію порівняння її значень. Оскільки першу проекцію векторної функції (43), що має пріоритет, необхідно збільшувати, а другу – зменшувати, то два значення векторної цільової функції $U = (U_1, U_2)$ і $V = (V_1, V_2)$ порівняємо бінарною операцією «краще» $\prec$:

$$U \prec V = \begin{cases} 1, & \text{якщо } (U_1 > V_1) \vee (U_1 = V_1) \wedge (U_2 < V_2); \\ 0, & \text{якщо } (U_1 < V_1) \vee (U_1 = V_1) \wedge (U_2 \geq V_2). \end{cases} \quad (44)$$

Сенс цієї операції полягає в тому, що при різних перших складових векторної функції (43) в двох точках кращою вважається та, в якій перша складова, що має пріоритет, набуває більшого значення. При рівних значеннях першої складової кращою вважається та точка, в якій друга складова набуває меншого значення. Операцію (44) можна реалізувати за наступним алгоритмом.

Алгоритм 9. Порівняння значень векторної функції. Вхідні параметри: *U*, *V* – значення векторної функції. Вихідний параметр: *B* – бульовий результат порівняння.

Крок 1. Покласти $B := 0$.

Крок 2. Якщо $U_1 > V_1$, покласти $B := 1$ і перейти до кроку 5.

Крок 3. Якщо $U_1 < V_1$, перейти до кроку 5.
Крок 4. Якщо $U_2 < V_2$, покласти $B := 1$.
Крок 5. Кінець.

Для кількісної реалізації покрокового підходу модифікуємо прямі методи безумовної мінімізації функцій (методи Хука-Дживса і Нелдера-Міда) [8–10]. Зобразимо алгоритм модифікованого методу Хука-Дживса для оптимізації векторних функцій.

Алгоритм 10. Вхідні параметри: $\boldsymbol{x}$ – початкова точка, ε – допустима погрішність, $F(\boldsymbol{x})$ – векторна функція, $R(\boldsymbol{x}, F_{\boldsymbol{x}})$ – функція досліджуючого пошуку. Вихідні параметри: $\boldsymbol{x}, F_{\boldsymbol{x}}$ – краща точка пошуку і значення векторної функції в ній.
Крок 1. Покласти $F_{\boldsymbol{x}} := F(\boldsymbol{x})$; $\delta := 2$ (δ – крок досліджуючого пошуку).
Крок 2. Якщо $\delta < \varepsilon$, перейти до кроку 10.
Крок 3. Покласти $\delta := \delta/2$.
Крок 4. Покласти $(\boldsymbol{y}, F_{\boldsymbol{y}}) := R(\boldsymbol{x}, F_{\boldsymbol{x}})$.
Крок 5. Якщо $(F_{\boldsymbol{y}} \prec F_{\boldsymbol{x}})$, перейти до кроку 2.
Крок 6. Вирахувати $\boldsymbol{z} := 2\boldsymbol{y} - \boldsymbol{x}$; $F_{\boldsymbol{z}} := F(\boldsymbol{z})$.
Крок 7. Покласти $\boldsymbol{x} := \boldsymbol{y}$; $F_{\boldsymbol{x}} := F_{\boldsymbol{y}}$.
Крок 8. Вирахувати $(\boldsymbol{y}, F_{\boldsymbol{y}}) := R(\boldsymbol{z}, F_{\boldsymbol{z}})$.
Крок 9. Якщо $(F_{\boldsymbol{y}} \prec F_{\boldsymbol{x}})$, перейти до кроку 6, інакше – до кроку 2.
Крок 10. Кінець.

Тут $(\boldsymbol{x}, F_{\boldsymbol{x}})$ – краща точка попередньої інтеграції і значення функції в ній, $(\boldsymbol{y}, F_{\boldsymbol{y}})$ – краща точка досліджуючого пошуку і значення функції в ній, $(\boldsymbol{z}, F_{\boldsymbol{z}})$ – точка пошуку за зразком і значення функції в ній. На кроці 1 обчислюється значення цільової функції в початковій точці та ініціалізується крок досліджуючого пошуку. На кроці 2 перевіряється критерій зупину. Кроки 3–9 реалізують ітераційний цикл методу. На кроці 3 зменшується крок досліджуючого пошуку, а на кроці 4 здійснюється досліджуючий пошук типу 1. На кроці 5 порівнюються значення векторної функції в кінцевій точці досліджуючого пошуку типу 1 і базовій точці. На кроці 6 виконують пошук за зразком. На кроці 7 замінюється базова точка. На кроці 8 здійснюється досліджуючий пошук типу 2. На кроці 9 порівнюються значення векторної функції в кінцевій точці досліджуючого пошуку типу 2 і в новій базовій точці. Модифікований метод Хука-Дживса відрізняється від методу мінімізації скалярної цільової функції кроками 5 і 9, на яких замість операції «менше» застосовується операція «краще» (44). Модифікований досліджуючий пошук, що реалізовує функцію $R(\boldsymbol{x}, F_{\boldsymbol{x}})$, представимо алгоритмом.

Алгоритм 11. Вхідні параметри: $(\boldsymbol{x}, F_{\boldsymbol{x}})$, – базова точка, δ – крок пошуку, $F(\boldsymbol{x})$ – векторна функція. Вихідні параметри: $(\boldsymbol{y}, F_{\boldsymbol{y}})$ – краща точка пошуку.
Крок 1. Покласти $n := \dim \boldsymbol{x}$; $\boldsymbol{y} := \boldsymbol{x}$; $F_{\boldsymbol{y}} := F_{\boldsymbol{x}}$; $\boldsymbol{u} := \boldsymbol{y}$; $i := 1$.
Крок 2. Вирахувати $u_i := y_i + \delta$; $F_{\boldsymbol{u}} := F(\boldsymbol{u})$.
Крок 3. Якщо $(F_{\boldsymbol{u}} \prec F_{\boldsymbol{y}})$, покласти $y_i := u_i$; $F_{\boldsymbol{y}} := F_{\boldsymbol{u}}$ і перейти до кроку 6.

Крок 4. Вирахувати $u_i := y_i - \delta;\ F_{\boldsymbol{u}} := F(\boldsymbol{u})$.
Крок 5. Якщо $(F_{\boldsymbol{u}} \prec F_{\boldsymbol{y}})$, покласти $y_i := u_i;\ F_{\boldsymbol{y}} := F_{\boldsymbol{u}}$, інакше – $u_i := y_i$.
Крок 6. Якщо $i < n$, покласти $i := i + 1$ і перейти до кроку 2.
Крок 7. Кінець.

У цьому алгоритмі n – розмірність вектора змінних, i – номер змінної. На кроці 1 ініціалізуються параметри досліджуючого пошуку. Кроки 2–6 складають ітераційний цикл методу. На кроці 2 виконується досліджуваний пошук в позитивному напрямі. На кроці 3 порівнюються значення векторної функції в поточній точці пошуку і в кращій точці. Після невдалого пошуку в позитивному напрямі кроком 4 виконується досліджуючий пошук в негативному напрямі. На кроці 5 порівнюються значення векторної функції в поточній точці пошуку і в кращій точці. На кроці 6 перевіряється критерій зупину. Операція «краще» для значень векторної функції в цьому алгоритмі застосовується на кроках 3 і 5.

Пошук мінімуму функції методом Нелдера-Міда залежить від результату порівняння значень цільової функції в різних точках пошуку при визначенні кращої або гіршої вершини, розтягуванні або стискуванні багатогранника [6; 8–10]. Для застосування цього методу в оптимізації векторних цільових функцій порівняння значень скалярної цільової функції замінимо порівнянням значень векторних функцій (44). Модифікований метод Нелдера-Міда представимо наступним алгоритмом.

Алгоритм 12. Вхідні параметри: $\boldsymbol{x}$ – початкова точка, $F(\boldsymbol{x})$ – векторна функція, ε – допустима погрішність. Вихідні параметри: $(\boldsymbol{x}, F_{\boldsymbol{x}})$ – краща точка пошуку і значення векторної функції в ній.
Крок 1. Покласти $n := \dim \boldsymbol{x};\ m := n + 1;\ \delta := 1;\ j := 1;\ \boldsymbol{P}_m := \boldsymbol{x}$.
Крок 2. Вирахувати $F_{\boldsymbol{m}} := F(\boldsymbol{x})$.
Крок 3. Покласти $\boldsymbol{P}_j := \boldsymbol{x};\ P_{jj} := x_j + \delta$.
Крок 4. Вирахувати $F_j := F(\boldsymbol{P}_j)$.
Крок 5. Якщо $j < n$, покласти $j := j + 1$ і перейти до кроку 3.
Крок 6. Якщо $\delta < \varepsilon$, перейти до кроку 29.
Крок 7. Покласти $l := m;\ F_{\boldsymbol{x}} := F_{\boldsymbol{m}};\ h := m;\ F_{\boldsymbol{w}} := F_{\boldsymbol{m}};\ j := 1$.
Крок 8. Якщо $F_j \prec F_{\boldsymbol{x}}$, покласти $F_{\boldsymbol{x}} := F_j;\ l := j$.
Крок 9. Якщо $F_{\boldsymbol{w}} \prec F_j$, покласти $F_{\boldsymbol{w}} := F_j;\ h := j$.
Крок 10. Якщо $j < n$, покласти $j := j + 1$ і перейти до кроку 8.
Крок 11. Покласти $\boldsymbol{x} := \boldsymbol{P}_l;\ \boldsymbol{w} := \boldsymbol{P}_h$.
Крок 12. Вирахувати $\delta := \max\left|P_{ij} - x_i\right|$.
Крок 13. Вирахувати $\boldsymbol{c} := \left(\sum_{j=1}^{m} \boldsymbol{P}_j - \boldsymbol{w}\right)/n$.
Крок 14. Вирахувати $\boldsymbol{y} := 2\boldsymbol{c} - \boldsymbol{w};\ F_{\boldsymbol{y}} := F(\boldsymbol{y})$.
Крок 15. Якщо $F_{\boldsymbol{y}} \prec F_{\boldsymbol{x}}$, перейти до кроку 16, інакше – до кроку 19.
Крок 16. Вирахувати $\boldsymbol{z} := 2\boldsymbol{y} - \boldsymbol{c};\ F_{\boldsymbol{z}} := F(\boldsymbol{z})$.
Крок 17. Якщо $F_{\boldsymbol{z}} \prec F_{\boldsymbol{y}}$, покласти $\boldsymbol{P}_h := \boldsymbol{z};\ F_h := F_{\boldsymbol{z}}$, інакше – $\boldsymbol{P}_h := \boldsymbol{y};\ F_h := F_{\boldsymbol{y}}$.

Крок 18. Перейти до кроку 6.
Крок 19. Покласти $j := 1$.
Крок 20. Якщо $j \neq h$ і $F_y \prec F_j$, покласти $\boldsymbol{P}_h := \boldsymbol{y}$; $F_h := F_y$ і перейти до кроку 6.
Крок 21. Якщо $j < m$, покласти $j := j + 1$ і перейти до кроку 20.
Крок 22. Якщо $F_y \prec F_w$, покласти $\boldsymbol{w} := \boldsymbol{y}$; $F_w := F_y$.
Крок 23. Вирахувати $\boldsymbol{z} := 0{,}5(\boldsymbol{w} + \boldsymbol{c})$; $F_z := F(\boldsymbol{z})$.
Крок 24. Якщо $F_z \prec F_w$, покласти $\boldsymbol{P}_h := \boldsymbol{z}$; $F_h := F_z$ і перейти до кроку 6.
Крок 25. Покласти $j := 1$.
Крок 26. Якщо $j \neq l$, вирахувати $\boldsymbol{P}_j := 0{,}5(\boldsymbol{P}_j + \boldsymbol{x})$; $F_j := F(\boldsymbol{P}_j)$.
Крок 27. Якщо $j < m$, покласти $j := j + 1$ і перейти до кроку 26.
Крок 28. Перейти до кроку 26.
Крок 29. Кінець.

У цьому алгоритмі m означає кількість вершин багатогранника; δ – його розмір; $\boldsymbol{P}_j, F_j$ – вершина з номером j і значення векторної функції в ній; $\boldsymbol{x}, l, F_x$ – краща вершина, її індекс і значення векторної функції в ній; $\boldsymbol{w}, h, F_w$ – гірша вершина, її індекс і значення векторної функції в ній; c – центр тяжіння вершин за винятком гіршої; $\boldsymbol{y}$ і F_y – точка віддзеркалення і значення векторної функції в ній; $\boldsymbol{z}$ і F_z – точка розтягування або стискування і значення векторної функції в ній. Кроки 1 і 2 ініціалізують параметри та формують початкову вершину. Кроки 3–5 формують початковий багатогранник. Кроки 6–28 складають ітераційний цикл методу. На кроці 6 перевіряється критерій зупину. Кроки 7–11 визначають кращу і гіршу вершини. Крок 12 обчислює розмір багатогранника. На кроці 13 обчислюється центр вершин багатогранника, за винятком гіршої вершини. На кроках 14–21 виконуються віддзеркалення і розтягування багатогранника, а на кроках 22–28 – стискання і редукція багатогранника.

Висновки. 1. У статті розглянуто властивості векторної цільової функції для мінімізації інтегрованої квадратичної оцінки (ІКО) помилки системи управління захистом. На відміну від традиційного завдання функції ІКО тільки в області стійких систем управління, пропонована векторна цільова функція визначається в усьому просторі варійованих параметрів.

2. Запропоновано покроковий підхід встановлення достатньої кількості обмежень для переходу в область визначення ІКО і раціональний механізм його реалізації у вигляді пріоритетної оптимізації векторної цільової функції з використанням операції порівняння її значень.

3. Розроблено алгоритми модифікованих методів Хука-Дживса і Нелдера-Міда для оптимізації векторних функцій. Основна відмінність запропонованих модифікованих методів оптимізації векторних функцій від методів мінімізації скалярних функцій полягає у використанні бінарних операцій порівняння значень векторних функцій.

4. Застосування методів оптимізації векторної цільової функції дозволяє в єдиному обчислювальному процесі перейти з будь-якої початкової точки пошуку допустимих параметрів системи управління захистом інформаційних ресурсів в область стійкості системи управління і знайти в ній мінімум ІКО.

Список використаних джерел

1. Тискина Е. О. Моделирование системы управления защитой объекта / Е. О. Тискина, В. А. Хорошко // Інформаційна безпека людини, суспільства, держави. – 2010. – № 2(4). – С. 66–70.
2. Сергиенко И. В. Автоматизированные системы обработки данных / И. В. Сергиенко, И. Н. Парасюк, Н. И. Тукалевская. – К. : Наукова думка, 1976. – 256 с.
3. Основы автоматического управления / под ред. В. С. Пугачева. – М. : Наука, 1965. – 680 с.
4. Fuller A. The replacement of saturation constrains by energy constrains in control optimization theory / A. Fuller // International Journal of Control. – 1997. – № 3. – P. 201–277.
5. Кац А. М. К вопросу о вычислении квадратичного качества регулирования / А. М. Кац // Прикладная математика и механика. – 1952. – № 16, Вып. 3. – С. 382–384.
6. Тимченко А. А. Основы информатики системного проектирования объектов новой техники / А. А. Тимченко, А. А. Радионов. – К. : Наукова думка, 2000. – 152 с.
7. Острем К. Ю. Введение в стохастическую теорию управления / К. Ю. Острем ; пер. с англ. под ред. Н. С. Райбмана. – М. : Мир, 1973. – 322 с.
8. Химмельблау Д. Прикладное нелинейное программирование / Д. Химмельбау ; пер. с англ. под ред. М. С. Быховского. – М. : Мир, 1975. – 536 с.
9. Хеллман О. Введение в теорию оптимального поиска / О. Хеллман ; пер. с англ. под ред. Н. Н. Моисеева. – М. : Наука, 1985. – 248 с.
10. Хоменюк В. Б. Элементы теории многоцелевой оптимизации / В. Б. Хоменюк. – М. : Наука, 1983. – 254 с.
11. Майника Э. Алгоритмы оптимизации на сетях и графах / Э. Майника ; пер. с англ. под ред. Е. Масловского. – М. : Мир, 1981. – 324 с.

Аннотация: В статье рассматривается подход к оптимизации показателей качества системы защиты информации на основе модели системы автоматического управления. Предлагаются алгоритмы вычисления интегральных квадратичных оценок и векторной целевой функции для их минимизации с соблюдением критериев устойчивости.

Ключевые слова: система автоматического управления, весовой коэффициент, векторная целевая функция, интегральная квадратичная оценка.

Abstract: The paper presents an approach to optimize the quality parameters of information protection system based on the model of the automatic control. Algorithms for calculating the integral quadratic estimates and the vector of the purposive function to minimize them in compliance with the sustainability criteria are proposed.

Key words: automatic control system, weighting coefficient, vector purposive function, integral quadratic estimates.