В. П. Майданюк, А. М. Пєтух

ОБРОБКА СИГНАЛІВ

Міністерство освіти і науки, молоді та спорту України
Вінницький національний технічний університет

В. П. Майданюк, А. М. Пєтух

ОБРОБКА СИГНАЛІВ

Навчальний посібник

Вінниця
ВНТУ
2012

УДК 621.391 (075)
ББК 32.811.3я73
М 91

Рекомендовано до друку Вченою радою Вінницького національного технічного університету Міністерства освіти і науки, молоді та спорту України (протокол № 3 від 28.10.2010 р.)

Рецензенти:
В. А. Лужецький, доктор технічних наук, професор
В. П. Кожем'яко, доктор технічних наук, професор
В. М. Лисогор, доктор технічних наук, професор
А. С. Довбиш, доктор технічних наук, професор

Майданюк, В. П.
М91 Обробка сигналів: навчальний посібник / В. П. Майданюк, А. М. Пєтух. – Вінниця : ВНТУ, 2012. – 144 с.

В посібнику розглянуто основи обробки сигналів. Матеріал містить 10 розділів, які відповідають 10 двогодинним лекціям. Крім того, в додатках наведено основи роботи в середовищі MATLAB. Навчальний посібник призначений для студентів напрямів підготовки 6.050101 – «Комп'ютерні науки» та 6.050103 - «Програмна інженерія» для вивчення дисципліни "Обробка сигналів і зображень", а також окремих розділів інших споріднених дисциплін.

УДК 621.391 (075)
ББК 32.811.3я73

ЗМІСТ

ВСТУП 5
1 ЧАСОВА ФОРМА ПОДАННЯ СИГНАЛІВ 6
1.1 Загальна характеристика сигналів 6
1.2 Класифікація сигналів 7
1.3 Подання сигналів в часовій формі 8
1.4 Енергетичні характеристики дійсного сигналу 12
Контрольні запитання 13
2 СПЕКТРАЛЬНА ТЕОРІЯ СИГНАЛІВ 14
2.1 Узагальнена спектральна теорія сигналів 14
2.2 Ортогональне подання сигналів 14
Контрольні запитання 17
3 ГАРМОНІЧНИЙ АНАЛІЗ ПЕРІОДИЧНИХ СИГНАЛІВ 18
3.1 Гармонічний аналіз періодичних сигналів. Ряд Фур'є 18
3.2 Інші форми ряду Фур'є 20
Контрольні запитання 22
4 ГАРМОНІЧНИЙ АНАЛІЗ НЕПЕРІОДИЧНИХ СИГНАЛІВ 23
4.1 Пряме і зворотне перетворення Фур'є 23
4.2 Основні теореми про спектри 25
4.3 Розподіл енергії в спектрі сигналу 27
4.4 Випадковий процес як модель сигналу 29
Контрольні запитання і вправи 29
5 ПЕРЕТВОРЕННЯ НЕПЕРЕРВНИХ СИГНАЛІВ В ДИСКРЕТНІ 30
5.1 Теорема Котельникова 30
5.2 Періодичність спектра дискретної функції 33
Контрольні запитання і вправи 36
6 ДИСКРЕТНЕ ПЕРЕТВОРЕННЯ ФУР'Є 37
6.1 Основні формули дискретного перетворення Фур'є 37
6.2 Ідеї швидкого обчислення ДПФ 37
6.3 ШПФ Кулі-Тьюкі 38
Контрольні запитання і вправи 40
7 ШПФ КУЛІ-ТЬЮКІ З ПРОРІДЖЕННЯМ ЗА ЧАСТОТОЮ ТА ГРАФИ ШПФ 41
7.1 ШПФ Кулі-Тьюкі з прорідженням за частотою 41

7.2 Використання ШПФ для прискореного обчислення зворотного ДПФ 42
7.3 Графи ШПФ 43
7.4 Несинусоїдні ортогональні перетворення 45
Контрольні запитання і вправи 47
8 ЗАГАЛЬНІ ПОНЯТТЯ ПРО ЦИФРОВІ ФІЛЬТРИ 48
8.1 Z-перетворення 48
8.2 Цифрові фільтри – загальні поняття (частотна характеристика, імпульсна характеристика, класифікація) 48
8.3 Нерекурсивні фільтри зі симетричними коефіцієнтами 50
Контрольні запитання і вправи 52
9 СИНТЕЗ НЕРЕКУРСИВНИХ ЦИФРОВИХ ФІЛЬТРІВ 53
9.1 Синтез нерекурсивного фільтра низької частоти (ФНЧ) методом найменших квадратів 53
9.2 Явище Гіббса і функції вікна 55
9.3 Частота дискретизації і схеми допусків 59
9.4 Обчислення коефіцієнтів високочастотних, смугових і режекторних фільтрів 60
Контрольні запитання і вправи 65
10 РЕКУРСИВНІ ФІЛЬТРИ 66
10.1 Загальні поняття про рекурсивні фільтри 66
10.2 Типи РФ 69
10. 3 Каталог параметрів РФ 70
10.4 Частотне перетворення 71
Контрольні запитання і вправи 72
ВИСНОВКИ 73
ЛІТЕРАТУРА 74
СЛОВНИК ОСНОВНИХ ТЕРМІНІВ (GLOSSARY) 75
ДОДАТОК А ОСНОВИ РОБОТИ В МАТЛАБ 76
ДОДАТОК Б ОГЛЯД ФУНКЦІЙ MATLAB 126
ДОДАТОК В КОМПОНЕНТИ MATLAB 130
ДОДАТОК Г ПРОГРАМА SPTOOL 135

ВСТУП

Курси теорії сигналів *(signals)* в наш час займають одне з центральних місць серед дисциплін професійної підготовки не лише радіоінженерів - розробників радіотехнічних систем різного призначення, але і всіх спеціальностей, тією чи іншою мірою пов'язаних з реєстрацією, поданням, обробкою і використанням інформаційних даних різної природи. Це визначається тим, що інформація, разом з матерією і енергією, належить до фундаментальних філософських категорій природознавства і є однією з рушійних сил сучасного розвитку науки, техніки і людської цивілізації в цілому. Але інформація не існує в явній фізичній формі. Носіями інформації є сигнали в будь-якій формі їх матеріального подання в межах систем, поза якими поняття сигналів також не мають сенсу. Професійно грамотна і ефективна реєстрація інформації, її обробка, інтерпретація і використання можливі тільки при хороших знаннях теорії сигналів.

Даний курс містить десять невеликих розділів, кожний з яких відповідає окремій лекції, і додатки. Перші чотири розділи присвячені питанням обробки неперервних сигналів в часовій і частотній областях, наступні шість розділів охоплюють питання обробки сигналів в дискретній формі, де розглянуто дискретне перетворення Фур'є *(discrete Fourier transform - DFT)* та швидкі методи його обчислення, синтез нерекурсивних *(finite impulse response - FIR)* та рекурсивних *(infinite impulse response - IIR)* цифрових фільтрів *(digital filter).* Кожний розділ завершується спеціально підібраними контрольними запитаннями та вправами, які допоможуть кращому засвоєнню матеріалу та набуттю практичних навичок у використанні методів теорії обробки сигналів. Матеріали, що не ввійшли в склад навчального посібника, винесені для самостійної роботи студентів. В додатках наведені основні відомості із роботи в системі MATLAB *(MATrix LABoratory* - матрична лабораторія), яка з'явилося у 1984 р. і стала світовим стандартом в області наукових і технічних розрахунків.

Навчальний посібник призначений для студентів напрямів підготовки 6.050101 - «Комп'ютерні науки» та 6.050103 - «Програмна інженерія» і може використовуватись при вивченні дисципліни «Обробка сигналів і зображень», а також для окремих розділів інших споріднених дисциплін.

1 ЧАСОВА ФОРМА ПОДАННЯ СИГНАЛІВ

1.1 Загальна характеристика сигналів

В широкому значенні слова під сигналом розуміють матеріальний носій інформації. При цьому до сигналів відносять як природні сигнали, так і сигнали створені з певною метою. Природні сигнали, наприклад, космічні сигнали, світлові сигнали, які дозволяють бачити навколишній світ. Прикладом спеціальних створених сигналів можуть служити сигнали, які генеруються з метою вилучення інформації про зміни в об’єкті або процесі (еталонні сигнали) [1-2].

Ми будемо використовувати поняття «сигнал» у вузькому значенні, як сигнал спеціально створений для передачі повідомлення в інформаційній системі. Матеріальну основу сигналу складає, будь-який фізичний об’єкт або процес, який називається носієм інформації (повідомлення). Носій стає сигналом в процесі модуляції. Параметри носія, які змінюються в часі відповідно до повідомлення, що передається, називають інформативними.

Як носій інформації використовують коливання різної природи, частіше всього гармонічні, включаючи окремий випадок – постійний стан (ω=0). В технічних інформаційних системах найбільш широке розповсюдження отримали носії у вигляді коливань електричної напруги і струму. При використанні електричних гармонічних коливань інформативними можуть бути такі параметри, як амплітуда, частота і фаза.

Сигнали як фізичні процеси можна вивчати за допомогою приладів – осцилографів, вольтметрів і т. д. Однак такий емпіричний метод має недоліки: явища, що спостерігаються, мають частковий характер, що не дозволяє судити про їх фундаментальні властивості. Для того, щоб зробити сигнал об’єктом теоретичних досліджень, необхідно вказати спосіб їх математичного опису або іншими словами, створити математичну модель сигналу, що досліджується.

Математичною моделлю сигналу може бути, наприклад, функціональна залежність, аргументом якої є час: s(t), f(t) і т. д. Математична модель дозволяє абстрагуватись від конкретної природи носія сигналу. Ми отримуємо можливість описувати тільки ті властивості сигналу, які об’єктивно є найважливішими, ігноруючи при цьому вторинні

ознаки. Знаючи математичну модель сигналу, можна порівнювати ці сигнали між собою, виконувати їх класифікацією.

1.2 Класифікація сигналів

Детерміновані і випадкові сигнали

Якщо математична модель сигналу дозволяє точно передбачити миттєві значення сигналу в будь-які моменти часу, то сигнал називається детермінованим.

Сигнал, значення якого в будь-які моменти часу будуть випадковими величинами, називаючи випадковим.

Одновимірні і багатовимірні сигнали

Одновимірний сигнал є функцією одного аргументу $\{u(x), x \in X\}$. У більшості випадків аргументом функції $u(x)$ є час - $u(t)$.

Упорядкований набір n одновимірних сигналів утворює багатоканальний (багатовимірний або векторний) сигнал:

$$\overset{\rho}{u}(t) = \{u_1(t), u_1(t), ..., u_n(t)\}, \quad t \in T. \tag{1.1}$$

Або інша форма запису:

$$u(\overset{\nu}{x}) = u(x_1, x_2, ...x_n),$$

де число n – розмірність вектора $\overset{\nu}{u}(t)$.

Коли одна зі змінних розуміється як час, а три інші змінні простору, то такі сигнали називають просторово-часові:

$$u(\overset{\nu}{r}, t) = u(x, y, z, t). \tag{1.2}$$

Неперервні, дискретні і дискретно-неперервні сигнали

Сигнал вважається дискретним за даним параметром, якщо число значень, яке може прийняти цей параметр, зліченне (скінченне). Сигнал дискретний за одним параметром і неперервний за іншим називають дискретно-неперервним. Інакше сигнал вважається неперервним. Оскільки неперервні сигнали можуть мати розриви, то коректніше називати їх терміном континуальний. Відповідно до цього існують такі математичні моделі детермінованих сигналів:

- неперервна функція неперервного аргументу (рис. 1.1, а);
- неперервна функція дискретного аргументу (рис. 1.1, б);

- дискретна функція неперервного аргументу (рис. 1.1, в);
- дискретна функція дискретного аргументу (рис. 1.1, г).

Особлива різновидність останнього типу сигналів є цифрові сигнали. Для них характерно те, що відліки подані у формі чисел (двійкових чисел) [2-3].

1.3 Подання сигналів в часовій формі

При аналізі процесу проходження довільного детермінованого сигналу через лінійну систему або систему близьку до лінійної, його зручно подати у вигляді відповідним чином вибраної сукупності елементарних сигналів. Знаючи реакцію системи на елементарний сигнал, можна, користуючись методом суперпозиції, визначити реакцію системи на сигнал довільної форми. До елементарних детермінованих сигналів відносять одиничну функцію *(unit step function (Heaviside step function))*, ідеальний одиничний імпульс *(unit impulse function (Dirac delta))* і синусоїдальний елементарний сигнал. Одинична функція і одиничний імпульс використовується при часовому поданні сигналів [2-4].

Властивості одиничної функції визначаються таким співвідношенням (рис 1.2):

$$\sigma(t-\tau)=\begin{cases}0, \text{при } t<\tau\\ 1, \text{при } t\geq\tau\end{cases} \tag{1.3}$$

Властивості ідеального одиничного імпульсу (дельта-функції Дірака) визначаються таким співвідношенням:

$$\delta(t-\tau)=\begin{cases}0, & \text{при} \quad t\neq\tau\\ \infty, & \text{при} \quad t=\tau.\end{cases} \tag{1.4}$$

Причому:

$$\int_{-\infty}^{\infty}\delta(t-\tau)dt=1 \quad \text{і} \quad \int_{0}^{t}\delta(t-\tau)dt=\sigma(t-\tau), \quad 0<\tau<t\,. \tag{1.5}$$

Одиничний імпульс характеризується нескінченно малою тривалістю і нескінченно великим рівнем і площею рівною 1.

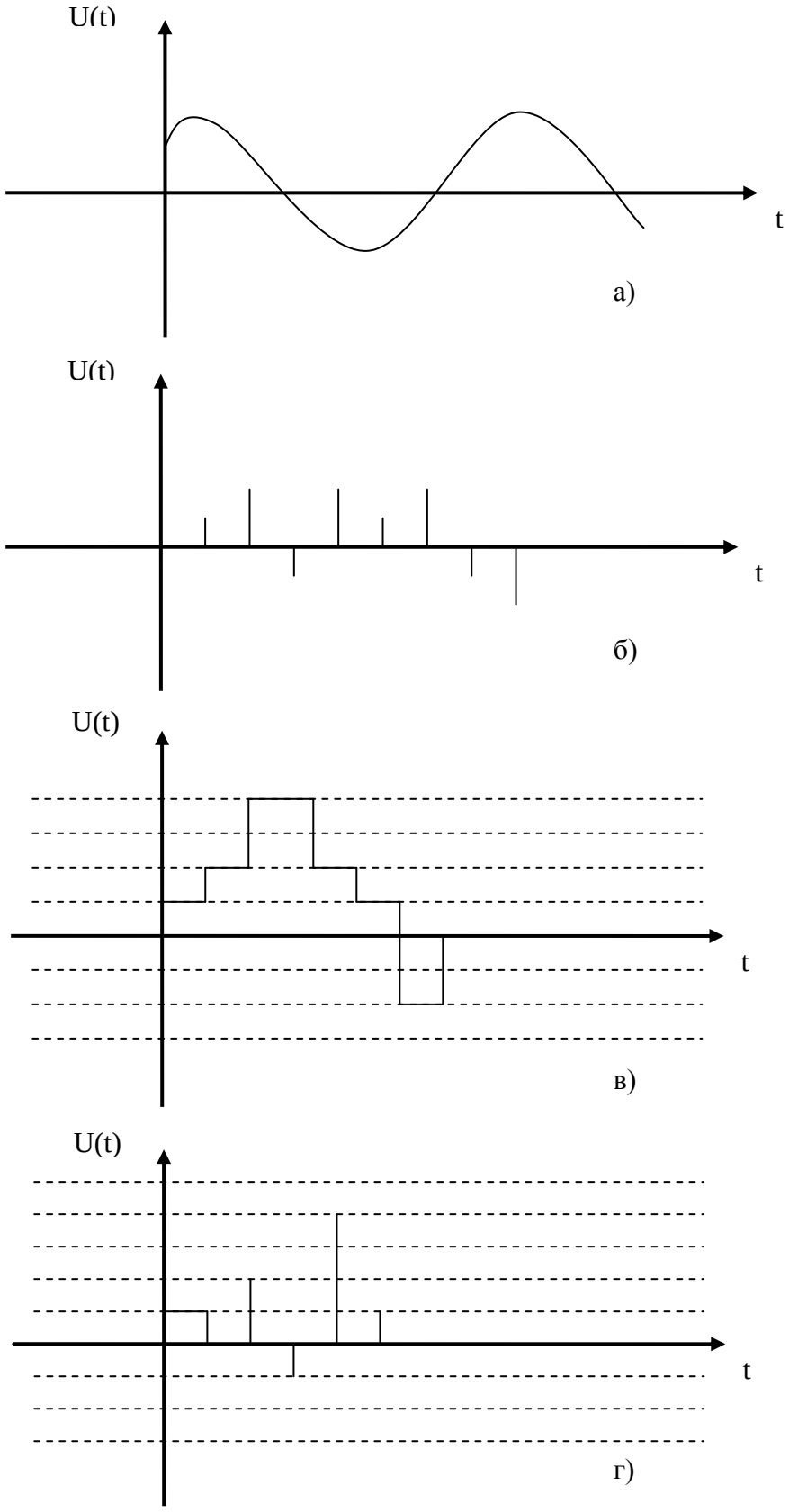


Рисунок 1.1 – Моделі детермінованих сигналів

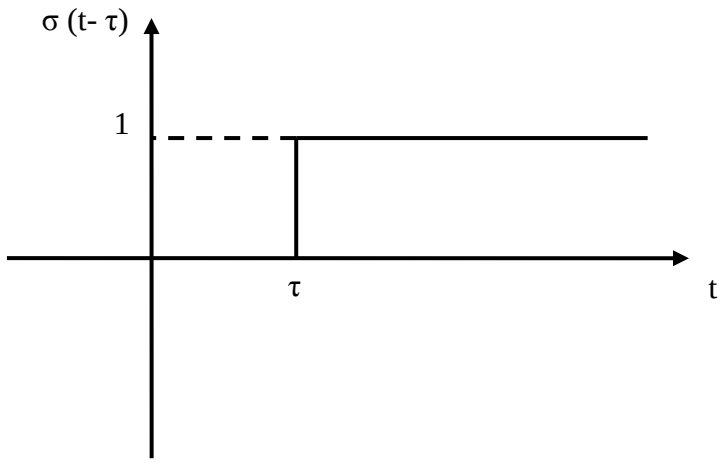


Рисунок 1.2 – Одинична функція

Розглянемо деякий сигнал s(t) (рис. 1.3). Нехай s(t)=0, при t<0. Нехай Δ, 2Δ, 3Δ - послідовні моменти часу і $\{s_1, s_2, s_3\}$ – відповідні значення сигналу. Якщо $s_0=s(0)$ – початкове значення, то поточне значення сигналу при будь-якому t приблизно дорівнює сумі ступінчастих функцій:

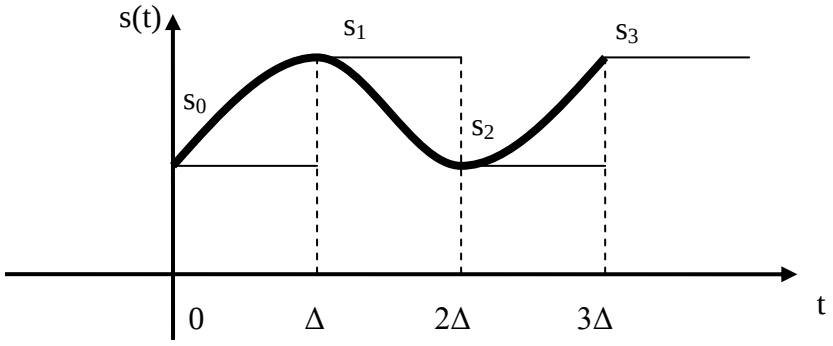


Рисунок 1.3 – Динамічне подання сигналу за допомогою одиничних функцій

$$s(t) = s_0\sigma(t) + (s_1 - s_0)\sigma(t - \Delta) + (s_2 - s_1)\sigma(t - 2\Delta) + (s_3 - s_2)\sigma(t - 3\Delta) + \ldots =$$

$$= s_0\sigma(t) + \sum_{k=1}^{\infty}(s_k - s_{k-1})\sigma(t - k\Delta) \qquad (1.6)$$

Якщо тепер крок $\Delta \to 0$, то $k\Delta \to \tau$, а $s_k - s_{k-1} \to \left(\frac{ds}{d\tau}\right)d\tau$. Таким чином, отримаємо:

$$s(t) = s_0\sigma(t) + \int_0^{\infty} \frac{ds}{d\tau}\sigma(t-\tau)d\tau. \qquad (1.7)$$

Вираз (1.7) дає динамічне подання сигналу за допомогою одиничних функцій (вираз динамічне подання підкреслює те, що процес розвивається в часі).

В іншому способі динамічного подання сигналу елементарними сигналами служать прямокутні імпульси (рис. 1.4). Ці імпульси безпосередньо примикають один до одного, утворюючи послідовність, вписану в криву або описану навкруг неї.

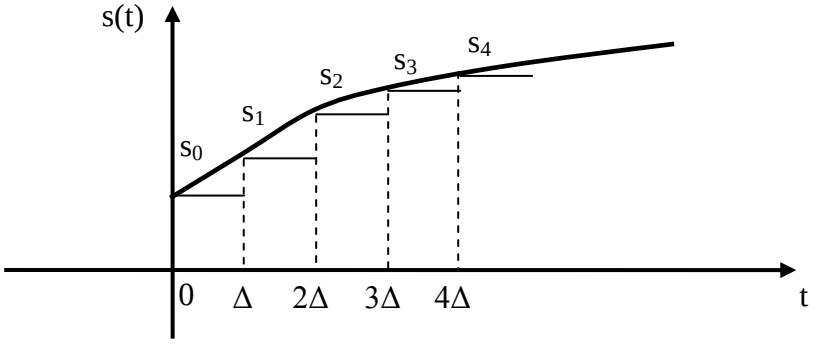


Рисунок 1.4 – Динамічне подання сигналу за допомогою одиничного імпульсу

Якщо s_k – значення сигналу в k-у відліку, то елементарний імпульс з номером k подається так:

$$\eta_k(t) = s_k[\sigma(t-t_k) - \sigma(t-t_k-\Delta)] \qquad (1.8)$$

Відповідно до принципу динамічного подання:

$$s(t) = \sum_{k=-\infty}^{\infty} \eta_k(t) \qquad (1.9)$$

В цій сумі відмінним від нуля буде тільки один член, який відповідає тому номеру k, який задовольняє нерівність: $t_k < t < t_{k+1}$.

Підставимо (1.8) в (1.9), помноживши і поділивши на величину кроку Δ, отримаємо:

$$s(t) = \sum_{k=-\infty}^{\infty} s_k \frac{1}{\Delta}[\sigma(t-t_k) - \sigma(t-t_k-\Delta)]\cdot\Delta \qquad (1.10)$$

Переходячи до границі при $\Delta \to 0$, замінимо суму інтегруванням за формальною змінною τ, диференціал якої $d\tau$ відповідає величині Δ.

Оскільки, $\lim_{\Delta\to 0}[\sigma(t-\tau)-\sigma(t-\tau-\Delta)]\frac{1}{\Delta}=\delta(t-\tau)$, отримаємо

$$s(t)=\int_{-\infty}^{\infty}s(\tau)\cdot\delta(t-\tau)d\tau \qquad (1.11)$$

Звідси витікає структурна схема системи, яка виконує вимірювання миттєвих значень аналогового сигналу s(t) (рис. 1.5). Система складається з двох ланок помножувача і інтегратора. Вимірювання буде тим точнішим, чим коротший реальний імпульс, який наближено подає дельта-функцію. В цьому полягає фільтруюча властивість дельта-функції.

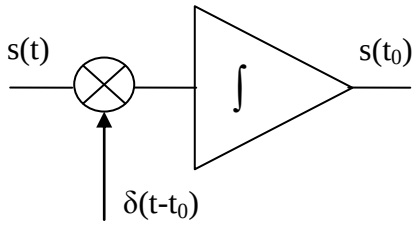


Рисунок 1.5 – Структурна схема системи для вимірювання миттєвих значень

1.4 Енергетичні характеристики дійсного сигналу

Дійсний сигнал s (t) має такі енергетичні характеристики [4].

1. Миттєва потужність:

$$p(t)=s^2(t). \qquad (1.12)$$

Якщо s(t) – напруга або струм, то p(t) – миттєва потужність, що виділяється на опорі в 1 Ом.

2. Енергія сигналу на інтервалі t_2, t_1:

$$E=\int_{t_1}^{t_2}p(t)dt=\int_{t_1}^{t_2}s^2(t)dt. \qquad (1.13)$$

Середня потужність сигналу на інтервалі t_2, t_1:

$$\frac{E}{t_2-t_1}=\frac{1}{t_2-t_1}\int_{t_1}^{t_2}s^2(t)dt=\overline{s^2(t)}. \qquad (1.14)$$

Контрольні запитання

1. Поясніть поняття сигнал у «широкому значенні».
2. Поясніть поняття сигнал у «вузькому значенні».
3. Що використовують як носій інформації?
4. Коли носій інформації стає сигналом?
5. Що таке модуляція?
6. Які ви знаєте види модуляції?
7. Які параметри носія називають інформативними?
8. Які є методи дослідження сигналів?
9. Які недоліки емпіричного методу дослідження сигналів?
10. Охарактеризуйте математичну модель сигналу.
11. Наведіть класифікацію сигналів.
12. Наведіть означення детермінованих сигналів.
13. Наведіть означення випадкових сигналів.
14. Які вам відомі математичні моделі детермінованих сигналів?
15. Чим відрізняється дискретний і цифровий сигнали?
16. Охарактеризуйте одновимірні та багатовимірні сигнали.
17. Дайте класифікацію дискретно-неперервних сигналів.
18. Які сигнали відносять до елементарних детермінованих сигналів?
19. Як визначити реакцію лінійної системи на сигнал довільної форми?
20. Які елементарні сигнали використовуються при часовій формі подання сигналів?
21. Охарактеризуйте властивості одиничної функції.
22. Яким співвідношенням визначаються властивості ідеального одиничного імпульсу?
23. Наведіть формулу динамічного подання сигналу за допомогою одиничної функції.
24. Наведіть формулу динамічного подання сигналу за допомогою одиничного ідеального імпульсу.
25. Які ланки містить система, що вимірює миттєві значення аналогового сигналу s(t) ?
26. Що таке миттєва потужність?
27. Як визначається енергія сигналу?
28. Наведіть енергетичні характеристики дійсного сигналу s (t).

2 СПЕКТРАЛЬНА ТЕОРІЯ СИГНАЛІВ

2.1 Узагальнена спектральна теорія сигналів

При аналізі проходження складного періодичного сигналу s(t) через інваріантні в часі лінійні системи його можна подати у вигляді зваженої суми базисних функцій $\varphi_k(t)$ [2]:

$$s(t) = \sum_k c_k \varphi_k(t), \qquad t \in [t_1, t_2] \tag{2.1}$$

де $[t_1, t_2]$ - інтервал існування сигналу.

При вибраному наборі базисних функцій $\varphi_k(t)$ сигнал s(t) повністю визначається сукупністю безрозмірних коефіцієнтів $c_к$, які називаються *дискретним спектром (spectr) сигналу.*

Для неперіодичного сигналу для будь-якого моменту часу справедливий такий вираз:

$$s(t) = \int_{-\infty}^{\infty} s(\alpha) \cdot \varphi(\alpha, t) d\alpha, \tag{2.2}$$

де $\varphi(\alpha, t)$ - базисна функція з неперервно змінюваним параметром α;

$s(\alpha)$ - спектральна щільність.

Аналогом коефіцієнтів $c_к$ є величина $s(\alpha)d\alpha$.

Для теоретичного аналізу базисні функції $\varphi_k(t)$ вибирають так, щоб вони мали простий аналітичний вираз, забезпечували швидку збіжність ряду (2.1) для будь-яких s(t) і дозволяли легко знаходити коефіцієнти $c_к$.

2.2 Ортогональне подання сигналів

Обчислення спектральних складових сигналу спрощується при виборі як базису системи ортогональних функцій.

Систему функцій $\psi_0(t), \psi_1(t), ..., \psi_k(t), ...$ називають ортогональною на відрізку $[t_a, t_b]$, якщо[1-2]:

$$\begin{cases} \int_{t_a}^{t_b} \psi_k(t)\psi_j(t)dt = 0, & k \neq j \\ \int_{t_a}^{t_b} \psi_k{}^2(t)dt \neq 0, & k = j. \end{cases} \qquad (2.3)$$

Якщо

$$\int_{t_a}^{t_b} \psi_j{}^2(t)dt = 1, \qquad (2.4)$$

то система функцій буде ортонормальною (ортонормованою).

При $\int_{t_a}^{t_b} \psi_j{}^2 dt = \mu_j > 0$ систему нормують, помножуючи $\psi_j(t) \cdot \frac{1}{\sqrt{\mu_j}}$.

І так, нехай:

$$s(t) = \sum_k c_k \cdot \psi_k(t), \qquad t \in [t_1, t_2], \qquad (2.5)$$

де $\Psi_k(t)$ - система ортонормованих функцій, інтервал $[t_1, t_2]$ знаходиться всередині відрізка ортогональності $[t_a, t_b]$. Помножимо праву і ліву частину на $\Psi_j(t)$ і проінтегруємо на інтервалі $[t_1, t_2]$:

$$\int_{t_1}^{t_2} s(t)\Psi_j(t)dt = \sum_k c_k \int_{t_1}^{t_2} \Psi_j(t)\Psi_k(t)dt \qquad (2.6)$$

В силу справедливості (2.3) всі інтеграли в правій частині (2.6) при $k \neq j$ рівні 0, а при $k = j$ рівні 1. Таким чином:

$$c_j = \int_{t_1}^{t_2} s(t)\psi_j(t)dt. \qquad (2.7)$$

Ряд (2.5) називають узагальненим рядом Фур'є *(Fourier series)*, а сукупність коефіцієнтів $c_к$ узагальненими коефіцієнтами Фур'є або дискретним спектром сигналу [2].

Зауваження 2.1. Нормою функції $\psi_j(t)$ називають таку величину:

$$\| \psi_j \| = \sqrt{\int_{t_a}^{t_b} \psi_j^2(t)dt} . \tag{2.8}$$

За аналогією і з урахуванням (1.13):

$$\| s \|^2 = \int_{t_a}^{t_b} s^2(t)dt = E . \tag{2.9}$$

Підставивши (2.5) замість s(t) одержимо:

$$E = \int_{t_a}^{t_b} (\sum_k c_k \psi_k(t))^2 dt = \sum_k c_k^2 \int_{t_a}^{t_b} \psi_k^2(t)dt = \sum_k c_k^2 \|\psi_k\|^2 . \tag{2.10}$$

Для ортонормованих функцій:

$$\|\psi_k\|^2 = 1 \text{ і } E = \sum_k c_k^2 . \tag{2.11}$$

Знайдемо c_k для випадку, коли система функцій $\psi_k(t)$ не є ортонормованою. Аналогічно попередньому випадку помножимо ліву і праву частину на $\psi_j(t)$ і проінтегруємо на інтервалі $[t_1, t_2]$:

$$\int_{t_1}^{t_2} s(t)\psi_j(t)dt = \sum_k c_k \int_{t_1}^{t_2} \psi_k(t)\psi_j(t)dt . \tag{2.12}$$

В силу справедливості (2.3) всі інтеграли в правій частині (2.12) при $k \neq j$ рівні 0, а при $k = j$ рівні:

$$\int_{t_1}^{t_2} \psi_j^2 dt = \|\psi_j\|^2, \qquad k = j. \tag{2.13}$$

Таким чином:

$$c_j = \frac{1}{\|\Psi_j\|^2} \int_{t_1}^{t_2} s(t)\Psi_j(t)dt . \tag{2.14}$$

Узагальнений ряд Фур'є при заданій системі функцій $\Psi_k(t)$ і фіксованому числі доданків ряду забезпечує найменшу

середньоквадратичну помилку, тобто найкращу апроксимацію [2]. Нехай середньоквадратична помилка оцінюється такою величиною:

$$M = \int_{t_a}^{t_b} [s(t) - \sum_k a_k \psi_k(t)]^2 dt. \qquad (2.15)$$

Нехай $a_k = c_k + b_k$, тоді отримаємо:

$$M = \int_{t_a}^{t_b} s^2(t)dt - 2\int_{t_a}^{t_b} (s(t)\sum_k a_k \psi_k(t_1))dt + \int_{t_a}^{t_b} \sum_k [a_k \psi_k(t)]^2 dt = \\ = \int_{t_a}^{t_b} s^2(t)dt - \sum_k c_k^2 \left\|\psi_j\right\|^2 + \sum_k b_k{}^2 \left\|\psi_j\right\|^2. \qquad (2.16)$$

Звідки видно, що M досягає мінімуму при $b_k = 0$.

Контрольні запитання

1. Наведіть основні положення узагальненої спектральної теорії.
2. Що таке дискретний спектр сигналу?
3. Що таке спектральна щільність?
4. Які відмінності між дискретним спектром та спектральною щільністю?
5. Яку систему функцій називають ортогональною?
6. Яка ортогональна система функцій є ортонормованою?
7. Охарактеризуйте узагальнений ряд Фур'є?
8. Що таке норма функції?
9. Як пов'язані енергія сигналу і норма функції?
10. Як знайти коефіцієнти узагальненого ряду Фур'є?
11. Що таке норма функцій?
12. Наведіть формулу знаходження коефіцієнтів узагальненого ряду Фур'є для випадку, коли система функцій $\psi_k(t)$ ортонормована.
13. Наведіть формулу знаходження коефіцієнтів узагальненого ряду Фур'є для випадку, коли система функцій $\psi_k(t)$ не є ортонормованою.
14. Як називають сукупність коефіцієнтів узагальненого ряду Фур'є?
15. Яку середньо-квадратичну помилку забезпечує ряд Фур'є?

3 ГАРМОНІЧНИЙ АНАЛІЗ ПЕРІОДИЧНИХ СИГНАЛІВ

3.1 Гармонічний аналіз періодичних сигналів. Ряд Фур'є

Розглянемо, які функції потрібно вибирати як базові при аналізі інваріантних в часі лінійних систем. Розв'язки завжди містять комплексні експоненціальні функції часу, оскільки вони інваріантні відносно операцій диференціювання і інтегрування [2-4]. Дійсно:

$$(e^{pt})' = pe^{pt}$$

$$\int e^{pt}dt = \frac{1}{p}e^{pt} + c.$$

Якщо $p = j\omega$, то маємо перетворення Фур'є. Якщо $p = s + j\omega$ - перетворення Лапласа.

Тобто, при розкладі періодичного сигналу s(t) в ряд Фур'є як ортогональну систему беруть функції $\ldots e^{-j2\omega_1 t}, e^{-j\omega_1 t}, 1, e^{j\omega_1 t}, e^{j2\omega_1 t} \ldots$ Інтервал ортогональності збігається з періодом $T = \frac{2\pi}{\omega_1}$ функції s(t). Таким чином:

$$s(t) = \sum_{k=-\infty}^{\infty} c_k e^{jk\omega_1 t}. \qquad (3.1)$$

Вираз (3.1) задає ряд Фур'є в комплексній формі.

З урахуванням того, що $\omega_1 = \frac{2\pi}{T}$ - частотний інтервал між сусідніми гармоніками:

$$s(t) = \sum_{k=-\infty}^{\infty} c_k e^{jk\frac{2\pi}{T}t}. \qquad (3.2)$$

З урахуванням (2.14) коефіцієнти $c_к$ визначаються так:

$$c_k = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} s(t)e^{-jk\omega_1 t}dt. \qquad (3.3)$$

Зауваження 3.1. Для системи функцій $\psi_k(t)$, які приймають комплексні значення, умова ортогональності така:

$$\int_{t_a}^{t_b} \psi_k(t) \cdot \psi_j^*(t)dt = 0 \qquad k \neq j. \tag{3.4}$$

При $k = j$, отримаємо квадрат норми функції:

$$\|\psi_k\|^2 = \int_{t_a}^{t_b} \psi_k(t) \cdot \psi_k^*(t)dt = \int_{t_a}^{t_b} |\psi_k(t)|^2 dt. \tag{3.5}$$

Коефіцієнти Фур'є у цьому випадку визначаються так:

$$c_j = \frac{1}{\|\psi_j\|^2} \int_{t_1}^{t_2} s(t)\psi_j^*(t)dt. \tag{3.6}$$

Якщо $\psi_j = e^{jk\omega_1 t}$, то

$$\|\Psi_k\|^2 = \int_{-\frac{T}{2}}^{\frac{T}{2}} e^{jk\omega_1 t} \cdot e^{-jk\omega_1 t}dt = T. \tag{3.7}$$

Що і пояснює наявність у виразі (3.3) множника 1/T.

Сукупність коефіцієнтів $c_к$ називають частотним спектром періодичного сигналу. Окремі складові гармоніками.

Коефіцієнти $c_к$ в загальному випадку є комплексними величинами. З урахуванням того, що

$$e^{-jk\omega_1 t} = \cos k\omega_1 t - j\sin k\omega_1 t, \tag{3.8}$$

отримаємо:

$$c_k = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} s(t)\cos(k\omega_1 t)dt - j\frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} s(t)\sin(k\omega_1 t)dt. \tag{3.9}$$

Позначимо:

$$\tilde{a}_k = \frac{1}{T}\int_{\frac{-T}{2}}^{\frac{T}{2}} s(t)\cos(k\omega_1 t)dt, \qquad (3.10)$$

$$\tilde{b}_k = \frac{1}{T}\cdot\int_{\frac{-T}{2}}^{\frac{T}{2}} s(t)\sin(k\omega_1 t)dt. \qquad (3.11)$$

Тоді:

$$\varphi_k = -\text{arctg}\frac{b_k}{a_k}, \qquad (3.12)$$

$$|c_k| = \sqrt{\tilde{a}_k^{\,2} + \tilde{b}_k^{\,2}}, \qquad (3.13)$$

$$c_k = |c_k| e^{-j\varphi_k}. \qquad (3.14)$$

Підставивши (3.14) в (3.2), отримаємо:

$$s(t) = \sum_{k=-\infty}^{\infty} |c_k| e^{j(k\omega_1 t - \varphi_k)}. \qquad (3.15)$$

3.2 Інші форми ряду Фур'є

Перейдемо до тригонометричної форми ряду Фур'є. Розкривши знак суми у виразі (3.15), одержимо [4]:

$$s(t) = \ldots|c_{-2}| e^{j(-2\omega_1 t - \varphi_{-2})} + |c_{-1}| e^{j(-2\omega_1 t - \varphi_{-1})} + |c_0| e^{-j\varphi_0} +$$
$$+ |c_1| e^{j(\omega_1 t - \varphi_1)} + |c_2| e^{j(2\omega_1 t - \varphi_2)} + \ldots$$

З урахуванням того, що:

$$\ldots|c_{-2}| = |c_2|, \quad |c_{-1}| = |c_1|\ldots,$$

$$\ldots\varphi_{-2} = -\varphi_2, \quad \varphi_{-1} = -\varphi_1, \quad \varphi_0 = 0\ldots,$$

отримаємо:

$$s(t) = \ldots + |c_2| e^{j(-2\omega_1 t + \varphi_2)} + |c_1| e^{j(-\omega_1 t + \varphi_1)} + |c_0| + |c_1| e^{j(\omega_1 t - \varphi_1)} +$$
$$+ |c_2| e^{j(2\omega_1 t - \varphi_2)} + \ldots = |c_0| + |c_2|[\cos(2\omega_1 t - \varphi_2) - j\sin(2\omega_1 t - \varphi_2) +$$
$$+ \cos(2\omega_1 t - \varphi_2) + j\sin(2\omega_1 t - \varphi_2)] + +|c_1|[\cos(\omega_1 t - \varphi_1) - j\sin(\omega_1 t - \varphi_1) +$$
$$+ \cos(\omega_1 t - \varphi_1) + j\sin(\omega_1 t - \varphi_1)] + \ldots = |c_0| + \sum_{k=1}^{\infty} 2|c_k| \cos(k\omega_1 t - \varphi_k).$$

Позначимо:

$$|c_0| = C_0 = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} s(t)dt,$$

$$2|c_k| = C_k.$$

Тоді вирази

$$s(t) = |c_0| + \sum_{k=1}^{\infty} 2|c_k| \cos(k\omega_1 t - \varphi_k),$$

$$s(t) = C_0 + \sum C_k \cos(k\omega_1 t - \varphi_k) \qquad (3.16)$$

задають тригонометричну форму ряду Фур'є.

Сукупність коефіцієнтів $C_к$ називають спектром амплітуд, а φ_k - спектром фаз. Спектр амплітуд і спектр фаз повністю визначають структуру частот спектра періодичного коливання. Для багатьох застосувань достатньо знати спектр амплітуд. Спектр періодичної функції має вигляд, наведений на рис. 3.1.

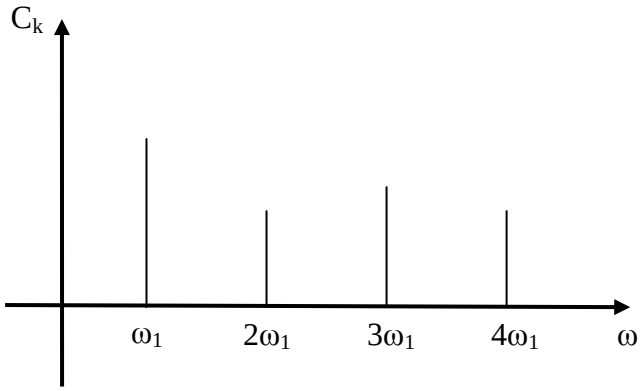


Рисунок 3.1 – Дискретний спектр періодичного коливання

Його називають лінійчастим або дискретним, або ще гармонічним. Він складається з рівновіддалених (еквідистантних) спектральних ліній. Частоти гармонік знаходяться у простих кратних співвідношеннях. Якщо функція s(t) парна, то в тригонометричному записі ряду залишаються лише косинусні складові, для непарної функції - тільки синусні.

Дискретний спектр можуть мати і неперіодичні коливання. Наприклад, складне коливання, яке є результатом додавання двох синусоїдальних коливань з нерівними частотами ω_1 і $\sqrt{2}\omega_1$. Його спектр складається з двох спектральних ліній.

Контрольні запитання

1. Які функції найчастіше вибирають базовими при аналізі лінійних систем?
2. Чому комплексні експоненціальні функції часу вибирають як базові при аналізі інваріантних в часі лінійних систем?
3. Наведіть умову ортогональності для комплексних функцій.
4. Наведіть вираз для ряду Фур'є в комплексній формі.
5. Що таке частотний спектр періодичного сигналу?
6. Наведіть формулу для обчислення коефіцієнтів частотного спектра періодичного сигналу.
7. Знайдіть спектр фаз з урахуванням того, що коефіцієнти частотного спектра є комплексними величинами.
8. Що таке гармоніка?
9. Наведіть вираз для ряду Фур'є в тригонометричній формі.
10. Як називають спектр періодичної функції?
11. Що таке спектр амплітуд?
12. Як пов'язаний частотний спектр зі спектром амплітуд і фаз?
13. Яку інформацію дає спектр фаз?
14. Чи завжди при обробці сигналів потрібно знати спектр фаз?
15. Поясніть термін «еквідистантний».
16. Чим відрізняється дискретний спектр від лінійчастого?
17. Чи можуть мати дискретний спектр неперіодичні коливання?
18. Наведіть умову, при якій неперіодичне коливання може мати дискретний спектр.

4 ГАРМОНІЧНИЙ АНАЛІЗ НЕПЕРІОДИЧНИХ СИГНАЛІВ

4.1 Пряме і зворотне перетворення Фур'є

Повернемося до комплексної форми ряду Фур'є:

$$s(t) = \sum_{k=-\infty}^{\infty} c_k e^{j\omega_1 kt} = \sum_{k=-\infty}^{\infty} c_k e^{j\frac{2\pi}{T}kt}, \tag{4.1}$$

$$c_k = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} s(t)e^{-j\frac{2\pi kt}{T}}dt. \tag{4.2}$$

Підставимо (4.2) в (4.1):

$$\omega_1 = \frac{2\pi}{T} \qquad T = \frac{2\pi}{\omega_1}$$

$$s(t) = \sum_{k=-\infty}^{\infty} e^{j\frac{2\pi}{T}kt}\frac{\omega_1}{2\pi}\int_{-\frac{T}{2}}^{\frac{T}{2}} s(t)e^{-j\frac{2\pi kt}{T}}dt. \tag{4.3}$$

Будемо розглядати неперіодичний сигнал як періодичний з нескінченним періодом, тобто $T \to \infty$, звідки:

$$\omega_1 \to d\omega$$

$$\frac{2\pi}{T}k \to \omega.$$

Сума перейде в інтеграл і вираз (4.3) прийме такий вигляд:

$$s(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{j\omega t}d\omega \int_{-\infty}^{\infty} s(t)e^{-j\omega t}dt. \tag{4.4}$$

Позначимо другий інтеграл як

$$S(j\omega) = \int_{-\infty}^{\infty} s(t)e^{-j\omega t}dt, \tag{4.5}$$

тоді

$$s(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{j\omega t} dt, \qquad (4.6)$$

де величина $S(j\omega)$ – спектральна щільність або спектральна характеристика функції $s(t)$.

Вираз (4.5) називають прямим перетворенням Фур'є *(Fourier transform)*, а вираз (4.6) - зворотним перетворенням Фур'є [4-5]. Ці дві формули називають ще основними формулами теорії спектрів.

З урахуванням того, що

$$\begin{aligned} e^{-j\omega t} &= \cos\omega t - j\sin\omega t, \\ S(j\omega) &= \int_{-\infty}^{\infty} s(t)\cos\omega t dt - j\int_{-\infty}^{\infty} s(t)\sin\omega t dt. \end{aligned} \qquad (4.7)$$

Позначимо

$$\begin{aligned} A(\omega) &= \int_{-\infty}^{\infty} s(t)\cos\omega t dt \\ B(\omega) &= \int_{-\infty}^{\infty} s(t)\sin\omega t dt, \end{aligned} \qquad (4.8)$$

тоді

$$S(j\omega) = A(\omega) - jB(\omega), \qquad (4.9)$$

де $A(\omega)$ – парна функція частоти;

$B(\omega)$ – непарна функція частоти.

Величину

$$S(\omega) = |S(j\omega)| = \sqrt{|A(\omega)|^2 + |B(\omega)|^2} \qquad (4.10)$$

називають спектральною щільністю амплітуд або амплітудно-частотною характеристикою (АЧХ), або спектром неперервного сигналу *(frequency response (FR))*. А величину

$$\varphi(\omega) = -\operatorname{arctg}\frac{B(\omega)}{A(\omega)} \qquad (4.11)$$

- фазочастотною характеристикою *(Phase response (PR))* суцільного спектра неперервного сигналу (ФЧХ).

З урахуванням (4.10) та (4.11) подамо комплексну величину $S(j\omega)$ в експоненціальній формі:

$$S(j\omega) = S(\omega)e^{-j\varphi(\omega)}. \tag{4.12}$$

Тоді

$$\begin{aligned} s(t) &= \frac{1}{2\pi}\int S(\omega)e^{-j\varphi(\omega)}e^{j\omega t}d\omega = \frac{1}{2\pi}\int_{-\infty}^{\infty} S(\omega)e^{j(\omega t-\varphi(\omega))}d\omega = \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} S(\omega)\cos[\omega(t)-\varphi(\omega)]d\omega + \frac{j}{2\pi}\int_{-\infty}^{\infty} S(\omega)\sin[\omega(t)-\varphi(\omega)]d\omega. \end{aligned} \tag{4.13}$$

Оскільки функція $\sin()$ непарна, то другий інтеграл у виразі (4.13) рівний нулю і

$$s(t) = \frac{1}{\pi}\int_{0}^{\infty} S(\omega)\cos[\omega t-\varphi(\omega)]dt. \tag{4.14}$$

Вираз (4.14) є тригонометричною формою перетворення Фур'є.

4.2 Основні теореми про спектри

1. Спектр суми функцій рівний сумі їх спектрів. Нехай

$$s(t) = s_1(t) + s_2(t).$$

Тоді

$$S(j\omega) = S_1(j\omega) + S_2(j\omega) \tag{4.15}$$

2. Якщо для функції $s(t)$ $S(j\omega)$ – комплексний спектр, то для похідної $s'(t)$ комплексний спектр визнається так:

$$S_{(+1)}(j\omega) = j\omega S(j\omega). \tag{4.16}$$

Для похідної n-го порядку:

$$S_{(+1)}(j\omega) = (j\omega)^n S(j\omega). \tag{4.17}$$

3. Якщо для $s(t)$ $S(j\omega)$ - комплексний спектр, то для інтеграла функції $s(t)$

$$s_{(-1)}(t) = \int_{-\infty}^{t} s(x)dx, \tag{4.18}$$

комплексний спектр визначається так:

$$S_{(-1)}(j\omega) = \frac{1}{j\omega} S(j\omega). \tag{4.19}$$

Функція $s(t)$ повинна відповідати такій умові:

$$\int_{-\infty}^{\infty} s(t)dt = 0. \tag{4.19}$$

4. Якщо для $s(t)$ $S(j\omega)$ - комплексний спектр, то для функції $s_1(t) = s(t-\tau)$, яка існує на інтервалі $(t_1+\tau, t_2+\tau)$, отримаємо:

$$S_\tau(j\omega) = \int_{t_1+\tau}^{t_2+\tau} s_1(t)e^{-j\omega t}dt = \int_{t_1+\tau}^{t_2+\tau} s(t-\tau)e^{-j\omega t}dt\,. \tag{4.20}$$

Введемо нову змінну інтегрування $t_n = t-\tau$ $t = t_n+\tau$ $dt = dt_n$. Тоді:

$$S_\tau(j\omega) = \int_{t_1}^{t_2} s(t_n)e^{-j\omega(t_n+\tau)}dt_n = e^{-j\omega\tau)} \int_{t_1}^{t_2} s(t_n)e^{j\omega t_n}dt_n = e^{-j\omega\tau}S(j\omega). \tag{4.21}$$

Тобто

$$S_\tau(j\omega) = e^{-j\omega\tau}S(j\omega). \tag{4.22}$$

Спектр не залежить від вибору початкового моменту відліку часу.

5. Нехай $s(t) = f(t)g(t)$, тобто сигнал $s(t)$ є добутком сигналів $f(t)$ та $g(t)$. Комплексний спектр для $s(t)$ визначається так:

$$S(j\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} G(\nu)F(\omega-\nu)d\nu. \tag{4.23}$$

Інтеграл в правій частині - згортка функцій G і F, які є комплексними спектрами для функцій $g(t)$ та $f(t)$, відповідно.

6. Складемо згортку *(convolution)* двох функцій f(t) і g(t):

$$s(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau. \qquad (4.24)$$

Тоді

$$S(j\omega) = F(j\omega)G(j\omega). \qquad (4.25)$$

Тобто, спектр згортки функцій дорівнює добутку спектрів цих функцій [5].

4.3 Розподіл енергії в спектрі сигналу

Відомо, що енергія сигналу на інтервалі $(-\infty,\infty)$ визначається так:

$$E = \int_{-\infty}^{\infty} s^2(t)dt. \qquad (4.26)$$

Оскільки

$$|S(j\omega)|^2 = S(j\omega)S(-j\omega), \qquad (4.27)$$

то

$$S(-j\omega) = \int_{-\infty}^{\infty} s(t)e^{j\omega t}dt. \qquad (4.28)$$

Тоді

$$\int_{-\infty}^{\infty} |s(j\omega)|^2 d\omega = \int_{-\infty}^{\infty} s(j\omega)(-j\omega)d\omega = \int_{-\infty}^{\infty} s(j\omega)\left[\int_{-\infty}^{\infty} s(t)e^{j\omega t}dt\right]d\omega =$$
$$= \int_{-\infty}^{\infty} s(t)\left[\int_{-\infty}^{\infty} s(j\omega)e^{j\omega t}d\omega\right]dt = 2\pi\int_{-\infty}^{\infty}|s(t)|^2 dt. \qquad (4.29)$$

Таким чином:

$$\int_{-\infty}^{\infty}|s(t)|^2 dt = \frac{1}{\pi}\int_{0}^{\infty}|S(j\omega)|^2 d\omega. \qquad (4.30)$$

Рівність (4.30) називають рівністю Парсеваля [2].

Приклад 4.1. Обчислити спектр прямокутного імпульсу (рис. 4.1):

$$s(t)=\begin{cases} h, & -\frac{\tau}{2}\le t\le\frac{\tau}{2} \\ 0, & t>\frac{\tau}{2},\quad t<-\frac{\tau}{2} \end{cases}$$

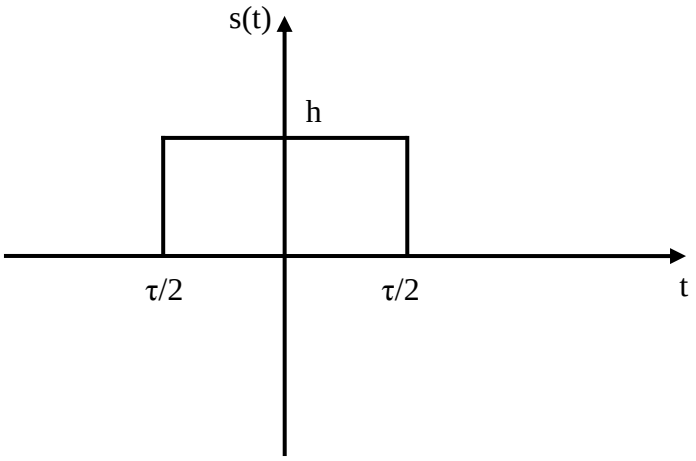


Рисунок 4.1 – Одиничний прямокутний імпульс

Розв'язування. Згідно з (4.5) спектр визначається так:

$$S(j\omega)=\int_{-\tau/2}^{\tau/2}s(t)e^{-j\omega t}dt=\int_{-\tau/2}^{\tau/2}he^{-j\omega t}dt=-\frac{h}{j\omega}e^{-j\omega t}\Big|_{-\tau/2}^{\tau/2}=q\ \frac{\sin\ \omega\tau/2}{\omega\tau/2},$$

де $q=h\tau$ - площа імпульсу.

Графік функції $S(\omega)=|S(j\omega)|$ наведено на рис. 4.2.

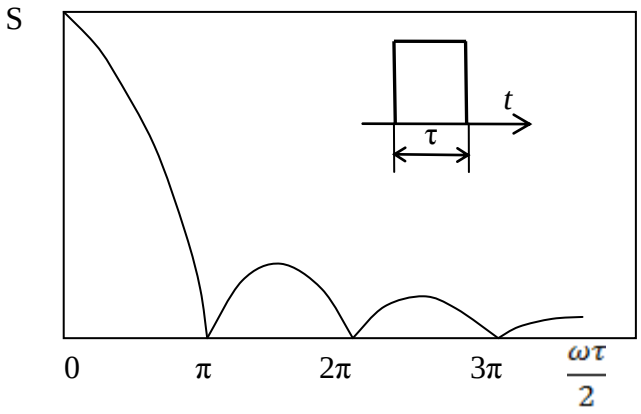


Рисунок 4.2 – Спектр прямокутного імпульсу

З рис. 4.2 видно, що спектр одиничного імпульсу має ту ж форму як і обвідна спектра періодичної послідовності таких імпульсів, причому чим більша τ тим вужчий спектр.

Інший важливий висновок: якщо тривалість сигналу обмежена, то його спектр необмежений і навпаки сигнал з обмеженим спектром триває нескінченно довго. Реальні сигнали обмежені в часі. У зв'язку з цим виникає необхідність ввести сигнали, які мають скінченну тривалість і спектр. Практичну тривалість $\tau_п$ і практичну ширину спектра $\omega_п$ вибирають із співвідношення, яке випливає з рівності Парсеваля (енергетичний критерій):

$$\frac{1}{\pi}\int_0^{\omega_п} |S(j\omega)|^2 d\omega = \frac{\eta}{\pi}\int_0^{\infty} |S(j\omega)|^2 d\omega . \qquad (4.31)$$

Коефіцієнт вибирають з діапазону $\eta = 0{,}9 - 0{,}99$. Тобто, практична ширина спектра – це ширина в якій знаходиться 90 % - 100 % енергії сигналу [2].

4.4 Випадковий процес як модель сигналу

Математичною моделлю сигналу при передачі і перетворенні сигналу є випадкова функція. Кожна детермінована функція розглядається як реалізація цього випадкового процесу [2].

Контрольні запитання і вправи

1. Наведіть основні формули теорії спектрів.
2. Що таке амплітудно-частотна характеристика?
3. Наведіть вираз для знаходження фазочастотної характеристики.
4. Наведіть вираз для тригонометричної форми перетворення Фур'є.
5. Чому дорівнює спектр суми функцій?
6. Визначити спектр добутку функцій?
7. Як знайти спектр згортки двох функцій?
8. Що визначає рівність Парсеваля?
9. Обчислити спектр імпульсу у формі рівнобедреного трикутника з основою τ і висотою h.
10. Обчислити спектр косинусоїдного імпульсу, який вирізаний з косинусоїди з періодом 2 τ і висотою h.

5 ПЕРЕТВОРЕННЯ НЕПЕРЕРВНИХ СИГНАЛІВ В ДИСКРЕТНІ

Хоча інформацію можна зберігати і передавати як у вигляді неперервних, так і дискретних сигналів, а саме - цифрових сигналів *(digital signal)*, перевага віддається дискретним сигналам, тому аналогові сигнали перетворюються в дискретні (частіше всього в цифрові). Для цього кожний неперервний сигнал піддається операції дискретизації за часом, квантуванню за рівнем та кодуванню (рівні квантування подаються двійковими числами), тобто аналого-цифровому перетворенню.

Правило вибору граничного кроку при рівномірній дискретизації з використанням моделі сигналу з обмеженим спектром сформулював В. А. Котельников.

5.1 Теорема Котельникова

Означення. Функція s(t), що допускає перетворення Фур'є і має неперервний спектр, обмежений смугою частот від 0 до $F_c = \frac{\omega_c}{2\pi}$, повністю визначається дискретним рядом своїх миттєвих значень, відрахованих через інтервали часу

$$\Delta t = \frac{1}{2Fc} \quad (\text{або } f_д \geq 2F_c), \tag{5.1}$$

де $f_д$ - частота дискретизації.

Доведення. Нехай функція s(t) має спектральну щільність $S(j\omega)$, причому $S(j\omega) = 0$ при $|\omega| > \omega_c$, де ω_c – найбільша частота у спектрі сигналу s(t). Використовуючи зворотне перетворення Фур'є, отримаємо:

$$s(t) = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} S(j\omega)e^{j\omega t}d\omega. \tag{5.2}$$

Для моментів часу $t_n = n\Delta t = \frac{n}{2F_c} = \frac{n\pi}{\omega_c}$, де n – ціле число, маємо:

$$s(t_n) = s(\frac{n\pi}{\omega_c}) = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} S(j\omega) e^{j\frac{n\pi}{\omega_c}\omega} d\omega \qquad (5.3)$$

Розкладемо функцію $S(j\omega)$ на інтервалі існування $(-\omega_c, \omega_c)$ в ряд Фур'є за частотами, періодично продовжуючи її з періодом $2\omega_c$:

$$(S(j\omega) = \sum_{n=-\infty}^{\infty} \underline{c_k} e^{j\frac{n\pi}{\omega_c}\omega}, \qquad (5.4)$$

де

$$\underline{c_k} = \frac{1}{2\omega_c}\int_{-\omega_c}^{\omega_c} S(j\omega) e^{-j\frac{n\pi}{\omega_c}\omega} d\omega. \qquad (5.5)$$

З урахуванням (5.5)

$$s(-\frac{n\pi}{\omega_c}) = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} S(j\omega) e^{-j\frac{n\pi}{\omega_c}\omega} d\omega. \qquad (5.6)$$

Порівнявши (5.5) і (5.6) одержимо:

$$\underline{c_k} = s(-\frac{n\pi}{\omega_c})\frac{\pi}{\omega_c} = \frac{\pi}{\omega_c} s(-n\Delta t). \qquad (5.7)$$

Підставимо (5.7) в (5.4):

$$S(j\omega) = \frac{\pi}{\omega_c}\sum_{n=-\infty}^{\infty} s(-n\Delta t) e^{j\omega n\Delta t}. \qquad (5.8)$$

Оскільки додавання виконується як за додатними, так і за від'ємними частотами, то знак перед «n» можна змінити на протилежний:

$$S(j\omega) = \frac{\pi}{\omega_c}\sum_{n=-\infty}^{\infty} s(n\Delta t) e^{-j\omega n\Delta t}. \qquad (5.9)$$

Підставивши останній вираз в (5.2), маємо:

$$s(t) = \frac{1}{2\omega_c}\int_{-\omega_c}^{\omega_c} \left\{ \sum_{n=-\infty}^{\infty} s(n\Delta t) e^{-j\omega n\Delta t} \right\} e^{j\omega t} d\omega. \qquad (5.10)$$

Змінивши порядок додавання і інтегрування отримаємо:

$$s(t)=\frac{1}{2\omega_c}\sum_{n=-\infty}^{\infty}s(n\Delta t)\int_{-\omega_c}^{\omega_c}e^{j\omega(t-n\Delta t)}d\omega. \qquad (5.11)$$

Обчислимо інтеграл:

$$\int_{-\omega_c}^{\omega_c}e^{j\omega(t-n\Delta t)}d\omega=\frac{1}{j(t-n\Delta t)}e^{j\omega(t-n\Delta t)}\Big|_{-\omega_c}^{\omega_c}=\frac{2\sin\omega_c(t-n\Delta t)}{(t-n\Delta t)}. \qquad (5.12)$$

З урахуванням чого

$$s(t)=\sum_{n=-\infty}^{\infty}s(n\Delta t)\frac{\sin\omega_c(t-n\Delta t)}{\omega_c(t-n\Delta t)}. \qquad (5.13)$$

І так, функція s(t) виражена через дискретні значення взяті в моменти часу $t_n=n\Delta t=\frac{\pi n}{\omega_c}$. Причому функція відліків

$$\frac{\sin\omega_c(t-n\Delta t)}{\omega_c(t-n\Delta t)}=\begin{cases}1, & t=n\Delta t\\ 0, & t=k\Delta t,\quad k\neq n,\end{cases} \qquad (5.14)$$

що і потрібно було довести. Тут роль коефіцієнта c_k виконують відліки $s(n\Delta t)$ функції s(t) [2].

Приклад 5.1. Визначити за теоремою Котельникова крок дискретизації детермінованої функції

$$s(t)=\begin{cases}2e^{-t}, & t\geq 0\\ 0, & t<0,\end{cases}$$

орієнтуючись на практичну ширину спектра з $\eta=0{,}95$.

Розв'язування. Визначимо комплексний спектр функції s(t) згідно з (4.5):

$$S(j\omega)=2\int_0^{\infty}e^{-t}e^{-j\omega t}dt=\frac{2}{1+j\omega}.$$

Тоді модуль комплексної функції $S(j\omega)$ такий:

$$S(\omega)=|S(j\omega)|=\frac{2}{\sqrt{1+\omega^2}}.$$

Практичну ширину визначимо зі співвідношення:

$$\frac{1}{\pi}\int_0^{\omega_п}|S(j\omega)|^2 d\omega = \frac{\eta}{\pi}\int_0^{\infty}|S(j\omega)|^2 d\omega.$$

Звідки маємо:

$$\frac{1}{\pi}\int_0^{\omega_п}\frac{4}{1+\omega^2}d\omega = \frac{\eta}{\pi}\int_0^{\infty}\frac{4}{1+\omega^2}d\omega,$$

$$\int_0^{\infty}\frac{1}{1+\omega^2}d\omega = \operatorname{arctg}\omega\Big|_0^{\infty} = \frac{\pi}{2},$$

$$\operatorname{arctg}\omega_п = \frac{0{,}95\pi}{2}.$$

Тоді

$$\omega_п = 13{,}1 \quad c^{-1},$$

$$\Delta t = \frac{\pi}{\omega_п} = \frac{\pi}{13{,}1} \quad c.$$

5.2 Періодичність спектра дискретної функції

Дискретизована функція може бути подана так:

$$\begin{aligned} s(k\Delta t) &= s(t)\varphi(k\Delta t), \\ \varphi(k\Delta t) &= \sum_{k=-\infty}^{\infty}\delta(t-k\Delta t), \end{aligned} \qquad (5.15)$$

де $\delta(t-k\Delta t)$ - одинична функція.

Оскільки $\varphi(k\Delta t)$ періодична функція, то вона може бути розкладена в ряд Фур'є

$$\varphi(k\Delta t) = \sum_{k=-\infty}^{\infty} c_k e^{jk\omega_c t} = \sum_{k=-\infty}^{\infty} c_k e^{jk\frac{2\pi}{\Delta t}t}. \qquad (5.16)$$

Відомо, що для періодичної послідовності одиничних імпульсів значення коефіцієнтів c_k визначаються так [1-2]:

$$c_k = \frac{1}{\Delta t}.$$

Тоді

$$\varphi(k\Delta t) = \frac{1}{\Delta t}\sum_{k=-\infty}^{\infty} e^{jk\frac{2\pi}{\Delta t}t}. \qquad (5.17)$$

З урахуванням того, що

$$S(j\omega) = \int_{-\infty}^{\infty} s(t)e^{-j\omega t}dt, \qquad (5.18)$$

для дискретизованої функції одержимо:

$$\begin{aligned} S_{\Delta t}(j\omega) &= \frac{1}{\Delta t}\int_{-\infty}^{\infty} s(t)\sum_{k=-\infty}^{\infty} e^{jk\frac{2\pi}{\Delta t}t}e^{-j\omega t}dt = \\ &= \frac{1}{\Delta t}\sum_{k=-\infty}^{\infty}\int_{-\infty}^{\infty} s(t)e^{-j(\omega-\frac{2k\pi}{\Delta t})t}dt = \frac{1}{\Delta t}\sum_{k=-\infty}^{\infty} S\left[j(\omega - k\frac{2\pi}{\Delta t})\right]. \end{aligned} \qquad (5.19)$$

Тобто, спектр дискретних сигналів періодичний.

Якщо сигнал має обмежений спектр з максимальною частотою ω_c і $\Delta t = \frac{\pi}{\omega_c}$, то отримаємо:

$$S_{\Delta t}(j\omega) = \frac{1}{\Delta t}\sum_{k=-\infty}^{\infty} S[j(\omega - 2k\omega_c)]. \qquad (5.20)$$

Графічно спектр сигналу, дискретизованого згідно з теоремою Котельникова, подано на рис. 5.1.

Якщо крок дискретизації

$$\Delta t_1 = \frac{\pi}{\omega} > \frac{\pi}{\omega_c}, \qquad (5.21)$$

то спектри будуть перекриватись (рис. 5.2) і початковий сигнал не буде відновлено після дискретизації з таким кроком [1-2].

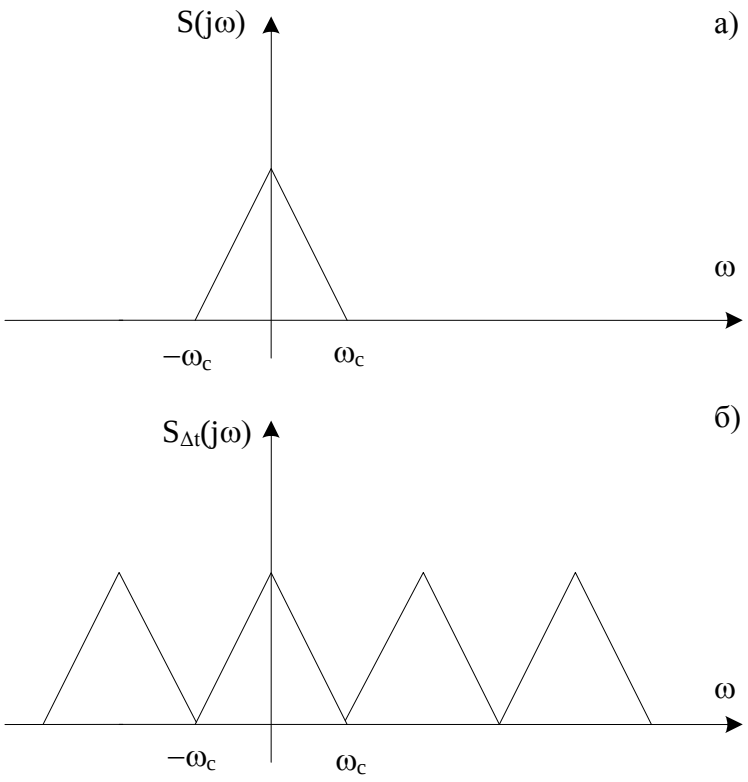


Рисунок 5.1 – Спектри сигналу: а) – спектр неперервного сигналу; б) спектр сигналу, дискретизованого згідно з теоремою Котельникова

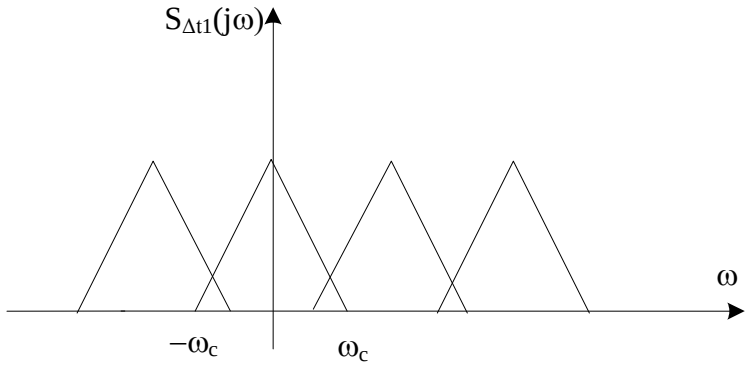


Рисунок 5.2 – Перекриття спектрів

Контрольні запитання і вправи

1. Хто сформулював правило вибору граничного кроку дискретизації для сигналів з обмеженим спектром?
2. Дайте формулювання теореми Котельникова.
3. Як називають теорему Котельникова в західній науковій літературі?
4. Який спектр мають дискретизовані сигнали?
5. Як залежить величина періоду спектра від частоти дискретизації?
6. Наведіть переваги і недоліки цифрової форми подання неперервних сигналів.
7. Який спектр мають дискретні сигнали у випадку, якщо крок дискретизації більший кроку згідно з теоремою Котельникова.
8. Визначити за теоремою Котельникова крок дискретизації детермінованої функції

$$s(t)=\begin{cases} e^{-2t}, & t\ge 0 \\ 0, & t<0, \end{cases}$$

орієнтуючись на практичну ширину спектра з $\eta = 0{,}9$.

9. Визначити за теоремою Котельникова крок дискретизації детермінованої функції

$$s(t)=\begin{cases} t\sin t^2, & t\ge 0 \\ 0, & t<0, \end{cases}$$

орієнтуючись на практичну ширину спектра з $\eta = 0{,}9$.

10. Визначити за теоремою Котельникова крок дискретизації детермінованої функції

$$s(t)=\begin{cases} t\cos t^2, & t\ge 0 \\ 0, & t<0, \end{cases}$$

орієнтуючись на практичну ширину спектра з $\eta = 0{,}95$.

6 ДИСКРЕТНЕ ПЕРЕТВОРЕННЯ ФУР'Є

6.1 Основні формули дискретного перетворення Фур'є

Дискретне перетворення Фур'є (ДПФ) широко застосовується в цифровій обробці сигналів *(digital signal processing (DSP))* та зображень. Пряме ДПФ виконується послідовністю дискретних величин скінченної тривалості. В результаті отримують також скінченної тривалості дискретну послідовність величин, які дають частотно-спектральне подання вказаної початкової послідовності. При зворотному ДПФ отримують початкову послідовність. Пряме і зворотне ДПФ визначаються такими виразами [6-7]:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}kn} \tag{6.1}$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi}{N}kn}, \tag{6.2}$$

де $k = 0, 1, \Lambda, N-1$; $n = 0, 1, \Lambda, N-1$,

N – кількість відліків в дискретній послідовності,

x(n) – значення n-го елемента дискретної послідовності.

Позначимо

$$W_N = e^{-j\frac{2\pi}{N}}, \tag{6.3}$$

тоді отримаємо:

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \tag{6.4}$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn} \tag{6.5}$$

$$k - 0, 1, \Lambda, N-1; \quad n = 0, 1, \Lambda, N-1.$$

6.2 Ідеї швидкого обчислення ДПФ

Основним недоліком ДПФ є велика кількість обчислень. Наприклад, для формули (6.4) необхідно виконати N^2 множень. Тому для обчислення ДПФ необхідно застосовувати швидкі алгоритми розрахунку (швидке перетворення Фур'є (ШПФ) - *Fast Fourier transform (FFT)*). Виникненням

швидких алгоритмів прийнято вважати 1965 рік, коли Кулі і Тьюкі представили свій алгоритм виконання ШПФ. Хоча ще раніше Гуд і Томас розробили інший алгоритм ШПФ. Наступним з'явився алгоритм Винограда та інші [6].

Розглянемо основні ідеї ШПФ. Нехай $N=2^k$ де k – ціле. При обчисленні ДПФ методом ШПФ початкова N-точкова послідовність розділяється на частини. Наприклад, розділивши послідовність на дві частини, в кожній з яких N/2 відліків, можна для кожної з них обчислити ДПФ, а потім виразити загальний ДПФ всієї N-точкової послідовності відліків через ДПФ однієї і другої частини. Можна дальше розділити ці підпослідовності на дві частини і т. д. Таким чином з'являється можливість виразити ДПФ всієї початкової послідовності з N відліків як функції від ДПФ двоточкових послідовностей. Цей метод обчислення ДПФ, який і названо ШПФ, дозволяє значно зменшити кількість арифметичних операцій [6-7].

6.3 ШПФ Кулі-Тьюкі

Розглянемо більш детально алгоритм ШПФ Кулі-Тьюкі. Відомо дві модифікації цього алгоритму:

- з проріджением за часом;
- з проріджением за частотою.

ШПФ Кулі-Тьюкі з проріджением за часом

Алгоритм з проріджением за часом ґрунтується на розділенні початкової послідовності на підпослідовності парних і непарних відліків [6]. Нехай

$$\begin{aligned} x_1(n) &= x(2n), \\ x_2(n) &= x(2n+1), \\ n &= 0,1,\ldots\frac{N}{2}-1. \end{aligned} \tag{6.6}$$

ДПФ для кожної з цих підпослідовностей такі:

$$X_1(k) = \sum_{n=0}^{\frac{N}{2}-1} x_1(n) W_{N/2}^{kn}, \tag{6.7}$$

$$X_2(k) = \sum_{n=0}^{\frac{N}{2}-1} x_2(n) W_{N/2}^{kn}. \qquad (6.8)$$

Оскільки $W_N = e^{-j\frac{2\pi}{N}}$, то

$$W_{N/2} = e^{-j\frac{2\pi}{N/2}} = (e^{-j\frac{2\pi}{N}})^2 = W_N^2. \qquad (6.9)$$

Тоді вирази (6.7) та (6.8) приймуть такий вигляд:

$$X_1(k) = \sum_{n=0}^{\frac{N}{2}-1} x_1(n) W_N^{2kn} \qquad (6.10)$$

$$X_2(k) = \sum_{n=0}^{\frac{N}{2}-1} x_2(n) W_N^{2kn} \qquad (6.11)$$

Перепишемо вираз (6.4), розділивши відліки для парних і непарних n:

$$X(k) = \sum_{k=0}^{N-1} x(n) W_N^{kn} = \sum_{n=0}^{\frac{N}{2}-1} x(2n) W_N^{2kn} + \sum_{n=0}^{\frac{N}{2}-1} x(2n+1) W_N^{k(2n+1)}, \qquad (6.12)$$

$$k = 0,1,\ldots,\frac{N}{2}-1.$$

Таким чином, з урахуванням виразів (6.6), (6.10 – 6.12) одержимо:

$$X(k) = X_1(k) + W_N^k X_2(k),$$

$$k = 0,1,\ldots,\frac{N}{2}-1. \qquad (6.13)$$

Вираз (6.13) називають основною формулою виконання ШПФ Кулі-Тьюкі за основою «2» з прорідженням за часом. Вона дозволяє визначити $X(k)$ тільки для k=0, 1, …, (N/2) – 1.

Для знаходження $X(k)$ при k=N/2, (N/2)+1, …, N – 1 розглянемо множник $W_{N/2}^{nk}$. Нехай $k = k - \frac{N}{2}$, тоді:

$$W_{N/2}^{n(k-\frac{N}{2})} = W_{N/2}^{nk} W_{\frac{N}{2}}^{\frac{nN}{2}} = W_{N/2}^{nk}. \qquad (6.14)$$

Тобто, це періодична функція з періодом N/2, тому

$$X(k) = X_1(k - \frac{N}{2}) + W_N^k X_2(k - \frac{N}{2}),$$
$$k = \frac{N}{2}, \frac{N}{2} + 1, \Lambda\ , N - 1. \qquad (6.15)$$

Зауваження 6.1. Термін за основою «2» означає, що довжина перетворення 2^k подається як $2 \cdot 2^{k-1}$. Аналогічно якщо довжина перетворення 4^k розкладається як $4 \cdot 4^{k-1}$, то говорять про перетворення за основою «4».

Контрольні запитання і вправи

1. Наведіть основні формули дискретного перетворення Фур'є.
2. Який основний недолік ДПФ?
3. Які основні ідеї швидкого обчислення ДПФ?
4. Коли з'явились швидкі алгоритми обчислення ДПФ?
5. Які ви знаєте швидкі алгоритми обчислення ДПФ?
6. Назвіть модифікації алгоритму ШПФ Кулі-Тьюкі.
7. На чому ґрунтується алгоритм ШПФ Кулі-Тьюкі з прорідженням за часом.
8. Наведіть основні формули виконання ШПФ Кулі-Тьюкі за основою «2» з прорідженням за часом.
9. Поясність значення термінів алгоритм ШПФ Кулі-Тьюкі за основою два, за основою чотири.
10. Обчислити дискретне перетворення Фур'є з використанням алгоритму ШПФ Кулі-Тьюкі з прорідженням за часом такої послідовності: 10, 12, 10, 10.
11. Обчислити дискретне перетворення Фур'є з використанням алгоритму ШПФ Кулі-Тьюкі з прорідженням за часом такої послідовності: 10, 10, 10, 10, 100, 100, 100, 100.
12. Обчислити дискретне перетворення Фур'є з використанням алгоритму ШПФ Кулі-Тьюкі з прорідженням за часом такої послідовності: 13, 13, 13, 13.
13. Обчислити дискретне перетворення Фур'є з використанням алгоритму ШПФ Кулі-Тьюкі з прорідженням за часом такої послідовності: 43, 43, 43, 43, 43, 43, 43, 43.

7 ШПФ КУЛІ-ТЬЮКІ З ПРОРІДЖЕННЯМ ЗА ЧАСТОТОЮ ТА ГРАФИ ШПФ

7.1 ШПФ Кулі-Тьюкі з проріджениям за частотою

ШПФ Кулі-Тьюкі з проріджениям за частотою ґрунтується на розподілі початкової послідовності на підпослідовності перших N/2 відліків і наступних N/2 відліків [6]. Нехай $N=2^k$, тоді:

$$\begin{aligned} x_1(n) &= x(n), \qquad n = 0,1,\Lambda\ ,\frac{N}{2}-1 \\ x_2(n) &= x(n+\frac{N}{2}), \quad n = 0,1,\Lambda\ ,\frac{N}{2}-1. \end{aligned} \tag{7.1}$$

І так, згідно з виразом (6.4):

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \qquad k = 0,1,\Lambda\ ,N-1.$$

Перепишемо цей вираз так:

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x(n) W_N^{nk} + \sum_{n=\frac{N}{2}}^{N-1} x(n) W_N^{nk}. \tag{7.2}$$

З урахуванням виразу (7.1)

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x_1(n) W_N^{nk} + \sum_{n=0}^{\frac{N}{2}-1} x_2(n) W^{(n+\frac{N}{2})k},$$

$$k = 0,1,\Lambda\ ,N-1. \tag{7.3}$$

Розглянемо останній вираз для парних і непарних k:

$$X(2k) = \sum_{n=0}^{\frac{N}{2}-1} x_1(n) W_N^{n2k} + \sum_{n=0}^{\frac{N}{2}-1} x_2(n) W_N^{(n+\frac{N}{2})2k},$$

$$X(2k+1) = \sum_{n=0}^{\frac{N}{2}-1} x_1(n) W_N^{n(2k+1)} + \sum_{n=0}^{\frac{N}{2}-1} x_2(n) W_N^{(n+\frac{N}{2})(2k+1)},$$

$$k = 0, \Lambda, \frac{N}{2} - 1. \qquad (7.4)$$

Враховуючи, що

$$W_N^{Nk} = 1; \quad W_N^{\frac{N}{2}(2k+1)} = -1. \qquad (7.5)$$

Отримаємо

$$X(2k) = \sum_{n=0}^{\frac{N}{2}-1} [x_1(n) + x_2(n)] W_N^{2nk},$$

$$X(2k+1) = \sum_{n=0}^{\frac{N}{2}-1} [x_1(n) - x_2(n)] W_N^{n} W_N^{2nk},$$

$$k = 0, 1, \Lambda, \frac{N}{2} - 1. \qquad (7.6)$$

7.2 Використання ШПФ для прискореного обчислення зворотного ДПФ

Відомо, що

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-nk},$$

$$W_N = e^{-j\frac{2\pi}{N}}$$

перетворюються до вигляду

$$x(n) = \frac{1}{N} \left[\sum_{k=0}^{N-1} X^*(k) W_N^{nk} \right]^*,$$

$$n = 0, \Lambda, N-1 \qquad (7.7)$$

де $[\ ]^*$ - комплексна спряжена величина;

$X^*(k)$ - величина комплексно спряжена $X(k)$.

Тобто, може бути обчислена з використанням прямого ДПФ [6].

7.3 Графи ШПФ

Розглянемо приклад ШПФ з проріджанням за часом при N=8, тоді згідно з (6.6), (6.10-6.11):

$$x_1(n) = x(2n),$$
$$x_2(n) = x(2n+1),$$
$$X_1(k) = \sum_{n=0}^{3} x_1(n) W_8^{2nk} = \sum_{n=0}^{3} x(2n) W_8^{2nk},$$
$$X_2(k) = \sum_{n=0}^{3} x_2(n) W_8^{2nk} = \sum_{n=0}^{3} x(2n+1) W_8^{2nk}.$$

Розіб'ємо послідовності $x_1(n)$ і $x_2(n)$ на парні і непарні підпослідовності:

$$x_{11}(n) = x_1(2n) = x(4n),$$
$$x_{12}(n) = x_1(2n) = x(4n+2),$$
$$x_{21}(n) = x_2(2n) = x(4n+1),$$
$$x_{22}(n) = x_2(2n+1) = x(4n+3).$$

Обчислимо для кожної підпослідовності ДПФ згідно з (6.10-6.11):

$$X_{11}(k) = \sum_{n=0}^{1} x(4n) W_4^{2nk} = \sum_{n=0}^{1} x(4n) W_8^{4nk}, \qquad (7.8)$$

$$X_{12}(k) = \sum_{n=0}^{1} x(4n+2) W_8^{4nk}, \qquad (7.9)$$

$$X_{21}(k) = \sum_{n=0}^{1} x(4n+1) W_8^{4nk}, \qquad (7.10)$$

$$X_{22}(k) = \sum_{n=0}^{1} x(4n+3) W_8^{4nk}, \qquad (7.11)$$

$$k = 0, 1.$$

Знайдемо ДПФ підпослідовностей $x_1(n)$ і $x_2(n)$ згідно з (6.13) і (6.15):

$$X_1(k) = X_{11}(k) + W_8^{2k} X_{12}(k), \quad k = 0, 1, \qquad (7.12)$$

$$X_1(k) = X_{11}(k-2) + W_8^{2k} X_{12}(k-2), \quad k = 2, 3, \qquad (7.13)$$

$$X_2(k) = X_{21}(k) + W_8^{2k} X_{22}(k), \quad k = 0, 1, \qquad (7.14)$$

$$X_2(k) = X_{21}(k-2) + W_8^{2k} X_{22}(k-2), \quad k = 2, 3. \quad (7.15)$$

Знайдемо загальний ДПФ послідовності $x(n)$:

$$X(k) = X_1(k) + W_8^k X_2(k), \quad k = 0, 1, 2, 3, \quad (7.16)$$

$$X(k) = X_1(k-4) + W_8^k X_2(k-4), \quad k = 4, 5, 6, 7. \quad (7.17)$$

Зобразимо графічно процес обчислення ШПФ (рис. 7.1). Де стрілка без додаткового позначення указує, що просто передається значення, а стрілка з позначенням W_N^m під нею указує на передачу даної величини помноженої на W_N^m. Кружками позначені або початкові точки, або при підході до кружка двох ліній – точки виконання операцій додавання. Розташуємо вхідні відліки у двійково-інверсному порядку [6]. Тобто,

$$u(000) = x(000),$$
$$u(001) = x(100),$$
$$u(010) = x(010),$$
$$u(011) = x(110),$$
$$u(100) = x(001),$$
$$u(101) = x(101),$$
$$u(110) = x(011),$$
$$u(111) = x(111).$$

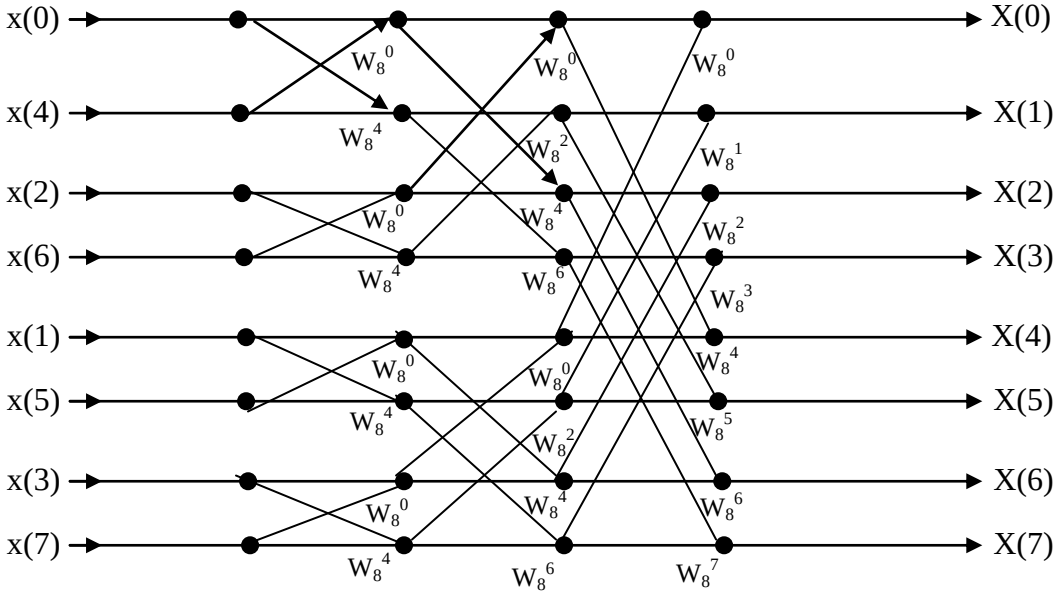
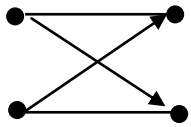


Рисунок 7.1 – Граф ШПФ з прорідженням за часом

Для кожного зі ступенів ШПФ граф має такий вигляд:



Цей граф нагадує метелика («бабочку»), тому цю операцію називають «метелик» (або російською «бабочка») [6].

Зауваження 7.1. Граф ШПФ з проріджениям за частотою відрізняється від графа ШПФ з проріджениям за часом тим, що вхідні дані розташовуються в звичайному порядку, а вихідні значення X (k) в двійково-інверсному порядку [6].

7.4 Несинусоїдні ортогональні перетворення

Крім синусоїдних перетворень типу перетворення Фур'є, в цифровій обробці сигналів знаходять застосування і несинусоїдні перетворення. Найбільш відомим і важливим несинусоїдним ортогональним перетворенням є перетворення Уолша-Адамара, відмінною особливістю якого є те, що розрахунок базується лише на операціях додавання.

В процесі перетворення використовується послідовно упорядкована квадратна матриця Адамара H, елементи якої можуть приймати лише значення +1 або -1, а відповідні рядки і стовбці є взаємно ортогональними. Матриця перетворення H для двох випадкових величин матиме вигляд [6]:

$$\mathrm{H} = \begin{vmatrix} 1 & 1 \\ 1 & -1 \end{vmatrix} \tag{7.18}$$

Якщо з елементів матриці H скласти базисні вектори

$$\mathrm{e}_1 = [\mathrm{h}_{11}, \mathrm{h}_{12}] = [1,1] \;\; \mathrm{e}_2 = [\mathrm{h}_{21}, \mathrm{h}_{22}] = [1,-1],$$

то вони характеризуватимуть поворот ортогональної системи координат на 45° відносно одиничного базису (рис. 7.2). Якщо випадкові величини x_1 і x_2 мають кореляційну залежність, то проекція вектора X $[x_1, x_2]^T$ на базисний вектор e_1 буде, в середньому, більша, ніж проекція цього ж вектора на базисний вектор e_2. Завдяки цьому інформація буде зосереджена в першому коефіцієнті перетворення. Другий коефіцієнт служить для уточнення подання вектора X в новому базисі.

При збільшенні розмірності простору велика частка інформації буде зосереджена на малому числі коефіцієнтів, які потрібно зберегти з найменшими втратами. Решту коефіцієнтів можна або відкинути, або квантувати з більшим кроком.

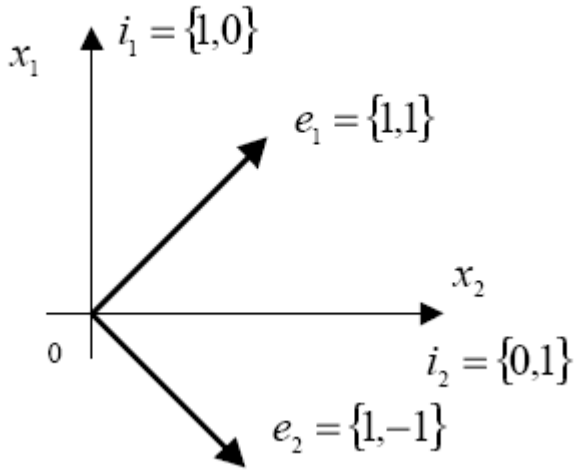


Рисунок. 7.2 - Базисні вектори перетворення Адамара

Матрицю Адамара розмірності 4×4 елементи легко побудувати з матриці Адамара H розмірністю 2×2 елементи:

$$\mathrm{H} = \begin{vmatrix} \mathrm{H} & \mathrm{H} \\ \mathrm{H} & -\mathrm{H} \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{vmatrix} \qquad (7.19)$$

Користуючись співвідношенням (7.19), можна побудувати матрицю Адамара будь-якої розмірності $N = 2^n$, де n - будь-яке ціле додатне число.

Перевагою такого перетворення є простота реалізації і низька обчислювальна складність. Для обчислення перетворення Уолша-Адамара використовуються ті ж швидкі алгоритми, що і для ДПФ, наприклад алгоритм Кулі-Т'юкі.

Це перетворення даватиме добрі результати для кусково-постійних функцій, оскільки базисні функції перетворення Адамара виділяють постійні складові сигналу. В результаті для кусково-постійних сигналів

велика частка коефіцієнтів перетворення Адамара буде близька до нуля, що приведе до зменшення ентропії перетворених даних і збільшення коефіцієнта ущільнення. Проте на практиці такі сигнали зустрічаються досить рідко.

Контрольні запитання і вправи

1. На чому ґрунтується ШПФ Кулі-Тьюкі з проріджением за частотою.
2. Наведіть розрахункові формули для обчислення ШПФ Кулі-Тьюкі з проріджением за частотою.
3. Як обчислити зворотне ДПФ з використанням ШПФ?
4. Як графічно подати процес обчислення ДПФ з використанням ШПФ?
5. Як розташовують вхідні відліки в графах ШПФ з проріджением за часом.
6. Чим відрізняється граф ШПФ з проріджением за часом від графа ШПФ з проріджением за частотою.
7. Побудувати граф ШПФ з проріджением за часом при N = 4.
8. Побудувати граф ШПФ з проріджением за частотою при N = 4.
9. Побудувати граф ШПФ з проріджением за частотою при N = 8.
14. Обчислити дискретне перетворення Фур'є з використанням алгоритму ШПФ Кулі-Тьюкі з проріджением за частотою такої послідовності: 13, 0, 13, 0.
10. Обчислити дискретне перетворення Фур'є з використанням алгоритму ШПФ Кулі-Тьюкі з проріджением за частотою такої послідовності: 43, 43, 43, 43, 0, 0, 0, 0.
11. З використанням графа ШПФ з проріджением за часом обчислити дискретне перетворення Фур'є такої послідовності: 100, 200, 100, 200.
12. З використанням графа ШПФ з проріджением за часом обчислити дискретне перетворення Фур'є такої послідовності: 10, 20, 30, 40, 10, 20, 30, 40.
13. Яке перетворення серед несинусоїдних є найбільш відомим?
14. Яка особливість перетворення Уолша-Адамара?
15. Які переваги несинусоїдних ортогональних перетворень?

8 ЗАГАЛЬНІ ПОНЯТТЯ ПРО ЦИФРОВІ ФІЛЬТРИ

1. Z-перетворення
2. Цифрові фільтри – загальні поняття (частотна характеристика, імпульсна характеристика, класифікація)
3. Нерекурсивні фільтри з симетричними коефіцієнтами.

8.1 Z-перетворення

Z-перетворення *(Z-transform)* визначається так:

$$X(Z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n} . \qquad (8.1)$$

Z-перетворення є аналогом перетворення Фур'є для дискретного за часом сигналу. Воно включає дискретне перетворення Фур'є як окремий випадок [5].

8.2 Цифрові фільтри – загальні поняття (частотна характеристика, імпульсна характеристика, класифікація)

Цифровий фільтр (ЦФ) це пристрій, що перетворює цифрові послідовності [5, 8-11]. В загальному випадку процес роботи ЦФ описується таким різницевим рівнянням:

$$\sum_{l=0}^{M} b_l y_{n-l} = \sum_{k=0}^{N} a_k x_{n-k} . \qquad (8.2)$$

Нехай $b_0=1$, тоді отримаємо:

$$y_n = \sum_{k=0}^{N} a_k x_{n-k} - \sum_{l=1}^{M} b_l y_{n-l} , \qquad (8.3)$$

де *n, n-l, n-k* – номери відліків цифрової послідовності;

y,x – значення елементів цифрової послідовності в заданих точках.

Якщо в рівнянні (8.3) є коефіцієнти $b_l \neq 0$, то такий ЦФ називається рекурсивним фільтром (РФ), а якщо всі $b_l=0$ (для $l=1, 2,\ldots,M$), то такий ЦФ називається нерекурсивним фільтром (НФ).

Якщо *X(Z)* - Z-перетворення вхідної послідовності x_n (або *x(n)*), а *Y(Z)* – *Z*-перетворення вихідної послідовності фільтра y_n, то передавальна функція фільтра визначається так:

$$H(Z) = \frac{Y(Z)}{X(Z)}. \tag{8.4}$$

Нехай $Z = e^{j\omega T_a}$, де T_a – період дискретизації вхідної послідовності, то отримаємо комплексну частотну характеристику фільтра:

$$H(e^{j\omega T_a}) = \frac{Y(e^{j\omega T_a})}{X(e^{j\omega T_a})}. \tag{8.5}$$

А модуль $\left|H(e^{j\omega T_a})\right|$ - амплітудно-частотна характеристика (АЧХ) фільтра.

Реакція фільтра при нульових початкових умовах на одиничний вхідний сигнал утворює імпульсну характеристику фільтра *(impulse response function (IRF))* $h(nT_a)$. Одиничний вхідний сигнал визначається так:

$$\tilde{\delta}(nT) = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0. \end{cases} \tag{8.6}$$

В залежності від характеру імпульсної характеристики ЦФ прийнято ділити на два класи:

- СІХ-фільтри (КИХ, FIR),
- НІХ-фільтри (БИХ, IIR).

СІХ-фільтри – це фільтри зі скінченною імпульсною характеристикою *(finite impulse response (FIR))*, а НІХ-фільтри – фільтри з нескінченною імпульсною характеристикою *(infinite impulse response (IIR))*. Всі нерекурсивні фільтри – це СІХ-фільтри, а майже всі рекурсивні фільтри є НІХ-фільтрами.

Частотна і імпульсна характеристики фільтра пов’язані парою перетворень Фур’є [10]:

$$H(e^{j\omega T_a}) = \sum h(nT_a) e^{-jn\omega T_a},$$

$$h(nT_a) = \frac{T_a}{2\pi} \int_{-\frac{\pi}{Ta}}^{\frac{\pi}{Ta}} H(e^{j\omega T_a}) e^{jn\omega T_a} d\omega. \qquad (8.7)$$

8.3 Нерекурсивні фільтри зі симетричними коефіцієнтами

Нерекурсивний фільтр описується таким різницевим рівнянням:

$$y_n = \sum_{k=0}^{N} a_k x_{n-k}. \qquad (8.8)$$

Нехай *k* змінюється від *–N* до *N,* тоді:

$$y_n = \sum_{k=-N}^{N} a_k x_{n-k}. \qquad (8.9)$$

Якщо $a_{-k}=a_k$ *(або* $a_{-к}= -a_к$*),* то отримаємо фільтр з лінійною фазовою характеристикою. Подамо на вхід даного фільтра одиничний імпульс. Як видно з рівняння (8.9) на його виході з'являться коефіцієнти фільтра як вихідний сигнал. Тобто:

$$y_n = \sum_{k=-\infty}^{\infty} h_k x_{n-k}, \qquad (8.10)$$

де h_k – імпульсна характеристика фільтра (імпульсний відгук).

Тобто, вихідний сигнал нерекурсивного фільтра – згортка вхідного сигналу з імпульсною характеристикою фільтра [5].

Приклад 8.1. Знайти АЧХ та навести структурну схему такого фільтра:

$$\mathbf{y_n = (x_n + x_{n+1} + x_{n-1}) / 3}$$

Розв'язування. Відомо, що передавальна функція фільтра визначається так:

$$H(Z) = \frac{Y(Z)}{X(Z)},$$

де Y(Z) – Z-перетворення вихідної послідовності фільтра;

X(Z) - Z-перетворення вхідної послідовності фільтра.

З урахуванням цього знайдемо Y(Z):

$$Y(Z) = \frac{X(Z) + X(Z) * Z^{-1} + X(Z) * Z}{3} = \frac{X(Z)\left[1 + Z^{-1} + Z\right]}{3}.$$

Тоді передавальна функція H(Z) визначається так:

$$H(Z) = \frac{Y(Z)}{X(Z)} = \frac{1 + Z^{-1} + Z}{3}.$$

Для знаходження частотної характеристики фільтра підставимо $Z = e^{j\omega T_a}$, де T_a - період дискретизації вхідного сигналу. Отримаємо:

$$H(e^{j\omega T_a}) = \frac{1 + e^{-j\omega T_a} + e^{j\omega T_a}}{3}.$$

З урахуванням того, що

$$e^{j\omega T_a} = \cos\omega T_a + j\sin\omega T_a,$$

$$e^{-j\omega T_a} = \cos\omega T_a - j\sin\omega T_a,$$

отримаємо

$$H(e^{j\omega T_a}) = \frac{1 + \cos\omega T_a - j\sin\omega T_a + \cos\omega T_a + j\sin\omega T_a}{3} =$$

$$= \frac{1 + 2\cos\omega T_a}{3}.$$

Амплітудно-частотна характеристика визначається як модуль частотної характеристики:

$$| H(e^{j\omega T_a}) | = \left| \frac{1 + 2\cos\omega T_a}{3} \right|.$$

Структурна схема фільтра така (рис. 8.1):

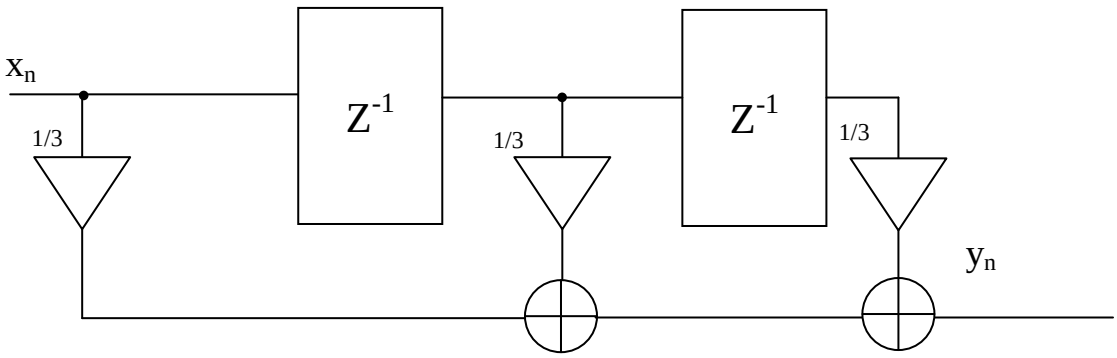


Рисунок 8.1 - Структурна схема фільтра

Як видно з прикладу, в загальному випадку передавальна функція для такого типу фільтрів визначається так [5]:

$$H(Z) = \sum_{k=-N}^{k=N} a_k z^{-k}. \qquad (8.11)$$

А частотна характеристика при $z = e^{j\omega T_a}$ така:

$$H(e^{j\omega T_a}) = \sum_{k=-N}^{k=N} a_k e^{-jk\omega T_a}. \qquad (8.12)$$

Контрольні запитання і вправи

1. Охарактеризуйте Z-перетворення.
2. Що таке цифровий фільтр?
3. Якими рівняннями описується процес роботи ЦФ?
4. Який фільтр називають нерекурсивним?
5. Який фільтр називають рекурсивним?
6. Охарактеризуйте передавальну, частотну та амплітудно-частотну характеристики фільтра.
7. Що таке передавальна характеристика фільтра?
8. Що таке імпульсна характеристика фільтра?
9. Що таке СІХ-фільтр?
10. Що таке НІХ-фільтр?
11. Дайте класифікацію ЦФ в залежності від характеру імпульсної характеристики.
12. Який імпульсний відгук нерекурсивного фільтра?
13. Який імпульсний відгук рекурсивного фільтра?
14. Як зв’язані частотна і імпульсна характеристики фільтра?
15. Згорткою яких сигналів є вихідний сигнал нерекурсивного фільтра?
16. Як знайти АЧХ фільтра, якщо відома комплексна частотна характеристика?
17. Як знайти АЧХ фільтра, якщо відомо різницеве рівняння, яке описує фільтр.
18. Знайти АЧХ та навести структурну схему такого фільтра:
$$y_n = 2x_n - x_{n-1} - x_{n+1}.$$
19. Знайти АЧХ та навести структурну схему такого фільтра:
$$y_n = x_n + x_{n-1} - 2y_{n-1}.$$
20. Знайти АЧХ та навести структурну схему такого фільтра:
$$y_n = -x_n + 2x_{n-1} - x_{n+1}.$$
21. Знайти АЧХ та навести структурну схему такого фільтра:
$$y_n = x_n + x_{n-1} - 3y_{n-2}.$$

9 СИНТЕЗ НЕРЕКУРСИВНИХ ЦИФРОВИХ ФІЛЬТРІВ

1. Синтез нерекурсивного фільтра низької частоті (ФНЧ).
2. Явище Гіббса і функції вікна.
3. Частота дискретизації і схеми допусків.
4. Обчислення коефіцієнтів високочастотних, смугових і режекторних фільтрів.

9.1 Синтез нерекурсивного фільтра низької частоти (ФНЧ) методом найменших квадратів

Відомо, що періодичну за часом функцію можна апроксимувати рядом Фур'є:

$$s(t) = \sum_{k=-\infty}^{\infty} c_k e^{jk\omega_1 t}, \tag{9.1}$$

де

$$c_k = \frac{1}{T} \int_{-T/2}^{T/2} s(t) e^{-jk\omega_1 t} dt. \tag{9.2}$$

А тепер розглянемо частотну характеристику фільтра, яка згідно з виразом (8.12) така:

$$H(e^{j\omega T_a}) = \sum_{k=-N}^{k=N} a_k e^{-jk\omega T_a},$$

де a_k - коефіцієнти фільтра.

Це також ряд Фур'є тільки в частотній області. Замінивши змінну інтегрування t на ω і період T на $2\pi/T_a$ одержимо:

$$a_k = \frac{T_a}{2\pi} \int_{-\pi/T_a}^{\pi/T_a} H(j\omega) e^{jk\omega T_a} d\omega. \tag{9.3}$$

Нехай бажаним є ФНЧ *(Low-pass filter (LPF))* з такою амплітудно-частотною характеристикою:

$$H_w(j\omega) = \begin{cases} 1, & \text{якщо} \quad \omega \le |\omega_g| \\ 0, & \text{якщо} \quad \omega > |\omega_g| \end{cases}. \tag{9.4}$$

Ряд Фур'є (8.12) найбільш точно за середньоквадратичним критерієм апроксимує бажану АЧХ H_w, коли коефіцієнти фільтра a_k є коефіцієнтами ряду Фур'є. Оскільки H_w є парною функцією, то з урахуванням формул Ейлера отримаємо:

$$a_k = a_{-k} = \frac{T_a}{\pi} \int_0^{\pi/Ta} H_w(j\omega) \cos k\omega T_a d\omega.$$

А з урахуванням виразу (9.4):

$$a_k = a_{-k} = \frac{T_a}{\pi} \int_0^{\omega_g} \cos k\omega T_a d\omega = \frac{T_a}{\pi} \frac{\sin k\omega_g T_a}{kT_a} = \frac{\omega_g T_a}{\pi} \frac{\sin k\omega_g T_a}{k\omega_g T_a}. \quad (9.5)$$

Оскільки, $\omega_g = 2\pi f_g$ і $T_a = \frac{1}{f_a}$ введемо

$$\Omega_g = \frac{\omega_g}{f_a} = 2\pi \frac{f_g}{f_a}. \quad (9.6)$$

Тоді з (9.5) отримаємо:

$$a_k = a_{-k} = \frac{\Omega_g}{\pi} \frac{\sin k\Omega_{g_a}}{k\Omega_g} = \frac{\Omega_g}{\pi} Si(k\Omega_g), \quad (9.7)$$

де $Si(k\Omega_g) = \frac{\sin k\Omega_{g_a}}{k\Omega_g}$ - інтегральний синус, причому Si(0) = 1 [5].

Приклад 9.1. Ідеальний ФНЧ апроксимувати фільтром з *N=3* для граничної частоти f_g *=25 Гц*, якщо частота дискретизації f_a*= 100 Гц.*

Розв'язування. Знайдемо Ω_g згідно з (9.6):

$$\Omega_g = 2\pi \frac{f_g}{f_a} = 2\pi \frac{25}{100} = \frac{\pi}{2}.$$

Коефіцієнти фільтра визначаємо згідно з (9.7):

$$a_0 = \frac{\Omega_g}{\pi} \frac{\sin k\Omega_{g_a}}{k\Omega_g} = \frac{1}{2},$$

$$a_1 = a_{-1} = \frac{1}{2} \frac{\sin \frac{1}{2}\pi}{\frac{1}{2}\pi} = \frac{1}{\pi},$$

$$a_2 = a_{-2} = \frac{1}{2} \frac{\sin 2\frac{1}{2}\pi}{2\frac{1}{2}\pi} = 0,$$

$$a_3 = a_{-3} = \frac{1}{2}\frac{\sin 3\frac{1}{2}\pi}{3\frac{1}{2}\pi} = -\frac{1}{3\pi}.$$

Даний фільтр описується таким різницевим рівнянням:

$$y_n = \frac{1}{2}x_n + \frac{1}{\pi}(x_{n-1} + x_{n+1}) - \frac{1}{3\pi}(x_{n-3} + x_{n+3}).$$

А комплексна частотна характеристика такого фільтра така (рис. 9.1):

$$H(j\omega) = \frac{1}{2} + \frac{2}{\pi}\cos\omega T_a - \frac{2}{3\pi}\cos 3\omega T_a.$$

Вибране відношення граничної частоти до частоти дискретизації 1/4 має перевагу, оскільки парні коефіцієнти фільтра зникають. Підсилення постійної складової сигналу цього фільтра менше 1.

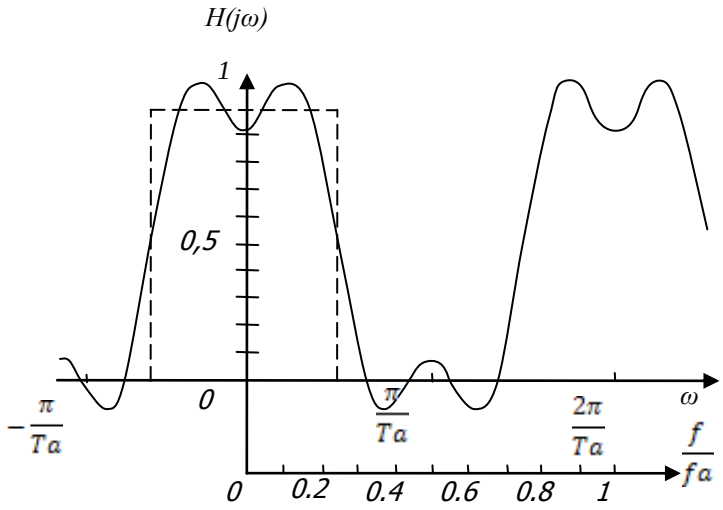


Рисунок 9.1 - Передавальна функція ФНЧ при N= 3 та $f_a = 4f_g$, параметри якого визначені за методом найменших квадратів

9.2 Явище Гіббса і функції вікна

Можна було б очікувати, що відхилення між ідеальним ФНЧ та розрахованим тим незначніше, чим більше коефіцієнтів було враховано в останньому. Це припущення відповідає дійсності для суми відхилень, але не для кожної різниці. В передавальній функції апроксимуючого фільтра є постійні викиди, що становлять приблизно 10%. Амплітуда викидів

незалежна від кількості коефіцієнтів фільтра. При зростаючому *N* відхилення тільки наближаються до фронту імпульсу фільтра (рис. 9.2). Така поведінка отримала назву явища Гіббса [5, 8].

Викиди в передавальній функції ФНЧ пояснюються стрибком ідеальної передавальної функції при ω_g. При застосуванні функції вікна *w(t)* можна ці викиди обмежити за рахунок крутості фільтра. В цьому разі різницеве рівняння фільтра має такий вигляд:

$$y_n = \sum_{k=-N}^{N} a_k w_k x_{n-k}. \tag{9.8}$$

А частотна характеристика такого ФНЧ така:

$$H(e^{j\omega T_a}) = \sum_{k=-N}^{k=N} a_k w_k e^{-jk\omega T_a} = \sum_{k=-N}^{k=N} a_k w_k \cos k\omega T_a. \tag{9.9}$$

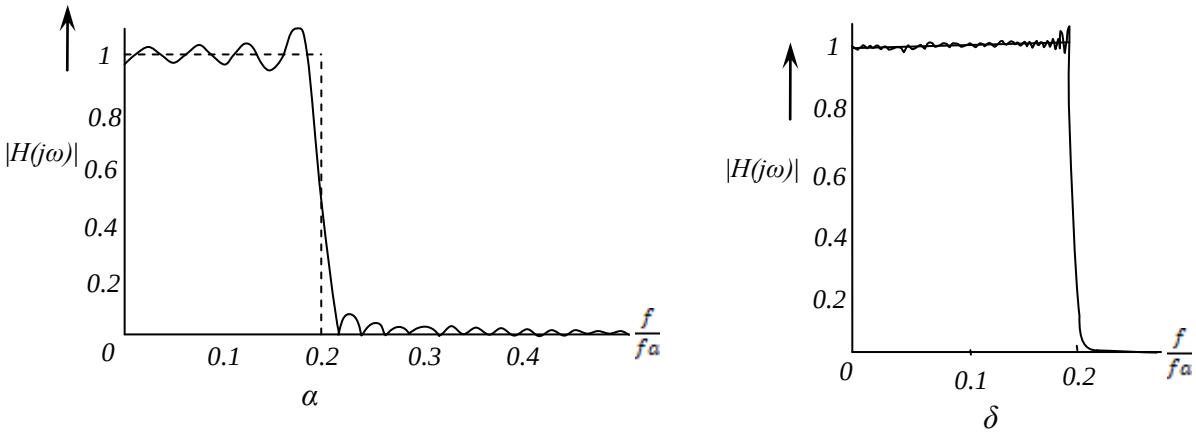


Рисунок 9.2 - Амплітудна характеристика FIR-ФНЧ: а – при N = 20 і $f_a = 5f_g$; б - при N = 200 і $f_a = 5f_g$

За згладжувальні функції можна взяти дзвонуваті функції косинуса. Рівняння функції вікна фон-Ганна запишемо як:

$$wk = \frac{1}{2}(1 + \cos\frac{\pi k}{N}). \tag{9.10}$$

Для ФНЧ за (9.8) при $N = 3$ дістаємо коефіцієнти згладжування:

$$\begin{aligned}
&k = 0; \quad w_0 = \frac{1}{2}(1+1) = 1; \\
&k = 1; \quad w_1 = w_{-1} = \frac{1}{2}(1+\cos\frac{\pi}{3}) = \frac{1}{2}(1+\frac{1}{2}) = 0{,}75; \\
&k = 2; \quad w_2 = w_{-2} = \frac{1}{2}(1+\cos\frac{2\pi}{3}) = \frac{1}{2}(1-\frac{1}{2}) = 0{,}25; \\
&k = 3; \quad w_3 = w_{-3} = \frac{1}{2}(1+\cos\frac{3\pi}{3}) = \frac{1}{2}(1-1) = 0.
\end{aligned} \tag{9.11}$$

Коефіцієнти a_k фільтра та коефіцієнти вікна фон-Ганна w_k для FIR-ФНЧ при $N = 20$ і $f_a = 5\ f_g$ наведені в табл. 9.1.

Іншу згладжувальну функцію запропонував К. Ланчос (C. Lanczos). Можна показати, що пульсація в передавальній функції має період першого відкинутого або останнього залишеного члена ряду Фур'є. Обчислення середнього значення протягом цього періоду може значно зменшити пульсацію. Для цього необхідно перемножити коефіцієнти a_k фільтра з σ-коефіцієнтами:

$$\sigma(N,k) = \frac{\sin(\pi k/N)}{\pi k/N}. \tag{9.12}$$

Згладжений за функцією Ланчоса фільтр має такий вигляд:

$$y_n = \sum_{k=-N}^{N} a_k \frac{\sin(\pi k/N)}{\pi k/N} x_{n-k}. \tag{9.13}$$

Передавальні функції незгладженого та згладженого фільтрів зображені на рис. 9.3.

При використанні функції вікна в зонах пропускання та запирання за рахунок крутості фронту імпульсу зменшується пульсація передавальної функції.

Таблиця 9.1 - Коефіцієнти a_k фільтра та коефіцієнти вікна фон-Ганна w_k для FIR-ФНЧ при N = 20 і $f_a = 5\ f_g$

k	**a_k**	**w_k**	**$a_k \cdot w_k$**
0	0,4000	1,0000	0,4000
1	0,3027	0,0038	0,3009
2	0,0935	0,9755	0,0913
3	-0,0624	0,9455	0,0590
4	-0,0757	0,9045	0,0685
5	0,0000	0,8536	0,0000
6	0,0505	0,7939	0,0401
7	0,0267	0,7270	0,0194
8	-0,0234	0,6545	0,0153
9	-0,0336	0,5782	0,0194
10	0,0000	0,5000	0,0000
11	0,0275	0,4218	0,0116
12	0,0156	0,3455	0,0054
13	0,0144	0,2730	0,0039
14	-0,0216	0,2061	0,0045
15	0,0000	0,1464	0,0000
16	0,0189	0,0955	0,0018
17	0,0110	0,0545	0,0006
18	-0,0104	0,0245	0,0003
19	-0,0159	0,0062	0,0001
20	0,0000	0,0000	0,0000

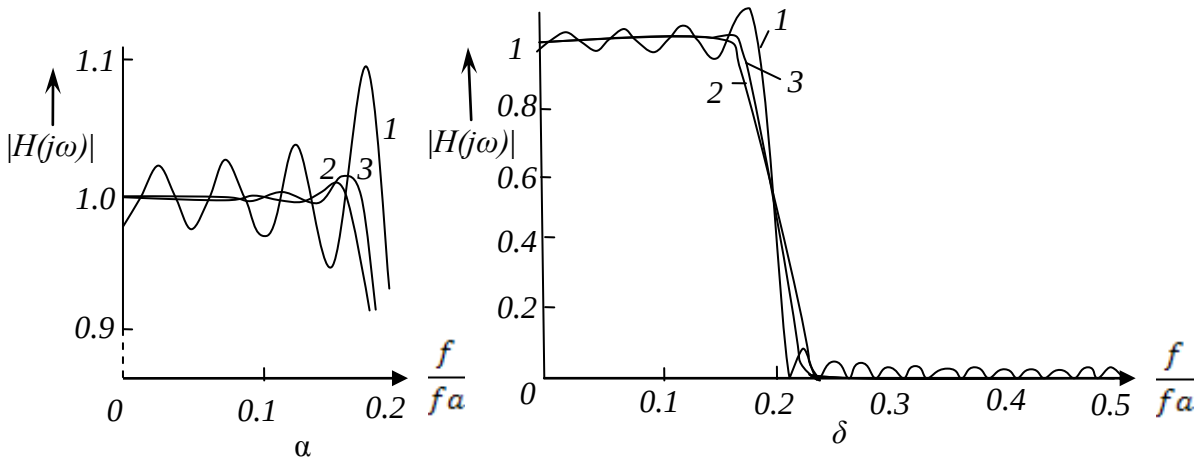


Рис. 9.3. Амплітудні характеристики FIR-ФНЧ при N = 20 і $f_a = 5f_g$: а - незгладженого фільтра; б - згладженого фільтра; 1 - прямокутне вікно без згладжування; 2 - згладжування за допомогою вікна фон-Ганна; 3 - згладжування за допомогою σ-коефіцієнтів рівняння (9.12)

9.3 Частота дискретизації і схеми допусків

При нерекурсивному ФНЧ частота дискретизації повинна набагато перевищувати граничну [5]. Прийнятними є коефіцієнти від 4 до 12. Порівняння рис. 9.2 та 9.4 показує, що при дуже високій частоті дискретизації бажана передавальна функція буде недостатньо наближена.

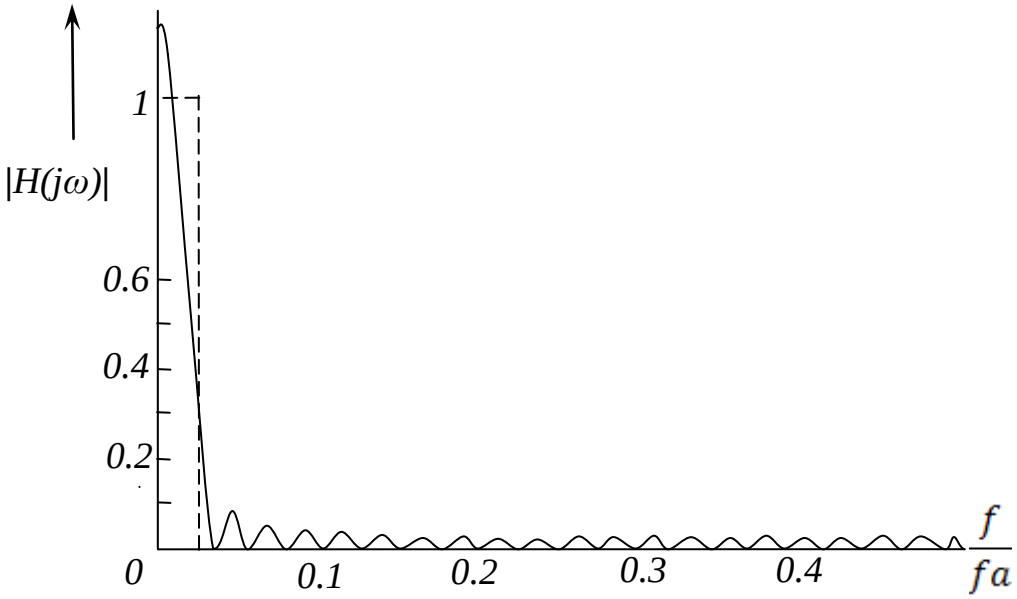


Рисунок 9.4 - Амплітудна характеристика FIR-ФНЧ при N = 20 і fa = 5 fg

Частотна характеристика ФНЧ має форму, зображену на рис. 9.5. У зонах пропускання та запирання можуть бути пульсації. Максимальні верхні та нижні викиди можна виразити відповідно через значення δ та ε (рис. 9.5). Вони визначають також частоти пропускання f_D та запирання f_S. Фільтр буде тим кращий, чим незначнішою буде пульсація і меншою відстань між f_D та f_S. У зоні пропускання $f < f_D$ амплітуди послаблюються максимально до 1 - δ, в зоні запирання $f > f_S$ - мінімально до ε.

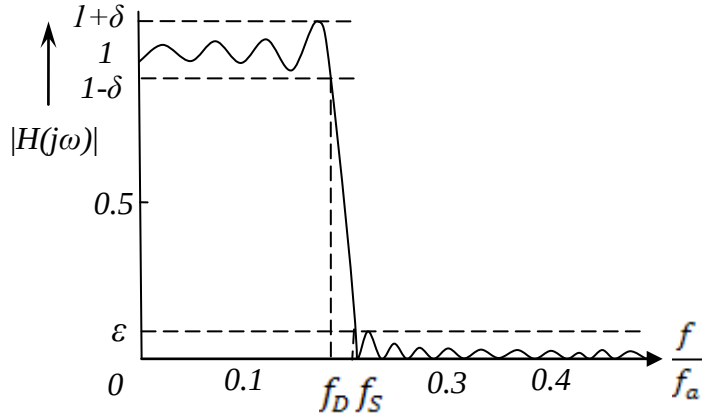


Рисунок 9.5 - Схема допусків амплітудно-частотної характеристики ФНЧ

9.4 Обчислення коефіцієнтів високочастотних, смугових і режекторних фільтрів

Коефіцієнти високочастотних *(high-pass filter (HPF))*, смугових *(band-pass filter (BPF))* та режекторних *(Band-rejection filter)* фільтрів можна обчислити з коефіцієнтів ФНЧ на основі теореми додавання перетворень Фур'є [5]. Якщо в частотній області додати спектральні функції, то дістанемо відповідну результуючу функцію в часовій області як суму вагових функцій, тобто коефіцієнтів фільтра.

Перетворення ФНЧ в ФВЧ. Всечастотний фільтр *(allpass filter)* пропускає, не послаблюючи, всі частоти. Його рівняння

$$y_n = x_n \qquad (9.14)$$

має тільки один коефіцієнт $a_0 = 1$. Всі інші коефіцієнти дорівнюють нулю, тобто, $a_k = 0$ при $k \neq 0$. Якщо відняти від спектра всечастотного фільтра спектр ФНЧ, то залишиться спектр ФВЧ з такою ж граничною частотою як у ФНЧ (рис. 9.6). Для коефіцієнтів фільтра це означає:

Коефіцієнт ФВЧ = коефіцієнт всечастотного фільтра мінус коефіцієнт ФНЧ

$$a_{k,ФВЧ} = a_{k,ВФ} - a_{k,ФНЧ}. \qquad (9.15)$$

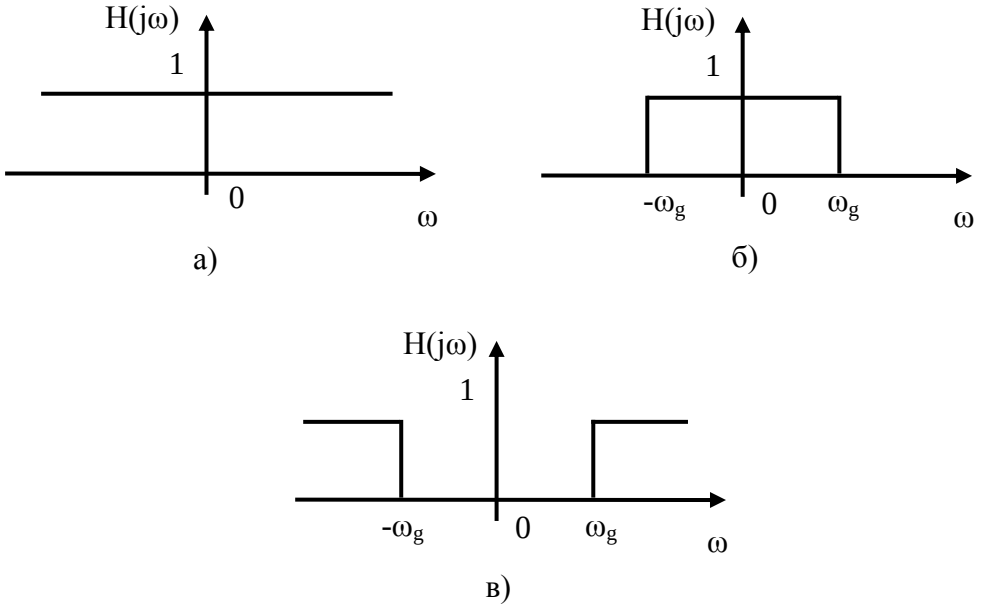


Рисунок 9.6 - Перетворення передавальних функцій ФНЧ в ФВЧ: а – АЧХ ідеального всечастотного фільтра; б - АЧХ ідеального ФНЧ; в - АЧХ ідеального ФВЧ

Приклад 9.2. Визначити параметри ФВЧ з граничною частотою f_g = 20 Гц у вигляді нерекурсивного фільтра при N= 3. Частота дискретизації fa = 100 Гц.

Розв'язування. Нормована гранична частота

$$\Omega_g = 2\pi\frac{f_g}{f_a} = 2\pi\frac{20}{100} = 0{,}4\pi.$$

Коефіцієнти ФНЧ обчислюються за рівнянням (9.7). Вони наведені в табл. 9.2 разом з добутими з них коефіцієнтами ФВЧ. Отже, бажаний ФВЧ з 2N+1 = 7 коефіцієнтами (рис. 9.7) має вигляд:

$$y_n = 0{,}6x_n - 0{,}3(x_{n-1} + x_{n+1}) - 0{,}09(x_{n-2} + x_{n+2}) + 0{,}06(x_{n-3} + x_{n+3}).$$

Таблиця 9.2 - Коефіцієнти FIR-ФВЧ

k	a_k (ВЧ)	a_k (ФНЧ)	a_k (ФВЧ)
0	1	0,4	0,6
± 1	0	0,3	-0,3
±2	0	0,09	-0,09
±3	0	- 0,06	0,06

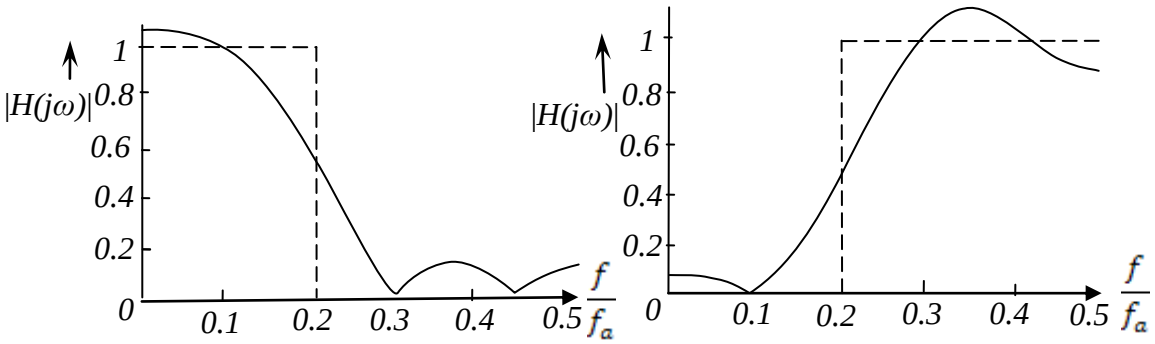


Рисунок 9.7 - Модулі передавальних функцій: *а* - ФНЧ, *б* - ФВЧ

Перетворення ФНЧ у смуговий фільтр. За основу візьмемо ФНЧ з граничною частотою $\omega_в$, що відповідає верхній частоті шуканого смугового фільтра. Із спектра ФНЧ віднімемо спектр другого ФНЧ з меншою граничною частотою $\omega_н$. Внаслідок залишиться спектр смугового фільтра з смугою пропускання між $\omega_н$ та $\omega_в$ (рис. 9.8). Коефіцієнти смугового фільтра обчислюються за формулою:

$$a_{k,СФ} = a_{k,ФНЧв} - a_{k,ФНЧн} \qquad (9.16)$$

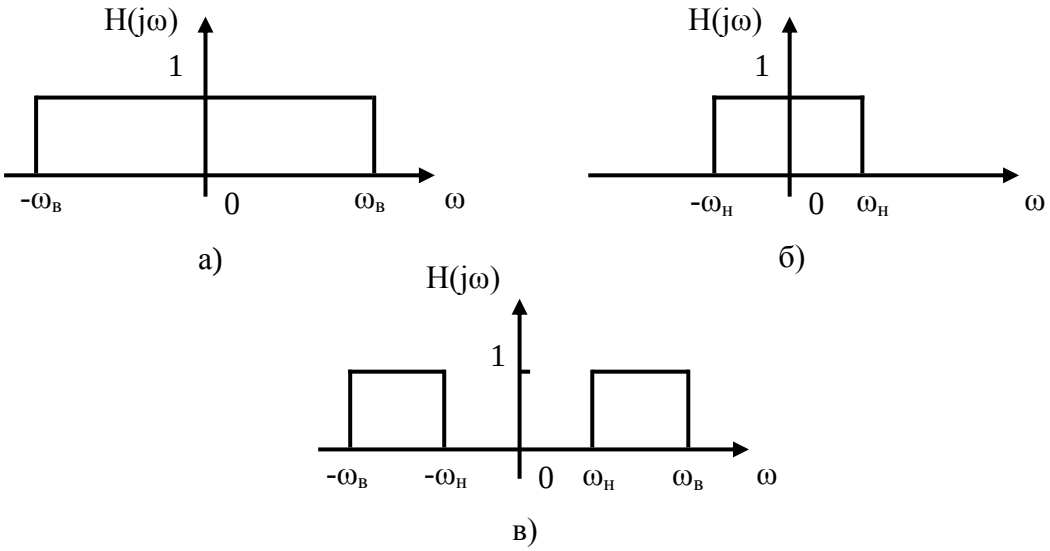


Рисунок 9.8 - Перетворення передавальних функцій двох ФНЧ у передавальну функцію смугового фільтра: а – АЧХ ідеального ФНЧ для верхньої частоти смугового фільтра; б - АЧХ ідеального ФНЧ для нижньої частоти смугового фільтра; в - АЧХ ідеального смугового фільтра

Приклад 9.3. Визначити коефіцієнти цифрового смугового фільтра з N = 3 для нижньої частоти $f_н$ = 5 кГц, верхньої $f_в$ = 10 кГц та частоти дискретизації f_a = 50 кГц.

Розв'язування. Коефіцієнти ФНЧ обчислювались за рівнянням (9.7), а коефіцієнти смугового фільтра за рівнянням (9.16) і наведені в табл. 9.3 Смуговий фільтр (рис. 9.9) описується таким рівнянням:

$$y_n = -0{,}16(x_{n+3} + x_{n-3}) - 0{,}06(x_{n+2} + x_{n-2}) + 0{,}11(x_{n+1} + x_{n-1}) + 0{,}20x_n.$$

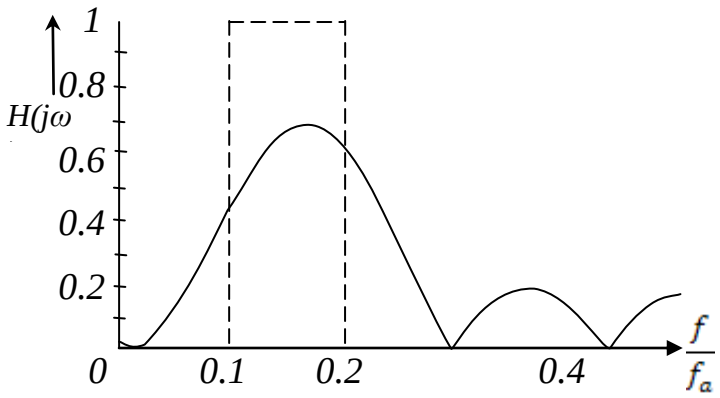


Рис. 9.9 - Амплітудна характеристика смугового фільтра

Таблиця 9.3- Коефіцієнти смугового та режекторного фільтрів

k	$a_{kВЧ}$	$a_{kФНЧв}$	$a_{kФНЧн}$	$a_{kСФ}$	$a_{kРФ}$
0	1	0,4	0,2	0,2	0,8
1	0	0,3	0,19	0,11	-0.11
2	0	0,09	0,15	-0,06	0,06
3	0	-0.06	0,1	-0,16	0,16

Перетворення ФНЧ в режекторний фільтр. Якщо зі спектра всечастотного фільтра відняти спектр смугового, то залишиться спектр режекторного фільтра (рис. 9.10). Для коефіцієнтів фільтра це означає:

$$a_{k,РФ} = a_{k,ВФ} - a_{k,СФ},$$
$$a_{k,РФ} = a_{k,ВЧ} - (a_{k,ФНЧв} - a_{k,ФНЧн}). \qquad (9.17)$$

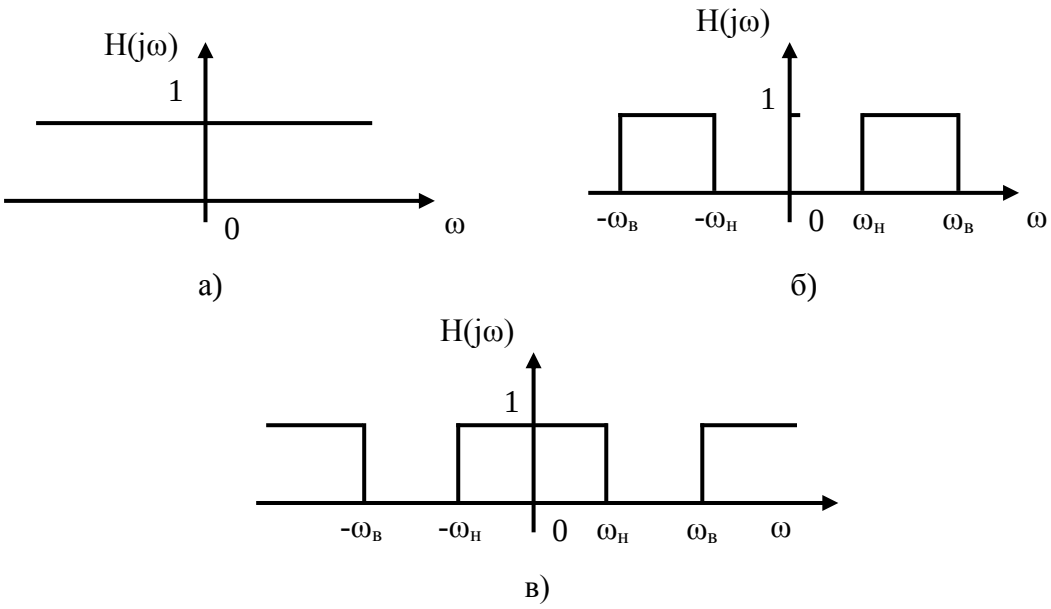


Рисунок 9.10 - Перетворення передавальних функцій всечастотного та смугового фільтрів у передавальну функцію режекторного фільтра: а – АЧХ ідеального всечастотного фільтра; б - АЧХ ідеального смугового фільтра; в - АЧХ ідеального режекторного фільтра

Приклад 9.4. Визначити коефіцієнти цифрового режекторного фільтра з параметрами, що відповідають прикладу 9.3.

Розв'язування. На підставі вже обчислених коефіцієнтів ФНЧ параметри режекторного фільтра визначаються за рівнянням (9.17) (табл. 9.3). Отже, режекторний фільтр (рис. 9.11) зі сімома коефіцієнтами описується рівнянням:

$$y_n = 0{,}16(x_{n+3} + x_{n-3}) + 0{,}06(x_{n+2} + x_{n-2}) - 0{,}11(x_{n+1} + x_{n-1}) + 0{,}80x_n.$$

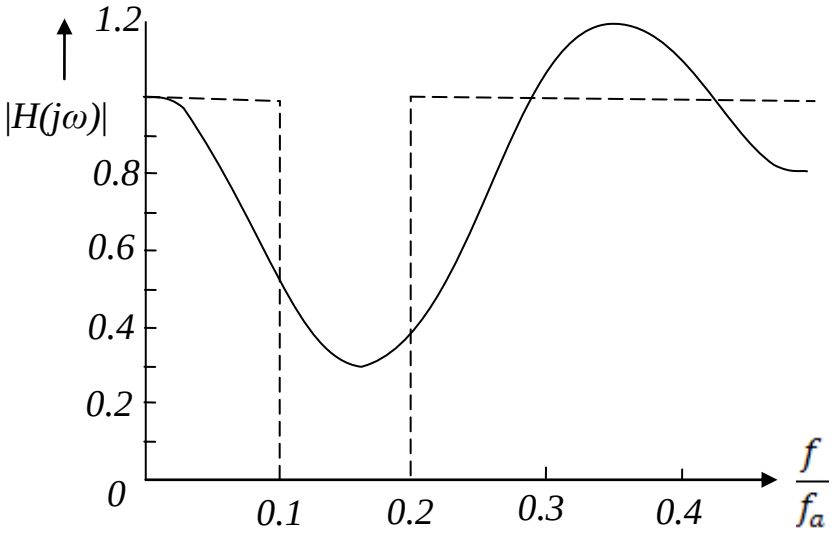


Рисунок 9.11 – АЧХ режекторного фільтра

Контрольні запитання і вправи

1. Охарактеризуйте фільтр низької частоти.
2. Охарактеризуйте фільтр високої частоти.
3. Наведіть АЧХ ідеального смугового фільтра та поясніть його призначення.
4. Наведіть АЧХ ідеального режекторного фільтра та поясніть його призначення.
5. Наведіть вирази для обчислення коефіцієнтів нерекурсивного ФНЧ.
6. Охарактеризуйте синтез нерекурсивного фільтра низької частоти (ФНЧ) методом найменших квадратів.
7. Що таке явище Гіббса?
8. В чому причина пульсацій АЧХ?
9. Які ви знаєте методи боротьби з пульсаціями АЧХ?
10. Для чого використовуються згладжувальні функції фон-Ганна та Ланчоса?
11. Наведіть різницеве рівняння фільтра зі згладжувальними функціями.
12. Яка повинна бути частота дискретизації для нерекурсивного ФНЧ?
13. Що таке всечастотний фільтр?
14. Як обчислити коефіцієнти цифрового ФВЧ?
15. Як обчислити коефіцієнти смугового фільтра?
16. Як обчислити коефіцієнти режекторного фільтра?
17. Ідеальний ФНЧ апроксимувати фільтром з $N = 2$ для граничної частоти $f_g = 50$ *Гц*, якщо частота дискретизації $f_a = 200$ *Гц*.
18. Визначити параметри ФВЧ з граничною частотою f_g = 50 Гц у вигляді нерекурсивною фільтра при N = 2. Частота дискретизації f_a = 200 Гц.
19. Визначити коефіцієнти цифрового смугового фільтра з N = 2 для нижньої частоти $f_н$ = 50 кГц, верхньої $f_в$ = 100 кГц та частоти дискретизації f_a = 500 кГц.
20. Визначити коефіцієнти цифрового режекторного фільтра з параметрами, що відповідають вправі 19.

10 РЕКУРСИВНІ ФІЛЬТРИ

10.1 Загальні поняття про рекурсивні фільтри

Рекурсивні фільтри визначаються так [5, 8-11]:

$$y_n = \sum_{k=0}^{N} a_k x_{n-k} - \sum_{k=1}^{M} b_k y_{n-k} . \qquad (10.1)$$

Більше з чисел M або N вказує на порядок фільтра. Блок-схема рекурсивного фільтра згідно з рівнянням (10.1) наведена на рис. 10.1.

Тобто, це фільтр зі зворотними зв'язками, у якому вихідний сигнал є зваженою сумою вхідних значень та попередніх вихідних значень. Причому зворотний зв'язок за вихідним сигналом може бути причиною нестабільної роботи фільтра.

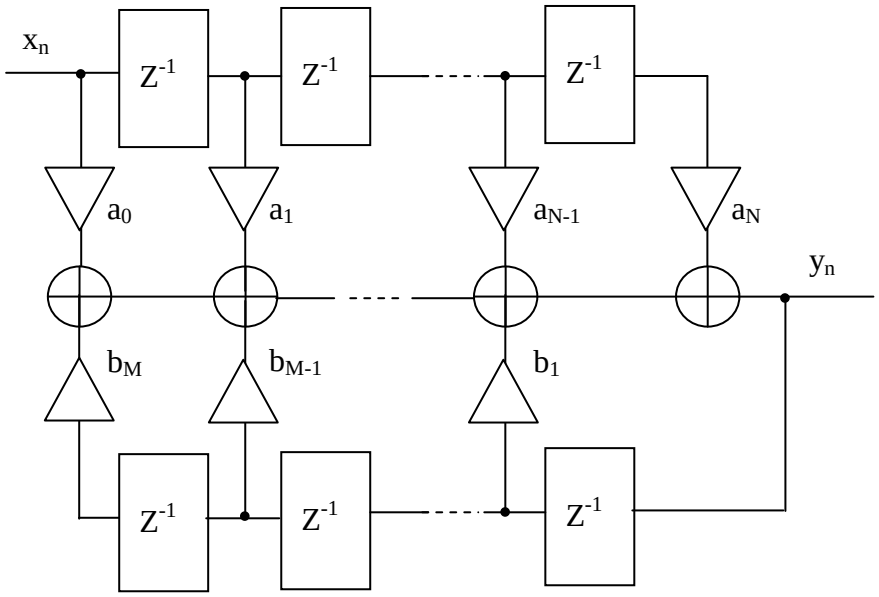


Рисунок 10.1 - Структурна схема рекурсивного фільтра

Імпульсний відгук. Нехай рекурсивний фільтр визначається таким рівнянням:

$$y_n = x_n + 0{,}5y_{n-1}.$$

А на вхід фільтра подано одиничний імпульс при нульових початкових умовах, тобто:

$$x_n = 1, \quad \text{для} \quad n = 0,$$

$$x_n = 0, \quad \text{для} \quad n \neq 0,$$

$$y_{0\text{-}1} = 0.$$

Тоді вихідні значення фільтра такі:

$$n = 0 : x_0 = 1; y_0 = x_0 + 0{,}5y_{0-1} = x_0 = 1,$$
$$n = 1 : x_1 = 0; y_1 = x_1 + 0{,}5y_0 = 0{,}5x_0 = 0{,}5,$$
$$n = 2 : x_2 = 0; y_2 = x_2 + 0{,}5y_1 = 0{,}5 * 0{,}5x_0 = 0{,}25,$$
$$\ldots$$
$$n = k : x_k = 0; y_k = x_k + 0{,}5y_{k-1} = (0{,}5)^k x_0 = (0{,}5)^k.$$

Отже, на вихідний сигнал y_k в момент часу kT_a все ще впливає вхідний сигнал x_0, що діяв в момент часу t = 0 (рис. 10.2). Тобто, рекурсивний фільтр має пам’ять і нескінченний імпульсний відгук:

$$y(kT_a) = (0{,}5)^k x_0.$$

Такі фільтри називаються фільтрами з нескінченним імпульсним відгуком, або IIR-фільтром.

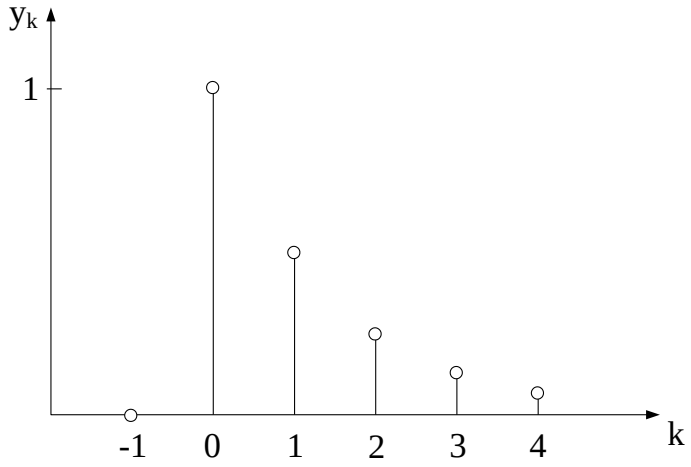


Рисунок 10.2 – Імпульсний відгук рекурсивного фільтра

Передавальна функція РФ. Відомо, що передавальна функція фільтра визначається як відношення Z-перетворень вихідного і вхідного сигналів:

$$H(z) = \frac{Y(z)}{X(z)}. \qquad (10.2)$$

Знайдемо Z-перетворення вихідного сигналу фільтра. З рівняння (10.1) одержимо:

$$Y(z) = \sum_{k=0}^{N} a_k z^{-k} X(z) - \sum_{k=1}^{M} b_k Y(z) z^{-k}.$$

Звідки:

$$Y(z) = \frac{X(z)\sum_{k=0}^{N} a_k z^{-k}}{1 + \sum_{k=1}^{M} b_k z^{-k}}.$$

Тоді з урахуванням (10.2) передавальна функція рекурсивного фільтра визначається так:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{N} a_k z^{-k}}{1 + \sum_{k=1}^{M} b_k z^{-k}}. \qquad (10.3)$$

Тепер комплексна частотна і амплітудно-частотна характеристики можуть бути визначені як показано в підрозділі 8.2.

Приклад 10.1. Нехай рекурсивний фільтр визначається таким різницевим рівнянням: $y_n = a_0 x_n - b_1 y_{n-1}$. Знайти АЧХ фільтра.

Розв'язування. З урахуванням того, що N = 0, M = 1 з виразу (10.3) знайдемо:

$$H(z) = \frac{a_0}{1 + b_1 z^{-1}}.$$

Для знаходження частотної характеристики фільтра підставимо в цей вираз $z = e^{j\omega T_a}$. Тоді одержимо:

$$H(j\omega) = \frac{a_0}{1 + b_1 e^{-j\omega T_a}}.$$

Оскільки $e^{-j\omega T_a} = \cos\omega T_a - j\sin\omega T_a$, то

$$H(j\omega) = \frac{a_0}{1 + b_1\cos\omega T_a - b_1\sin\omega T_a}.$$

Амплітудно-частотна характеристика фільтра така:

$$|H(j\omega)| = \frac{a_0}{\sqrt{(1+b_1\cos\omega T_a)^2 + (b_1\sin\omega T_a)^2}} = \frac{a_0}{\sqrt{1+b_1^2+2b_1\cos\omega T_a}}.$$

Нехай a_0=1 а b_1= - 0,5:

$$|H(j\omega)| = \frac{1}{\sqrt{1+0{,}25-\cos\omega T_a}}.$$

Це рекурсивний ФНЧ (рис. 10.3) з періодичною АЧХ.

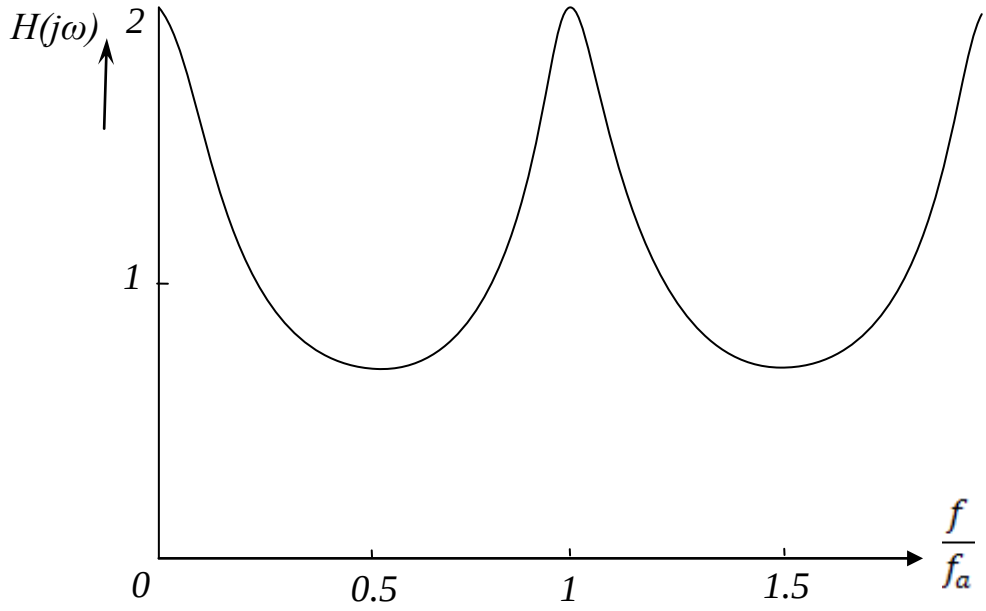


Рисунок 10.3 – АЧХ рекурсивного фільтра

10.2 Типи РФ

Рекурсивні фільтри характеризуються рядом властивостей. Це такі як пульсації в зоні запирання і пропускання, крутість характеристики при переході від зони пропускання до зони запирання, придатність для передачі імпульсів та ін. Жоден з фільтрів не відповідає усім перерахованим вимогам одночасно.

Розглянемо декілька типів фільтрів, які використовуються найчастіше [5].

1. ФНЧ Баттерворта (степеневий фільтр - *Butterworth filter*). Оптимізований для максимально плоскої амплітудної характеристики в зоні пропускання. Крутість зростає з порядком N фільтра. У реакції на стрибок помітні випади, тому цей фільтр спотворює форму імпульсів.

2. ФНЧ Бесселя *(Bessel filter)*. Амплітудна характеристика цього фільтра спадає менш круто в порівнянні з попереднім. Добре передає форму прямокутних імпульсів.
3. ФНЧ Чебишева *(Chebyshev filter)*. Має пульсації в області пропускання або запирання з певною максимальною амплітудою. Крутіший в порівнянні з попередніми перехід із зони пропускання в зону запирання. Реакція на стрибок характеризується сильними викидами.
4. ФНЧ Кауера *(Cauer filter)*. АЧХ має пульсації як в зоні пропускання, так і запирання і значну крутість фронту імпульсу АЧХ. Для додержання заданих допусків порядок фільтра Кауера знаходиться нижче, ніж у інших типах фільтрів. Для відтворення імпульсів непридатний.

10. 3 Каталог параметрів РФ

При синтезі рекурсивного цифрового фільтра за основу, як правило, беруть відповідний аналоговий фільтр, передавальна функція якого через білінійне перетворення ($p = l\frac{z-1}{z+1}$, де $l = ctg\pi\frac{f_g}{f_a}$) відображається на z-площину і в результаті одержують коефіцієнти цифрового фільтра. Цей шлях складний.

В [5] уже наведені в таблицях коефіцієнти прототипів ФНЧ (С. 300 – 303). Значення, що стосуються прототипів, помічаються зверху зірочкою, причому частота дискретизації в 4 рази більша, ніж гранична частота і фільтри нормовані, тобто:

$$f_a^* = 4f_g^*$$
$$b_0^* = 1 \qquad (10.4)$$

З цих таблиць вибирають необхідний фільтр і перераховують АЧХ прототипу H(z*) за допомогою частотного перетворення в АЧХ шуканого фільтра H(z). Якщо не відповідає вимогам, то вибирають інший і т. д.

Каскадне перетворення. ФНЧ високих порядків можуть бути реалізовані шляхом послідовного включення ФНЧ першого або другого порядку. Передавальна функція загального фільтра є добутком передавальних функцій часткових фільтрів:

$$H(z^*) = \prod_{i=0}^{P} \frac{a_{0i}^* + a_{1i}^* z^{*-1} + a_{2i}^{*-2}}{b_{0i}^* + b_{1i}^* z^{*-1} + b_{2i}^* z^{*-2}}. \qquad (10.5)$$

10.4 Частотне перетворення

На основі прототипу ФНЧ можна отримати і інші типи фільтрів за допомогою частотного перетворення.

1. Прототип ФНЧ→ФНЧ

$$z^{*-1} = \frac{z^{-1} - d}{1 - dz^{-1}}, \quad d = \frac{\sin(\pi \frac{f_g^*}{f_a^*} - \pi \frac{f_g}{f_a})}{\sin(\pi \frac{f_g^*}{f_a^*} + \pi \frac{f_g}{f_a})}. \qquad (10.6)$$

2. Прототип ФНЧ→ФВЧ

$$z^{*-1} = \frac{z^{-1} + d}{1 + dz^{-1}}, \quad d = \frac{\cos(\pi \frac{f_g^*}{f_a^*} + \pi \frac{f_g}{f_a})}{\sin(\pi \frac{f_g^*}{f_a^*} + \pi \frac{f_g}{f_a})}. \qquad (10.7)$$

3. Прототип ФНЧ→ СФ (смуговий фільтр)

$$z^{*-1} = \frac{z^{-2} - \frac{2dk}{k+1} z^{-1} + \frac{k-1}{k+1}}{\frac{k-1}{k+1} z^{-2} - \frac{2dk}{k+1} z^{-1} + 1}, \quad d = \frac{\cos(\pi \frac{f_в}{f_a} + \pi \frac{f_н}{f_a})}{\cos(\pi \frac{f_в}{f_a} - \pi \frac{f_н}{f_a})}, \qquad (10.8)$$

$$k = \operatorname{ctg}(\pi \frac{f_в}{f_a} - \pi \frac{f_н}{f_a}) \operatorname{tg} \pi \frac{f_g^*}{f_a^*}.$$

4. Прототип ФНЧ→ РФ (режекторний фільтр)

$$z^{*-1} = \frac{z^{-2} - \frac{2d}{k+1}z^{-1} + \frac{k-1}{k+1}}{\frac{1-k}{1+k}z^{-2} - \frac{2d}{k+1}z^{-1} + 1}, \quad d = \frac{\cos(\pi\frac{f_в}{f_a} + \pi\frac{f_н}{f_a})}{\cos(\pi\frac{f_в}{f_a} + \pi\frac{f_н}{f_a})}, \qquad (10.9)$$

$$k = tg(\pi\frac{f_в}{f_a} - \pi\frac{f_н}{f_a})tg\pi\frac{f_g^*}{f_a^*}.$$

Порядок знаходження коефіцієнтів фільтра такий.

1. Визначити тип фільтра і його порядок і взяти коефіцієнти прототипу фільтра з таблиць (наприклад з [5], С. 300 - 303).
2. Задати передавальну функцію прототипу H(z*) у вигляді (10.5).
3. За допомогою частотних перетворень (10.6 – 10.9) з H(z*) отримаємо передавальну функцію бажаного фільтра H (z) і нормуємо її так, щоб коефіцієнт $b_0 = 1$.
4. Вибрати з передавальної функції H (z) коефіцієнти фільтра і задати рівняння фільтра.
5. Проконтролювати, щоб фільтр відповідав заданим вимогам. Якщо ні, то вибирається інший прототип і знову виконуються всі пункти, наведені вище.

Контрольні запитання і вправи

1. Наведіть різницеве рівняння рекурсивного фільтра.
2. Що таке порядок фільтра?
3. Який імпульсний відгук мають рекурсивні фільтри?
4. Охарактеризуйте РФ різного типу.
5. Що таке каталог параметрів рекурсивних фільтрів?
6. Для чого використовується частотне перетворення?
7. Наведіть порядок знаходження коефіцієнтів рекурсивного фільтра.
8. Знайдіть АЧХ та наведіть структурну схему такого рекурсивного фільтра: $y_n = 0{,}5\, x_n + 0{,}5\, x_{n-1} – 2y_{n-1} + y_{n-2}$.
9. Синтезуйте ФВЧ Баттерворта другого порядку при f_g = 10 кГц та частоті дискретизації f_a = 50 кГц.

ВИСНОВКИ

Знання основ обробки сигналів стає в останні роки обов'язковим для інженерів-програмістів у зв'язку з широким застосуванням цифрових методів обробки, зберігання та передачі аналогових сигналів. Сучасні універсальні процесори здатні в реальному часі обробляти живе відео та звук та багато інших сигналів, що дає можливість ці задачі вирішувати програмним способом. З'явилась маса прикладних задач в галузі програмування, які потребують знання основ теорії обробки сигналів. В першу чергу це задачі кодування зображень і звуку, які ґрунтуються на ортогональних перетвореннях, фільтрація сигналів, розпізнавання образів. Причому маємо на увазі як використання відомих алгоритмів і стандартів, таких, наприклад, як стандарти MPEG (передача і кодування живого відео і звуку) або JPEG (кодування фотографічних зображень), так і пошук нових підходів, який потребує розробки програмних засобів для їх моделювання. Розробка систем Інтернет-телебачення або Інтернет-радіо також потребує знання основ обробки сигналів. Ще однією важливою задачею є розробка систем для спілкування за технологією P2P (peer-to-peer - рівний-рівному), де звукові та відеоповідомлення передаються безпосередньо з комп'ютера на комп'ютер. І звичайно комп'ютерні ігри, які без зображення і звуку взагалі неможливі.

Цей навчальний посібник, хоча і охоплює лише початкові знання з теорії обробки сигналів, є доброю основою для подальшого самостійного вивчення теорії обробки сигналів або окремих розділів цієї дисципліни. В цьому вам допоможе список літератури наведений в даному посібнику, зокрема з прикладних питань обробки сигналів [12-17], а також сайти Інтернет, які спеціалізуються на розповсюдженні безкоштовної навчальної інформації про методи теорії обробки сигналів. Можна відзначити сайт www.twirpx.com, який дає всебічну інформації про методи обробки сигналів та про сучасний стан досліджень в даній області. Є також ряд сайтів, які дозволяють безкоштовно для некомерційного використання скачувати відсканованні посібники, підручники, монографії з теорії інформації як ті, що давно видавались і їх практично неможливо знайти в паперовому вигляді, так і ті, що тільки вийшли з друку. Деякі з них такі: www.mirknig.com, www.knigka.info, www.ihtik.lib.ru, http://lib.prometey.org, www.litportal.kiev.ua, http://physicsbooks.narod.ru, http://www.4tivo.com.

ЛІТЕРАТУРА

1. Кузьмин И. В. Основи теории информации и кодирования / И. В. Кузьмин, В. А. Кедрус. – К. : Вища шк., 1986. – 238 с.
2. Дмитриев В. И. Прикладная теория информации / Дмитриев В. И. – М. : Высш. шк., 1989. – 420 с.
3. Баскаков С. И. Радиотехнические цепи и сигналы: учебник для вузов / Баскаков С. И. - М. : Высшая школа, 1988. – 448 с.
4. Гоноровский И. С. Радиотехнические цепи и сигналы / Гоноровский И. С. – М. : Радио и связь, 1986. – 512 с., ил.
5. Бабак В. П. та ін. Обробка сигналів: підручник / Бабак В. П., Хандецький В. С., Шрюфер Е. – К. : Либідь, 1996. – 392 с.
6. Залманзон Л. А. Преобразование Фурье, Уолша, Хаара и их применение в управлении, связи и других областях / Залманзон Л. А. – М. : Наука, 1989. – 496 с.
7. Блейхут Р. Быстрые алгоритмы цифровой обработки сигналов / Р. Блейхут; пер. с англ. И. И. Грушко. – М. : Мир, 1989. – 448 с.
8. Рабинер Л. Теория и применение цифровой обработки сигналов / Л. Рабинер, Б. Гоулд. – М. : Мир, 1978. – 848 с.
9. Гольденберг Л. М. Цифровая обработка сигналов: учебное пособие для вузов / Гольденберг Л. М. – М. : Радио и связь,1990. – 256 с.
10. Хемминг Р. В. Цифровые фильтры / Хемминг Р. В. – М. : Недра, 1987. – 221с.
11. Сергиенко А. Б. Цифровая обработка сигналов / Сергиенко А. Б. – СПб. : Питер, 2003. – 608 с.
12. Айфичер Э. Цифровая обработка сигналов. Практический подход / Э. Айфичер, Б. Джервис. – М. : "Вильямс", 2004. - 992 с.
13. Даджион Д. Цифровая обработка многомерных сигналов / Д. Даджион, Р. Мерсеро. – М. : Мир, 1988. – 488 с.
14. Коуэн К. Адаптивные фильтры / К. Коуэн , П. Грант. – М. : Мир, 1988. – 392 с.
15. Майданюк В. П. Кодування та захист інформації: навчальний посібник / Майданюк В. П. – Вінниця : ВНТУ, 2009. – 164 с.
16. Сэломон Д. Сжатие данных, изображений и звука / Сэломон Д. – М. : Техносфера, 2004. – 368 с.
17. Гонсалес Р. Цифровая обработка зображений / Р. Гонсалес, Р. Вудс. – М. : Техносфера, 2005. – 1072 с.

СЛОВНИК ОСНОВНИХ ТЕРМІНІВ (GLOSSARY)

Амплітудно-частотна характеристика – Frequency response (FR)
Z-перетворення – Z-transform
Всечастотний фільтр – Allpass filter
Дискретне косинусне перетворення (ДКП) – Discrete cosine transform(DCT)
Дискретне перетворення Фур'є – Discrete Fourier transform (DFT)
Імпульсна характеристика – impulse response function (IRF)
Лінійне перетворення – Linear transform
Нерекурсивний фільтр – Finite impulse response filter (FIR-filter)
Нескінченна імпульсна характеристика (НІХ) – Infinite impulse response (IIR)
Одинична імпульсна функція (дельта-функція, δ-функція Дірака, діраківська дельта) – Unit impulse function (Dirac delta or Dirac's delta)
Одинична ступінчата функція (функція Хевісайда) - Unit step function (Heaviside step function)
Перетворення Фур'є – Fourier transform
Режекторний фільтр (РФ) – Band-rejection filter
Рекурсивний фільтр – Infinite impulse response filter (IIR-filter)
Ряд Фур'є – Fourier series
Згортка – Convolution
Сигнал – Signal
Скінченна імпульсна характеристика (СІХ) – Finite impulse response (FIR)
Смуговий фільтр (СФ) – Band-pass filter (BPF)
Смугово-загороджувальний фільтр (СЗФ, теж саме, що РФ) - Band-stop filter
Спектр – Spectr , Spectrum
Унітарне перетворення – Unitary transform
Фазочастотна характеристика (ФЧХ) – Phase response (PR)
Фільтр – Filter
Фільтр Баттерворта – Butterworth filter
Фільтр Бесселя – Bessel filter
Фільтр високої частоти (ФВЧ) – High-pass filter (HPF)
Фільтр Калмана – Kalman filter
Фільтр низької частоти (ФНЧ) – Low-pass filter (LPF)
Фільтр Чебишева – Chebyshev filter
Цифрова обробка сигналів (ЦОС) – Digital signal processing (DSP)
Цифровий фільтр – Digital filter
Швидке перетворення Фур'є (ШПФ) – Fast Fourier transform (FFT)

ДОДАТОК А
ОСНОВИ РОБОТИ В МАТЛАБ

Система MATLAB (скорочення від MATrix LABoratory — матрична лабораторія) з'явилася у 1984 р. і стала світовим стандартом в області наукових і технічних розрахунків. Основна причина популярності, ймовірно, криється в тому, що MATLAB дав інженерам і ученим саме те, що їм було потрібно, - можливість з неперевершеною легкістю застосовувати до довільних даних, поданих у вигляді векторів і матриць, різноманітні чисельні алгоритми. Зручна мова програмування, в якій завдяки матричній орієнтації системи значно зменшилась необхідність в циклах, ще більше розширила сферу застосування MATLAB. У даному додатку наведені основні відомості про роботу з MATLAB. Цієї інформації вистачає для того, щоб познайомитися із системою і потім продовжити освоювати її самостійно. За докладнішими відомостями звертайтеся до книг, цілком присвячених MATLAB, а також, до довідкової системи і документації [11]. Інформація, що наводиться в даному додатку, відповідає версії MATLAB 6.5.

Робота в інтерактивному режимі

Після запуску MATLAB на екрані з’явиться вікно, показане на рис. А.1. MATLAB - інтерактивна діалогова система, тому велика частина її головного вікна призначена для введення команд і виведення результатів. Ця область називається командним вікном - Command Window. Спробуємо обчислити значення будь-якого виразу:

```
>> sqrt(cos(pi/12)^2 + 1)
ans =
1.3903
```

Отже, MATLAB може працювати як дуже великий і потужний калькулятор, обчислюючи значення математичних виразів. При цьому арифметичні операції записуються традиційно, для піднесення до степеня використовується знак ^, порядком обчислень можна управляти за допомогою круглих дужок, і в круглих же дужках записуються аргументи функцій, що викликаються (у даному прикладі використовуються функції косинуса cos і квадратного кореня sqrt).

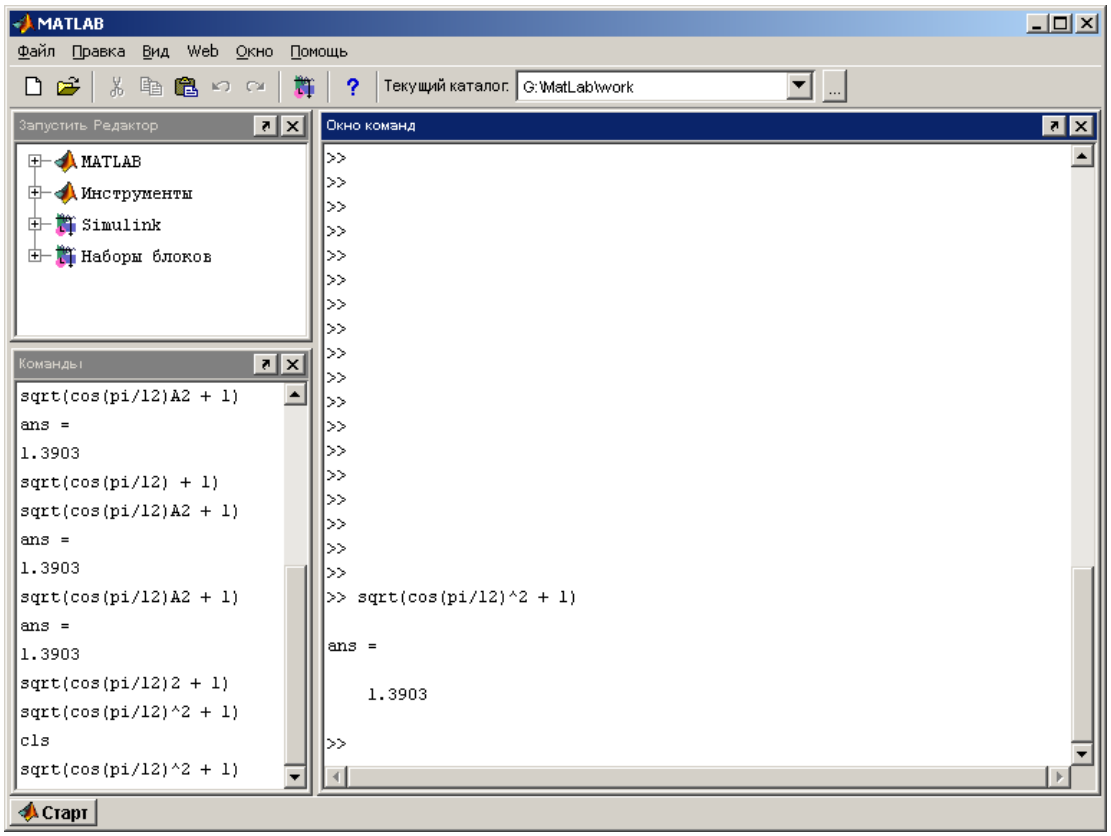


Рисунок А.1 – Головне вікно МАТЛАБ

Оскільки ми не присвоїли результат обчислень ніякій змінній, MATLAB автоматично створив для нього змінну з ім'ям ans і показав її значення (рис. А.1).

Проте головна цінність MATLAB полягає в можливості одночасно оперувати декількома значеннями. Введемо таку команду:

```
>> x = 1:0.5:5
x =
Columns 1 through 7
1.0000   1.5000   2.0000   2.5000   3.0000   3.5000   4.0000
Columns 8 through 9
4.5000    5.0000
```

Ми створили змінну x, яка є вектор-рядком, що містить 9 елементів. Вираз, що стоїть праворуч від знака рівності, генерує арифметичну прогресію, перший член якої дорівнює 1, різниця між сусідніми елементами дорівнює 0,5, а останній член не перевершує 5.

Імена змінних MATLAB чутливі до регістра символів, так що x і X - це дві різні змінні. Проте імена функцій до регістра не чутливі, оскільки фактично є іменами файлів (див. далі розділ «Створення функцій»).

Над отриманим вектором можна виконувати математичні операції:

```
>> x * 2
ans =
2   3   4   5    6   7    8   9   10
```

Можна застосовувати до векторів математичні функції:

```
>> exp(x)
ans =
Columns 1  through 7
2.7183   4.4817   7.3891   12.1825   20.0855   33.1155   54.5982
Columns 8  through 9
90.0171   148.4132
```

Але обчислення такого виразу дає несподіваний результат:

```
>> exp(x) /x
ans =
16.6289
```

А цей приклад взагалі викличе появу повідомлення про помилку:

```
>> x * x
??? Error using ==> *
Inner matrix dimensions must agree.
```

Річ у тому, що операції множення, ділення і піднесення до степеня в MATLAB виконуються не поелементно, а за правилами матричних операцій. Матричні правила забороняють множення рядка на рядок, тому видано повідомлення про несумісність розмірів перемножуваних матриць.

Несумісність розмірів матриць - ймовірно, одна з помилок, що найчастіше зустрічаються. Є дві головні причини її виникнення. По-перше, може виявитися неправильною орієнтація будь-якого з використовуваних у виразі масивів (приклад - спроба скласти вектор-рядок і вектор-стовпець). По-друге, будь-який з проміжних результатів, що на ваш погляд є числом (скаляром), може насправді виявитися матрицею - скажімо, через тонкощі

роботи функції, що викликається, або через особливості оброблюваних даних.

Щоб провести поелементні дії, до знака операції необхідно додати точку спереду (.*, ./ або .^). Тепер ми можемо отримати бажані результати, переписавши два останні приклади:

```
>> exp(x)./ x
ans =
Columns I through 7
2.7183   2.9878   3.6945   4.8730   6.6952   9.4616   13.6495
Columns 8 through 9
20.0038   29.6826
>> x .* x
ans =
Columns 1 through 7
1.0000   2.2500   4.0000   6.2500   9.0000   12.2500   16.0000
Columns 8 through 9
20.2500   25.0000
```

Щоб продемонструвати саме матричне множення, створимо другий вектор, причому не рядок, а стовпець:

```
>> y = x'
y =
1.0000
1.5000
2.0000
2.5000
3.0000
3.5000
4.0000
4.5000
5.0000
```

Апостроф в MATLAB позначає ермітове з’єднання - поєднання транспонування з комплексним з’єднанням. В даному випадку вектор x дійсний, тому комплексне з’єднання нічого не змінює. Для виконання транспонування без комплексного з’єднання до апострофа необхідно додати точку(.').

Тепер у нас є вектор-рядок x і вектор-стовпець y, розмірності яких дозволяють перемножувати їх, причому у будь-якому порядку:

```
 >> x * y
ans =
96
>> y * x
ans =
  Columns 1 through 7
    1.0000    1.5000    2.0000    2.5000    3.0000    3.5000    4.0000
    1.5000    2.2500    3.0000    3.7500    4.5000    5.2500    6.0000
    2.0000    3.0000    4.0000    5.0000    6.0000    7.0000    8.0000
    2.5000    3.7500    5.0000    6.2500    7.5000    8.7500   10.0000
    3.0000    4.5000    6.0000    7.5000    9.0000   10.5000   12.0000
    3.5000    5.2500    7.0000    8.7500   10.5000   12.2500   14.0000
    4.0000    6.0000    8.0000   10.0000   12.0000   14.0000   16.0000
    4.5000    6.7500    9.0000   11.2500   13.5000   15.7500   18.0000
    5.0000    7.5000   10.0000   12.5000   15.0000   17.5000   20.0000

  Columns 8 through 9
    4.5000    5.0000
    6.7500    7.5000
    9.0000   10.0000
   11.2500   12.5000
   13.5000   15.0000
   15.7500   17.5000
   18.0000   20.0000
   20.2500   22.5000
   22.5000   25.0000
```

Відповідно до правил матричних операцій при множенні рядка на стовпець вийшло число, а при множенні стовпця на рядок - квадратна матриця.

Вектори і матриці можна формувати і з окремих елементів, при цьому елементи в рядку розділяються пропусками або комами, між рядками ставиться крапка з комою, а все в цілому обмежується квадратними дужками:

```
>> M=[1,2; 3,4]
M =
    1    2
    3    4
```

Крапка з комою має і ще одне важливе застосування - розміщена в кінці рядка, вона блокує виведення результату обчислень на екран:

```
>> M2=M*M;
```

Дізнатися значення змінної можна просто, ввівши її ім'я:

```
>> M2
M2 =
    7   10
   15   22
```

І на закінчення першого знайомства наведемо нескладний приклад, пов'язаний з тематикою книги - обробкою сигналів. Спершу задамо частоту дискретизації Fs рівною 8 кГц:

```
>> Fs = 8e3:
```

Тепер сформуємо вектор значень часу, що охоплює з цією частотою дискретизації інтервал завдовжки в одну секунду:

```
>> t = 0:1/Fs:1:
```

Розрахуємо для цих моментів часу значення періодичної послідовності двополярних прямокутних імпульсів з частотою надходження f_0, рівною 50 Гц. Для цього обчислимо значення синусоїди потрібної частоти і застосуємо до них знакову функцію sign:

```
>> f0 = 50:
>> s = sign(sin2*pi*f0*t)):
```

Тепер підготуємо фільтр для обробки нашого сигналу. Хай це буде ФНЧ Баттерворта 4-го порядку з частотою зрізу f_1, рівною 200 Гц. Такий фільтр розраховується за допомогою функції butter з пакета розширення Signal Processing:

```
>> [b,a]=butter (4, f1*2/Fs)
b =
  1.0e-003 *
    0.0312    0.1250    0.1874    0.1250    0.0312
a =
    1.0000   -3.5897    4.8513   -2.9241    0.6630
```

Тут ми вперше стикаємося з цікавою особливістю MATLAB: на відміну від більшості мов програмування, функції MATLAB можуть повертати декілька результатів (у даному прикладі це вектори b і a - коефіцієнти поліномів чисельника і знаменника функції передачі розрахованого фільтра).

Вихідні параметри функції перераховуються через кому в квадратних дужках. При використанні такої функції у складі математичного виразу значенням функції є перший із списку повернутих нею результатів.

Тепер можна обробити сигнал s фільтром, що описується векторами b і a. Для цього призначена функція filter:

```
>> s2 = filter(b, a, s);
```

Нарешті поглянемо на отримані результати, вивівши графіки перших 200 значень вхідного і вихідного сигналів за допомогою функції plot:

```
>> plot(t(1:200), s(1:200). T(1:200). s2(1:200), ':')
```

Результуючий графік, який виводиться в окремому графічному вікні (figure window), показаний на рис. А. 2. Докладніше про функцію plot буде розказано далі, в розділі «Графіка» даного додатка.

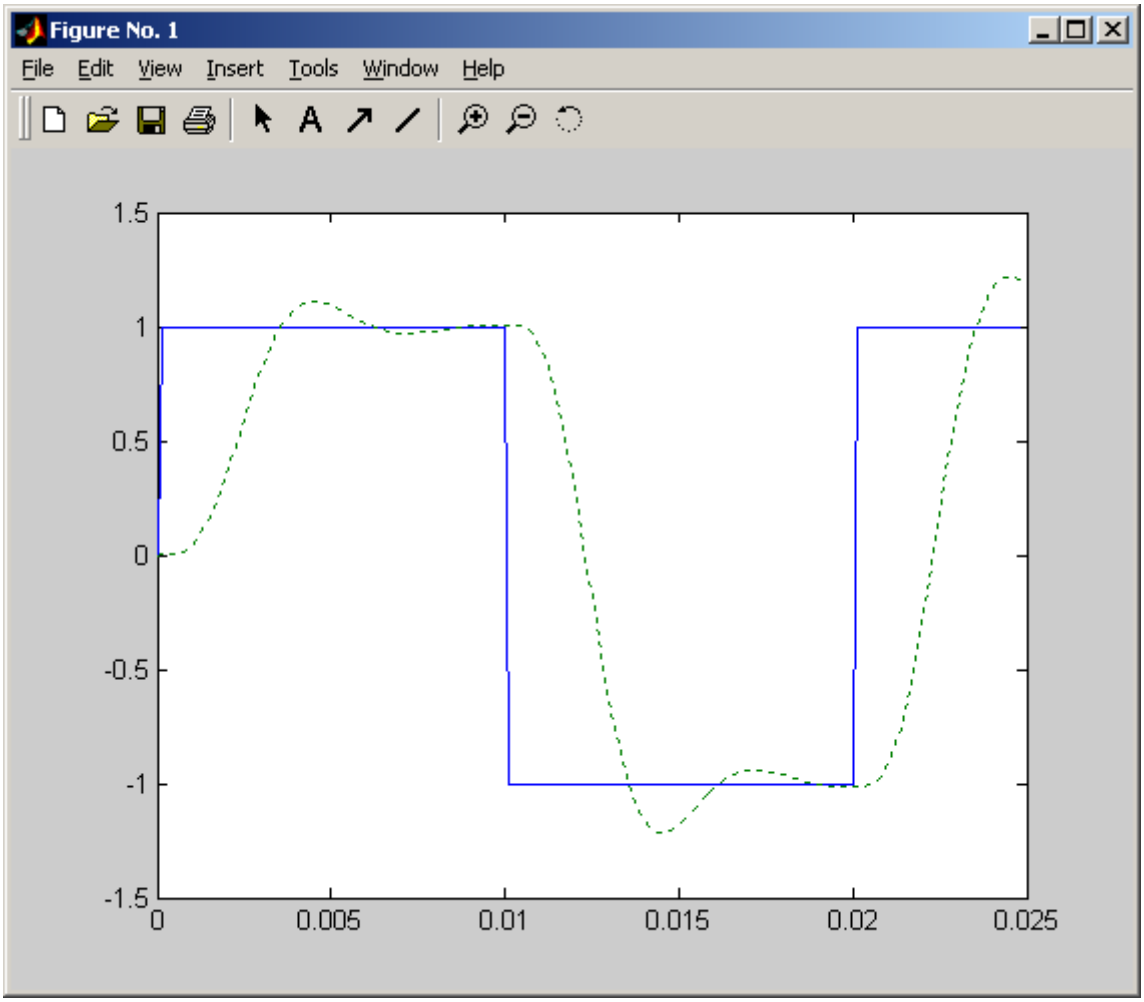


Рисунок А. 2 - Графіки вхідного (суцільна лінія) і вихідного (пунктирна лінія) сигналів дискретного фільтра

Інтерфейс головного вікна

Необхідно відзначити головне - MATLAB був і залишається системою командного рядка, і всі елементи інтерфейсу, які з'явилися в п'ятій і особливо в шостій версії, є лише засобом доступу до деяких команд за допомогою маніпулятора типу «миша». Проте як і раніше абсолютно всі необхідні операції можна виконувати з командного рядка.

На рис. А. 3 зліва від командного вікна видно чотири вікна, кожне з яких, у свою чергу, має дві вкладки. Включити всі п'ять вікон можна, вибравши пункт меню на вкладці *Вид* - *«Вид - Схема Рабочего стола – Пять Панелей».*

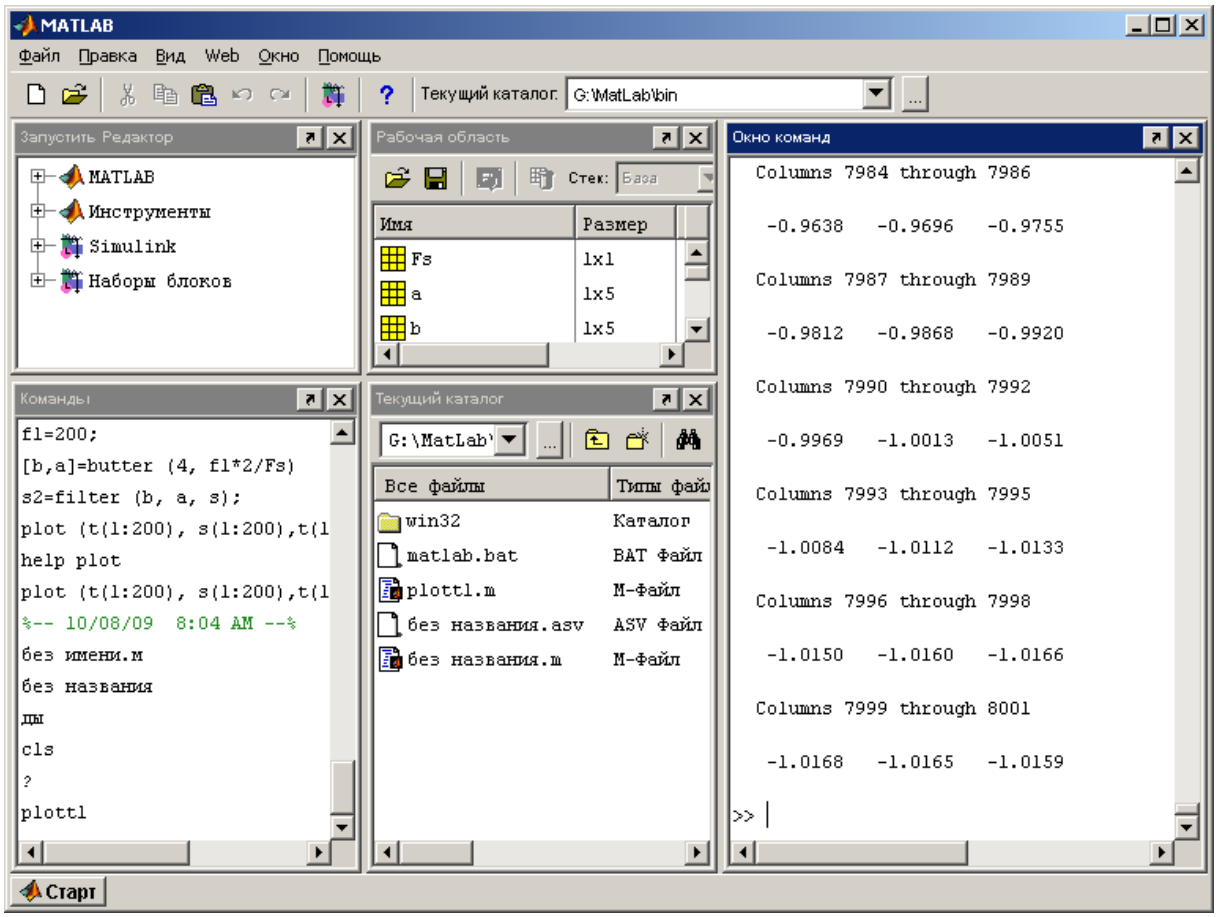


Рисунок А. 3 – П’ять панелей головного вікна

На вкладці *Команды (Command History)* поданий список команд, що вводилися, причому цей список охоплює не лише поточний сеанс роботи, але і попередні. Команди зі списку можна повторно виконувати (подвійним клацанням миші) або перетягувати в командний рядок для редагування. Ще один спосіб доступу до попередніх використаних команд - їх «перегортання» в командному рядку за допомогою клавіш ↑ і ↓.

Вкладка *Робочая область (Workspace)* демонструє вміст робочої області пам’яті MATLAB. Тут подані імена змінних (Name), їх розмір в елементах (Size), число займаних байтів пам’яті (Bytes) і тип даних (Class). Подвійне клацання на імені змінної викликає візуальний редактор масивів - Array Editor.

Вкладка *Запустить Редактор (Launch Pad)* показує список встановлених компонентів MATLAB і надає зручний доступ до їх документації, демонстраційних прикладів і спеціалізованого графічного середовища.

Вкладка *Текущий каталог (Current Directory)* дозволяє працювати з файлами поточного каталога - відкривати їх для редагування, перейменовувати, видаляти, запускати MATLAB-програми.

Що стосується панелі інструментів головного вікна MATLAB (рис. А. 3), відзначимо лише список, що розкривається, *Текущий каталог (Current Directory)*, в якому перераховані робочі каталоги, що використалися. При роботі з декількома MATLAB-проектами цей список дає можливість зручного перемикання між ними. Вибрати новий робочий каталог дозволяє кнопка з трьома крапками, розташована праворуч від списку.

Масиви

Для простоти можна вважати, що все, з чим оперує MATLAB, є масивами, які складаються з комплексних чисел, дійсна і уявна частини яких подані в 8-байтовому форматі double. Навіть звичайне число (скаляр) з погляду MATLAB є матрицею розміром 1×1.

MATLAB - інтерпретуюча система, тому масиви не потрібно якось спеціально описувати. Розмір масиву може змінюватися в ході роботи - якщо присвоїти значення елементу масиву, вказавши індекс, що перевищує поточний розмір, MATLAB збільшить масив, доповнивши його нулями. Зменшити поточний розмір масиву можна тільки примусово, видаливши частину його елементів.

При кожній зміні розміру масиву MATLAB заново виділяє для нього блок пам’яті. Подивимося, як змінюється розмір масиву в процесі роботи, набравши у головному вікні MATLAB декілька команд. Спершу створимо числову змінну:

```
>> x=5
x =
    5
```

Змінна x трактується системою як масив. Переконаємося в цьому за допомогою функції size, що повертає вектор, який містить кількість рядків і стовпців масиву:

```
>> size (x)
ans =
    1    1
```

Окрім функції size, що повертає вектор розмірів масиву по всіх вимірах, є функція length, що дозволяє визначити максимальний з розмірів масиву. Особливо вона зручна при визначенні довжини одновимірних векторів.

Отже, з погляду MATLAB x - це масив 1×1. Зробимо цю змінну «справжнім» масивом, присвоївши значення, скажімо, елементу, розташованому в п'ятому стовпці четвертого рядка:

```
>> x(4,5)=6
x =
     5     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     6
```

Нумерація елементів масивів в MATLAB починається з одиниці, як це прийнято при записі матриць в математиці, а не з нуля, як, скажімо, в мовах програмування C/C++. При зверненні до елемента масиву індекси указуються через кому в круглих дужках. Перевіримо за допомогою функції size змінну x:

```
>> size (x)
ans =
     4     5
```

Тобто, змінна x стала масивом розміром 4×5.

При спробі звернутися до елемента з індексами, що виходять за поточний розмір масиву, видається повідомлення про помилку:

```
>> x(2,7)
??? Index exceeds matrix dimensions.
```

Щоб зменшити розмір масиву, потрібно видалити з нього деяку кількість рядків або стовпців. Для цього потрібно просто присвоїти відповідному фрагменту масиву «порожнє» значення []:

```
>> x (2, :) = []
x =
    5    0    0    0    0
    0    0    0    0    0
    0    0    0    0    6
```

Тут ми використовували як другий індекс символ двокрапки (:), що означає «всі елементи масиву уздовж даної розмірності». Таким чином, ми дістали доступ не до одного елемента масиву, а до цілого фрагмента. Можливість роботи з фрагментами масивів, що наявна в MATLAB, відсутня в більшості мов програмування, тому її варто розглянути докладніше. Для використання в подальших прикладах сформуємо квадратну матрицю A розміром 5×5 - «магічний квадрат»:

```
>> A = magic (5)
A =
   17   24    1    8   15
   23    5    7   14   16
    4    6   13   20   22
   10   12   19   21    3
   11   18   25    2    9
```

Спершу виділимо з цього масиву прямокутний блок, що тягнеться від другого до третього рядка і від другого до четвертого стовпця:

```
>> A (2:3, 2:4)
ans =
    5    7   14
    6   13   20
```

Таким чином можна виділити з масиву довільний прямокутний блок. Якщо блок, що виділяється, по одному з вимірів збігається з розміром початкового масиву, замість задання діапазону індексів можна вказати просто двокрапку (ми вже стикалися з цим при видаленні фрагмента масиву):

```
>> A (3:4, :)
ans =
    4     6    13    20    22
   10    12    19    21     3
```

Деякі відступи від загального синтаксису можливі і тоді, коли фрагмент, що виділяється, «упирається» в межу масиву справа або знизу. В цьому випадку для позначення останнього (максимального) можливого значення індексу за будь-якою розмірністю можна використовувати ключове слово end. Це значення можна використовувати і при обчисленні індексів. Наприклад, виділимо з масиву A блок розміром 2×3, розташований в правому нижньому куті:

```
>> A (end - 1:end, end-2:end)
ans =
   19    21     3
   25     2     9
```

Фрагмент, що виділяється, не обов'язково має бути суцільним. Виділимо з масиву A другий і четвертий елементи першого і останнього рядків:

```
>> A([1 end], [2 4])
ans =
   24     8
   18     2
```

Для виділення з масиву абсолютно довільного набору елементів доведеться вдатися до звернення за лінійним (тобто одновимірним) індексом. Елементи двовимірних масивів при цьому нумеруються по стовпцях. Виділимо з масиву A елементи, що лежать на крос-діагоналі:

```
>> A ([5 9 13 17 21])
ans =
   11    12    13    14    15
```

Для автоматичного формування одновимірних індексів з двовимірних призначена функція sub2ind. Застосуємо її для отримання одновимірних індексів елементів масиву A, що лежать на його крос-діагоналі

```
>> x = 1:5;          % номери стовпців
>> y=5:-1:1;         % номери рядків
>> ind=sub2ind(size (A), y, x)
ind =
    5    9   13   17   21
```

Отримані значення фрагментів масивів можна не лише зберігати у вигляді нових змінних, з ними можна проводити обчислення, так би мовити, «на місці». Як приклад збільшимо в два рази значення елементів першого стовпця масиву A:

```
>> A(:, 1)=A (:, 1)*2
A =
   34   24    1    8   15
   46    5    7   14   16
    8    6   13   20   22
   20   12   19   21    3
   22   18   25    2    9
```

Ми розглянули лише основні прийоми роботи з масивами в MATLAB. Є ще безліч функцій, що реалізовують перетворення масивів, матричні операції і так далі. Частина цих функцій перерахована в додатку Б, а докладніша інформація міститься в довідковій системі і документації.

Інші типи даних

Окрім одновимірних векторів і двовимірних матриць MATLAB підтримує ряд інших типів даних. До них відносяться багатовимірні масиви, рядки, структури, масиви комірок, а також об'єкти. Докладний розгляд реалізації об'єктно-орієнтованих концепцій в MATLAB виходить за рамки цього короткого опису, мінімальні відомості про роботу з

об'єктами наведені в [11] у розділі 7 при обговоренні функцій пакета Filter Design. Решта перерахованих типів даних розглядається нижче.

Багатовимірні масиви

Багатовимірні масиви є природним узагальненням двовимірних масивів. Для звернення до їх елементів використовується необхідне число індексів, проте заповнювати багатовимірний масив числами і виразами доводиться по частинах, оскільки синтаксис MATLAB дозволяє записувати поелементний вміст тільки для одно- і двовимірних масивів. Як приклад створимо тривимірний масив розміром $2\times2\times2$:

```
>> x(:,:,1) = [1 2;3 4];     %перший «шар»
>> x(:,:,2) = [5 6;7 8];      %другий «шар»
>> x
x(:,:,1) =
    1    2
    3    4
x(:,:,2) =
    5    6
    7    8
```

Виведення значень багатовимірного масиву MATLAB теж здійснює по «шарах».

Багато функцій MATLAB, орієнтованих на роботу з векторами, при отриманні вхідного параметра у вигляді двовимірного масиву обробляють його стовпці незалежно. При використанні як параметра багатовимірного масиву така обробка вестиметься уздовж першого виміру, розмір масиву уздовж якого не дорівнює одиниці. Наприклад, для створеного в наведеному прикладі тривимірного масиву були б незалежно оброблені двоелементні вектори x(:,1, 1), x(:, 2,1), x(:,1,2) і x(:,2,2). Крім того, багато функцій дозволяють явно вказати, уздовж якої розмірності слід вибирати вектори з багатовимірного масиву. Відповідний параметр в документації зазвичай позначається ідентифікатором DIM. Як приклад можна навести функції швидкого перетворення Фур'є fft і ifft, що мають такий варіант синтаксису:

y = fft(x, N, DIM) і x = ifft(y, N, DIM).

Є і функції спеціально призначені для обробки багатовимірних даних. Зазвичай ідентифікатори таких функції відрізняються від імен їх одновимірних родичів доданою в кінці буквою <<n>>. Прикладом знову-таки можуть служити функції швидкого перетворення Фур'є, у яких є багатовимірні варіанти fftn і ifftn.

Рядки

Для задання рядкових констант використовуються апострофи:

```
>> s1 = 'Digital'
>> s2 = 'Signal'
>> s3 = 'Processing'.
```

З погляду MATLAB рядками є масиви символів. Тому, наприклад, операція з'єднання (конкатенації) рядків виконується не за допомогою оператора додавання, а з використанням синтаксису горизонтального з'єднання матриць:

```
>> s4 = [s1 ' ' s2 ' ' s3]
s4 =
Digital Signal Processing
```

Основні функції роботи з рядками перераховані в розділі «Strfun» (додаток Б).

Структури

Структури (structure) дозволяють об'єднати в одній змінній різнорідні дані (поля - field) і здійснювати доступ до них за іменами. У мові С відповідний тип даних теж називається структурою, а в мові Pascal - записом (record).

У MATLAB для створення структури не потребується спеціальних оголошень - досить присвоїти значення будь-якому полю. Імена полів

указуються після імені змінної через крапку. Наприклад, для зберігання інформації, що витягується з wav-файла, можна використовувати таку структуру:

```
>> w.name = 'g:\windows\media\tada.wav'; % ім'я файла
>> w.Fs = 22050; % частота дискретизації
>> w.bits = 16; % число бітів на відлік
>> w.channels =2; % число каналів
>> w.samples = 42752; % число відліків
>> w.data = wavread(w.name); % двовимірний масив відліків
```

MATLAB показує значення змінних-структур таким чином:

```
>> w
w =
        name: 'g:\windows\media\tada.wav'
        Fs: 22050
        bits: 16
        channels: 2
        samples: 42752
        data: [42752x2 double]
```

Значення полів, що мають короткі подання, виводяться повністю, а для великих масивів указуються тільки розміри і тип даних.

Можна створювати і масиви структур, для цього, як завжди, використовуються індекси. Перетворимо змінну w на двоелементний масив структур:

```
> w(2).name = 'g:\windows\media\chord.wav'
w =
        1x2 struct array with fields:
        name
        Fs
        bits
        channels
        samples
        data
```

Для масивів структур MATLAB показує тільки розміри масиву і список полів.

Масиви комірок

Масиви комірок (cell array), так само як і структури, дозволяють об'єднати в одній змінній різнорідні дані. Проте звернення до цих даних проводиться не за іменами полів, а за числовими індексами. Щоб відрізнити масив комірок від звичайного масиву, його індекси записуються не в круглих дужках, а у фігурних. Як приклад подамо ту ж інформацію з wav-файла, яку ми використовували при розгляді структур, у вигляді масиву комірок:

```
>> w{1}= 'g:\windows\media\tada.wav'; % ім'я файла
>> w{2}= 22050; % частота дискретизації
>> w{3}=16; % число бітів на відлік
>> w{4}=2; % число каналів
>> w{5}= 42752; % число відліків
>> w{6}= wavread(w{1}); % двовимірний масив відліків
```

Наведений приклад не має особливого практичного сенсу - в даному випадку значно зручніше був би доступ за іменами полів. Проте масиви комірок зручні в тих випадках, коли потрібно створити масив з векторів різної довжини або матриць різного розміру. Прикладом може служити зберігання в масиві комірок інформації про фільтр, складений з послідовно включених секцій різного порядку.

MATLAB показує вміст масивів комірок таким чином:

```
>> w
w =
  Columns 1 through 5
    [1x25 char]    [22050]    [16]    [2]    [42752]
  Column 6
    [42752x2 double]
```

Як і в разі структур, значення елементів, що мають короткі подання, виводяться повністю, а для великих масивів указуються тільки розміри і

тип даних. В даному випадку «дуже довгим» для виведення виявився і текстовий рядок.

Програмування

Інтерактивна робота з MATLAB в режимі командного рядка дуже зручна при освоєнні системи і при необхідності поекспериментувати, щоб випробувати різні підходи до аналізу і обробки наявних даних. Проте із досвідом, а також при рішенні конкретної задачі ви відмітите, що дуже часто набираєте одні і ті ж послідовності команд. Це означає, що настав час перетворити такі послідовності на програми (сценарії) і функції MATLAB.

Цими програмами і функціями є текстові (ASCII) файли з розширенням .m, у яких записані команди і оператори MATLAB. В принципі такі файли можуть створюватися і модифікуватися за допомогою будь-якого ASCII-редактора, але, починаючи з версії 5, в MATLAB з'явився свій редактор, що забезпечує, зокрема, колірне виділення синтаксичних елементів мови, а також містить інтегрований налагоджувач.

Програми і функції

Текстові файли, що містять оператори MATLAB, можуть бути двох типів: програми (script; інколи використовується дослівний переклад - «сценарій») і функції (function). Між ними є такі відмінності:

- функції мають заголовок function, а програми - заголовок script або зовсім не мають заголовка;

- функції можуть приймати вхідні параметри і повертати результати обчислень, а програми - ні;

- програми використовують робочу область пам'яті середовища MATLAB, а кожна функція при виклику створює свою власну робочу область пам'яті і спілкується із зовнішнім світом тільки через параметри і глобальні змінні (див. далі);

- функції можуть викликатися з програм і інших функцій, а програму можна запустити тільки вручну, набравши її ім'я в командному рядку MATLAB.

Отже, як вже було сказано, програмами і функціями MATLAB є текстові файли із записом команд, операторів і викликів функцій MATLAB. У цих файлах можуть міститися будь-які команди MATLAB, зокрема ті, які зазвичай не застосовуються в командному рядку (оператори циклів, умовні оператори і т. п.; використання цих засобів в командному рядку можливе, але приводить до необхідності набирати багато тексту, так що зручніше виявляється зберігати такі фрагменти у вигляді файлів програм).

У одному рядку можна записувати декілька операторів. Для їх розділення слід використовувати крапку з комою (якщо необхідно подавити виведення результатів обчислень в командне вікно) або кому (якщо це виведення потрібно дозволити).

Довгі рядки для поліпшення читання програми можна ділити на частини. Ознакою того, що запис оператора продовжується на наступному рядку, є три крапки в кінці рядка. Зручно використовувати цю можливість, наприклад, для наочнішого запису поелементно заданої матриці:

```
M = [a b c d; ...
     b b c d; ...
     c c c d; ...
     d d d d];
```

Хороша програма неможлива без коментарів. Для позначення коментарів в MATLAB використовується знак % - за коментар вважається частина рядка праворуч від нього. Засоби створення багаторядкових коментарів в MATLAB не передбачені. При необхідності закоментувати великий фрагмент тексту або, навпаки, зняти з нього знаки коментаря можна скористатися відповідно командами Comment і Uncomment меню Text редактора-налагоджувача (див. далі).

Редактор-налагоджувач М-файлів

Для запуску редактора-налагоджувача М-файлів необхідно створити новий М-файл або відкрити існуючий. Для цього використовуються команди Новий (New) > М-файл (M-file) і Відкрити (Open) меню Файл

(File) головного вікна MATLAB або відповідні кнопки панелі інструментів. Вікно редактора-налагоджувача показане на рис. А. 4.

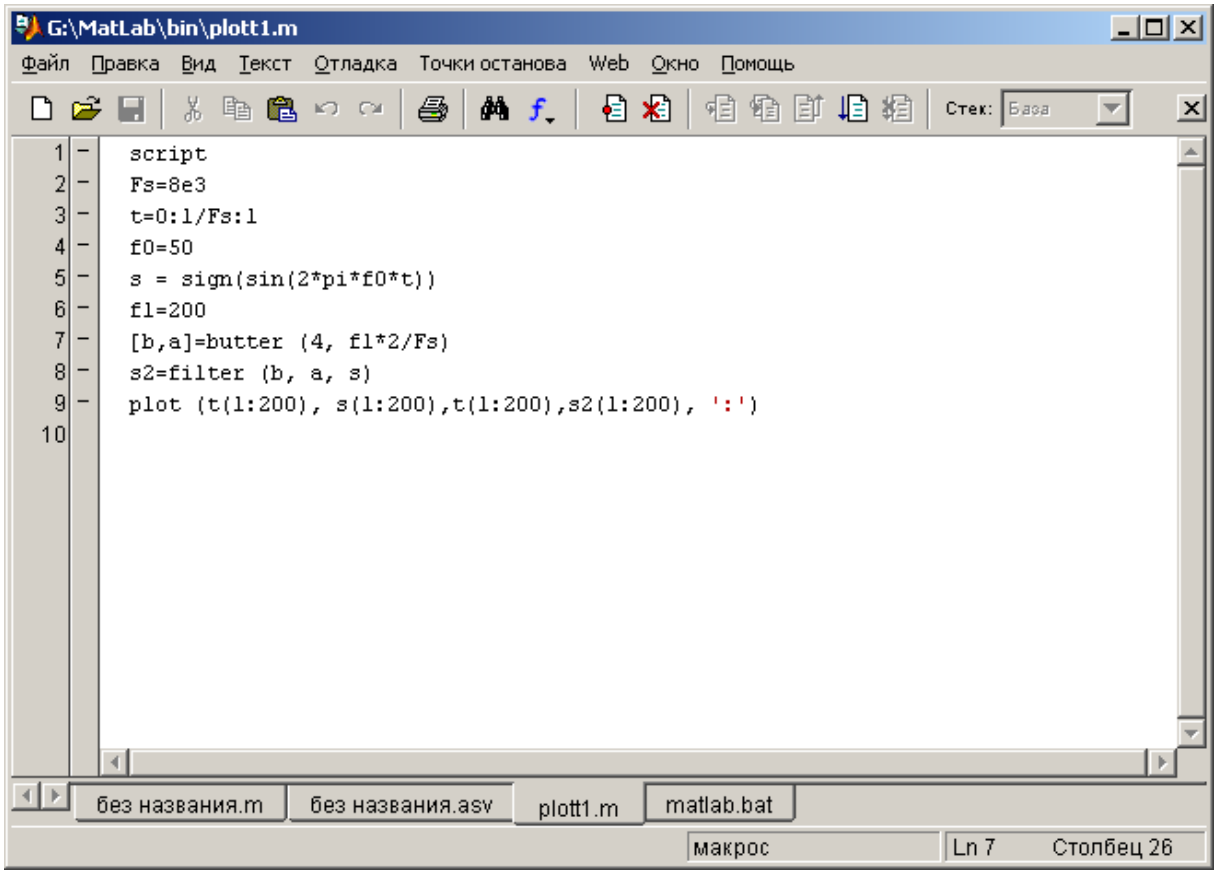


Рисунок А. 4 – Редактор-налагоджувач М-файлів

Функції редагування і роботи з файлами, що забезпечуються редактором-налагоджувачем М-файлів, є типовими для текстових редакторів. Із специфічних можливостей слід зазначити кольорове виділення синтаксичних елементів мови програмування MATLAB, а також команди меню *Текст (Text)*, що дозволяють закоментувати виділений фрагмент тексту (Comment), прибрати знаки коментарю (Uncomment), перевірити відповідність правих і лівих дужок (Balance Delimiters), а також автоматично відформатувати лівий край виділеного фрагмента тексту (Smart Indent).

Права частина панелі інструментів містить кнопки управління налагоджувачем.

Створення функцій

Текст М-файла функції повинен починатися із заголовка function, що має такий вигляд:

function [yl. y2. ...] = fname(xl. x2. ...)

Тут fname - ім'я функції, xl, x2 і так далі - вхідні параметри, yl, y2 і так далі - вихідні параметри. Вхідні або вихідні параметри можуть бути відсутніми. Насправді ім'я функції визначається не рядком fname, а ім'ям, під яким збережений М-файл. З цієї ж причини імена функцій, на відміну від імен змінних, не чутливі до регістра символів.

Як приклад створимо функцію closest, яка прийматиме два вхідні параметри, вектор і число, і вибиратиме з вектора значення, максимально близьке до другого (числового) параметра. Для цього, відкривши вікно редактора/налагоджувача командою меню File > New > M-file, введіть такий текст:

```
function y = closest(A. x)
% Вибір з масиву А елемента, найближчого до х
[tmp, ind] = min(abs(A - x));
y = A(ind);
```

Ввівши текст, збережіть файл під ім'ям closest.m.

Тепер коротко прокоментуємо цей програмний код. Перший рядок - це заголовок функції, складений згідно з наведеним раніше шаблоном. Другий рядок, що є коментарем, - це вбудована довідка про використання нашої функції. За командою help fname в командне вікно виводяться рядки коментарю, розташовані у файлі fname.m безпосередньо після заголовка функції. Перевіримо це:

```
> help closest
Вібір з масиву А елемента, найближчого до х.
```

Таким чином, створювати вбудовану довідку для власних функцій зовсім нескладно.

Далі за допомогою функції min визначається номер елемента, що цікавить нас, і в останньому рядку вихідному параметру *y* присвоюється значення цього елемента.

Змінні ind і tmp, створені усередині функції, є локальними. Вони створюються всередині функції і знищуються при завершенні її роботи. Якщо змінні з такими ж іменами існували в робочій області пам'яті MATLAB до виклику функції, їх значення не зміняться.

Функція спілкується із зовнішнім світом тільки через свої параметри і глобальні змінні. Щоб оголосити змінну x глобальною, необхідно використовувати в головній програмі і у всіх функціях, які повинні мати доступ до даної змінної, таку директиву:

global x

Повернення з функції проводиться після досягнення кінця її файла. Для дострокового повернення можна використовувати оператора return. У будь-якому випадку результатами роботи функції є значення, які вихідні параметри мали на момент повернення.

Шлях пошуку

Для того, щоб програма або функція була доступна із середовища MATLAB, система має бути здатна знайти відповідний М-файл. Пошук файлів здійснюється таким чином: спочатку є видимим поточний робочий каталог (його ім'я показане на панелі інструментів головного вікна на рис. А. 2), а потім каталоги, що входять в шлях пошуку (MATLAB search path). Коли користувач починає створювати власні М-файли, постає питання про те, де їх доцільно розміщувати. Можна дати таку рекомендацію: файли, що відносяться до різних проектів, слід розміщувати в окремих каталогах, не додаючи ці каталоги в шлях пошуку. Поточний каталог (Current Directory) панелі інструментів головного вікна забезпечує зручність переміщення між каталогами різних проектів. Якщо ж в процесі роботи у вас починає формуватися власна бібліотека функцій, що використовуються в декількох проектах, має сенс зібрати їх в один каталог і додати його ім'я в шлях пошуку (для виклику редактора шляху пошуку використовується команда Шлях (Set Path) меню Файл (File) головного вікна MATLAB). Тоді ці функції будуть доступні у будь-який момент, незалежно від того, який каталог є поточним.

Логічні умови

Як логічні умови в умовних операторах і циклах while (див. далі) можуть використовуватися числові скалярні значення. При цьому нульове значення трактується як «хибність» (false), а будь-яке ненульове - як «істина» (true).

Крім того, для формування умов часто використовуються оператори порівняння = = (рівно), -= (не рівно), < (менше), > (більше), <= (менше або рівно), >= (більше або рівно). Слід мати на увазі, що для масивів вони проводять поелементне порівняння, повертаючи масив такого ж розміру, як порівнювані аргументи. Результуючий масив містить одиниці там, де порівняння елементів дало виконання умови, і нулі там, де умова порівняння виконано не було.

Для формування логічних умов корисні також такі функції:

- all(x) - повертає 1, якщо всі елементи x відмінні від нуля;
- any(x) - повертає 1, якщо хоч би один елемент x відмінний від нуля;
- isequal(x, y) - повертає 1, якщо значення x і y збігаються. На відміну від конструкції all(x==y), використання даної функції не приведе до помилки, якщо розміри x і y не збігаються;
- isempty(x) — повертає 1, якщо x є порожньою матрицею (тобто має розмір 0×0).

Є ще досить багато функцій перевірки різноманітних умов, імена яких мають вид is.... За докладнішою інформацією потрібно звернутись до довідкової системи.

Умовний оператор

Умовний оператор в MATLAB реалізовується за допомогою ключових слів if, else і end:

```
if cond
% гілка, що виконується, якщо cond істинно
else
% гілка, що виконується, якщо cond помилково
End
```

Для оператора if, вкладеного в гілку else іншого (зовнішнього) оператора if, є скорочена форма запису з використанням ключового слова elseif:

```
if cond1
% гілка, що виконується, якщо cond1 істинно
elseif cond2
% гілка, що виконується, якщо cond1 помилково, а cond2 істинно
else
% гілка, що виконується, якщо condl і cond2 помилкові
end
```

Аналогічним чином можна використовувати elseif кілька разів.

Оператор вибору

Якщо в умовному операторові вибирається одна з двох можливих альтернатив, то оператор вибору реалізовує розгалуження програми на декілька шляхів. Вибір конкретного шляху залежить від значення заданої змінної. Запис оператора вибору в MATLAB проводиться за допомогою ключових слів switch, case, otherwise і end:

```
switch x
case x1
% гілка, що виконується, якщо x рівне x1
case x2
% гілка, що виконується, якщо x рівне x2
otherwise
% гілка, що виконується, якщо всі умови case не дотримуються
end
```

При виконання оператора switch значення x по черзі порівнюється з x1, x2 і так далі При виявленні першого збігу виконуються оператори відповідної гілки, після чого проводиться вихід з оператора вибору. На відміну від аналогічного оператора мови C оператори break в кінці кожної гілки в MATLAB не потрібні.

Цикли

Для реалізації циклів в MATLAB є два оператори — for і while. У операторові for змінна-лічильник циклу по черзі набуває значень елементів деякого вектора:

```
for k = x
% тіло циклу
end
```

Оператори, що входять в тіло циклу, виконуватимуться при значенні змінної k, рівному x (1), потім x(2) і так далі до x(end). Після завершення циклу значення k залишається рівним x(end).

Зрозуміло, найчастіше в циклі for використовується послідовний перебір цілочислових значень лічильника циклу:

```
for k = 1:N
```

Проте при необхідності можна використовувати дробове значення кроку

```
for k = -1:0.01:1
або перебір довільних значень:
for k = [1 10 pi 9 ln(3)]
```

Вивчаючи програмування, люди, як правило, набувають звички використовувати як лічильники циклів змінні з іменами i, j. Однак в MATLAB ці змінні мають значення уявної одиниці, тому одним з варіантів є використання двобуквенних імен лічильників: ii, jj і так далі.

Другий тип циклів реалізовується за допомогою оператора while. Тіло циклу виконується, поки умова cond залишається істинною (тобто значення cond відмінне від нуля):

```
while cond
% тіло циклу
end
```

Якщо cond спочатку має нульове значення, тіло циклу не буде виконано жодного разу.

Щоб реалізувати дострокове завершення циклу, використовується оператор break. Зрозуміло, він повинен застосовуватися у поєднанні з умовним оператором, наприклад так:

```
for k = 1:N
% перша половина тіла циклу
% перевірка додаткової умови завершення
if cond
break
end
% друга половина тіла циклу
…
end
```

Оператора break можна використовувати для реалізації циклів з перевіркою умови завершення не на початку (як робить оператора while), а в довільному місці циклу. Для цього за допомогою оператора while створюється «вічний» цикл, а його завершення проводиться оператором break:

```
while 1 % «вічний» цикл
% перша половина тіла циклу
…
% перевірка умови завершення
if cond
break
end
% друга половина тіла циклу
…
end
```

Окрім оператора break для управління виконанням програми усередині циклу є оператор continue. Розміщений усередині циклу for або while цей оператор передає управління на перевірку умови завершення

циклу, пропускаючи фрагмент тіла циклу, який при цьому залишився. Це дозволяє зробити код наочнішим, уникнувши використання довгого оператора if.

Функції зі змінним числом параметрів

При виклику функції можна використовувати меншу кількість вхідних і вихідних параметрів, ніж вказано в заголовку функції. При цьому розраховані функцією «зайві» вихідні параметри просто ігноруються. Що стосується не вказаних вхідних параметрів, то функція видасть повідомлення про помилку, коли спробує звернутися до одного з них. Проте усередині функції доступні відомості про використане при виклику число вхідних і вихідних параметрів, що дозволяє залежно від цього управляти алгоритмом роботи функції.

Число переданих вхідних параметрів можна дізнатися за допомогою функції nargin, а число очікуваних вихідних параметрів - за допомогою функції nargout. У простому випадку функція nargin може використовуватися для задання вхідним параметрам значень за замовчуванням:

```
function y = nargin_demo(x, N, mu, fid)
if nargin < 4
fid = 1;
end
if nargin < 3
mu = 0.2;
end
if nargin < 2
N = 256;
end
```

При виклику такої функції можна опускати один, два або три останні вхідні параметри, аналогічно тому, як це дозволяють робити мова C++ і останні версії мови Object Pascal:

```
y = nargin_demo(x, N, mu, fid)% повний варіант
```

y = nargin_demo(x, N, mu)
y = nargin_demo(x, N)
y = nargin_demo(x)

Замість опущених параметрів будуть використані задані в тексті функції значення за замовчуванням.

Аналогічним чином можна використовувати і функцію nargout - наприклад, щоб заощадити час, не обчислюючи не потрібні вихідні величини. Можна використовувати функції nargin і nargout і для серйознішого управління логікою роботи - скажімо, змінювати використовуваний алгоритм залежно від числа вхідних і вихідних параметрів.

Введення і виведення даних

Простим способом виведення значення змінної в командне вікно MATLAB є вказівка її імені без крапки з комою в кінці. Аналогічну дію виконує функція disp(name), відмінність полягає тільки в тому, що на екран не виводитиметься ідентифікатор змінної name.

Форматоване виведення

Щоб виводити в командне вікно інформацію в акуратнішому вигляді, слід скористатися функціями форматованого виведення. Це функції sprintf (формування рядкової змінної з форматованим текстом) і fprintf (виведення форматованого тексту в потік виведення). Синтаксис обох функцій дуже схожий і практично не відрізняється від синтаксису відповідних функцій мови С. Коротко розглянемо лише функцію fprintf:

fprintf (fid, 'format', xl, x2....)

Тут fid - числовий ідентифікатор потоку виведення. При нульовому значенні виведення не проводиться, значення 1 відповідає стандартному потоку виведення (виведення в командне вікно), значення 2 - стандартному потоку виведення повідомлень про помилки (за замовчуванням це теж виведення в командне вікно). Для запису інформації у файл як fid

необхідно використовувати значення, отримане від функції fopen при відкритті файла (робота з файловими потоками в даному додатку не розглядається). Даний параметр можна опустити, тоді за замовчуванням виведення проводитиметься в командне вікно.

Рядковий параметр 'format' задає формат виведення. Цей рядок повинен містити фіксований текст, що виводиться, і специфікатори формату, на місце яких будуть підставлені значення змінних xl, x2 і так далі. Специфікатори формату починаються зі знаку %, за яким ідуть цифри, що задають число символів для виведення, і буква, що визначає тип формата. Повна інформація про можливі специфікатори формата є в HTML-довідці, тут же наведемо лише декілька конкретних прикладів:

%d - виведення цілого числа, кількість позицій вибирається автоматично;

%.3f - виведення числа у форматі з фіксованою комою, після коми виводиться три десяткові знаки, кількість позицій для цілої частини вибирається автоматично;

%-15.3e - виведення числа в експоненціальному форматі, мантиса має три десяткові знаки після коми, поле виведення займає 15 символів, текст вирівнюється по лівому краю поля.

Для виведення спеціальних символів можна використовувати комбінації:

\n - переведення рядка;
\r - повернення каретки;
\t - табуляція;
\b - забивання;
\\ - зворотна похила риска;
%% - символ відсотка;
‘’ - одиничний апостроф.

Тепер використовуємо перераховані специфікатори і деякі з символів:

```
>> n = 15;
>> err=128.3567;
>> value = 1234567890;
```

```
>> fprintf (1, 'Крок %d, помилка %.3f \nx"= %-15.3e m', n, err, value)
Крок 15, помилка 128.357
x'= 1.235e+009      m
```

Збереження і завантаження значень змінних

MATLAB дозволяє зберігати змінні у вигляді MAT-файлів (вони мають розширення .mat), які потім можуть бути знову завантажені. Для збереження всього вмісту робочої області пам'яті використовується команда

```
>> save filename
```

Всі змінні, що існують в даний момент, будуть записані в розташований в поточному каталозі файл filename.mat.

Щоб зберегти значення тільки деяких змінних, слід вказати їх імена після імені файла:

```
>> save filename x y asd ...
```

У файл filename.mat будуть записані тільки значення змінних x, y, asd і так далі

Завантаження збережених змінних проводиться командою load:

```
>> load filename
```

Файл filename.mat має бути доступний, тобто він повинен знаходитися в поточному каталозі або в одному з каталогів, внесених до списку для пошуку файлів (див. раніше розділ «Шлях пошуку»).

Робота з налагоджувачем

Налагоджувальні засоби зібрані в правій частині панелі інструментів вікна редактора/налагоджувача (рис. А. 4), а також в *меню Отладка (Debug)* і *Точки останова (Breakpoints).*

Кнопки панелі інструментів мають таке призначення:

- *Установить/очистить точку останова (Set/clear breakpoint)* - установлення або скидання точки переривання в поточному рядку;

- *Очистить все точки останова (Clear all breakpoints)* - скидання всіх точок переривання;

- *Шаг (Step)* - виконання одного рядка програми, при цьому виклик функції вважається як один оператор і заходження всередину функції, що викликається, не проводиться;

- *Шаг в (Step in)* - виконання одного рядка програми із заходженням всередину функцій, що викликаються;

- *Шаг вне (Step out)* - виконання частини поточної функції, що залишилася, зупинення програми проводиться після повернення з функції;

- *Запуск (Continue)* - продовження виконання програми;

- *Выход из режима отладки (Exit Debug Mode)* - завершення відлагодження;

- *Стек (Stack)* - список, що розкривається, дозволяє перемикатися між робочими просторами MATLAB і викликаними функціями.

При зупиненні програми в налагоджувальному режимі (у встановленій точці переривання або при покроковому виконанні) в головному вікні з'являється налагоджувальна підказка K>>.

При цьому стають доступними робочі області пам'яті всіх викликаних в даний момент функцій. За замовчуванням поточною є робоча область пам’яті останньої викликаної функції. Перемикатися між робочими областями пам’яті всього стека функцій можна за допомогою списку Stack панелі інструментів, що розкривається (див. рис. А. 4). Такий же список є в панелі інструментів вкладки Workspace (див. рис. А. 3). Область пам’яті середовища MATLAB фігурує в списку під ім'ям Base.

Крім того дізнатися значення змінної можна, затримавши покажчик миші на її ідентифікаторі у вікні редактора-налагоджувача. З'явиться "спливаюча" підказка, що відображує значення змінної. Цей спосіб має сенс застосовувати тільки для скалярних змінних і масивів невеликого розміру, оскільки для великих масивів відображується тільки частина вмісту або тільки розмір і тип даних.

Оптимізація MATLAB-програм

Під оптимізацією в даному випадку маємо на увазі підвищення швидкості роботи функцій і програм. У MATLAB застосовні всі або майже всі загальні прийоми оптимізації програм (типу винесення з циклу назовні

всіх обчислень, не залежних від лічильника циклу). Проте у MATLAB є і деяка специфіка, яку слід враховувати при оптимізації коду. При необхідності підвищити швидкодію MATLAB-програми слід враховувати два аспекти:

- MATLAB - це інтерпретувальна, а не компілююча система;
- функції векторно-матричних операцій вбудовані в ядро системи і виконуються гранично швидко.

Найзагальніша рекомендація, яка дається практично в будь-якій книзі, присвяченій MATLAB, - прагнути уникати використання циклів, замінюючи їх векторно-матричними операціями і поелементними операціями над масивами. Наведемо простий приклад, обчисливши 10000 значень функції sin двома способами - за допомогою циклу і шляхом виконання поелементної операції над вектором. Хронометраж вестимемо за допомогою функції tic і toc. Перша з них фіксує початок відліку часу, а друга виводить на екран час (у секундах), що пройшов після tic. Ось відповідний програмний код:

```
>> % обчислення в циклі
>> % знищуємо всі змінні
>> clear all
>> tic
>> for k = 1:10000  x(k) = sin(k); end
>> toc
elapsed_time =
3.7900

>> % векторна операція
>> % знищуємо всі змінні
>> clear all
>> tic
>> x = sin (1:10000);
>> toc
elapsed_time =
0.0500
```

Відмова від використання циклу прискорила обчислення приблизно в 75 разів.

Час виконання фрагмента програми може залежати від того, існують використовувані змінні або їх доводиться створювати (див. далі обговорення питань, пов'язаних з виділенням пам'яті). Тому для підвищення чистоти експерименту в наведеному лістингу перед початком розрахунків всі наявні змінні знищуються командою clear all. Цю команду можна також використовувати для вибіркового знищення змінних - в цьому випадку замість слова all необхідно вказати імена змінних, що знищуються, розділяючи їх пропусками: clear x y z....

Проте не слід сприймати дану пораду як абсолютну догму. Не всі алгоритми добре піддаються векторизації, і інколи спроба штучно звести алгоритм до послідовності матричних операцій може потребувати створення проміжних матриць великого розміру. Необхідність виділення пам’яті для зберігання цих матриць може звести нанівець переваги відмови від циклів.

Отже, без циклів все-таки обійтися не вдається, але при необхідності їх використання слід звернути увагу ще на один аспект - виділення пам'яті для зберігання змінних. Мова програмування MATLAB, будучи мовою надвисокого рівня, що інтерпретується, приховує від користувача операції, пов'язані з виділенням і звільненням пам'яті. В результаті ми найчастіше навіть не замислюємося над тим, що насправді відбувається при виконанні простого (з погляду користувача) оператора привсвоєння. Розглянемо невеликий приклад:

```
for k = 1:N
% які-небудь обчислення
x(k) = ...
end
```

Якщо змінна x не існувала до початку циклу, то при кожному проході циклу відбувається збільшення числа елементів в ній. В результаті при кожному виконанні оператора присвоєння система повинна виконати таке:

- виділити пам'ять під змінну x нового розміру;
- скопіювати туди старий вміст змінної;

- звільнити пам'ять, відведену раніше під зберігання старого значення x;

- нарешті виконати власне оператор присвоєння для x(k).

Операції виділення і звільнення пам'яті виконуються досить повільно, тому необхідність пробляти їх при кожному проході циклу може сильно уповільнити роботу програми, особливо якщо нарощувані масиви мають великий розмір і якщо в циклі їх використовується декілька.

Як приклад, в циклі заповнюватимемо два вектори випадковими гаусовими числами, виводячи на екран час, який був потрібен для додавання кожних 1000 елементів:

```
>> clear all  % знищуємо всі змінні
        tic
        for k = 1:1e4
        x(k) = randn;
        y(k) = randn;
        if rem(k, 1e3)==0
        t=toc;
        fprintf (1, '%d:  %.3f  seconds\n', k, t)
        tic
        end
end
1000:  0.016  seconds
2000:  0.031  seconds
3000:  0.031  seconds
4000:  0.250  seconds
5000:  0.156  seconds
6000:  0.235  seconds
7000:  0.297  seconds
8000:  0.172  seconds
9000:  0.359  seconds
10000:  0.406  seconds
```

Як бачите, із зростанням розміру масивів швидкість роботи дуже падає - для додавання останньої тисячі елементів знадобилося в 25 разів більше часу, чим для першої. Вирішенням цієї проблеми є попереднє виділення пам'яті під заповнювані в циклі масиви. Для цього необхідно створити змінну потрібного розміру, скажімо, за допомогою функції zeros, що формує масив, заповнений нулями. Модифікуємо попередній приклад:

```
>> clear all % знищуємо всі змінні
        x = zeros(1, 1e4); % попередньо
        y = zeros(1, 1e4); % створюємо масиви
        tic
        for k = 1:1e4
        x(k) = randn;
        y(k) = randn;
        if rem(k, 1e3)==0
        t=toc;
        fprintf (1, '%d:  %.3f  seconds\n', k, t)
        tic
        end
end
1000:  0.016  seconds
2000:  0.016  seconds
3000:  0.000  seconds
4000:  0.015  seconds
5000:  0.000  seconds
6000:  0.016  seconds
7000:  0.000  seconds
8000:  0.015  seconds
9000:  0.000  seconds
10000:  0.000  seconds
```

Тобто, робота циклу проходить рівномірно, не уповільнюючись. Нульові значення, кілька разів отримані при виконанні останнього прикладу, пояснюються тим, що функція toc вимірює час дискретними порціями, розмір яких лежить між 50 і 60 мс.

При використанні оператора while число проходів циклу заздалегідь не відоме. Тому не відомий і підсумковий розмір нарощуваних в такому циклі масивів. У цій ситуації можна застосувати компромісний підхід, збільшуючи масиви усередині циклу, але не при кожному проході, а зрідка, зате великими кусками. Виходить код на зразок такого:

```
sizejncr = 10000; % дискретність приросту
x = zeros (1, sizejncr); % створення масивів
y = zeros(1, sizejncr);
k = 1; % лічильник елементів
while cond
% перевіряєм, чи не вичерпана поточна довжина масивів
if k>length(x)
% виділяємо додаткову пам’ять
x = [x zeros(1, sizejncr)];
y = [y zeros(1, sizejncr)];
end
% будь-які обчислення
x(k) = ...
y(k) = ...
k = k+1; % збільшуємо значення лічильника
end
% після завершення циклу видаляємо нульові «хвости»
x(k; end) = []:
y(k; end) = [];
```

Даний підхід дозволяє істотно прискорити обчислення, якщо число проходів циклу заздалегідь не відоме.

До цих пір ми вимірювали час виконання фрагментів програми за допомогою функцій tic і toc. Вони достатньо корисні, але в MATLAB є і серйозніший хронометруючий інструмент - утиліта профілізації, що викликається командою profile on. Як приклад, створимо декілька варіантів функції ind_diff, що обчислює квадратну матрицю, значення елементів якої дорівнюють різниці їх індексів: Aij = i - j. Єдиним вхідним параметром є розмір N створюваної матриці. Перша версія коду буде найбільш прямолінійною - з використанням двох вкладених циклів:

```
function y = ind_diff(N)
for k = 1:N
        for l = 1:N
                y(k,l) = k - l;
        end
end
```

Рисунок А. 5 - Підсумковий звіт про профілізацію

Зберігаємо текст функції у вигляді файла ind_diff.m, включаємо профілізацію і викликаємо функцію:

```
>> profile on
>> y = ind_diff(100);
```

Для виведення результатів профілізації використовується така команда:

```
>> profile report
```

MATLAB згенерує звіт в HTML-форматі і покаже його у встановленому в системі web-браузері (рис. А. 5). У підсумковому звіті показані загальний час профілізації, число викликаних функцій і час, витрачений на їх виконання. В даному випадку окрім самої функції profile викликалася тільки наша функція ind_diff. Щоб отримати докладну інформацію про результати профілізації цієї функції, необхідно клацнути на її імені в таблиці. Після переходу за гіперпосиланням відкриється сторінка деталізованої інформації (рис. А. 6). Найбільший інтерес тут викликають виведені під таблицею окремі рядки лістингу функції, поряд з якими показаний абсолютний (у секундах) і відносний (у відсотках) час їх виконання.

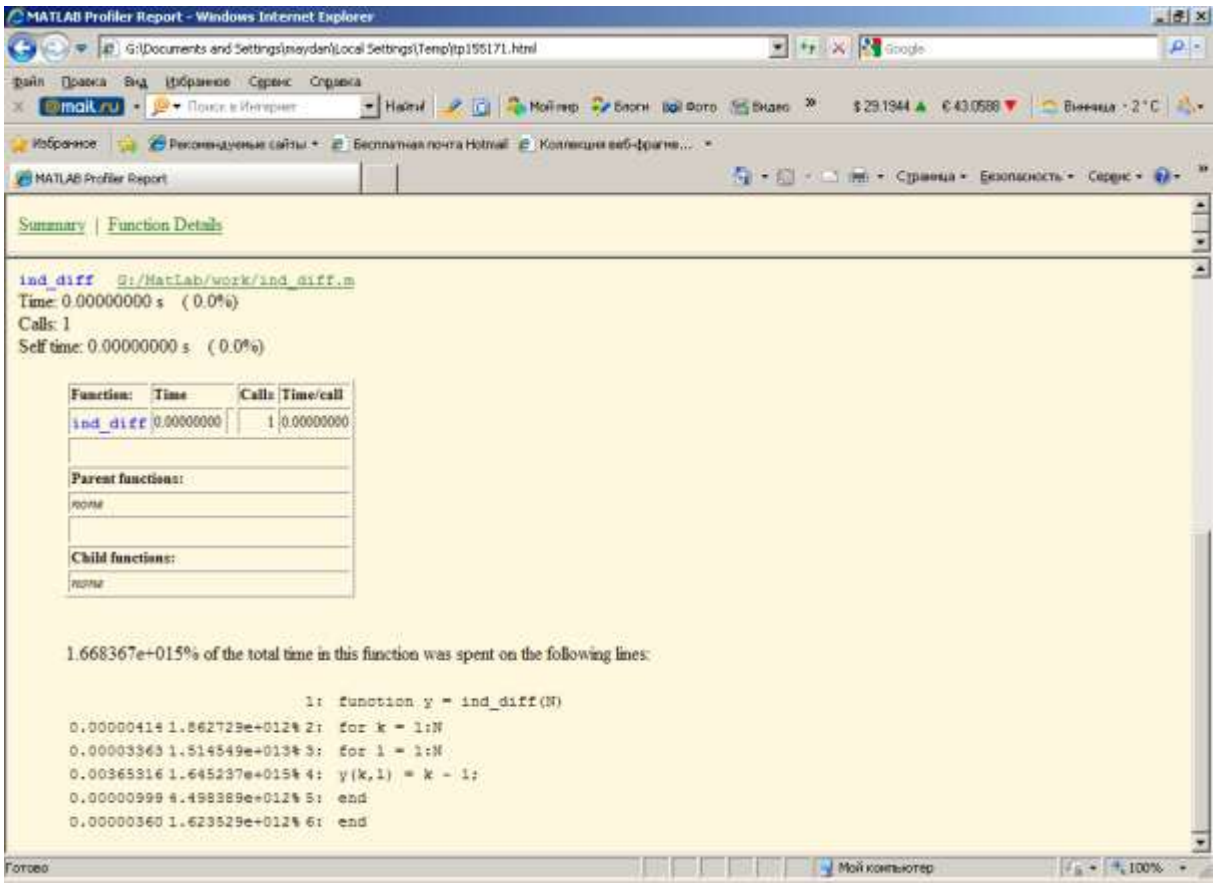


Рисунок А. 6 - Звіт про результати профілізації - детальна інформація про функції, що викликаються

Проаналізуємо отримані результати. Якийсь час витрачено на обслуговування циклів (рядки 5 і 6), але основні витрати (71 %) припали на

оператор присвоювання (рядок 4). Виділимо заздалегідь пам'ять під створюваний масив:

```
function y = ind_diff1(N)
        y = zeros(N, N);
        for k = 1:N
                for l = 1:N
                y(k,l) = k - l;
        end
end
```

Зберігаємо нову версію і знову включаємо профілізацію:

```
>> profile on
>> y = ind_diff1(300);
>> profile report
```

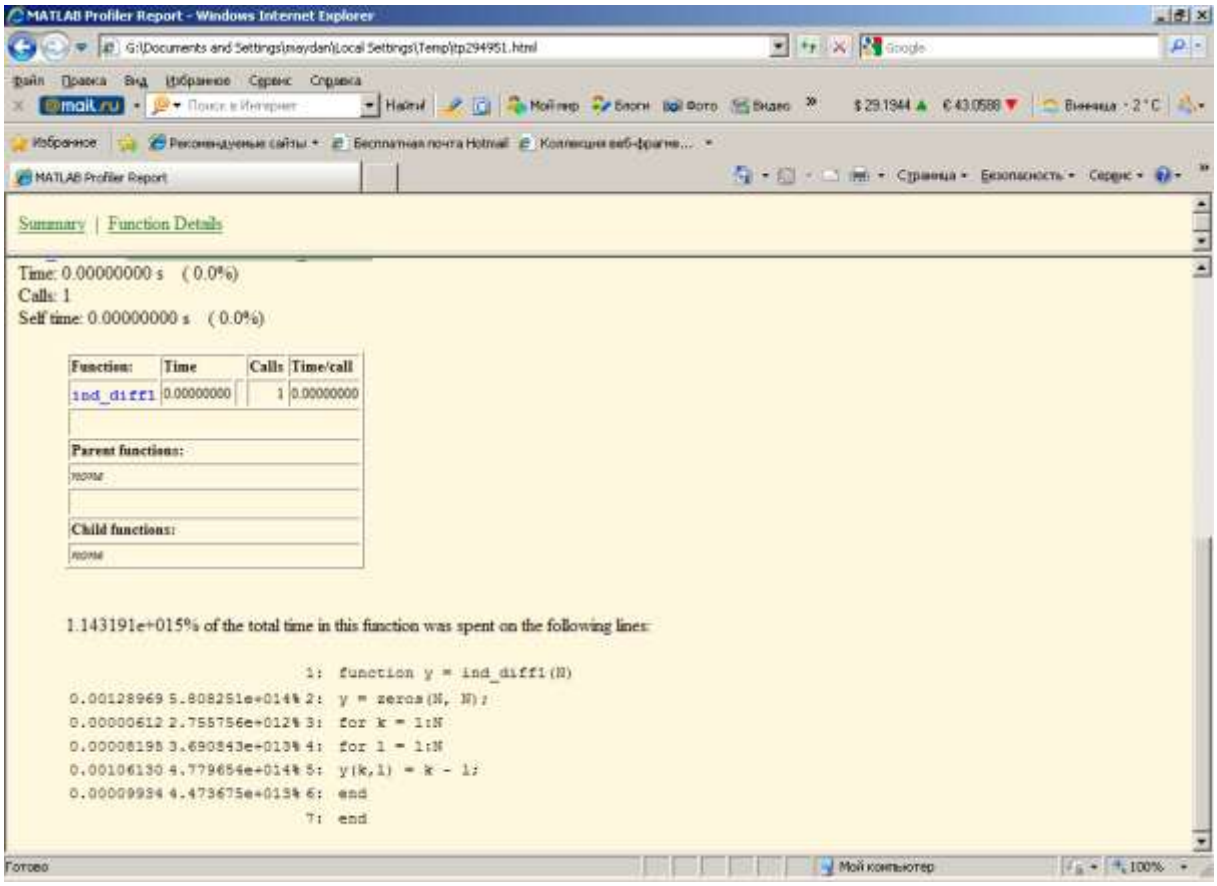


Рисунок А. 7 - Результати профілізації оптимізованого коду

Детальний звіт показано на рис. А. 7. Як видно, час, що витрачається на виконання присвоювання усередині циклу, істотно зменшився. Завершуємо роботу з утилітою профілізації: >> profile off

Час виконання даної функції можна ще більше зменшити, відмовившись від циклів зовсім. Якщо вам дійсно потрібно створити таку матрицю, слід зауважити, що уздовж її діагоналей, паралельних головній, стоять однакові числа, і скористатися функцією toeplitz:

y = toeplitz (0:N-l, -(0:N-1);

Це, мабуть, найшвидший спосіб.

Графіка

Говорячи про графічні засоби MATLAB, перш за все потрібно відзначити таке. MATLAB - матрична програма, і її графічні команди можуть лише різноманітними способами візуалізувати вектори і матриці. На практиці це означає, що точки, відповідні елементам векторів і матриць, можуть з'єднуватися лише прямими лініями - ніякого згладжування або інтерполяції проводитися не буде. Якщо ви хочете, щоб графік виглядав плавнішим, поклопочіться про те, щоб в масиві, який візуалізується, було більше точок. Якщо будується графік функції, уменшіть крок між сусідніми значеннями її аргументу. Якщо необхідно згладити експериментально отримані дані, для яких отримати додаткові точки важко, слід скористатися функціями інтерполяції (interp*, spline, pchip, griddata*) або апроксимації (polyfit; крім того, є графічний інтерфейс апроксимації, що викликається командою Basic Fitting з меню Tools графічних вікон). За докладнішими відомостями необхідно звернутись до документації MATLAB.

Двовимірна графіка

Основним засобом двовимірної графіки є функція plot. Найбільш типовий варіант її використання виглядає таким чином: plot (x, y). Тут x і y - вектори однакової довжини, які задають координати точок, що виводяться на графік. За замовчуванням точки з'єднуються суцільними лініями синього кольору. Типи та колір ліній, символів і точок можна

змінити, є ряд функцій для цього. Як приклад, побудуємо графік функції y = sin(πx)/(πx), яка в MATLAB має ім'я sinc (рис. А.8):

```
>> x = -10:0.1:10; % значення координати x
>> y = sinc(x);  % значення координати y
>> plot(x, y)
```

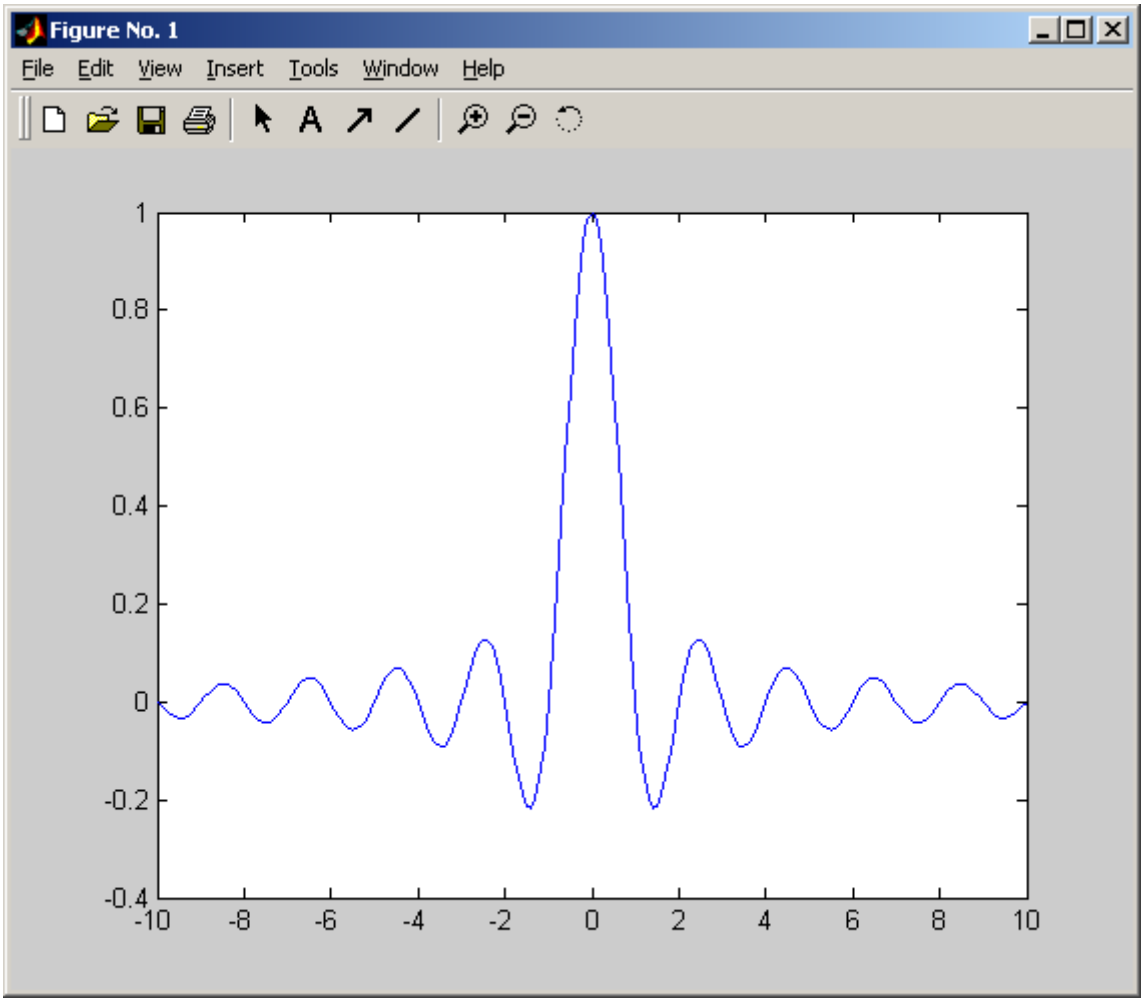


Рисунок А. 8 - Графік, побудований функцією plot

Якщо на екрані ще не було графіків, функція plot створить графічне вікно. Якщо на екрані вже були графічні вікна, графік буде виведений в поточне вікно - те, яке останнім виводилося на передній план. Про те, як можна створити декілька графічних вікон, буде розказано далі, в розділі «Одночасне виведення декількох графіків».

Можна використовувати функцію plot з одним аргументом - plot(y). В цьому випадку вектор y задає вертикальні координати крапок, а по горизонталі відкладаються номери елементів вектора.

Графік MATLAB є векторним графічним об'єктом, для перетворення якого на екранні піксели може знадобитися значний об'єм обчислень. Якщо ви змінюєте розмір вікна, розвертаючи його, перетягуючи його межі або використовуючи команду *Размер* системного меню вікна, всі обчислення, необхідні для побудови растрового образу графіка, системі доведеться виконати наново. Перерисовування графіка після зміни розмірів його вікна потребує стільки ж часу, скільки і первинна побудова. Майте це на увазі, якщо ваш графік містить велику кількість елементів.

З панеллю управління графічного вікна можна познайомиться за допомогою вкладки Help (рис. А. 8).

Інші різновиди двовимірних графіків

Окрім функції plot є ще цілий ряд функцій з аналогічним синтаксисом, що дозволяють отримати інші різновиди двовимірних графіків. Нижче перераховані тільки деякі з цих функцій, найбільш корисні в інженерних і наукових застосуваннях:

- semilogx(x, y) - графік з логарифмічним масштабом по осі x;
- semilogy(x, y) - графік з логарифмічним масштабом по осі y;
- loglog(x, y) - графік з логарифмічним масштабом по обох осях;
- plotyy(xl, yl, x2, y2) - виведення залежностей y1 від x1 і y2 від x2 з роздільним оцифруванням вертикальних осей (для першого графіка оцифрування вертикальної осі наноситься зліва, для другого - справа);
- stem(x, y) - графік у вигляді «стеблинок»;
- stairs(x. y) - графік у вигляді ступінчастої лінії;
- polar(phi, r) - графік в полярних координатах (phi - кутові координати точок в радіанах, r - відповідні їм радіуси).

Тривимірна графіка

У MATLAB є функція з ім’ям plot3, але основним засобом тривимірної графіки є не вона. Річ у тому, що функція plot3(x, y, z) є просто тривимірним аналогом функції plot, тобто вона дозволяє

побудувати лінію в тривимірному просторі за координатами точок, що задаються векторами x, y і z.

На практиці набагато частіше необхідно будувати графіки функцій двох змінних у вигляді поверхонь або ліній рівного рівня. Для побудови тривимірних поверхонь служить функція surf, що має такий синтаксис: surf(X, Y, Z). Тут X, Y і Z - двовимірні масиви однакового розміру. Функція будує поверхню, що складається з чотирикутних комірок. Координати кутів кожної комірки задаються значеннями чотирьох сусідніх елементів масивів X, Y і Z з індексами (i, j), (i, j+1), (i+l, j) і (i+l, j+1). Колір комірок (точніше, індекс кольору в поточній палітрі) за замовчуванням змінюється пропорційно координаті z. Із способу задання параметрів для функції surf видно, що її можливості не обмежуються побудовою двокоординатних функціональних залежностей. Дійсно, функція surf дозволяє будувати довільні тривимірні поверхні. Проте ця гнучкість дещо ускладнює побудову звичайних графіків функцій двох змінних. Щоб побудувати графік функціонально заданої поверхні z = f(x, y), масиви X і Y, які передаються функції surf, повинні задавати сітку (як правило, рівномірну). Значення масиву Z розраховуються за формулою функціональної залежності з використанням поелементних операцій над масивами X і Y. Масиви X і Y в даному випадку найзручніше формувати за допомогою спеціальної функції meshgrid, що має такий синтаксис: [X, Y] = meshgrid(x, y). Тут x і y - вектори, що задають набори значень x- і y-координат. Формовані масиви X і Y мають length(y) рядків і length(x) стовпців. Рядки масиву X є копіями вектора x, а стовпці масиву Y - копіями вектора y. Щоб пояснити роботу функції meshgrid, передамо їй два цілочислові вектори:

```
>> [X, Y] = meshgrid(l:5, -3:3)
X =
    1    2    3    4    5
    1    2    3    4    5
    1    2    3    4    5
    1    2    3    4    5
    1    2    3    4    5
    1    2    3    4    5
    1    2    3    4    5
```

```
Y =
    -3    -3    -3    -3    -3
    -2    -2    -2    -2    -2
    -1    -1    -1    -1    -1
     0     0     0     0     0
     1     1     1     1     1
     2     2     2     2     2
     3     3     3     3     3
```

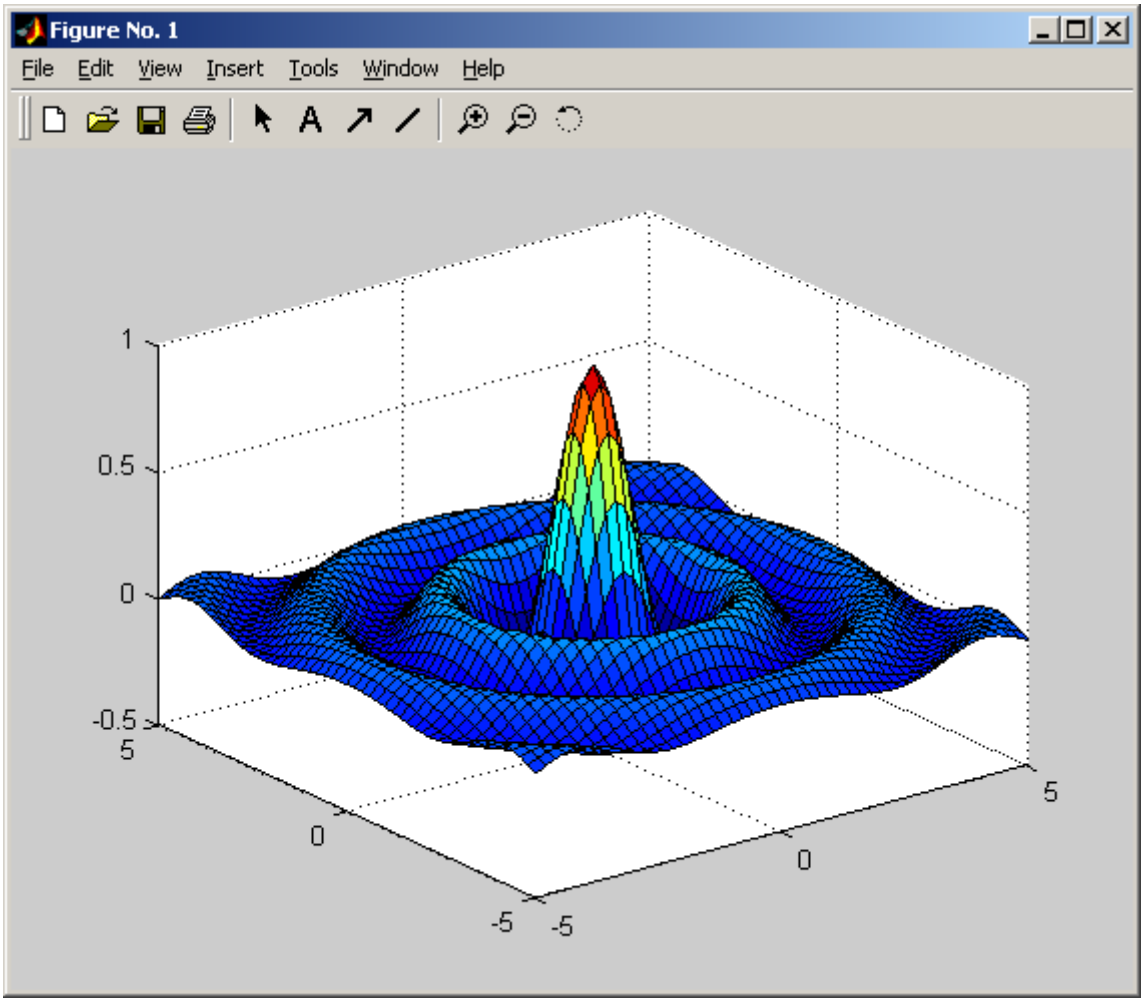


Рисунок А. 9 - Тривимірна поверхня, побудована за допомогою функції surf

Як приклад, побудуємо графік функції

$$z(x,y)=\frac{\sin(\pi\sqrt{x^2+y^2})}{\pi\sqrt{x^2+y^2}}$$

при зміні x і y від -5 до 5 з кроком 0,2 (рис. А. 9):

```
>> x = -5:0.2:5;
>> [X, Y] = meshgrid(x);
>> Z = sinc(sqrt(X.^2+Y.^2));
>> surf(X, Y, Z)
>> colormap gray
```

У наведеному фрагменті коду функція meshgrid використовується з одним вхідним параметром. Це скорочений варіант виклику, еквівалентний meshgrid(x, x).

Для побудови графіка функції двох змінних у вигляді ліній рівного рівня служить функція contour, яка має такий синтаксис:

[c, h] = contour(X, Y, Z, V).

Тут Z - двовимірний масив, що містить значення функції, а параметри X і Y можуть бути як масивами, отриманими від функції meshgrid, так і векторами координат, що передаються функції meshgrid. За відсутності цих параметрів як координати використовуються номери рядків і стовпців матриці Z. Параметр V може бути числом (тоді він задає кількість ліній рівня, що виводяться) або вектором (тоді цей параметр трактується як набір значень функції для побудови ліній рівного рівня). За відсутності даного параметра значення рівня вибираються автоматично. Результати c і h, що повертаються цією функцією, є структури даних, які необхідно передати іншій функції, clabel, для нанесення оцифрування на лінії рівня: clabel(c,h).

Інші різновиди тривимірних графіків

Окрім функцій surf і contour є ще цілий ряд функцій з аналогічним синтаксисом, що дозволяють отримати інші різновиди тривимірних графіків. Нижче перераховані тільки деякі з цих функцій, найбільш корисні в інженерних і наукових застосуваннях:

- surfc(X, Y, Z) - комбінація функцій surf і contour. Лінії рівного рівня виводяться на нижній координатній площині;

- mesh(X, Y, Z) - побудова поверхні у вигляді сітки із зафарбованими ребрами і незафарбованими чотирикутними комірками;

- meshc(X, Y, Z) - комбінація функцій mesh і contour. Лінії рівного рівня виводяться на нижній координатній площині;

- meshz(X, Y, Z) - те ж, що mesh, але з країв побудованої сітчастої поверхні вниз спадає «завіса»;

- waterfall (X, Y. Z) - те ж, що mesh, але ребра, що розділяють комірки, проводяться тільки уздовж осі x. В результаті об'ємне тіло виглядає «нарізаним на скибочки»;

- stem3(X, Y, Z) - виведення тривимірного графіка у вигляді «стеблинок», що починаються при z = 0 в точках, які задаються масивами X і Y. Висота «стеблинок» визначається масивом Z;

- contourf(X, Y, Z) - те ж, що contour, але простір між лініями рівного рівня зафарбований в різні кольори залежно від значень Z;

- contour3(X, Y, Z) - те ж, що contour, але лінії рівного рівня рисуються не в одній площині, а при відповідних значеннях z-координати;

- pcolor(X, Y, Z) - будує двовимірний графік, що є сіткою з чотирикутних комірок, координати вершин яких задаються масивами X і Y. Комірки зафарбовані кольорами, що визначаються значеннями елементів масиву Z. Той же результат можна отримати, якщо для поверхні, побудованою функцією surf, встановити точку огляду точно зверху.

В більшості випадків параметри оформлення графіків, вибрані за замовчуванням, виявляються цілком прийнятними. Проте інколи для підвищення виразності зображення виникає необхідність змінити тип і колір ліній, нанести на графік пояснювальні написи і тому подібне. MATLAB має ряд функцій, які дозволяють змінювати зовнішній вигляд графіка. Для знайомства з ними необхідно скористатися документацією MATLAB.

Одночасне виведення декількох графіків

Вивести на екран декілька графіків одночасно можна по-різному - в різних графічних вікнах, в різних зонах одного вікна або ж в загальних осях координат. Залежно від цього використовуються різні засоби MATLAB.

Щоб вивести декілька графіків в різних графічних вікнах, потрібно створити ці графічні вікна за допомогою функції figure або команди меню File > New > Figure головного вікна MATLAB (у меню графічних вікон є

коротший варіант - команда File > New Figure). Після створення нового вікна воно стає поточним, і подальше графічне виведення прямує саме в нього. Зробити конкретне графічне вікно поточним можна, вивівши його на передній план і потім перемкнувшись в командне вікно MATLAB. Наведені нижче команди створять два графіки в різних вікнах:

```
>> plot (xl, yl)
>> figure
>> plot(x2, y2)
```

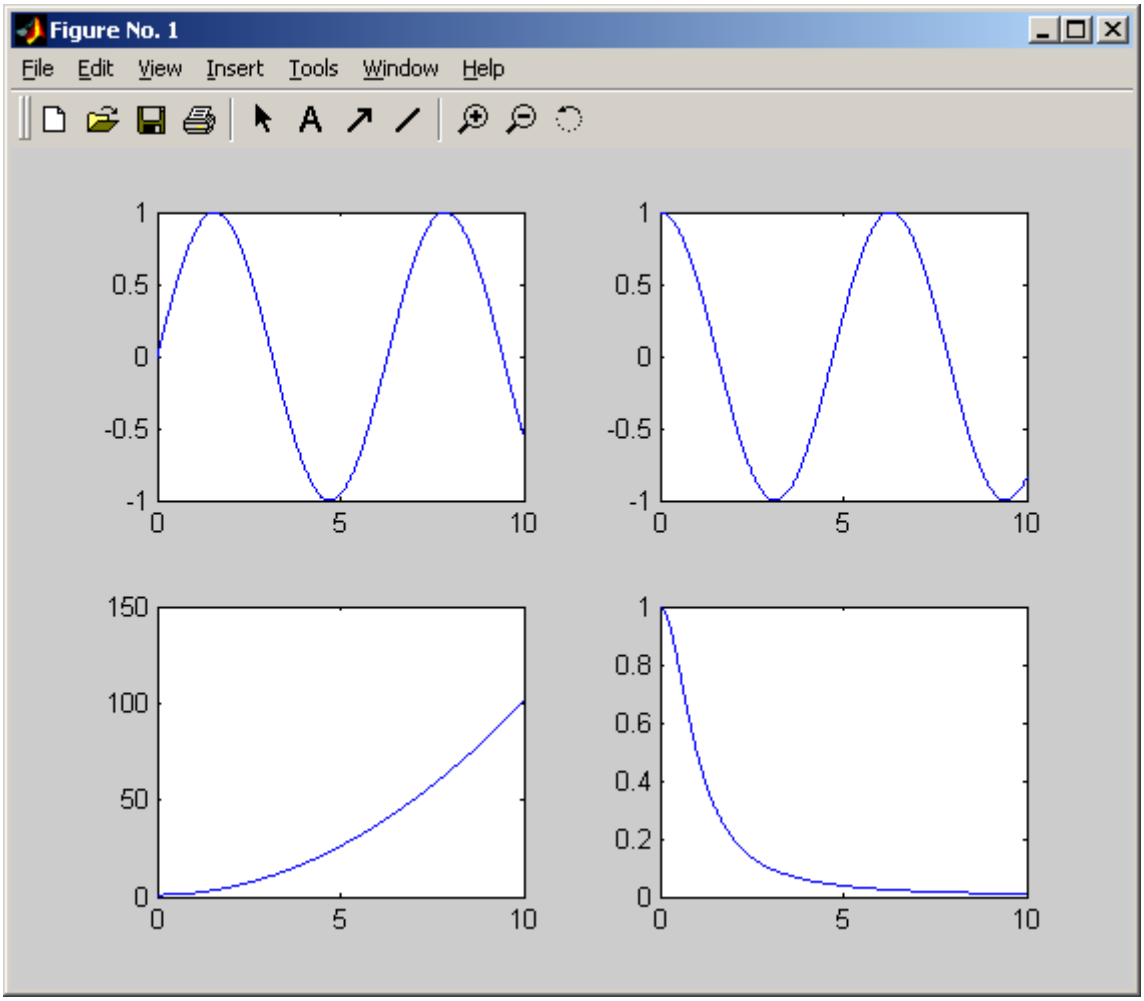


Рисунок А. 10 - Результат використання команди subplot

Для виведення декількох графіків в різних зонах одного вікна використовується команда subplot, що має такий синтаксис:

subplot(Rows, Cols, N).

В результаті графічне вікно буде розбито на клітинки у вигляді матриці, що має Rows рядків і Cols стовпців, і N-а клітинка (на відміну від нумерації елементів в матрицях, нумерація клітинок здійснюється по рядках) стає поточною. Наведені нижче команди створюють чотири графіки, розташовуючи їх в одному графічному вікні в два рядки і два стовпці (рис. А.10):

```
>> t = 0:0.01:10;
>> subplot(2, 2, 1)
>> plot(t, sin(t))
>> subplot(2, 2, 2)
>> plot(t, cos(t))
>> subplot(2, 2, 3)
>> plot(t, t.^2+1)
>> subplot(2, 2, 4)
>> plot(t, 1./(t.^2+1))
```

Нарешті вивести декілька графіків в загальних осях теж можна різними способами. По-перше, якщо параметр функції plot, який задає значення координати y, є матрицею, то будуються окремі графіки для кожного стовпця матриці, наприклад:

```
>> x = (0:0.1:10)';
>> plot(x, [sin(x) cos(x)])
```

По-друге, при виклику функції plot можна перерахувати декілька пар значень x- і y-координат: plot (xl, y1, x2, y2 …). При цьому для кожної пари (xi, yi) буде побудований свій графік. Тому той же результат, що і в попередньому прикладі, можна отримати і таким чином:

```
>> plot (x, sin(x), x, cos(x)).
```

По-третє, можна використовувати команду hold on, що включає режим збереження вмісту поточних координатних осей при виведенні в них нових графіків. Тоді нові команди plot будуватимуть графіки поверх тих, що є, не затираючи їх. Для усунення цього режиму використовується команда hold off. Перепишемо наш приклад ще раз, використовуючи команди hold (див. рис. А. 11):

```
>> x = (0:0.1:10);
>> plot(x, sin(x))
>> hold on
>> plot(x, cos(x))
>> hold off
```

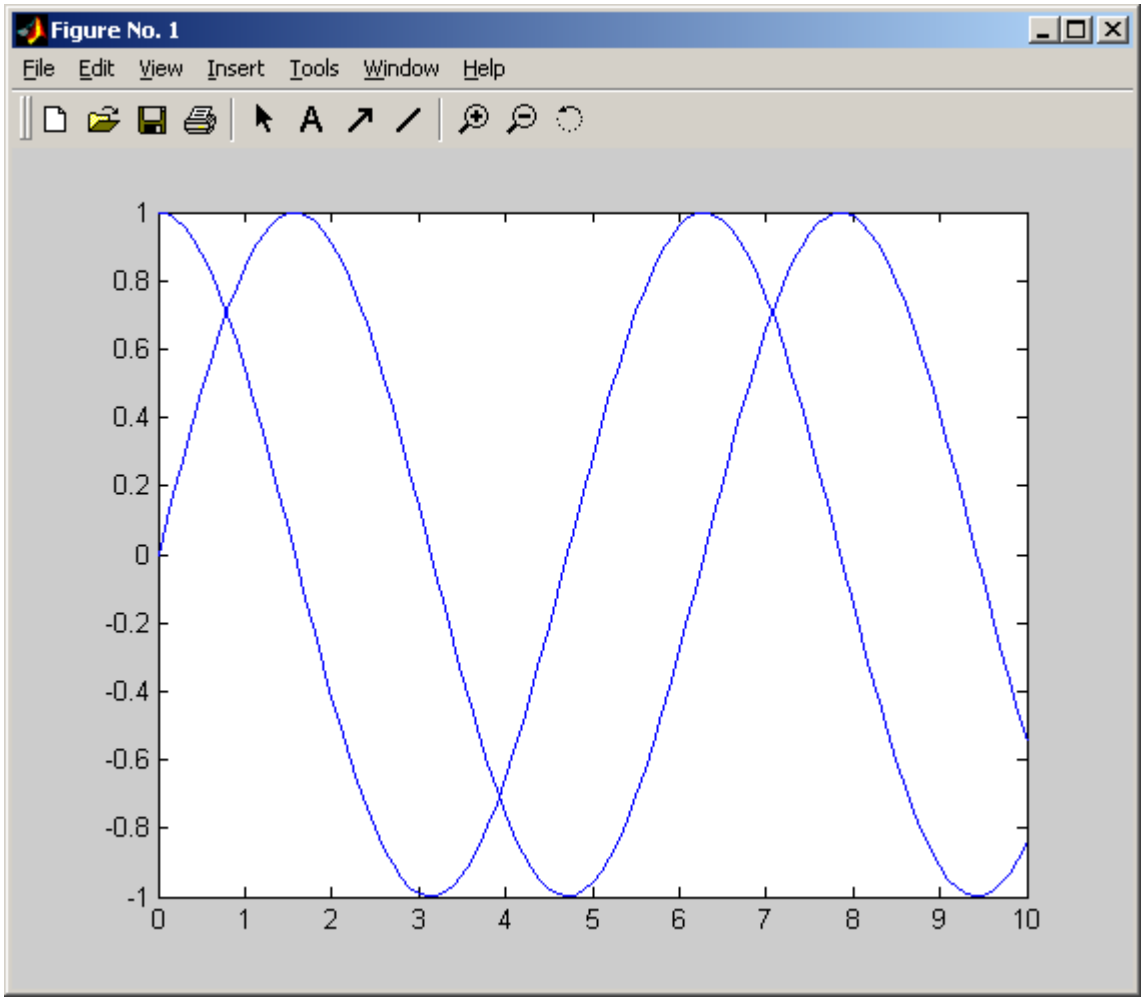


Рисунок А. 11 - Декілька графіків в загальних осях.

ДОДАТОК Б

ОГЛЯД ФУНКЦІЙ MATLAB

Цих функцій дуже багато, щоб можна було навести хоч би короткий опис для кожної з них. Тому нижче наводиться лише список тематичних розділів довідки MATLAB з короткою їх характеристикою. За докладнішим описом слід звертатися до довідкової системи і документації MATLAB. Розділи перераховані в алфавітному порядку імен їх каталогів, а не у порядку значущості. Отримати довідку про конкретний розділ можна, набравши в командному рядку такий текст: >> help ім'я_розділу. В результаті буде виведений список функцій даного розділу з короткими коментарями щодо їх призначення.

Audio

Містить функції роботи зі звуковою картою комп'ютера.

Datafun

У цьому розділі зосереджені різноманітні функції аналізу даних. Більшість з них орієнтована на роботу з векторним аргументом; матриці обробляються по стовпцях.

Datatypes

Містить засоби підтримки типів даних MATLAB.

Demos

У даному розділі зібрані демонстраційні приклади.

Elfun

Як видно з назви розділу, в ньому містяться функції, що реалізовують елементарні математичні функції. В більшості випадків ці функції вбудовані в ядро системи, так що в M-файлах міститься тільки текст довідки.

Elmat

У даному розділі зібрані функції елементарних матричних операцій.

Funfun

Містить функції пошуку мінімумів і нулів функцій, числового інтегрування та розв’язування диференціальних рівнянь. Всі ці операції потребують створення М-файла функції, з якою вестиметься робота, тому даний розділ і отримав таке ім’я - «Function functions».

General

Даний розділ містить функції загального призначення. Сюди відносяться команди роботи з довідковою системою, управління робочою областю пам’яті, відкриття і збереження файлів, налагодження шляху пошуку файлів, управління командним вікном, засоби виклику команд операційної системи, а також інші налагоджувальні засоби.

Graph2d

У даному розділі містяться засоби двовимірної графіки. Найважливіші з них розглянуто в додатку А.

Graph3d

У даному розділі містяться засоби тривимірної графіки. Найважливіші з них розглянуто в додатку А.

Graphics

У цьому розділі зібрані засоби підтримки дескрипторної графіки (Handle Graphics®) MATLAB. Коротко її ідею можна описати таким чином. Всі графічні функції, такі як figure, plot та інші, можуть повертати результат - число або декілька чисел, які є дескрипторами (handle), створених функцією графічних об'єктів, - графічних вікон, координатних осей, графіків і так далі. Ці дескриптори використовуються для отримання програмного доступу до графічних об'єктів і їх властивостей. Докладніший опис дескрипторної графіки виходить за рамки тематики даної книги. Необхідна інформація міститься у довідковій системі MATLAB і книзі «Using MATLAB Graphics», що постачається у складі документації.

Iofun

До даної категорії відносяться засоби файлового введення/виведення. Найважливіші функції роботи з файлами розглянуті в додатку А.

Lang

У даному розділі зібрані конструкції мови програмування MATLAB. Основні елементи мови були описані в додатку А.

Matfun

У даному розділі зібрані матричні функції лінійної алгебри.

Ops

Даний розділ присвячений операторам і символам, що мають для MATLAB спеціальне значення, зокрема, функції роботи з бітовими поданнями додатних цілих чисел.

Polyfun

У даному розділі зібрані функції інтерполяції і роботи з поліномами.

Sparfun

У деяких наочних областях доводиться мати справу з матрицями дуже великого розміру, у яких відмінна від нуля лише мала частина елементів. Такі матриці називаються розрідженими (sparse). На зберігання нульових елементів і виконання математичних операцій з ними даремно витрачається багато ресурсів комп'ютера. У даному розділі зібрані функції, що дозволяють зберігати матриці і виконувати матричні операції безпосередньо в розрідженому вигляді, тобто зберігати тільки значення і індекси ненульових елементів матриць.

Specfun

Як виявляється з назви, в цьому розділі знаходяться спеціальні математичні функції.

Specgraph

У даному розділі зібрані спеціалізовані графічні функції. Сюди відносяться різноманітні діаграми (area, bar, barh, pie і ін.), а також багато інших. Особливий інтерес викликають функції, імена яких починаються з букв ez (від «easy» - «легкий») - ezplot, ezpolar, ezcontour, ezcontourf, ezgraph3, ezmesh, ezmeshc, ezplot3, ezsurf і ezsurfc. Вони дозволяють

будувати графіки відповідних типів (див. п. «Графіка» додатка А), задаючи не масиви даних, а рядки, в яких записані формули для обчислень. Також в даному розділі є функції читання і запису графічних файлів, тривимірної візуалізації і анімації.

Strfun

До даного розділу відносяться функції роботи з рядками (див. додаток А).

Timefun

У даному розділі зібрані функції роботи з датами і часом.

Uitools

MATLAB дозволяє створювати програми, що мають графічний призначений інтерфейс користувача. У даному розділі бібліотеки зібрані засоби, що забезпечують відповідні функції. Питання створення призначеного для користувача інтерфейсу виходять далеко за рамки тематики даної книги, скажемо лише, що запуск конструктора форм проводиться функцією guide. За додатковою інформацією звертайтеся до довідкової системи і документації MATLAB.

Verctrl

Функції даного розділу забезпечують взаємодію MATLAB зі системами контролю версій файлів (version control system, VCS).

Winfun

У даному розділі зібрані функції взаємодії MATLAB з операційною системою Windows через інтерфейси DDE і ActiveX. Роботу з ActiveX демонструють два приклади: mwsamp (створення елемента управління ActiveX) і sampev (створення обробника події для сервера ActiveX; ця функція використовується прикладом mwsamp).

ДОДАТОК В

КОМПОНЕНТИ MATLAB

MATLAB - складна система, і вона постійно розвивається, набуваючи все нових можливостей. Головна сила MATLAB - це численні пакети розширення, орієнтовані на вирішення завдань в різних областях. У даному додатку перераховані компоненти MATLAB і наведені їх короткі описи. Розділення компонентів на категорії відповідає розділенню пропонованому фірмою MathWorks.

MATLAB

Окрім власне ядра MATLAB до складу системи входить цілий ряд компонентів, що надають їй додаткові можливості:

- MATLAB - ядро системи MATLAB («мова технічних обчислень», так називають свій продукт розробники);
- MATLAB Compiler - засіб перетворення M-файлів в програмний код мовою C/C++;
- MATLAB C/C++ Graphics Library - бібліотека графічних функцій MATLAB для використання в програмах на C/C++;
- MATLAB C/C++ Math Library - бібліотека математичних функцій MATLAB для використання в програмах на C/C++;
- MATLAB Excel Builder - засіб створення модулів розширення для Microsoft Excel;
- MATLAB Report Generator - засіб створення документації для MATLAB-додатків і даних;
- MATLAB Runtime Server - засіб перетворення MATLAB-додатків в автономні програмні продукти;
- MATLAB Web Server - засіб розробки і розповсюдження MATLAB-додатків, що запускаються через інтернет;
- MATRIXVB - інтерфейс для виклику функцій MATLAB з Microsoft Visual Basic.

Пакети розширення MATLAB

Пакетами розширення MATLAB (toolbox) є набори функцій, об'єднаних загальною тематикою і орієнтованих на вирішення завдань конкретної предметної області.

Математика і аналіз

- Curve Fitting Toolbox - функції апроксимації;

- Optimization Toolbox - функції, що реалізовують різноманітні числові методи оптимізації;

- Statistics Toolbox - функції статистичного моделювання і перевірки гіпотез;

- Neural Network Toolbox - функції моделювання нейронних мереж;

- Symbolic/Extended Symbolic Math Toolbox - реалізація аналітичних розрахунків з використанням ядра програми Maple;

- Partial Differential Equation Toolbox - розв’язування диференціальних рівнянь в частинних похідних;

- PLS_Toolbox - аналіз і моделювання часової динаміки хімічних процесів методом часткових найменших квадратів (Partial Least Squares, PLS);

- Mapping Toolbox - пакет розширення для візуалізації географічних карт і роботи з іншою географічною інформацією;

- Spline Toolbox - функції сплайнової апроксимації і інтерполяції;

- Structural Dynamics Toolbox - аналіз коливань і вібрацій в механічних системах методом скінченних елементів;

- Virtual Reality Toolbox - засіб створення віртуальних тривимірних моделей, керованих із середовища MATLAB або Simulink, на основі мови VRML.

Експорт і імпорт даних

- Data Acquisition Toolbox - засіб взаємодії з устаткуванням аналогового і цифрового введення/виведення даних;

- Instrument Control Toolbox — засіб взаємодії MATLAB із сучасною вимірювальною апаратурою через протоколи GPIB і VISA;

- Database Toolbox - засіб обміну даними з реляційними базами даних;

- Excel Link - засіб взаємодії MATLAB з Microsoft Excel;

- Portable Graph Object - засіб, що дозволяє експортувати графіки MATLAB в Microsoft Word у вигляді об'єктів ActiveX.

Обробка сигналів і зображень

- Signal Processing Toolbox - функції аналізу і обробки сигналів;

- Image Processing Toolbox - функції аналізу і обробки зображень;

- Communications Toolbox - функції аналізу і моделювання систем зв'язку;

- Frequency Domain System Identification Toolbox - функції ідентифікації лінійних динамічних систем в частотній області;

- System Identification Toolbox - функції ідентифікації лінійних динамічних систем, тобто створення їх математичних моделей на основі виміряних вхідних і вихідних сигналів;

- Wavelet Toolbox - функції аналізу, очищення від шуму і компресії сигналів і зображень з використанням вейвлет-алгоритмів;

- Filter Design Toolbox - функції аналізу і синтезу фільтрів, зокрема з урахуванням ефектів квантування і арифметики з фіксованою комою;

- Motorola DSP Developer's Kit - функції моделювання і тестування коду для сигнальних процесорів серіїв 56300 і 56600 фірм Motorola;

- Developer's Kit for Texas Instruments™ DSP - функції моделювання і тестування коду для цифрових сигнальних процесорів фірми Texas Instruments.

Розробка систем управління

- Control System Toolbox - функції моделювання і аналізу систем автоматичного управління (систем зі зворотним зв'язком);

- Fuzzy Logic Toolbox - функції моделювання і аналізу систем із нечіткою логікою;

- Robust Control Toolbox - функції моделювання і аналізу систем управління, стійких до випадкових дій;

- Analysis and Synthesis Toolbox - функції моделювання і аналізу систем управління з багатьма змінними за наявності невизначеності моделі;

- LMI Control Toolbox - функції моделювання і аналізу стійких систем управління з використанням методів оптимізації опуклих функцій;

- Model Predictive Control Toolbox - функції моделювання і аналізу систем управління з великим числом вхідних і вихідних параметрів за наявності обмежень;

- Polynomial Toolbox - функції, що реалізовують сучасні поліноміальні методи стосовно систем управління;

- QFT Frequency Domain Control Design Toolbox - функції моделювання і аналізу стійких систем управління в частотній області з використанням теорії кількісних зворотних зв'язків (Quantitative Feedback Theory).

Фінансове моделювання і аналіз

- Financial Toolbox - функції для моделювання фінансових даних і розробки алгоритмів фінансового аналізу;
- Financial Time Series Toolbox - функції для аналізу фінансових даних методом часових рядів;
- GARCH Toolbox - функції аналізу мінливості фінансових ринків з використанням одновимірних GARCH-моделей;
- Financial Derivatives Toolbox - функції аналізу і моделювання процентних ставок і ризиків;
- Datafeed Toolbox - засіб отримання фінансових даних від постачальників інформації в реальному часі.

Simulink

Хоча Simulink® є розширенням MATLAB, проте він надає стільки нових можливостей, що його слід розглядати як окремий програмний продукт. Simulink - це графічне середовище моделювання аналогових і дискретних систем. Моделювання проводиться шляхом перетягування блоків з вікон бібліотек у вікно створюваної моделі і налагодження зв’язків між ними. Велике число наявних бібліотек дозволяє моделювати найрізноманітніші електричні, механічні, гідравлічні і інші системи. Після створення моделі можна запустити процес моделювання. Simulink створить систему диференціальних рівнянь, що описують модель, і почне виконувати її розв’язування числовим методом.

Simulink є складною системою, що містить декілька окремо встановлюваних компонентів:

- Simulink - ядро системи Simulink, середовище моделювання систем безперервного і дискретного часу;
- Simulink Performance Tools - засоби, що спрощують роботу з великими Simulink-моделями і дозволяють виконувати їх оптимізацію;
- Stateflow® - засіб моделювання систем, керованих подіями;
- Stateflow Coder - засіб генерації С-кода на основі діаграм Stateflow;

- Real-Time Windows Target - середовище моделювання, що забезпечує виконання моделей Simulink і Stateflow на персональному комп'ютері в реальному часі (з частотою дискретизації до 10 кГц);

- Real-Time Workshop® - засіб генерації С-кода на основі Simulink-моделей;

- Real-Time Workshop Embedded Coder - засіб генерації високоякісного початкового коду для вбудовуваних систем;

- Real-Time Workshop Ada Coder - засіб генерації коду Ada 95 і Ada 83 на основі Simulink-моделей;

- xPC Target - засіб створення типових додатків реального часу, що використовують стандартні апаратні засоби персональних комп'ютерів;

- xPC Target Embedded Option - засіб створення і розповсюдження додатків реального часу, що використовують стандартні апаратні засоби персональних комп'ютерів;

- Simulink Report Generator - засіб створення документації для моделей Simulink і Stateflow;

- Requirements Management Interface - засіб зв’язування набору формальних технічних вимог з моделями Simulink, M-файлами і діаграмами Stateflow;

- SimMechanics - засоби моделювання механічних систем.

Набори блоків Simulink

Наборами блоків Simulink (blockset) є тематичні бібліотеки, орієнтовані на конкретні предметні області:

- CDMA Reference Blockset - моделювання систем мобільної телефонії IS-95A;

- Communications Blockset - блоки для моделювання систем зв'язку;

- Dials & Gauges Blockset - набір графічних блоків вимірювальних приладів для відображення сигналів і параметрів моделювання;

- DSP Blockset - моделювання систем цифрової обробки сигналів;

- Fixed-Point Blockset - моделювання арифметики з фіксованою комою;

- Nonlinear Control Design Blockset - моделювання нелінійних систем управління;

- Power System Blockset - моделювання енергетичних систем.

Докладніше про систему Simulink можна прочитати в [11].

ДОДАТОК Г

ПРОГРАМА SPTOOL

Програма SPTool (Signal Processing Tool) надає в розпорядження користувача графічне середовище для перегляду графіків сигналів і їх спектрів, розрахунку і аналізу фільтрів, а також фільтрації сигналів. Для запуску програми SPTool необхідно набрати її ім'я в командному рядку MATLAB: >> sptool. Головне вікно програми показано на рис. Г. 1.

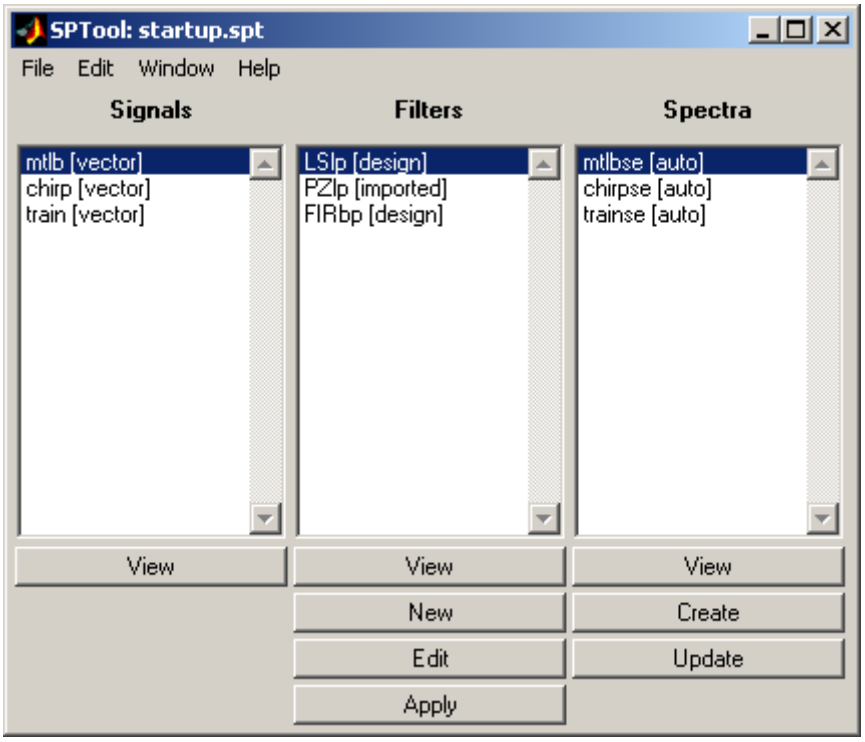


Рисунок Г. 1 - Програма SPTool - головне вікно

У трьох списках цього вікна перераховані завантажені в програму SPTool сигнали (Signals), фільтри (Filters) і спектри (Spectra). Розташовані під списками кнопки дозволяють виконувати різні операції.

Типовий набір дій, що виконуються за допомогою програми SPTool, включає такі операції:

- завантаження сигналу;
- проглядання графіка сигналу;
- спектральний аналіз сигналу;
- розрахунок фільтра;

- фільтрація сигналу.

Завантаження сигналу

Для завантаження сигналу в меню File головного вікна програми SPTool необхідно вибрати команду Import. З'явиться вікно Import to SPTool, показане на рис. Г. 2.

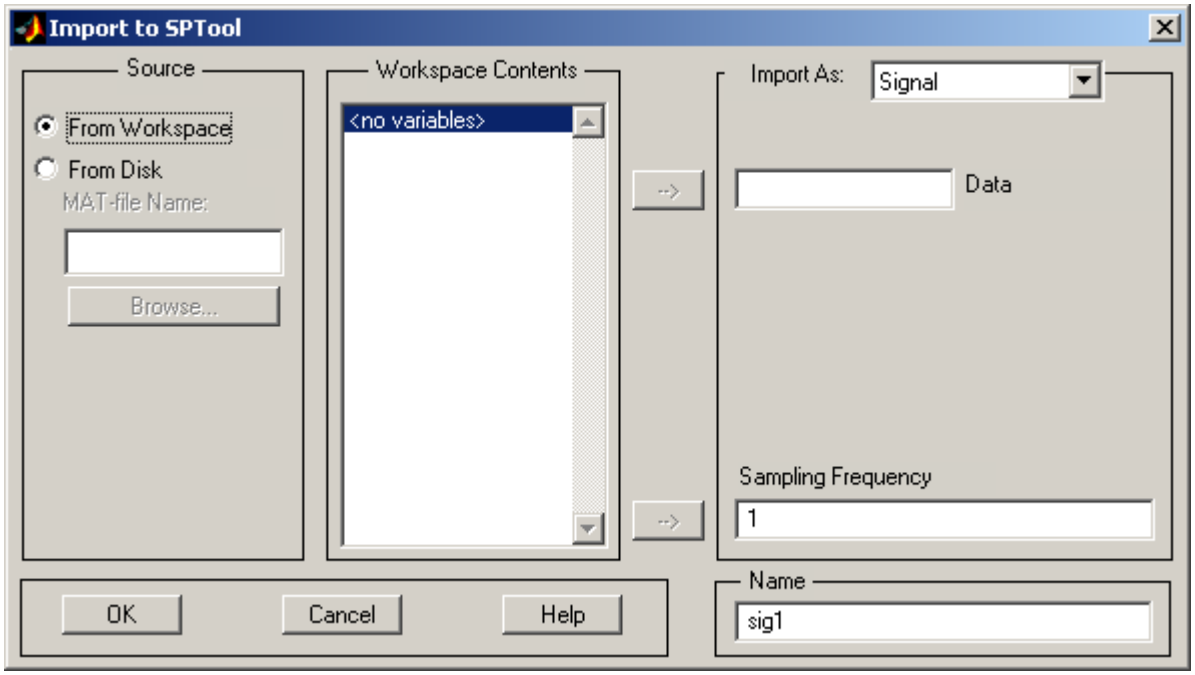


Рисунок Г. 2 - Програма SPTool - вікно імпорту даних

Перемикач Source дозволяє вказати місце зберігання завантажуваного сигналу - робочу область MATLAB (From Workspace) або MAT-файл (From Disk). У другому випадку стає доступним поле введення MAT-file Name, в яке потрібно ввести вручну або за допомогою кнопки Browse ім'я потрібного MAT-ФАЙЛА.

У списку Workspace Contents перераховані змінні, наявні в робочій пам'яті MATLAB в даний момент. Якщо як джерело був вибраний MAT-файл, цей список називатиметься File Contents і міститиме змінні, збережені у вибраному файлі.

У списку Import As, що розкривається, виберіть варіант Signal. Решта два варіанти, Filter і Spectrum, дозволяють імпортувати описи фільтрів і вже розрахованих спектрів для перегляду і аналізу.

Виберіть в списку змінну або уручну введіть в це поле ідентифікатор змінної, що містить відліки завантажуваного сигналу, і клацніть на кнопці «-->», розташованій поряд з полем введення Data (див. рис. Г. 2).

У полі введення Sampling Frequency за замовчуванням знаходиться значення 1. Його можна відредагувати вручну, а можна імпортувати, ввівши ідентифікатор змінної або вибравши його в списку і клацнувши на нижній кнопці «-->».

Нарешті, в полі введення Name можна відредагувати ім’я, під яким даний сигнал фігуруватиме в програмі SPTool. Зробивши це, клацніть на кнопці ОК. Імпортований сигнал з'явиться в списку Signals основного вікна програми (див. рис. Г. 1).

Перегляд графіка сигналу

Для перегляду графіка сигналу виберіть його в списку Signals (див. рис. Г. 1) і клацніть на кнопці View, що розташована під цим списком. З’явиться вікно Signal Browser, показане на рис. Г. 3.

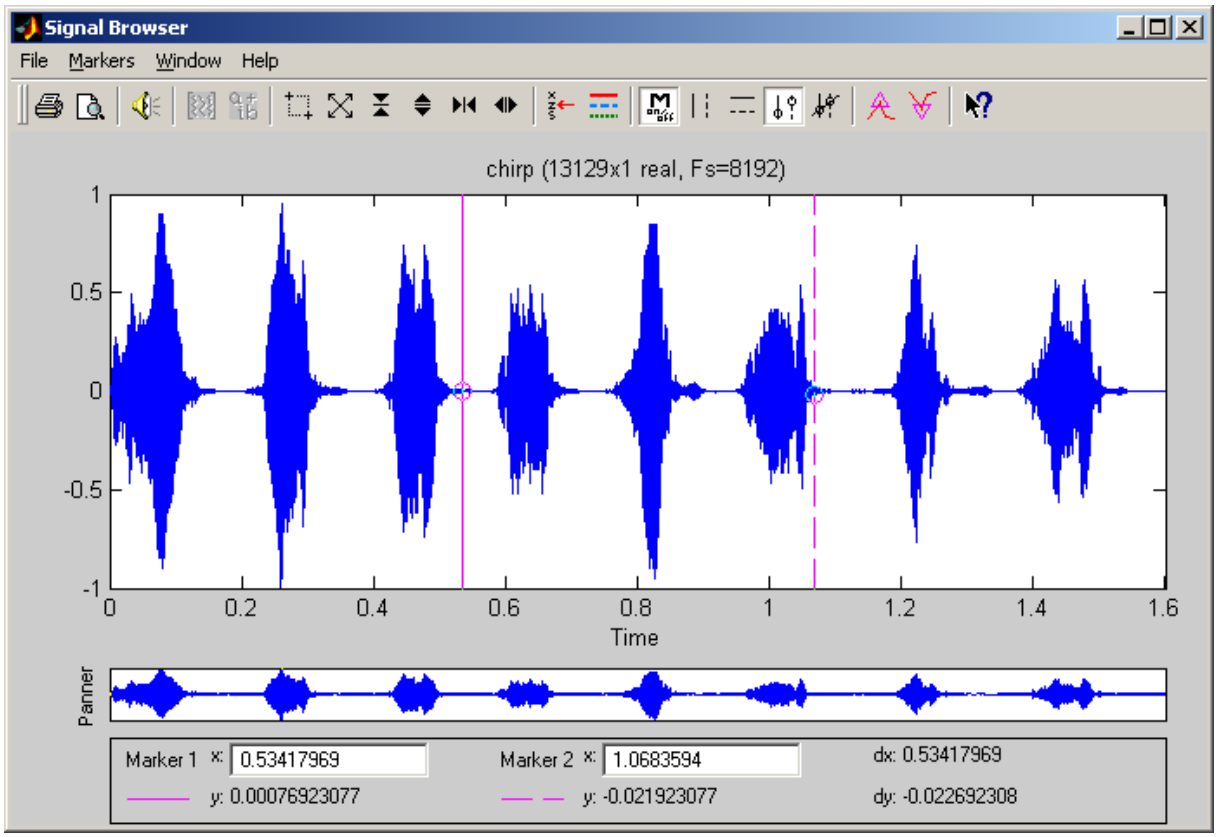


Рисунок Г. 3 - Програма SPTool - вікно перегляду сигналів

У даному вікні виводяться два графіки. Нижній, з написом Panner, показує панораму всього сигналу. Верхній графік спочатку теж відображує весь сигнал, але масштаб відображення можна збільшити, і тоді верхній графік показуватиме лише фрагмент сигналу, а положення цього фрагмента на загальній панорамі демонструється за допомогою прямокутника на нижньому графіку. Цей прямокутник можна перетягувати мишею, змінюючи тим самим ділянку огляду.

Кнопки панелі інструментів вікна Signal Browser дозволяють роздруковувати графік сигналу, відтворювати сигнал за допомогою звукової карти комп'ютера, управляти масштабом відображення, вибирати потрібний канал в разі багатоканального сигналу і задавати потрібний режим відображення маркерів.

На графік можна нанести два маркери, що дозволяють проводити над сигналом кількісні вимірювання. Маркери перетягуються за допомогою миші, а інформація про помічені ними відліки сигналу виводиться в нижній частині вікна.

Можна експортувати цю інформацію в робочу область пам'яті MATLAB у вигляді структури - для цього служить команда Export меню Markers.

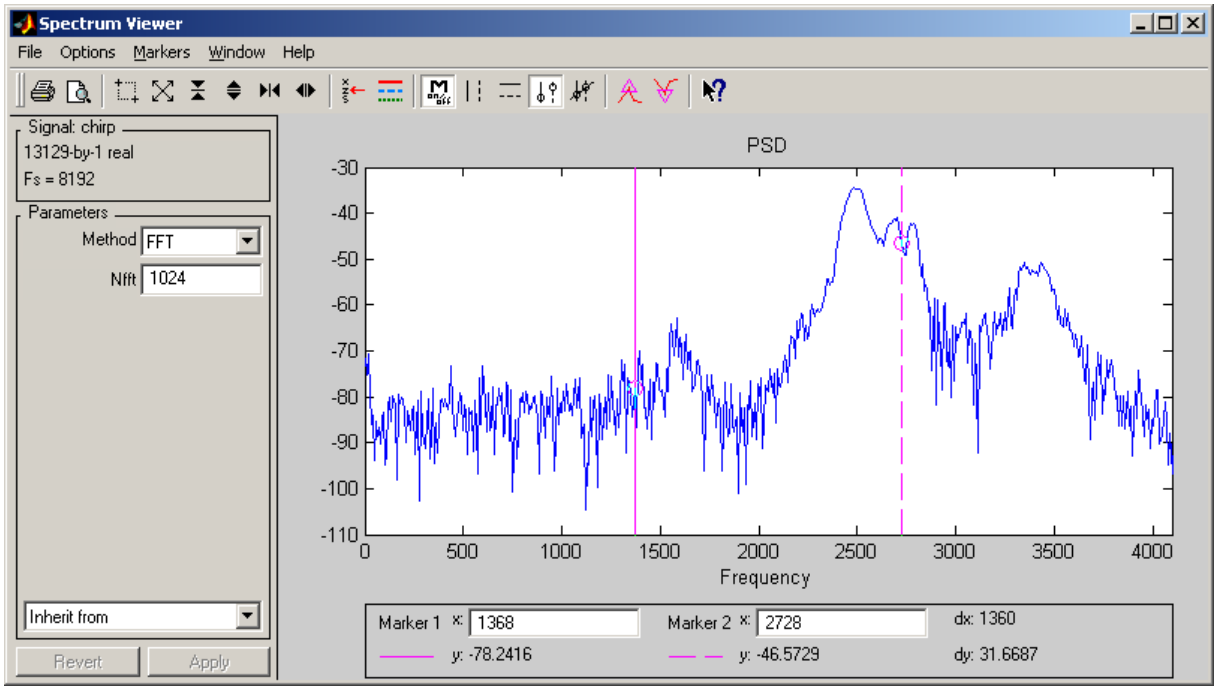


Рисунок Г. 4 - Програма SPTool - вікно спектрального аналізу

Спектральний аналіз сигналу

Для аналізу спектра сигналу, завантаженого в програму SPTool, виберіть потрібний сигнал в списку Signals головного вікна програми (див. рис. Г. 1) і клацніть на кнопці Create, розташованій під списком Spectra. З'явиться вікно Spectrum Viewer, показане на рис. Г. 4.

У лівій частині вікна вибирається метод спектрального аналізу і налагоджуються його параметри. Список, що розкривається, в якому спочатку виведений рядок Inherit from, дозволяє скопіювати повний набір налагоджень аналізатора спектра з іншого розрахунку, поданого в списку Spectra основного вікна програми.

Провівши налагодження параметрів аналізу, клацніть на кнопці Apply. Буде розрахована оцінка спектра сигналу і виведений відповідний графік. Кнопки панелі інструментів дозволяють виконувати ті ж операції, що були перераховані вище стосовно перегляду графіків сигналів. При перегляді спектра також можливе використання маркерів.

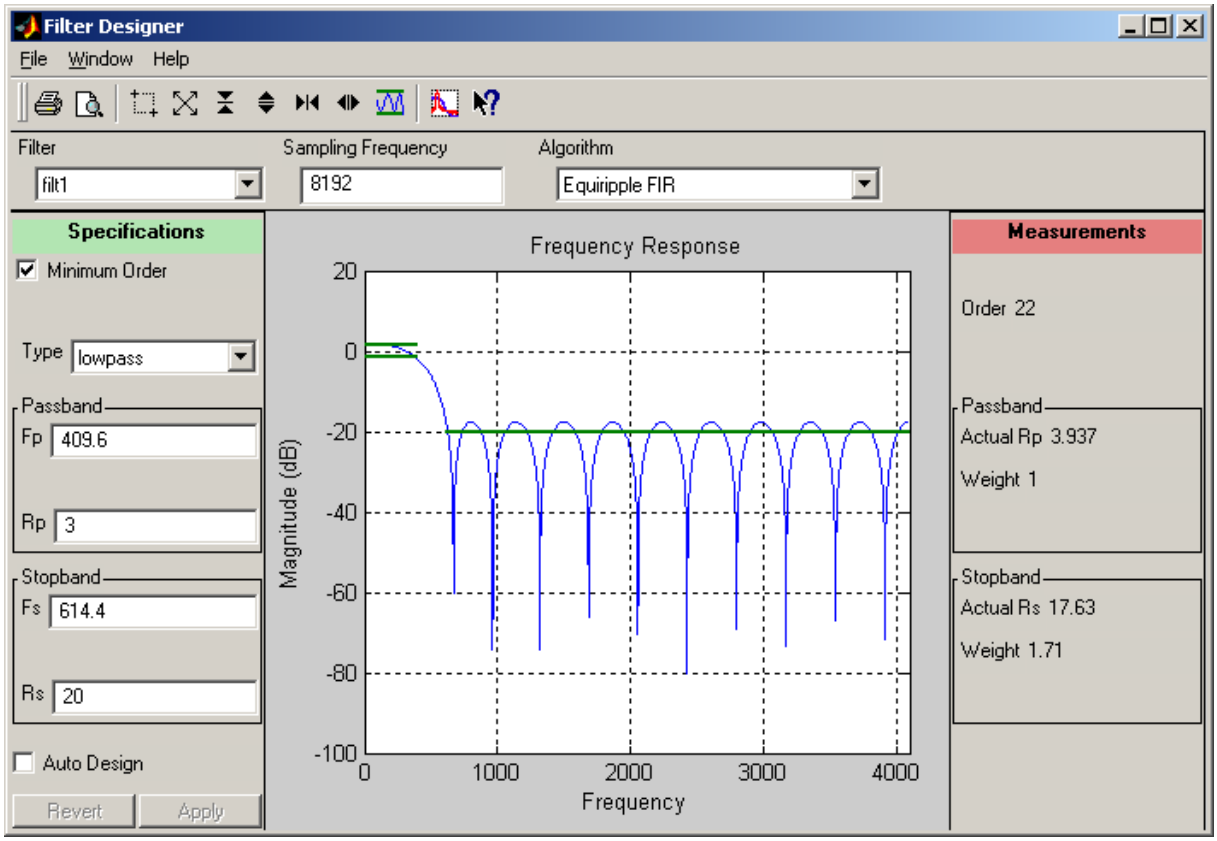


Рисунок Г. 5 - Програма SPTool - вікно розрахунку фільтрів

Розрахунок фільтра

Для розрахунку дискретного фільтра клацніть на кнопці New, розташованій під списком Filters в головному вікні програми SPTool (див. рис. Г. 1). Можна також змінити параметри вже розрахованого фільтра, вибравши його в списку Filters і клацнувши на кнопці Edit. З'явиться вікно Filter Designer, показане на рис. Г. 5.

У списку Algorithm, що розкривається, вибирається метод розрахунку, а в розділі Specifications задаються параметри фільтра, що синтезується. Більш докладніша інформація про методи синтезу дискретних фільтрів міститься в розділах 9-10 даного навчального посібника.

Вибравши метод розрахунку і задавши параметри фільтра, клацніть на кнопці Apply для виконання синтезу. В центрі вікна буде виведений графік АЧХ синтезованого фільтра, а в розділі Measurements показані його параметри.

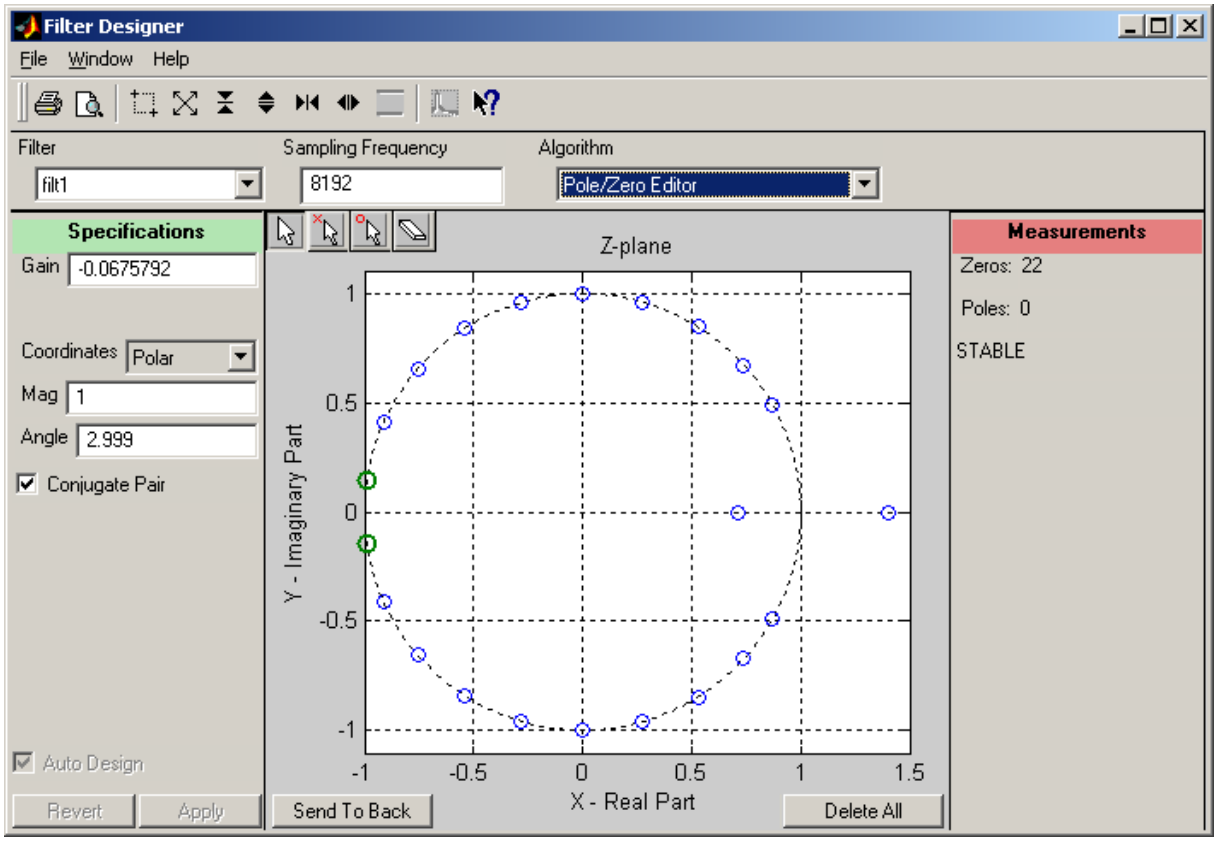


Рисунок Г. 6 - Програма SPTool - редагування розташування нулів і полюсів фільтра

Кнопки панелі інструментів дозволяють управляти масштабом відображення графіка. Крім того, можна вивести графік АЧХ в смузі пропускання великим планом, а також накласти на АЧХ графік одного із спектрів, перерахованих в списку Spectra основного вікна програми.

Можливості програми SPTool щодо синтезу фільтрів істотно обмежені в порівнянні з програмою FDATool. Єдина можливість, наявна в SPTool і відсутня в FDATool, - це пряме редагування розташування нулів і полюсів фільтра. Для такого редагування виберіть в списку Algorithm останній рядок - Pole/Zero Editor. Вікно Filter Designer прийме вигляд, показаний на рис. Г. 6.

У цьому режимі редагування фільтра можна переміщати нулі і полюси мишею, додавати і видаляти їх, автоматично формувати з них комплексно-пов'язані пари.

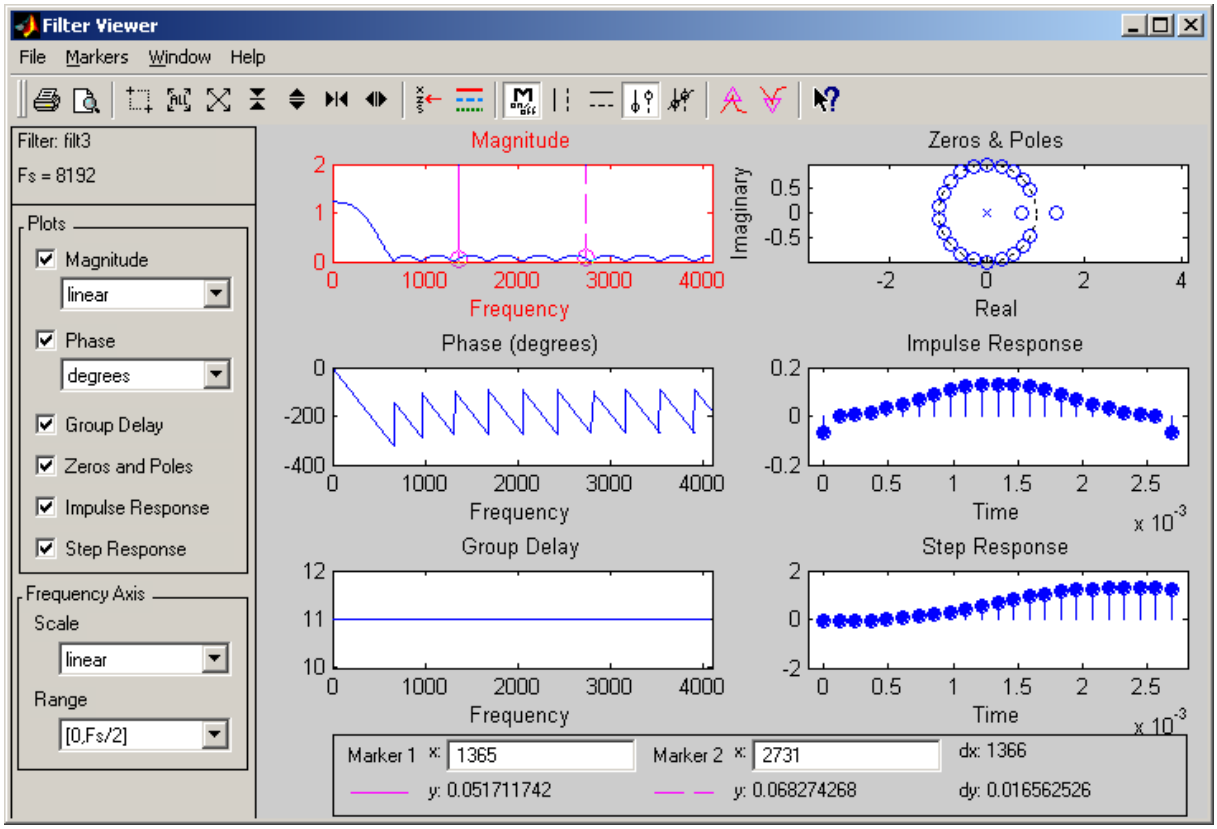


Рисунок Г. 7 - Програма SPTool - перегляд характеристик фільтра

Оскільки у вікні Filter Designer при редагуванні нулів і полюсів не виводиться графік АЧХ фільтра, при використанні даного режиму слід одночасно з вікном Filter Designer відкрити вікно Filter Viewer (рис. Г. 7).

При редагуванні фільтра у вікні Filter Designer його характеристики, що відображуються у вікні Filter Viewer, синхронно оновлюватимуться. Взагалі вікна програми SPTool не є модальними, тобто можна відкрити будь-яку їх кількість і вільно пересуватися між ними.

Перегляд характеристик фільтра

Для перегляду характеристик фільтра, завантаженого в програму SPTool, виберіть його в списку Filters основного вікна програми (див. рис. Г. 1) і клацніть на кнопці View, розташованій під цим списком. З'явиться вікно Filter Viewer, показане на рис. Г. 7.

У лівій частині вікна Filter Viewer розташована група прапорців Plots для вибору складу графіків, що відображуються. У розділі Frequency Axis можна вибрати частотний діапазон для перегляду характеристик і задати тип шкали частот - лінійний або логарифмічний.

Кнопки панелі інструментів дозволяють управляти режимом відображення і виведенням маркерів для поточного графіка, осі якого виділяються червоним кольором. Інформація про маркери, як завжди, виводиться в нижній частині вікна.

Фільтрація сигналу

Для пропускання сигналу через фільтр необхідно вибрати сигнал і фільтр відповідно в списках Signals і Filters основного вікна програми (див. рис. Г. 1), а потім клацнути на кнопці Apply, розташованій під списком Filters. З'явиться вікно Apply Filter, показане на рис. Г. 8.

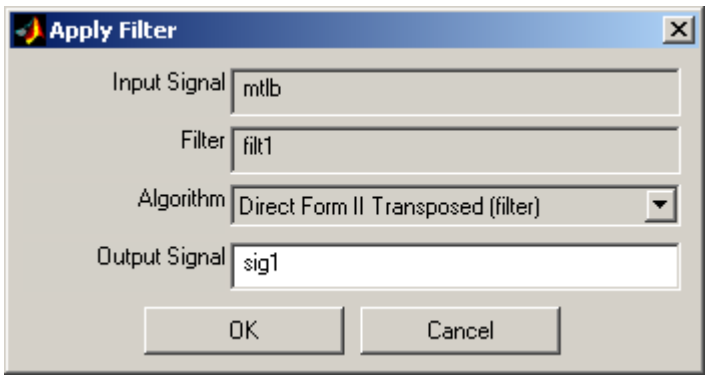


Рисунок Г. 8 - Програма SPTool - вікно застосування фільтра

Єдиним доступним полем введення в цьому вікні є поле Output Signal, в якому потрібно задати ім'я для вихідного сигналу. У списку Algorithm, що розкривається, можна вибрати функцію MATLAB для здійснення фільтрації - filter, filtfilt або fftfilt. Виконавши ці дії, клацніть на кнопці ОК. Буде розрахований вихідний сигнал, який з'явиться під вказаним ім'ям в списку Signals основного вікна програми.

Проглянути графік вихідного сигналу і виконати аналіз його спектра можна описаними раніше способами. Для одночасного перегляду графіків вхідного і вихідного сигналів потрібно вибрати їх обидва в списку Signals основного вікна програми (для цього при клацанні на вибраному сигналі необхідно, як це прийнято в Windows, натискувати клавішу Ctrl) і клацнути на кнопці View. У вікні Signal Browser, що відкрилося, будуть показані графіки обох вибраних сигналів (рис. Г. 9).

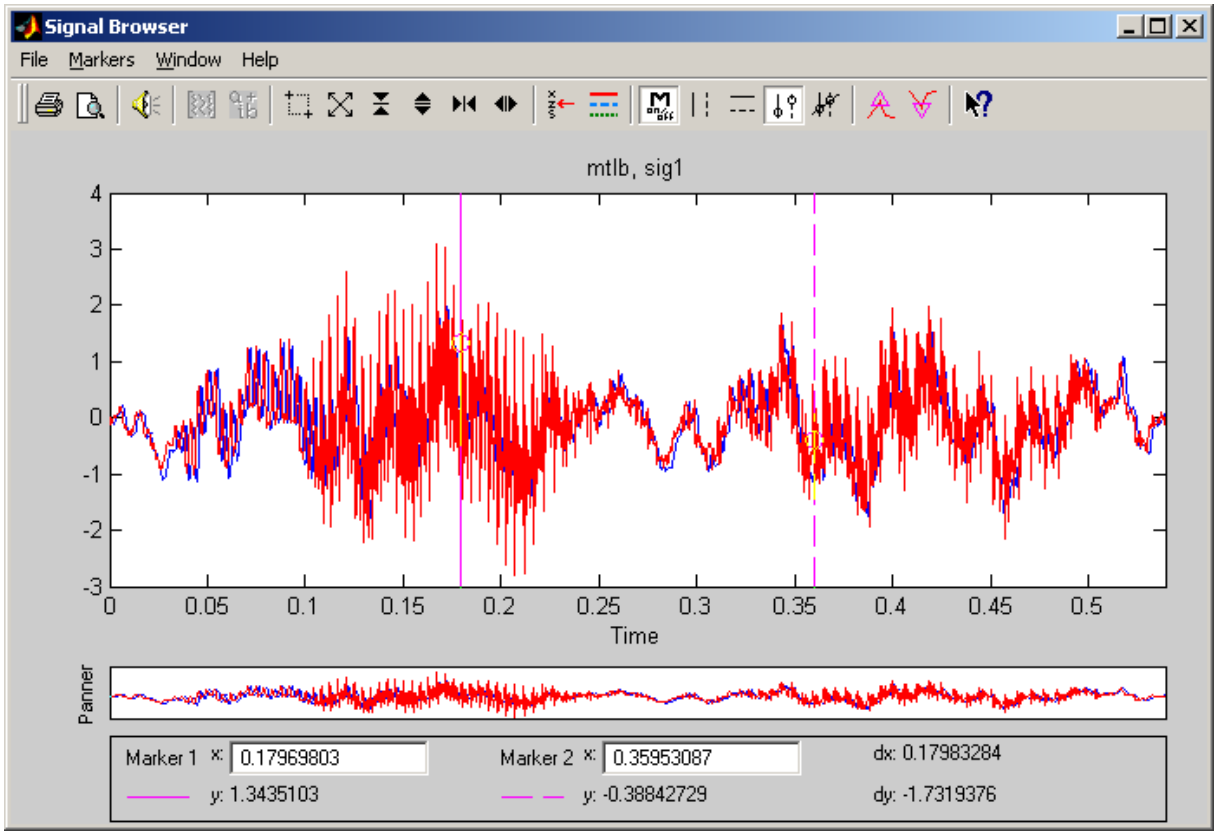


Рисунок Г. 9 - Програма SPTool - одночасний перегляд графіків вхідного і вихідного сигналів фільтра

Аналогічним чином можна переглядати декілька спектрів або характеристики декількох фільтрів одночасно.

Збереження результатів роботи

Сеанс роботи з програмою SPTool можна зберегти за допомогою команди Save Session або Save Session As з меню File основного вікна програми SPTool. Файли сеансів мають розширення spt. Завантажити збережений сеанс можна командою Open Session того ж меню File. Крім того можна експортувати сигнали, фільтри і спектри у вигляді структур даних. Для цього використовується команда Export з меню File основного вікна програми SPTool. Після вибору даної команди з'явиться вікно Export from SPTool, показане на рис. Г. 10.

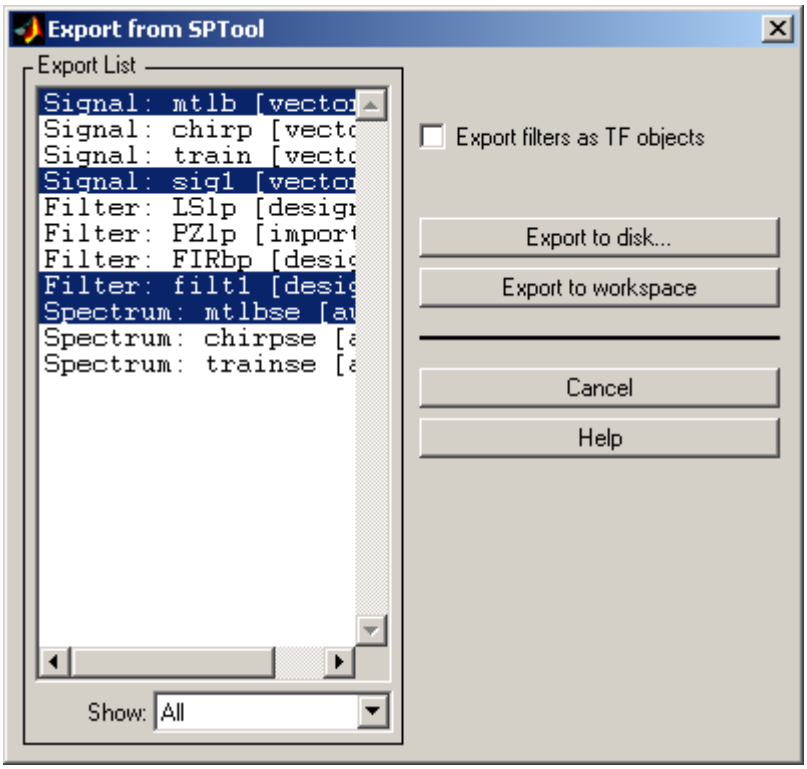


Рисунок Г. 10 - Програма SPTool - вікно експорту даних

У списку, що займає ліву частину цього вікна, необхідно вибрати об'єкти, що експортуються. Клацання на кнопці Export to workspace проведе експорт відповідних структур даних в робочу область пам’яті MATLAB, а для їх запису в MAT-файл необхідно скористатися кнопкою Export to disk.

Інформацію про склад структур даних, що експортуються, можна отримати з довідкової системи пакета Signal Processing.

Навчальне видання

Володимир Павлович Майданюк
Анатолій Михайлович Пєтух

ОБРОБКА СИГНАЛІВ

Навчальний посібник

Редактор В. Дружиніна
Коректор З. Поліщук
Оригінал-макет підготовлено В. Майданюком

Підписано до друк
Формат 29,7×42¼. Папір офсетний.
Гарнітура Times New Roman.
Друк різографічний. Ум. друк. арк.
Наклад 100 прим. Зам. №

Вінницький національний технічний університет,
науково-методичний відділ ВНТУ.
21021, м. Вінниця, Хмельницьке шосе, 95,
ВНТУ, к. 2201.
Тел. (0432) 59-87-36.
Свідоцтво суб'єкта видавничої справи
серія ДК №3516 від 01.07.2009 р.

Віддруковано у Вінницькому національному технічному університеті
в комп'ютерному інформаційно-видавничому центрі.
21021, м. Вінниця, Хмельницьке шосе, 95,
ВНТУ, ГНК, к. 114.
Тел. (0432) 59-87-38.
Свідоцтво суб'єкта видавничої справи
серія ДК №3516 від 01.07.2009 р.